链滴

coverity 部分规则提取

作者：nanolikeyou
原文链接：https://ld246.com/article/1539070494947
来源网站：链滴

全部checker

这里的json文件其实是--directive-file [安全选项] 获取包含若干用户配置指令的 JSON 文件的路径是一种DF.CUSTOM_CHECKER，可以照猫画虎自定义实现检查规则，支持java、c、JavaScript、noejs。

```
sink_for_checker
sink matching directive:
analysis/checkers/security/checkers/custom-dataflow/custom-dataflow.cpp
matched sanitizer for argument:
matched sanitizer for return value
Unexpected null pointer subs_msg
Unexpected null pointer sink_fn
Skipping sink due to 'Sanitize If Call In Same Method As Sink' directive (fn:
feasible inter-procedural path found
 The value is used unsafely in bytecode, which cannot be displayed.
A field as a sink is not supported.
ReadableProgramData is not supported yet.
WritableProgramData is not supported yet.
!dc->remediation_advice.empty()
str_equal(err.getIssueType(), dc->getIssueType())
ignoring sanitized value at sink
post_merge
post_clear
post_clone
post_assign
handleSanitizers
--
    "sink_for_checker" : "ANGULAR_EXPRESSION_INJECTION",
    "sink" : {
      "input" : "arg1", // of the caller function
      "to_callsite" : {
         "call_on" : {
            "from_mangled_function" : ".*__coverity_angjs__.Scope:.\\$watch$"
         }
      }
    },
    "requires_tainted_prefix" : false
  },
  // sink: [type Angular Scope].$watchCollection arg1
  {
    "sink_for_checker" : "ANGULAR_EXPRESSION_INJECTION",
    "sink" : {
      "input" : "arg1", // of the caller function
      "to_callsite" : {
         "call_on" : {
            "from_mangled_function" : ".*__coverity_angjs__.Scope:.\\$watchCollection$"
         }
      }
    },
    "requires_tainted_prefix" : false
  },
  // Bug 92845: sink: [type Angular Scope].$watchCollection arg1[any-array-index]
```

原文链接：coverity 部分规则提取

```
  // sink: [type Angular Scope].$http.$(eval|evalAsync|apply|applyAsync) arg1
  {
    "sink_for_checker" : "ANGULAR_EXPRESSION_INJECTION",
    "sink" : {
      "input" : "arg1", // of the caller function
      "to_callsite" : {
        "call_on" : {
          "from_mangled_function" : ".*__coverity_angjs__.Scope:.\\$(eval|evalAsync|apply|app
yAsync)$"
        }
      }
    },
    "requires_tainted_prefix" : false
  },
  // sink: [Angular service $interpolate or $parse] arg1
  {
    "sink_for_checker" : "ANGULAR_EXPRESSION_INJECTION",
    "sink" : {
      "input" : "arg1", // of the caller function
      "to_callsite" : {
        "call_on" : {
          "from_mangled_function" : ".*__coverity_angjs__.service_map.\\$(interpolate|parse)$

        }
      }
    },
    "requires_tainted_prefix" : false
  },
  // sink: [Angular service $sce].parseAs arg2
  {
    "sink_for_checker" : "ANGULAR_EXPRESSION_INJECTION",
    "sink" : {
      "input" : "arg2", // of the caller function
      "to_callsite" : {
        "call_on" : {
          "from_mangled_function" : ".*__coverity_angjs__.service_map.\\$sce.parseAs$"
        }
      }
    },
    "requires_tainted_prefix" : false
  },
  // sink: [Angular service $sce].parseAs(Html|Css|Url|ResourceUrl|Js) arg1
  {
    "sink_for_checker" : "ANGULAR_EXPRESSION_INJECTION",
    "sink" : {
      "input" : "arg1", // of the caller function
      "to_callsite" : {
        "call_on" : {
          "from_mangled_function" : ".*__coverity_angjs__.service_map.\\$sce.parseAs(Html|Cs
|Url|ResourceUrl|Js)$"
        }
      }
    },
    "requires_tainted_prefix" : false
```

```
    },
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
// 1) MongoDb actions requiring CSRF protection.
// 2) Mongoose actions requiring CSRF protection.
// 3) Sequelize actions requiring CSRF protection.
// 4) Bookshelf actions requiring CSRF protection.
// 5) orm actions requiring CSRF protection.
// 6) HANA XSC actions requiring CSRF protection.
// 7) Acl actions requiring CSRF protection.
// 8) Tedious actions requiring CSRF protection.
// 9) MSSql actions requiring CSRF protection.
--
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "write" } ],
            "read_from_object_of_type" : "HTMLDocument"
          }
        }
      }
    },
    // sink: [type HTMLDocument].writeln(x)
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "writeln" } ],
            "read_from_object_of_type" : "HTMLDocument"
          }
        }
      }
    },
    // sink: [type HTMLEmbedElement].setAttribute('src', code)
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg2",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "setAttribute" } ],
            "read_from_object_of_type" : "HTMLEmbedElement"
          },
          "when" : {
            "iregex_string" : "^(src)$",
            "only_if_arg_index" : 1
          }
        }
      },
    },
    // sink: [type HTMLEmbedElement].setAttributeNode('src', code)
```

```
// Deprecated
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNode" } ],
        "read_from_object_of_type" : "HTMLEmbedElement"
      },
      "when" : {
        "iregex_string" : "^(src)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLEmbedElement].setAttributeNS(namespace, 'src', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNS" } ],
        "read_from_object_of_type" : "HTMLEmbedElement"
      },
      "when" : {
        "iregex_string" : "^(src)$",
        "only_if_arg_index" : 2
      }
    }
  },
},
// sink: [type HTMLEmbedElement].setAttributeNodeNS(namespace, 'src', code)
// Deprecated
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNodeNS" } ],
        "read_from_object_of_type" : "HTMLEmbedElement"
      },
      "when" : {
        "iregex_string" : "^(src)$",
        "only_if_arg_index" : 2
      }
    }
  },
},
// sink: [type HTMLEmbedElement].src
{
```

```
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write" : [ { "property" : "src" } ],
    "write_to_object_of_type" : "HTMLEmbedElement"
  }
},
// sink: [type HTMLObjectElement].setAttribute('data', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttribute" } ],
        "read_from_object_of_type" : "HTMLObjectElement"
      },
      "when" : {
        "iregex_string" : "^(data)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLObjectElement].setAttributeNode('data', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNode" } ],
        "read_from_object_of_type" : "HTMLObjectElement"
      },
      "when" : {
        "iregex_string" : "^(data)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLObjectElement].setAttributeNS(namespace, 'data', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNS" } ],
        "read_from_object_of_type" : "HTMLObjectElement"
      },
      "when" : {
        "iregex_string" : "^(data)$",
        "only_if_arg_index" : 2
      }
```

```
    }
  },
},
// sink: [type HTMLObjectElement].setAttributeNodeNS(namespace, 'data', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNodeNS" } ],
        "read_from_object_of_type" : "HTMLObjectElement"
      },
      "when" : {
        "iregex_string" : "^(data)$",
        "only_if_arg_index" : 2
      }
    }
  },
},
// sink: [type HTMLObjectElement].data
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write" : [ { "property" : "data" } ],
    "write_to_object_of_type" : "HTMLObjectElement"
  }
},
// sink: [type HTMLScriptElement].setAttribute('innerText|textContent|text|src', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttribute" } ],
        "read_from_object_of_type" : "HTMLScriptElement"
      },
      "when" : {
        "iregex_string" : "^(innerText|textContent|text|src)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLScriptElement].setAttributeNode('innerText|textContent|text|src', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNode" } ],
        "read_from_object_of_type" : "HTMLScriptElement"
```

```
          },
          "when" : {
            "iregex_string" : "^(innerText|textContent|text|src)$",
            "only_if_arg_index" : 1
          }
        }
      },
    },
    // sink: [type HTMLScriptElement].setAttributeNS(namespace, 'innerText|textContent|text|sc', code)
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg3",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "setAttributeNS" } ],
            "read_from_object_of_type" : "HTMLScriptElement"
          },
          "when" : {
            "iregex_string" : "^(innerText|textContent|text|src)$",
            "only_if_arg_index" : 2
          }
        }
      },
    },
    // sink: [type HTMLScriptElement].setAttributeNodeNS(namespace, 'innerText|textContent text|src', code)
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg3",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "setAttributeNodeNS" } ],
            "read_from_object_of_type" : "HTMLScriptElement"
          },
          "when" : {
            "iregex_string" : "^(innerText|textContent|text|src)$",
            "only_if_arg_index" : 2
          }
        }
      },
    },
    // sink: [type HTMLScriptElement].innerText
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "write" : [ { "property" : "innerText" } ],
        "write_to_object_of_type" : "HTMLScriptElement"
      }
    },
    // sink: [type HTMLScriptElement].textContent
    {
```

```
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write" : [ { "property" : "textContent" } ],
    "write_to_object_of_type" : "HTMLScriptElement"
  }
},
// sink: [type HTMLScriptElement].text
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write" : [ { "property" : "text" } ],
    "write_to_object_of_type" : "HTMLScriptElement"
  }
},
// sink: [type HTMLScriptElement].src
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write" : [ { "property" : "src" } ],
    "write_to_object_of_type" : "HTMLScriptElement"
  },
  "requires_tainted_prefix" : true
},
// sink: [type HTMLStyleElement].setAttribute('innerText|textContent', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttribute" } ],
        "read_from_object_of_type" : "HTMLStyleElement"
      },
      "when" : {
        "iregex_string" : "^(innerText|textContent)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLStyleElement].setAttributeNode('innerText|textContent', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNode" } ],
        "read_from_object_of_type" : "HTMLStyleElement"
      },
      "when" : {
        "iregex_string" : "^(innerText|textContent)$",
        "only_if_arg_index" : 1
      }
```

```
      }
    },
  },
  // sink: [type HTMLStyleElement].setAttributeNS(namespace, 'innerText|textContent', code

  {
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "input" : "arg3",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "setAttributeNS" } ],
          "read_from_object_of_type" : "HTMLStyleElement"
        },
        "when" : {
          "iregex_string" : "^(innerText|textContent)$",
          "only_if_arg_index" : 2
        }
      }
    },
  },
  // sink: [type HTMLStyleElement].setAttributeNodeNS(namespace, 'innerText|textContent',
code)
  {
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "input" : "arg3",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "setAttributeNodeNS" } ],
          "read_from_object_of_type" : "HTMLStyleElement"
        },
        "when" : {
          "iregex_string" : "^(innerText|textContent)$",
          "only_if_arg_index" : 2
        }
      }
    },
  },
  // sink: [type HTMLStyleElement].innerText
  {
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "write" : [ { "property" : "innerText" } ],
      "write_to_object_of_type" : "HTMLStyleElement"
    }
  },
  // sink: [type HTMLStyleElement].textContent
  {
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "write" : [ { "property" : "textContent" } ],
      "write_to_object_of_type" : "HTMLStyleElement"
    }
```

```
},
// sink: [type HTMLElement].setAttribute('on+', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttribute" } ],
        "read_from_object_of_type" : "HTMLElement"
      },
      "when" : {
        "iregex_string" : "^(on[a-z]{3,40})$",
        "only_if_arg_index" : 1
      }
    }
  }
},
// sink: [type HTMLElement].setAttribute('innerHTML|outerHTML', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttribute" } ],
        "read_from_object_of_type" : "HTMLElement"
      },
      "when" : {
        "iregex_string" : "^(innerHTML|outerHTML)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLElement].setAttributeNode('innerHTML|outerHTML', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNode" } ],
        "read_from_object_of_type" : "HTMLElement"
      },
      "when" : {
        "iregex_string" : "^(innerHTML|outerHTML)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLElement].setAttributeNS(namespace, 'innerHTML|outerHTML', code)
{
```

```
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "input" : "arg3",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "setAttributeNS" } ],
          "read_from_object_of_type" : "HTMLElement"
        },
        "when" : {
          "iregex_string" : "^(innerHTML|outerHTML)$",
          "only_if_arg_index" : 2
        }
      }
    },
  },
  // sink: [type HTMLElement].setAttributeNodeNS(namespace, 'innerHTML|outerHTML', co
e)
  {
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "input" : "arg3",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "setAttributeNodeNS" } ],
          "read_from_object_of_type" : "HTMLElement"
        },
        "when" : {
          "iregex_string" : "^(innerHTML|outerHTML)$",
          "only_if_arg_index" : 2
        }
      }
    },
  },
  // sink: [type HTMLElement].outerHTML
  // Should be for the HTML_PCDATA context.
  {
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "write" : [ { "property" : "outerHTML" } ],
      "write_to_object_of_type" : "HTMLElement"
    }
  },
  // sink: [type HTMLScriptElement].innerHTML
  // implied by sink: [type HTMLElement].innerHTML
  // sink: [type HTMLStyleElement].innerHTML
  // implied by [type HTMLElement].innerHTML
  // sink: [type HTMLElement].innerHTML
  // Should be for the HTML_PCDATA context.
  {
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "write" : [ { "property" : "innerHTML" } ],
      "write_to_object_of_type" : "HTMLElement"
    }
```

原文链接：coverity 部分规则提取

```
},
// sink: [type HTMLAnchorElement].setAttribute('href', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttribute" } ],
        "read_from_object_of_type" : "HTMLAnchorElement"
      },
      "when" : {
        "iregex_string" : "^(href)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLAnchorElement].setAttributeNode('href', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNode" } ],
        "read_from_object_of_type" : "HTMLAnchorElement"
      },
      "when" : {
        "iregex_string" : "^(href)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLAnchorElement].setAttributeNS(namespace, 'href', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNS" } ],
        "read_from_object_of_type" : "HTMLAnchorElement"
      },
      "when" : {
        "iregex_string" : "^(href)$",
        "only_if_arg_index" : 2
      }
    }
  },
},
// sink: [type HTMLAnchorElement].setAttributeNodeNS(namespace, 'href', code)
{
```

```
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNodeNS" } ],
        "read_from_object_of_type" : "HTMLAnchorElement"
      },
      "when" : {
        "iregex_string" : "^(href)$",
        "only_if_arg_index" : 2
      }
    }
  },
},
// sink: [type HTMLAnchorElement].href
// Should be for the URL context.
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write" : [ { "property" : "href" } ],
    "write_to_object_of_type" : "HTMLAnchorElement"
  },
  "requires_tainted_prefix" : true
},
// sink: [type HTMLAreaElement].setAttribute('href', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttribute" } ],
        "read_from_object_of_type" : "HTMLAreaElement"
      },
      "when" : {
        "iregex_string" : "^(href)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLAreaElement].setAttributeNode('href', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNode" } ],
        "read_from_object_of_type" : "HTMLAreaElement"
      },
      "when" : {
        "iregex_string" : "^(href)$",
```

原文链接：coverity 部分规则提取

```
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLAreaElement].setAttributeNS(namespace, 'href', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNS" } ],
        "read_from_object_of_type" : "HTMLAreaElement"
      },
      "when" : {
        "iregex_string" : "^(href)$",
        "only_if_arg_index" : 2
      }
    }
  },
},
// sink: [type HTMLAreaElement].setAttributeNodeNS(namespace, 'href', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNodeNS" } ],
        "read_from_object_of_type" : "HTMLAreaElement"
      },
      "when" : {
        "iregex_string" : "^(href)$",
        "only_if_arg_index" : 2
      }
    }
  },
},
// sink: [type HTMLAreaElement].href
// Should be for the URL context.
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write" : [ { "property" : "href" } ],
    "write_to_object_of_type" : "HTMLAreaElement"
  },
  "requires_tainted_prefix" : true
},
// sink: [type HTMLIFrameElement].setAttribute('src', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
```

```
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "setAttribute" } ],
      "read_from_object_of_type" : "HTMLIFrameElement"
     },
     "when" : {
      "iregex_string" : "^(src)$",
      "only_if_arg_index" : 1
     }
    }
   },
},
// sink: [type HTMLIFrameElement].setAttributeNode('src', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "setAttributeNode" } ],
      "read_from_object_of_type" : "HTMLIFrameElement"
     },
     "when" : {
      "iregex_string" : "^(src)$",
      "only_if_arg_index" : 1
     }
    }
  },
},
// sink: [type HTMLIFrameElement].setAttributeNS(namespace, 'src', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "setAttributeNS" } ],
      "read_from_object_of_type" : "HTMLIFrameElement"
     },
     "when" : {
      "iregex_string" : "^(src)$",
      "only_if_arg_index" : 2
     }
    }
  },
},
// sink: [type HTMLIFrameElement].setAttributeNodeNS(namespace, 'src', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "setAttributeNodeNS" } ],
```

```
        "read_from_object_of_type" : "HTMLIFrameElement"
      },
      "when" : {
        "iregex_string" : "^(src)$",
        "only_if_arg_index" : 2
      }
    }
  },
},
// sink: [type HTMLIFrameElement].src
// Should be for the URL context.
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write" : [ { "property" : "src" } ],
    "write_to_object_of_type" : "HTMLIFrameElement"
  },
  "requires_tainted_prefix" : true
},
// sink: [type HTMLFrameElement].setAttribute('src', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttribute" } ],
        "read_from_object_of_type" : "HTMLFrameElement"
      },
      "when" : {
        "iregex_string" : "^(src)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLFrameElement].setAttributeNode('src', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNode" } ],
        "read_from_object_of_type" : "HTMLFrameElement"
      },
      "when" : {
        "iregex_string" : "^(src)$",
        "only_if_arg_index" : 1
      }
    }
  },
},
// sink: [type HTMLFrameElement].setAttributeNS(namespace, 'src', code)
```

```
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNS" } ],
        "read_from_object_of_type" : "HTMLFrameElement"
      },
      "when" : {
        "iregex_string" : "^(src)$",
        "only_if_arg_index" : 2
      }
    }
  },
},
// sink: [type HTMLFrameElement].setAttributeNodeNS(namespace, 'src', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "setAttributeNodeNS" } ],
        "read_from_object_of_type" : "HTMLFrameElement"
      },
      "when" : {
        "iregex_string" : "^(src)$",
        "only_if_arg_index" : 2
      }
    }
  },
},
// sink: [type HTMLFrameElement].src
// Should be for the URL context.
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write" : [ { "property" : "src" } ],
    "write_to_object_of_type" : "HTMLFrameElement"
  },
  "requires_tainted_prefix" : true
},
// sink: [type Window].open(x)
// Should be under the URL context.
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "open" } ],
        "read_from_object_of_type" : "Window"
      }
```

```
    }
  },
  "requires_tainted_prefix" : true
},
// sink: [type Window].openDialog(x)
// Should be under the URL context.
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "openDialog" } ],
        "read_from_object_of_type" : "Window"
      }
    }
  },
  "requires_tainted_prefix" : true
},
// sink: [type HTMLElement.insertAdjacentHTML(pos, text source)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "insertAdjacentHTML" } ],
        "read_from_object_of_type" : "HTMLElement"
      }
    }
  }
},
// sink: location
// Note: we cannot have [type Location] as sink.
// Should be for the URL context when assigned to a string or URL.
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write_path_off_global" : [ { "property" : "location" } ]
  },
  "requires_tainted_prefix" : true
},
// sink: [type Location].href
// Should be for URL context.
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "write" : [ { "property" : "href" } ],
    "write_to_object_of_type" : "Location"
  },
  "requires_tainted_prefix" : true
},
// sink: [type Location].assign(x)
// Should be for x under the URL context.
```

原文链接：coverity 部分规则提取

```
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "assign" } ],
        "read_from_object_of_type" : "Location"
      }
    }
  },
  "requires_tainted_prefix" : true
},
// sink: [type Location].replace(x)
// Should be for x under the URL context.
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "replace" } ],
        "read_from_object_of_type" : "Location"
      },
      "when" : {
        "is_max_index" : true,
        "only_if_arg_index" : 1
      }
    }
  },
  "requires_tainted_prefix" : true
},
// sink: eval(source)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read_path_off_global" : [ { "property" : "eval" } ]
      }
    }
  }
},
// sink: execScript(source)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "last_arg",
    "to_callsite" : {
      "call_on" : {
        "read_path_off_global" : [ { "property" : "execScript" } ]
      }
    }
```

原文链接：coverity 部分规则提取

```
  }
},
// sink: new Function(source)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "last_arg",
    "to_callsite" : {
      "new_on" : {
        "read_path_off_global" : [ { "property" : "Function" } ]
      }
    }
  }
},
// sink: new GeneratorFunction(source)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "new_on" : {
        "read_path_off_global" : [ { "property" : "GeneratorFunction" } ]
      }
    }
  }
},
// sink: setTimeout(source)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "Window",
        "read" : [ { "property" : "setTimeout" } ]
      }
    }
  }
},
// sink: setInterval(source)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "Window",
        "read" : [ { "property" : "setInterval" } ]
      }
    }
  }
},
// sink: setImmediate(source)
{
```

```
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_from_object_of_type" : "Window",
            "read" : [ { "property" : "setImmediate" } ]
          }
        }
      }
    }
    /*
    // Former approximated sinks.
    // sink: *.innerHTML
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "write_off_any" : { "property" : "innerHTML" }
      }
    },
    // sink: *.innerText
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "write_off_any" : { "property" : "innerText" }
      }
    },
    // sink: *.textContent
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "write_off_any" : { "property" : "textContent" }
      }
    },
    */
  ]
},
  "type" : "Coverity analysis configuration",
  "format_version" : 4,
  "language" : "javascript",
  "directives" : [
    // 2) jQuery sinks.
    // sink: jQuery(arg) on mangled name
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
             "from_mangled_function" : ".*:jQuery"
          }
        }
      }
    },
```

原文链接：coverity 部分规则提取

```
// sink: jQuery(arg) on jQuery
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
         "read_path_off_global" : [ { "property" : "jQuery" } ]
      }
    }
  }
},
// sink: jQuery(arg) on $
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
         "read_path_off_global" : [ { "property" : "$" } ]
      }
    }
  }
},
// sink: [type JQueryStatic].globalEval(code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "globalEval" } ],
        "read_from_object_of_type" : "JQueryStatic"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQueryStatic].parseHTML(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "parseHTML" } ],
        "read_from_object_of_type" : "JQueryStatic"
      }
    }
  }
},
// sink: [type JQuery].attr('on+' , code)
{
  "sink_for_checker" : "DOM_XSS",
```

原文链接：coverity 部分规则提取

```
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "attr" } ],
        "read_from_object_of_type" : "JQuery"
      },
      "when" : {
        "iregex_string" : "^(on[a-z]{3,40})$",
        "only_if_arg_index" : 1
      }
    }
  }
},
// sink: [type JQuery].attr('src' or 'href', code)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "attr" } ],
        "read_from_object_of_type" : "JQuery"
      },
      "when" : {
        "iregex_string" : "^(src|href)$",
        "only_if_arg_index" : 1
      }
    }
  },
  "requires_tainted_prefix" : true
},
// No check on the type of arg1
// sink: [type JQuery].add(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "add" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].has(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
```

```
        "read" : [ { "property" : "has" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].constructor(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "constructor" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].init(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "init" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].index(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "index" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].wrapAll(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
```

```
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "wrapAll" } ],
      "read_from_object_of_type" : "JQuery"
     }
    }
   }
},
// No check on the type of arg1
// sink: [type JQuery].wrapInner(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "wrapInner" } ],
      "read_from_object_of_type" : "JQuery"
     }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].wrap(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "wrap" } ],
      "read_from_object_of_type" : "JQuery"
     }
    }
  }
},
// No check on the types of args
// sink: [type JQuery].append(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "all_args",
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "append" } ],
      "read_from_object_of_type" : "JQuery"
     }
    }
  }
},
// No check on the types of args
// sink: [type JQuery].prepend(html_string)
{
  "sink_for_checker" : "DOM_XSS",
```

原文链接：coverity 部分规则提取

```
  "sink" : {
    "input" : "all_args",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "prepend" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the types of args
// sink: [type JQuery].before(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "all_args",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "before" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the types of args
// sink: [type JQuery].after(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "all_args",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "after" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].html(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "html" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].replaceWith(html_string)
```

```
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "replaceWith" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].appendTo(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "appendTo" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].prependTo(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "prependTo" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].insertBefore(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "insertBefore" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
```

原文链接：coverity 部分规则提取

```
// No check on the type of arg1
// sink: [type JQuery].insertAfter(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "insertAfter" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// No check on the type of arg1
// sink: [type JQuery].replaceAll(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "replaceAll" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQueryStatic].get(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "get" } ],
        "read_from_object_of_type" : "JQueryStatic"
      }
    }
  },
  "requires_tainted_prefix" : true
},
// sink: [type JQueryStatic].post(html_string)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "post" } ],
        "read_from_object_of_type" : "JQueryStatic"
      }
    }
  },
```

```
      "requires_tainted_prefix" : true
    },
    // sink: [type JQueryStatic].getScript(html_string)
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "getScript" } ],
            "read_from_object_of_type" : "JQueryStatic"
          }
        }
      },
      "requires_tainted_prefix" : true
    },
    // sink: [type JQuery].load(html_string)
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "load" } ],
            "read_from_object_of_type" : "JQuery"
          }
        }
      },
      "requires_tainted_prefix" : true
    }
  ]
},
  "type" : "Coverity analysis configuration",
  "format_version" : 4,
  "language" : "javascript",
  "directives" : [
    // 3) jQuery-UI sinks.
    // sink: [type JQuery].datepicker(arg.altField)
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg1",
        "path" : [ { "property" : "altField" } ],
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "datepicker" } ],
            "read_from_object_of_type" : "JQuery"
          }
        }
      }
    },
    // sink: [type JQuery].datepicker(arg.appendText)
    {
      "sink_for_checker" : "DOM_XSS",
```

```
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "appendText" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "datepicker" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].datepicker(arg.buttonText)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "buttonText" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "datepicker" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].datepicker(arg.closeText)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "closeText" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "datepicker" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].datepicker(arg.currentText)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "currentText" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "datepicker" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].datepicker(arg.weekHeader)
```

```
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "weekHeader" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "datepicker" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].datepicker(arg.yearSuffix)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "yearSuffix" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "datepicker" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].button(arg.label)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "label" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "button" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].dialog(arg.closeText)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "closeText" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "dialog" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
```

```
},
// sink: [type JQuery].dialog(arg.title)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "title" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "dialog" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].tooltip(arg.content)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "content" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "tooltip" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].effect(xx, arg.to)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "property" : "to" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "effect" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].draggable(arg.appendTo)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "appendTo" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "draggable" } ],
        "read_from_object_of_type" : "JQuery"
      }
```

```
    }
  }
},
// sink: [type JQuery].draggable(arg.containment)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "containment" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "draggable" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].resizable(arg.alsoResize)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "alsoResize" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "resizable" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].resizable(arg.containment)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "containment" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "resizable" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].sortable(arg.appendTo)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "appendTo" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "sortable" } ],
```

```
          "read_from_object_of_type" : "JQuery"
        }
      }
    }
  },
  // sink: [type JQuery].sortable(arg.containment)
  {
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "containment" } ],
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "sortable" } ],
          "read_from_object_of_type" : "JQuery"
        }
      }
    }
  },
  // sink: [type JQuery].position(arg.of)
  {
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "of" } ],
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "position" } ],
          "read_from_object_of_type" : "JQuery"
        }
      }
    }
  },
  // sink: [type JQuery].position(arg.within)
  {
    "sink_for_checker" : "DOM_XSS",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "within" } ],
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "position" } ],
          "read_from_object_of_type" : "JQuery"
        }
      }
    }
  }
 ]
},
 "type" : "Coverity analysis configuration",
 "format_version" : 4,
 "language" : "javascript",
 "directives" : [
  // 4) jQuery-ajax sinks.
```

原文链接：coverity 部分规则提取

```
    //
    // Approximation: We can't check the type of arg
--
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "ajax" } ],
            "read_from_object_of_type" : "JQueryStatic"
          }
        }
      },
      "requires_tainted_prefix" : true
    },
    // sink: [type JQueryStatic].ajax(arg.url)      // Version 1.0
    // sink: [type JQueryStatic].ajax(url, arg.url) // Version 1.5
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "last_arg",
        "path" : [ { "property" : "url" } ],
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "ajax" } ],
            "read_from_object_of_type" : "JQueryStatic"
          }
        }
      },
      "requires_tainted_prefix" : true
    },
    // sink: [type JQueryStatic].ajax(arg.jsonpCallback)      // Version 1.0
    // sink: [type JQueryStatic].ajax(url, arg.jsonpCallback) // Version 1.5
    {
      "sink_for_checker" : "DOM_XSS",
      "sink" : {
        "input" : "last_arg",
        "path" : [ { "property" : "jsonpCallback" } ],
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "ajax" } ],
            "read_from_object_of_type" : "JQueryStatic"
          }
        }
      }
    }
  ]
},
  "type" : "Coverity analysis configuration",
  "format_version" : 4,
  "language" : "javascript",
  "directives" : [
    // 5) Bootstrap sinks. BZ 95227.
    // sink: [type JQuery].affix(arg.target)
```

```
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "target" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "affix" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].popover(arg.template)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "template" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "popover" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].popover(arg.viewport)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "viewport" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "popover" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].tooltip(arg.template)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "template" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "tooltip" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
```

原文链接：coverity 部分规则提取

```
},
// sink: [type JQuery].tooltip(arg.viewport)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "viewport" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "tooltip" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].modal(arg.remote)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "remote" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "modal" } ],
        "read_from_object_of_type" : "JQuery"
      }
    }
  }
},
// sink: [type JQuery].popver(arg.html, arg.content)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "content" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "popover" } ],
        "read_from_object_of_type" : "JQuery"
      },
    }
  }
},
// sink: [type JQuery].popver(arg.html, arg.title)
{
  "sink_for_checker" : "DOM_XSS",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "title" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "popover" } ],
        "read_from_object_of_type" : "JQuery"
      },
```

```
        }
       }
      },
      // sink: [type JQuery].tooltip(arg.html, arg.title)
      {
       "sink_for_checker" : "DOM_XSS",
       "sink" : {
        "input" : "arg1",
        "path" : [ { "property" : "title" } ],
        "to_callsite" : {
         "call_on" : {
          "read" : [ { "property" : "tooltip" } ],
          "read_from_object_of_type" : "JQuery" },
        }
       }
      },
    ]
  },
    "type" : "Coverity analysis configuration",
    "format_version" : 5,
    "language" : "javascript",
    "directives" : [
      // JSON -> [type JSONStatic]
      // Model is identical to the one in the report for bug 94888. Nothing new
      // or modified seems necessary
      {
--
       "sink_for_checker" : "EL_INJECTION",
       "sink" : {
        "methods" : {
         "matching" : "org\\.springframework\\.web\\.servlet\\.tags\\.EvalTag\\.setExpression\\(
ava\\.lang\\.String\\).*"
        },
        "param_index" : 1
       }
      },
      // ### SpEL
      // http://docs.spring.io/spring/docs/3.1.4.RELEASE/javadoc-api/org/springframework/exp
ession/spel/standard/SpelExpressionParser.html
      /*
      // protected.
      //   Also, only implemented in InternalSpelExpressionParser.doParseExpression.
      //   That in turn calls SpelExpression.<init>(String, ...), which is modeled.
      //   Ignoring.
      {
       "sink_for_checker": "EL_INJECTION",
       "sink" : {
        "methods": {
         "matching" : "org\\.springframework\\.expression\\.spel\\.(standard\\.SpelExpressionP
rser|common\\.TemplateAwareExpressionParser)\\.doParseExpression\\(java\\.lang\\.String.*"
        },
        "param_index": 1
       }
      },
```

原文链接：coverity 部分规则提取

```
    */
    // public
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.springframework\\.expression\\.spel\\.(standard\\.SpelExpression
arser|common\\.TemplateAwareExpressionParser)\\.parseRaw\\(java\\.lang\\.String.*"
          }
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.expression\\.spel\\.standard\\.SpelExpression\\
<init>\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "and" : [
            {
              "implemented_in_class" : {
                "with_super" : {
                  "matching" : "org\\.springframework\\.expression\\.ExpressionParser.*"
                }
              }
            },
            {
              "matching" : ".*parseExpression\\(java\\.lang\\.String.*"
            }
          ]
        },
        "param_index" : 1
      }
    },
    // #### Variant of SpEL, bean access
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.context\\.expression\\.StandardBeanExpression
esolver\\.evaluate\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
```

```
    },
    // ### MVEL
    // Not being explicit on types since they use String, char[], Object, etc.
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.mvel(2)?\\.MVEL\\.(eval|evalToString|evalToBoolean|compileExpres
ion|executeExpression)\\(.*"
        },
        "param_index" : 1
      }
    },
    // MVEL1 Only
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.mvel\\.TemplateInterpreter\\.(eval|evalToString|parse|<init>)\\(.*"
        },
        "param_index" : 1
      }
    },
    // MVEL2 Only
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.mvel2\\.MVELInterpretedRuntime\\.<init>\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.mvel2\\.templates\\.TemplateRuntime\\.(<init>|eval)\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.mvel2\\.templates\\.TemplateCompiler\\.(compileTemplate|<init>)
\(.*"
        },
        "param_index" : 1
      }
    },
    // ### Apache commons EL
    // http://commons.apache.org/proper/commons-el/
```

原文链接：coverity 部分规则提取

```
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.apache\\.commons\\.el\\.ExpressionEvaluatorImpl\\.(parse(AndR
nder|ExpressionString))\\(java\\.lang\\.String.*"
          }
        },
        "param_index" : 1
      }
    },
    /*
    // Commons EL JavaDoc is a lie; there's no 1st param as Object in the jar...
    // http://commons.apache.org/proper/commons-el/apidocs/org/apache/commons/el/Exp
essionEvaluatorImpl.html#evaluate(java.lang.Object, java.lang.Class, javax.servlet.jsp.el.Variabl
Resolver, javax.servlet.jsp.el.FunctionMapper)
    {
      "sink_for_checker": "EL_INJECTION",
      "sink" : {
        "methods": {
          "overrides": {
            "matching" : "org\\.apache\\.commons\\.el\\.ExpressionEvaluatorImpl\\.evaluate\\(ja
a\\.lang\\.Object.*"
          }
        },
        "param_index": 1
      }
    },
    */
    // ### Apache commons JEXL
    // JEXL2
    // http://commons.apache.org/proper/commons-jexl/
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.commons\\.jexl2\\.(JexlEngine|UnifiedJEXL)\\.(createExpr
ssion|createScript|parse)\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.commons\\.jexl2\\.UnifiedJEXL\\$Template\\.<init>\\(.*"
        },
        "param_index" : 2
      }
    },
    // Workarounds for bz 50829, 57354. These methods just pass the tainted parameter to th
 above Template ctor
```

```
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.commons\\.jexl2\\.UnifiedJEXL\\.createTemplate\\(java\\.
ang\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.commons\\.jexl2\\.UnifiedJEXL\\.createTemplate\\(java\\.
ang\\.String, java\\.io\\.Reader.*"
        },
        "param_index" : 2
      }
    },
    // JEXL1
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.apache\\.commons\\.jexl\\.ScriptFactory\\.createScript\\(.*"
          }
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.apache\\.commons\\.jexl\\.ExpressionFactory\\.createExpression\\
java\\.lang\\.String\\).*"
          }
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "EL_INJECTION",
      "sink" : {
        "methods" : {
          "and" : [
            {
              "implemented_in_class" : {
                "with_super" : {
                  "matching" : "org\\.apache\\.commons\\.jexl\\.JexlExprResolver.*"
                }
```

```
            }
          },
          {
            "matching" : ".*evaluate\\(org\\.apache\\.commons\\.jexl\\.JexlContext, java\\.lang\\
String\\).*"
          }
        ]
      },
      "param_index" : 2
    }
  },
  // ### javax.el
  // shouldn't be able to arbitrarily control EL string
--
    "sink_for_checker" : "EL_INJECTION",
    "sink" : {
      "methods" : {
        "overrides" : {
          "matching" : "javax\\.el\\.ExpressionFactory\\.create(Value|Method)Expression\\(javax
\.el\\.ELContext, java\\.lang\\.String.*"
        }
      },
      "param_index" : 2
    }
  },
  {
    "sink_for_checker" : "EL_INJECTION",
    "sink" : {
      "methods" : {
        "overrides" : {
          "matching" : "javax\\.servlet\\.jsp\\.el\\.ExpressionEvaluator\\.(evaluate|parseExpressi
n)\\(.*"
        }
      },
      "param_index" : 1
    }
  }
  /*
  // Commented out below for now. These are abstract classes that need to have
  // some type of implementation. The above sink should be good enough and
  // benefits from ignoring the implementation in case it doesn't do anything.
  //
  // shouldn't be able to arbitrarily control EL string
  {
    "sink_for_checker": "EL_INJECTION",
    "sink" : {
      "methods": {
        "matching" : "javax\\.el\\.VariableMapper\\.(set|resolve)Variable\\(java\\.lang\\.String.*

      },
      "param_index": 1
    }
  },
  // shouldn't be able to arbitrarily control prefix
```

原文链接：coverity 部分规则提取

```
    {
      "sink_for_checker": "EL_INJECTION",
      "sink" : {
        "methods": {
          "matching" : "javax\\.el\\.FunctionMapper\\.resolveFunction\\(java\\.lang\\.String, java
\.lang\\.String\\).*"
        },
        "param_index": 1
      }
    },
    // shouldn't be able to arbitrarily control method name
    {
      "sink_for_checker": "EL_INJECTION",
      "sink" : {
        "methods": {
          "matching" : "javax\\.el\\.FunctionMapper\\.resolveFunction\\(java\\.lang\\.String, java
\.lang\\.String\\).*"
        },
        "param_index": 1
      }
    }
    */
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 7,
  "language" : "any",
  "directives" : [
    //
    // # HTTP Header Injection
    // We create two checkers to separate the injection
    // in the name of the header or in the content.
    // Note that the content of the header is only
    // important when it's something like `cookie`, etc.
--
      "sink_for_checker" : "HEADER_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "javax\\.servlet\\.http\\.HttpServletResponse(Wrapper)?\\.(add|set)Heade
\\(.*"
        },
        "param_index" : 1
      }
    },
    // ### Spring MVC
    {
      "sink_for_checker" : "HEADER_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.http\\.HttpHeaders\\.(set|add)\\(.*"
        },
        "param_index" : 1
      }
    },
```

```
    // ### Vert.x
    {
      "sink_for_checker" : "HEADER_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "io\\.vertx\\.core\\.http\\.HttpServerResponse\\.putHeader\\(.*"
        },
        "param_index" : 1
      }
    },
    /*
    // This sink:
    // ```
    //   org.springframework.http.HttpHeaders.setAll()
    // ```
    // takes a Map<String, String> in parameter. It's
    // unlikely to have the keys tainted, so we assign
    // this sink to the HEADER_VALUE_INJECTION only.
    {
      "sink_for_checker": "HEADER_INJECTION",
      "sink" : {
        "methods": {
          "matching" : "org\\.springframework\\.http\\.HttpHeaders\\.setAll\\(.*"
        },
        "param_index": 2
      }
    },
    */
    // ### Struts 1 and Struts 2
    // There doesn't seem to be any method to set the
    // headers. They play directly with the JEE
    // ### JSF
    // NOP
    // ### GWT
    {
      "sink_for_checker" : "HEADER_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.google\\.gwt\\.http\\.client\\.RequestBuilder\\.setHeader\\(.*"
        },
        "param_index" : 1
      }
    }
  ]
},
// ### C# ASP.NET directives
  "type" : "Coverity analysis configuration",
  "format_version" : 4,
  "language" : "C#",
  "directives" : [
    // ## Sinks
    {
      "sink_for_checker" : "HEADER_INJECTION",
      "sink" : {
```

```
        "methods" : {
          "named" : "System.Web.HttpResponse::AddHeader(System.String, System.String)Syst
m.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "HEADER_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpResponse::AppendHeader(System.String, System.String)S
stem.Void"
        },
        "param_index" : 1
      }
    },
    // ### Note: there is no inheritence relationship between HttpResponse and HttpRespons
Base
    //     See MSDN .NET HttpResponseWrapper for more details
    //     Also note that ASP.NET MVC Controller::Response is HttpResponseBase
    //     The short version is that this class 'contains-a' HttpResponse rather than 'is-a'
    {
      "sink_for_checker" : "HEADER_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpResponseBase::AddHeader(System.String, System.String)
ystem.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "HEADER_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpResponseBase::AppendHeader(System.String, System.Stri
g)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "HEADER_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpResponseWrapper::AddHeader(System.String, System.Stri
g)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "HEADER_INJECTION",
```

```
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpResponseWrapper::AppendHeader(System.String, System
String)System.Void"
        },
        "param_index" : 1
      }
    }
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
// Table of Contents:
// 0) Custom Dataflow Checker specification for HEADER_INJECTION_BUDA.
// 1) JavaScript HEADER_INJECTION_BUDA sinks.
// 0) Custom Dataflow Checker specification for HEADER_INJECTION_BUDA.
  "type" : "Coverity analysis configuration",
  "format_version" : 10,
  "language" : "any",
  "directives" : [
    {
      "dataflow_checker_name" : "HEADER_INJECTION",
      "dataflow_checker_internal_name" : "__HEADER_INJECTION_BUDA",
--
      "sink_for_checker" : "HEADER_INJECTION",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "setRequestHeader" } ],
            "read_from_object_of_type" : "XMLHttpRequest"
          }
        }
      }
    },
    // 2) Built-in HEADER_INJECTION_BUDA_SERVERJS sinks.
    // [type:ExpressResponse].append(<xx>, val)
    {
      "sink_for_checker" : "HEADER_INJECTION",
        "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "append" } ],
            "read_from_object_of_type" : "ExpressResponse"
          }
        }
      }
    },
    // [type:ExpressResponse].type(arg1)
    {
      "sink_for_checker" : "HEADER_INJECTION",
        "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
```

```
        "read" : [ { "property" : "type" } ],
        "read_from_object_of_type" : "ExpressResponse"
      }
    }
  }
},
// [type:ExpressResponse].set(<xx>, val)
{
  "sink_for_checker" : "HEADER_INJECTION",
    "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "set" } ],
        "read_from_object_of_type" : "ExpressResponse"
      }
    }
  }
},
// TODO: to be enabled by Bug 86837
// // [type:ExpressResponse].set({ <xx> : val } )
// {
//   "sink_for_checker" : "HEADER_INJECTION",
//     "sink" : {
//       "input" : "arg1",
//       "path" : [ { "any_property_key" : true } ],
//       "to_callsite" : {
//         "call_on" : {
//           "read" : [ { "property" : "set" } ],
//           "read_from_object_of_type" : "ExpressResponse"
//         }
//       }
//     }
// },
// TODO: to be enabled by Bug 86837
// // [type:ExpressResponse].sendFile( { headers: { <xx> : val } } )
// {
//   "sink_for_checker" : "HEADER_INJECTION",
//     "sink" : {
//       "input" : "arg2",
//       "path" : [ { "property" : "headers" }, { "any_property_key" : true } ],
//       "to_callsite" : {
//         "call_on" : {
//           "read" : [ { "property" : "sendFile" } ],
//           "read_from_object_of_type" : "ExpressResponse"
//         }
//       }
//     }
// },
// TODO: to be enabled by Bug 86837
// // [type:ExpressResponse].links( { <xx> : val } )
// {
//   "sink_for_checker" : "HEADER_INJECTION",
//     "sink" : {
```

```
//         "input" : "arg1",
//         "path" : [ { "any_property_key" : true } ],
//         "to_callsite" : {
//            "call_on" : {
//               "read" : [ { "property" : "links" } ],
//               "read_from_object_of_type" : "ExpressResponse"
//            }
//         }
//     }
// },
// TODO: to be enabled by Bug 86837
// // [type:HttpResponse].format({ <xx> : val } )
// {
//   "sink_for_checker" : "HEADER_INJECTION",
//    "sink" : {
//         "input" : "arg1",
//         "path" : [ { "any_property_key" : true } ],
//         "to_callsite" : {
//            "call_on" : {
//               "read" : [ { "property" : "format" } ],
//               "read_from_object_of_type" : "ExpressResponse"
//            }
//         }
//     }
// },
// 2) for http.response
// TODO: to be enabled by Bug 86837
// [type:HttpResponse].addTrailers({ <xx> : val } )
// {
//   "sink_for_checker" : "HEADER_INJECTION",
//    "sink" : {
//         "input" : "arg1",
//         "path" : [ { "any_property_key" : true } ],
//         "to_callsite" : {
//            "call_on" : {
//               "read" : [ { "property" : "addTrailers" } ],
//               "read_from_object_of_type" : "http.ServerResponse"
//            }
//         }
//     }
// },
// TODO: to be enabled by Bug 86837
// [type:HttpResponse].writeHead(status, { <xx> : val } )
// {
//   "sink_for_checker" : "HEADER_INJECTION",
//    "sink" : {
//         "input" : "last_arg",
//         "path" : [ { "any_property_key" : true } ],
//         "to_callsite" : {
//            "call_on" : {
//               "read" : [ { "property" : "writeHead" } ],
//               "read_from_object_of_type" : "http.ServerResponse"
//            }
//         }
```

```
    //     }
    // },
    // [type:HttpResponse].setHeader(<xx>, val)
    {
      "sink_for_checker" : "HEADER_INJECTION",
       "sink" : {
       "input" : "arg1",
       "to_callsite" : {
         "call_on" : {
           "read" : [ { "property" : "setHeader" } ],
           "read_from_object_of_type" : "http.ServerResponse"
         }
       }
      }
    },
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
// 0) Custom Dataflow Checker specification for COOKIE_INJECTION.
// 1) COOKIE_INJECTION sinks.
  "type" : "Coverity analysis configuration",
  "format_version" : 8,
  "language" : "javascript",
  "directives" : [
    // 0) Custom Dataflow Checker specification for COOKIE_INJECTION.
    {
      "dataflow_checker_name" : "COOKIE_INJECTION",
--
      "sink_for_checker" : "COOKIE_INJECTION",
      "sink" : {
        "write" : [ { "property" : "cookie" } ],
        "write_to_object_of_type" : "HTMLDocument"
      },
      "requires_tainted_prefix" : true
    }
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 7,
    "language" : "any",
    "directives" : [
      //
      // # Java code injection checker
      // We'll use the same set of libraries as we have
      // for the XPath injection checker
      //
      // CWE ID: 95
      {
        "dataflow_checker_name" : "JAVA_CODE_INJECTION",
        "languages" : {
--
        "sink_for_checker" : "JAVA_CODE_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "javassist\\.CtMethod\\.make\\(java\\.lang\\.String.*"
          },
          "param_index" : 1
```

```
      }
    },
    {
      "sink_for_checker" : "JAVA_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "javassist\\.CtBehavior\\.(addCatch|insert(Before|After)|setBody)\\(java\\.l
ng\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "JAVA_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "javassist\\.CtBehavior\\.insertAt\\(int, java\\.lang\\.String\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "JAVA_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "javassist\\.CtBehavior\\.insertAt\\(int, boolean, java\\.lang\\.String\\).*"
        },
        "param_index" : 3
      }
    },
    // ### Sinks for BZ 57875
    {
      "sink_for_checker" : "JAVA_CODE_INJECTION",
      "sink" : {
        "methods" : {
          // This method is deprecated
          "matching" : "java\\.lang\\.ClassLoader\\.defineClass\\(byte\\[\\], int, int\\).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "JAVA_CODE_INJECTION",
      "sink" : {
        // Second parameter is the class definition
        "methods" : {
          "matching" : "java\\.lang\\.ClassLoader\\.defineClass\\(java\\.lang\\.String.*"
        },
        "param_index" : 2
      }
    }
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
```

```
    "format_version" : 7,
    "language" : "any",
    "directives" : [
      //
      // # JCR injection checker
      //
      // CWE ID: 20
      {
        "dataflow_checker_name" : "JCR_INJECTION",
--
        "sink_for_checker" : "JCR_INJECTION",
        "sink" : {
          "methods" : {
            "overrides" : {
              "matching" : "javax\\.jcr\\.query\\.QueryManager\\.createQuery\\(java\\.lang\\.String,
ava\\.lang\\.String\\).*"
            }
          },
          "param_index" : 1
        }
      }
    ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 7,
    "language" : "Java",
    "directives" : [
      //
      // # JSP dynamic include checker
      //
      // CWE ID: 94
      {
--
        "sink_for_checker" : "JSP_DYNAMIC_INCLUDE",
        "sink" : {
          "methods" : {
            "matching" : "org\\.apache\\.taglibs\\.standard\\.tag\\.(rt|el)\\.core\\.ImportTag\\.setUr
\\(.*"
          },
          "param_index" : 1
        }
      },
      // ### JSP tags
      // #### <jsp:include url="TAINTED" />
      {
        "sink_for_checker" : "JSP_DYNAMIC_INCLUDE",
        "sink" : {
          "methods" : {
            "matching" : "org\\.apache\\.jasper\\.runtime\\.JspRuntimeLibrary\\.include\\(.*"
          },
          "param_index" : 3
        }
      },
      {
```

原文链接：coverity 部分规则提取

```
    "sink_for_checker" : "JSP_DYNAMIC_INCLUDE",
    "sink" : {
     "methods" : {
      "overrides" : {
       "matching" : "javax\\.servlet\\.jsp\\.PageContext\\.(include|forward)\\(.*"
      }
     },
     "param_index" : 1
    }
   },
   // ### Struts1 tags
   // Note that these are technically are early warnings.
   // These get written to the output only when bean:write is invoked on these
   // properties
   // #### <bean:include page="TAINTED" />
   {
    "sink_for_checker" : "JSP_DYNAMIC_INCLUDE",
    "sink" : {
     "methods" : {
      "matching" : "org\\.apache\\.struts\\.taglib\\.bean\\.IncludeTag\\.setPage\\(.*"
     },
     "param_index" : 1
    }
   },
   // #### <bean:include href="TAINTED" />
   {
    "sink_for_checker" : "JSP_DYNAMIC_INCLUDE",
    "sink" : {
     "methods" : {
      "matching" : "org\\.apache\\.struts\\.taglib\\.bean\\.IncludeTag\\.setHref\\(.*"
     },
     "param_index" : 1
    }
   },
   // #### <bean:include forward="TAINTED" />
   {
    "sink_for_checker" : "JSP_DYNAMIC_INCLUDE",
    "sink" : {
     "methods" : {
      "matching" : "org\\.apache\\.struts\\.taglib\\.bean\\.IncludeTag\\.setForward\\(.*"
     },
     "param_index" : 1
    }
   },
   // #### <bean:include page="TAINTED" />
   // http://struts.apache.org/release/1.2.x/api/org/apache/struts/taglib/bean/IncludeTag.ht
l#setName(java.lang.String)
   // Deprecated method: refer above
   {
    "sink_for_checker" : "JSP_DYNAMIC_INCLUDE",
    "sink" : {
     "methods" : {
      "matching" : "org\\.apache\\.struts\\.taglib\\.bean\\.IncludeTag\\.setName\\(.*"
     },
```

```
        "param_index" : 1
      }
    },
    // ### Struts2 tags
    // #### <s:include value="TAINTED" />
    {
      "sink_for_checker" : "JSP_DYNAMIC_INCLUDE",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.struts2\\.components\\.Include\\.setValue\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "JSP_DYNAMIC_INCLUDE",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.struts2\\.views\\.jsp\\.IncludeTag\\.setValue\\(.*"
        },
        "param_index" : 1
      }
    }
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 7,
  "language" : "any",
  "directives" : [
    //
    // # JSP SQL injection inclusion
    //
    // CWE ID: 89
    {
      "dataflow_checker_name" : "JSP_SQL_INJECTION",
      "languages" : {
--
      "sink_for_checker" : "JSP_SQL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.taglibs\\.standard\\.tag\\.(rt|el)\\.sql\\.(QueryTag|Update
ag)\\.setSql\\(.*"
        },
        "param_index" : 1
      }
    }
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 7,
  "language" : "any",
  "directives" : [
    //
    // # LDAP injection checker
```

```
    //
    // CWE ID: 90
    {
      "dataflow_checker_name" : "LDAP_INJECTION",
      "languages" : {
--
      "sink_for_checker": "LDAP_INJECTION",
      "sink" : {
        "methods": {
          "and": [
            {
              "implemented_in_class": {
                "with_super": {
                  "matching" : "javax\\.naming\\.directory\\.DirContext.*"
                }
              }
            },
            {
              "matching": "search\\((java\\.lang\\.String|javax\\.naming\\.Name), java\\.lang\\.Stri
g"
            }
          ]
        },
        "param_index": 2
      }
    },
    */
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "javax\\.naming\\.directory\\.DirContext\\.search\\((java\\.lang\\.String|j
vax\\.naming\\.Name), java\\.lang\\.String.*"
          }
        },
        "param_index" : 2
      }
    },
    /*
    // Should be detected by the above
    {
      "sink_for_checker": "LDAP_INJECTION",
      "sink" : {
        "methods": {
          "matching" : "org\\.springframework\\.ldap\\.pool\\.DelegatingDirContext\\.search\\(.*

        },
        "param_index": 2
      }
    },
    */
    // ### Spring LDAP
    {
```

```
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.springframework\\.ldap\\.core\\.simple\\.SimpleLdapOperations\\
(authenticate|search|searchForObject)\\(.*"
          }
        },
        "param_index" : 2
      }
    },
    /*
    // Should be detected by the above
    {
      "sink_for_checker": "LDAP_INJECTION",
      "sink" : {
        "methods": {
          "matching" : "org\\.springframework\\.ldap\\.core\\.simple\\.SimpleLdapTemplate\\.(a
thenticate|search|searchForObject)\\(.*"
        },
        "param_index": 2
      }
    },
    */
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.ldap\\.core\\.LdapTemplate\\.(authenticate|sear
h|searchForObject)\\(.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.ldap\\.filter\\.HardcodedFilter\\.<init>\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.springframework\\.ldap\\.odm\\.core\\.OdmManager\\.search\\(.
"
          }
        },
        "param_index" : 3
      }
```

```
    },
    /*
    // Should be detected by the above
    {
      "sink_for_checker": "LDAP_INJECTION",
      "sink" : {
        "methods": {
          "matching" : "org\\.springframework\\.ldap\\.odm\\.core\\.impl\\.OdmManagerImpl\\.
earch\\(.*"
        },
        "param_index": 3
      }
    },
    */
    // ### Spring Security
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.security\\.ldap\\.SpringSecurityLdapTemplate\\.
searchForSingleAttributeValues|searchForSingleEntry)\\(.*"
        },
        "param_index" : 2
      }
    },
    // ### Apache Directory LDAP
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.apache\\.directory\\.ldap\\.client\\.api\\.LdapConnection\\.search
\(.*"
          }
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.apache\\.directory\\.ldap\\.client\\.api\\.LdapAsyncConnection\\.
earchAsync\\(.*"
          }
        },
        "param_index" : 2
      }
    },
    // ### UnboundID
    // https://www.unboundid.com/products/ldap-sdk/docs/javadoc/index.html
    {
      "sink_for_checker" : "LDAP_INJECTION",
```

```
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "com\\.unboundid\\.ldap\\.sdk\\.LDAPInterface\\.search\\(java\\.lang\\.String, com\\.unboundid\\.ldap\\.sdk\\.SearchScope, java\\.lang\\.String.*"
          }
        },
        "param_index" : 3
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "com\\.unboundid\\.ldap\\.sdk\\.LDAPInterface\\.search\\(java\\.lang\\.String, com\\.unboundid\\.ldap\\.sdk\\.SearchScope, com\\.unboundid\\.ldap\\.sdk\\.DereferenePolicy, int, int, boolean, java\\.lang\\.String,.*"
          }
        },
        "param_index" : 7
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "com\\.unboundid\\.ldap\\.sdk\\.LDAPInterface\\.search\\(com\\.unboundid\\.ldap\\.sdk\\.SearchResultListener, java\\.lang\\.String, com\\.unboundid\\.ldap\\.sdk\\.SearchScope, java\\.lang\\.String,.*"
          }
        },
        "param_index" : 4
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "com\\.unboundid\\.ldap\\.sdk\\.LDAPInterface\\.search\\(com\\.unboundid\\.ldap\\.sdk\\.SearchResultListener, java\\.lang\\.String, com\\.unboundid\\.ldap\\.sdk\\.SearchScope, com\\.unboundid\\.ldap\\.sdk\\.DereferencePolicy, int, int, boolean, java\\.lang\\.String,.*"
          }
        },
        "param_index" : 8
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
```

```
        "overrides" : {
          "matching" : "com\\.unboundid\\.ldap\\.sdk\\.LDAPInterface\\.searchForEntry\\(.*"
        }
      },
      "param_index" : 3
    }
  },
  {
    "sink_for_checker" : "LDAP_INJECTION",
    "sink" : {
      "methods" : {
        "overrides" : {
          "matching" : "com\\.unboundid\\.ldap\\.sdk\\.LDAPInterface\\.searchForEntry\\(.*"
        }
      },
      "param_index" : 6
    }
  },
  {
    "sink_for_checker" : "LDAP_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "com\\.unboundid\\.ldap\\.sdk\\.migrate\\.ldapjdk\\.LDAPConnection\\.s
arch\\(java\\.lang\\.String, int, java\\.lang\\.String.*"
      },
      "param_index" : 3
    }
  },
  {
    "sink_for_checker" : "LDAP_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "com\\.unboundid\\.ldap\\.sdk\\.migrate\\.ldapjdk\\.LDAPUrl\\.<init>\\(.*

      },
      "param_index" : 6
    }
  },
  {
    "sink_for_checker" : "LDAP_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "com\\.unboundid\\.ldap\\.sdk\\.SearchRequest\\.<init>\\(java\\.lang\\.St
ing, com\\.unboundid\\.ldap\\.sdk\\.SearchScope, java\\.lang\\.String.*"
      },
      "param_index" : 3
    }
  },
  {
    "sink_for_checker" : "LDAP_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "com\\.unboundid\\.ldap\\.sdk\\.SearchRequest\\.<init>\\(com\\.unbou
did\\.ldap\\.sdk\\.SearchResultListener, java\\.lang\\.String, com\\.unboundid\\.ldap\\.sdk\\.S
```

```
archScope, java\\.lang\\.String.*"
          },
          "param_index" : 4
        }
      },
      {
        "sink_for_checker" : "LDAP_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "com\\.unboundid\\.ldap\\.sdk\\.SearchRequest\\.<init>\\(java\\.lang\\.St
ing, com\\.unboundid\\.ldap\\.sdk\\.SearchScope, com\\.unboundid\\.ldap\\.sdk\\.DereferencPolicy, int, int, boolean, java\\.lang\\.String.*"
          },
          "param_index" : 7
        }
      },
      {
        "sink_for_checker" : "LDAP_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "com\\.unboundid\\.ldap\\.sdk\\.SearchRequest\\.<init>\\(com\\.unbou
did\\.ldap\\.sdk\\.SearchResultListener, java\\.lang\\.String, com\\.unboundid\\.ldap\\.sdk\\.SarchScope, com\\.unboundid\\.ldap\\.sdk\\.DereferencePolicy, int, int, boolean, java\\.lang\\.String.*"
          },
          "param_index" : 8
        }
      },
      {
        "sink_for_checker" : "LDAP_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "com\\.unboundid\\.ldap\\.sdk\\.SearchRequest\\.<init>\\(com\\.unbou
did\\.ldap\\.sdk\\.SearchResultListener, com\\.unboundid\\.ldap\\.sdk\\.Control\\[\\], java\\.lag\\.String, com\\.unboundid\\.ldap\\.sdk\\.SearchScope, com\\.unboundid\\.ldap\\.sdk\\.DererencePolicy, int, int, boolean, java\\.lang\\.String.*"
          },
          "param_index" : 9
        }
      },
      {
        "sink_for_checker" : "LDAP_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "com\\.unboundid\\.ldap\\.sdk\\.SearchRequest\\.setFilter\\(.*"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "LDAP_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "com\\.unboundid\\.ldap\\.sdk\\.Filter\\.create\\(.*"
```

```
        },
        "param_index" : 1
      }
    },
    // ------------------------------------------------
    // Sanitizer directives
    // ------------------------------------------------
    // The analysis doesn't have an LDAP context parser. This could cause some
    // FNs if the wrong encoder is applied to an incorrect context.
    /*
    // Pointless, the first param is not a trackable type
    {
      "sanitizer_for_checker" : "LDAP_INJECTION",
      "sanitizer" : {
        "methods" : {
          "matching" : "org\\.apache\\.directory\\.shared\\.ldap\\.model\\.filter\\.AbstractExpr
ode\\.escapeFilterValue\\("
        },
--
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::set_Filter(System.String)System
Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::.ctor(System.String)System.Vo
d"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::.ctor(System.DirectoryServices
DirectoryEntry,System.String)System.Void"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::.ctor(System.String,System.Strin
g[])System.Void"
```

```
        },
        "param_index" : 1
      }
    },
    // XXX: Controlling the retrieved properties is a defect, too.
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::.ctor(System.String,System.Stri
g[])System.Void"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::.ctor(System.DirectoryServices
DirectoryEntry,System.String,System.String[])System.Void"
        },
        "param_index" : 2
      }
    },
    // XXX: Controlling the retrieved properties is a defect, too.
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::.ctor(System.DirectoryServices
DirectoryEntry,System.String,System.String[])System.Void"
        },
        "param_index" : 3
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::.ctor(System.String,System.Stri
g[],System.DirectoryServices.SearchScope)System.Void"
        },
        "param_index" : 1
      }
    },
    // XXX: Controlling the retrieved properties is a defect, too.
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::.ctor(System.String,System.Stri
g[],System.DirectoryServices.SearchScope)System.Void"
        },
```

```
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::.ctor(System.DirectoryServices
DirectoryEntry,System.String,System.String[],System.DirectoryServices.SearchScope)System.Vo
d"
        },
        "param_index" : 2
      }
    },
    // XXX: Controlling the retrieved properties is a defect, too.
    {
      "sink_for_checker" : "LDAP_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.DirectoryServices.DirectorySearcher::.ctor(System.DirectoryServices
DirectoryEntry,System.String,System.String[],System.DirectoryServices.SearchScope)System.Vo
d"
        },
        "param_index" : 3
      }
    },
    // ------------------------------------------------
    // Sanitizer directives
    // ------------------------------------------------
    // XXX: Deprecated
    {
      "sanitizer_for_checker" : "LDAP_INJECTION",
      "sanitizer" : {
        "methods" : {
          "named" : "Microsoft.Security.Application.Encoder::LdapEncode(System.String)System.
tring"
        },
        "param_index" : 1
      }
    },
--
      "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
      "sink" : {
        "input" : "arg1",
        "path" : [ {"property": "insertOne"}, {"property": "document"}, { "any_property" : true } ],
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "bulkWrite" } ],
            "read_from_object_of_type" : "MongoDbCollection"
          },
        }
      }
    },
    {
```

```
      "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
      "sink" : {
        "input" : "arg1",
        "path" : [ {"property": "updateOne"}, {"property": "update"}, {"property": "$set"}, { "any_p
operty" : true } ],
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "bulkWrite" } ],
            "read_from_object_of_type" : "MongoDbCollection"
          },
        }
      }
    },
    {
      "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
      "sink" : {
        "input" : "arg1",
        "path" : [ {"property": "updateMany"}, {"property": "update"}, {"property": "$set"}, { "any
property" : true } ],
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "bulkWrite" } ],
            "read_from_object_of_type" : "MongoDbCollection"
          },
        }
      }
    },
    {
      "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
      "sink" : {
        "input" : "arg1",
        "path" : [ {"property": "replaceOne"}, {"property": "replacement"}, { "any_property" : true
],
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "bulkWrite" } ],
            "read_from_object_of_type" : "MongoDbCollection"
          },
        }
      }
    },
    {
      "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "bulkWrite" } ],
            "read_from_object_of_type" : "MongoDbCollection"
          },
        }
      }
    },
    // sinks: [type MongoDBCollection].findAndModify(..,..,x)
```

原文链接：coverity 部分规则提取

```
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg3",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findAndModify" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      },
    }
  }
},
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg3",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findAndModify" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      },
    }
  }
},
// sinks: [type MongoDBCollection].findAndReplace(..,x)
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndReplace" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndReplace" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].findOneAndUpdate(..,x)
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
```

```
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndUpdate" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndUpdate" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].insert(x)
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "insert" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "insert" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].insertMany(x)
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg1",
```

```
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "insertMany" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     }
    }
   }
},
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "insertMany" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     }
    }
   }
},
// sinks: [type MongoDBCollection].insertOne(x)
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "insertOne" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     }
    }
   }
},
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "insertOne" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     }
    }
   }
},
// sinks: [type MongoDBCollection].replaceOne(..,x)
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
```

```
      "call_on" : {
        "read" : [ { "property" : "replaceOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "replaceOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].save(x)
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "save" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "save" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].update(..,x)
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "update" } ],
```

```
      "read_from_object_of_type" : "MongoDbCollection"
     }
    }
   }
},
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "update" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].updateMany(..,x)
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "updateMany" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "updateMany" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].updateOne(..,x)
{
  "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "updateOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
```

原文链接：coverity 部分规则提取

```
        }
       }
      },
      {
       "sink_for_checker" : "MONGODB_MASS_ASSIGNMENT",
       "sink" : {
        "input" : "arg2",
        "to_callsite" : {
         "call_on" : {
          "read" : [ { "property" : "updateOne" } ],
          "read_from_object_of_type" : "MongoDbCollection"
         }
        }
       }
      },
    ]
  },
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 10,
    "language" : "javascript",
    "directives" : [
     // Custom Dataflow Checker specification for MONGODB_QUERY_INJECTION.
     {
       "dataflow_checker_name" : "MONGODB_QUERY_INJECTION",
--
       "sink_for_checker" : "MONGODB_QUERY_INJECTION",
       "sink" : {
        "input" : "arg1",
        "to_callsite" : {
         "call_on" : {
          "read" : [ { "property" : "bulkWrite" } ],
          "read_from_object_of_type" : "MongoDbCollection"
         },
        }
       }
      },
      {
       "sink_for_checker" : "MONGODB_QUERY_INJECTION",
       "sink" : {
        "input" : "arg1",
        "path" : [ { "property" : "updateOne"}, { "property" : "filter"}, {"any_property": true} ],
        "to_callsite" : {
         "call_on" : {
          "read" : [ { "property" : "bulkWrite" } ],
          "read_from_object_of_type" : "MongoDbCollection"
         },
        }
       }
      },
      {
       "sink_for_checker" : "MONGODB_QUERY_INJECTION",
       "sink" : {
        "input" : "arg1",
```

```
    "path" : [ { "property" : "updateMany"}, { "property" : "filter"}, {"any_property": true} ],
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "bulkWrite" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     },
    }
   }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "deleteOne"}, { "property" : "filter"}, {"any_property": true} ],
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "bulkWrite" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     },
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "deleteMany"}, { "property" : "filter"}, {"any_property": true} ],
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "bulkWrite" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     },
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "replaceOne"}, { "property" : "filter"}, {"any_property": true} ],
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "bulkWrite" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     },
    }
  }
},
// sinks: [type MongoDBCollection].count(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
```

```
      "call_on" : {
        "read" : [ { "property" : "count" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      },
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "count" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      },
    }
  }
},
// sinks: [type MongoDBCollection].deleteMany(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "deleteMany" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "deleteMany" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].deleteOne(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
```

```
        "read" : [ { "property" : "deleteOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "deleteOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].distinct(..,x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "distinct" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "distinct" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].find(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "find" } ],
        "read_from_object_of_type" : "MongoDbCollection"
```

```
      }
     }
    }
  },
  {
    "sink_for_checker" : "MONGODB_QUERY_INJECTION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "find" } ],
          "read_from_object_of_type" : "MongoDbCollection"
        }
      }
    }
  },
  // sinks: [type MongoDBCollection].findAndModify(x)
  {
    "sink_for_checker" : "MONGODB_QUERY_INJECTION",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "findAndModify" } ],
          "read_from_object_of_type" : "MongoDbCollection"
        }
      }
    }
  },
  {
    "sink_for_checker" : "MONGODB_QUERY_INJECTION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "findAndModify" } ],
          "read_from_object_of_type" : "MongoDbCollection"
        }
      }
    }
  },
  // sinks: [type MongoDBCollection].findAndRemove(x)
  {
    "sink_for_checker" : "MONGODB_QUERY_INJECTION",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "findAndRemove" } ],
          "read_from_object_of_type" : "MongoDbCollection"
        }
      }
    }
```

原文链接：coverity 部分规则提取

```
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findAndRemove" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].findOne(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].findOneAndDelete(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndDelete" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
```

```
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndDelete" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].findOneAndReplace(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndReplace" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndReplace" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].findOneAndUpdate(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndUpdate" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
```

```
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndUpdate" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].group(..,x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "group" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "group" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
// sinks: [type MongoDBCollection].mapReduce(..,..,x)
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg3",
    "path" : [ {"property": "query"}, { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "mapReduce" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg3",
```

```
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "mapReduce" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     }
    }
   }
},
// sinks: [type MongoDBCollection].remove(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "remove" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "remove" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     }
    }
  }
},
// sinks: [type MongoDBCollection].removeMany(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
     "call_on" : {
      "read" : [ { "property" : "removeMany" } ],
      "read_from_object_of_type" : "MongoDbCollection"
     }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
     "call_on" : {
```

```
        "read" : [ { "property" : "removeMany" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].removeOne(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "removeOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "removeOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].replaceOne(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "replaceOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "replaceOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
```

```
      }
    }
  }
},
// sinks: [type MongoDBCollection].updateMany(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "updateMany" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "updateMany" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
// sinks: [type MongoDBCollection].updateOne(x)
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "path" : [ { "any_property" : true } ],
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "updateOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
  }
},
{
  "sink_for_checker" : "MONGODB_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "updateOne" } ],
        "read_from_object_of_type" : "MongoDbCollection"
      }
    }
```

原文链接：coverity 部分规则提取

```
        }
      },
    ]
  },
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 7,
    "language" : "any",
    "directives" : [
      // ## Checker definition.
      {
        "dataflow_checker_name" : "NOSQL_QUERY_INJECTION",
--
        "sink_for_checker" : "NOSQL_QUERY_INJECTION",
        "sink" : {
          "methods" : {
            "overrides" : {
              "matching" : "com\\.mongodb\\.DB\\.(doEval|eval)\\(.*"
            }
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "NOSQL_QUERY_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "org\\.bson\\.BsonDocument\\.parse\\(.*"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "NOSQL_QUERY_INJECTION",
        "sink" : {
          "methods" : {
            "named" : "com.mongodb.client.model.Filters.where(java.lang.String)org.bson.convers
ons.Bson"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "NOSQL_QUERY_INJECTION",
        "sink" : {
          "methods" : {
            "named" : "com.mongodb.DBCollection.group(com.mongodb.DBObject, com.mongo
b.DBObject, com.mongodb.DBObject, java.lang.String)com.mongodb.DBObject"
          },
          "param_index" : 4
        }
      },
      {
        "sink_for_checker" : "NOSQL_QUERY_INJECTION",
```

原文链接：coverity 部分规则提取

```
      "sink" : {
        "methods" : {
          "named" : "com.mongodb.DBCollection.group(com.mongodb.DBObject, com.mongo
b.DBObject, com.mongodb.DBObject, java.lang.String, java.lang.String)com.mongodb.DBObje
t"
        },
        "param_index" : 4
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "com.mongodb.DBCollection.group(com.mongodb.DBObject, com.mongo
b.DBObject, com.mongodb.DBObject, java.lang.String, java.lang.String)com.mongodb.DBObje
t"
        },
        "param_index" : 5
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "com.mongodb.DBCollection.group(com.mongodb.DBObject, com.mongo
b.DBObject, com.mongodb.DBObject, java.lang.String, java.lang.String, com.mongodb.ReadPr
ference)com.mongodb.DBObject"
        },
        "param_index" : 4
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "com.mongodb.DBCollection.group(com.mongodb.DBObject, com.mongo
b.DBObject, com.mongodb.DBObject, java.lang.String, java.lang.String, com.mongodb.ReadPr
ference)com.mongodb.DBObject"
        },
        "param_index" : 5
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.mongodb\\.(async\\.)?client\\.MongoCollection\\.mapReduce\\(ja
a\\.lang\\.String, java\\.lang\\.String\\).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
```

```
      "sink" : {
        "methods" : {
          "matching" : "com\\.mongodb\\.(async\\.)?client\\.MongoCollection\\.mapReduce\\(ja
a\\.lang\\.String, java\\.lang\\.String\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.mongodb\\.(async\\.)?client\\.MongoCollection\\.mapReduce\\(ja
a\\.lang\\.String, java\\.lang\\.String, java\\.lang\\.Class\\).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.mongodb\\.(async\\.)?client\\.MongoCollection\\.mapReduce\\(ja
a\\.lang\\.String, java\\.lang\\.String, java\\.lang\\.Class\\).*"
        },
        "param_index" : 2
      }
    },
    // ### Morphia
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "org.bson.types.CodeWScope.<init>(java.lang.String, org.bson.BSONObject
void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "org.mongodb.morphia.query.Query.where(java.lang.String)org.mongodb.
orphia.query.Query"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "org.mongodb.morphia.query.Query.filter(java.lang.String, java.lang.Object)
```

```
rg.mongodb.morphia.query.Query"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "NOSQL_QUERY_INJECTION",
        "sink" : {
          "methods" : {
            "named" : "org.mongodb.morphia.query.FieldEnd.contains(java.lang.String)java.lang.O
ject"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "NOSQL_QUERY_INJECTION",
        "sink" : {
          "methods" : {
            "named" : "org.mongodb.morphia.query.FieldEnd.containsIgnoreCase(java.lang.String)
ava.lang.Object"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "NOSQL_QUERY_INJECTION",
        "sink" : {
          "methods" : {
            "named" : "org.mongodb.morphia.Datastore.mapReduce(org.mongodb.morphia.Mapr
duceType, org.mongodb.morphia.query.Query, java.lang.String, java.lang.String, java.lang.Stri
g, java.util.Map, java.lang.Class)org.mongodb.morphia.MapreduceResults"
          },
          "param_index" : 3
        }
      },
      {
        "sink_for_checker" : "NOSQL_QUERY_INJECTION",
        "sink" : {
          "methods" : {
            "named" : "org.mongodb.morphia.Datastore.mapReduce(org.mongodb.morphia.Mapr
duceType, org.mongodb.morphia.query.Query, java.lang.String, java.lang.String, java.lang.Stri
g, java.util.Map, java.lang.Class)org.mongodb.morphia.MapreduceResults"
          },
          "param_index" : 4
        }
      },
      {
        "sink_for_checker" : "NOSQL_QUERY_INJECTION",
        "sink" : {
          "methods" : {
            "named" : "org.mongodb.morphia.Datastore.mapReduce(org.mongodb.morphia.Mapr
duceType, org.mongodb.morphia.query.Query, java.lang.String, java.lang.String, java.lang.Stri
g, java.util.Map, java.lang.Class)org.mongodb.morphia.MapreduceResults"
```

```
        },
        "param_index" : 5
      }
    },
    // ### Jongo
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "org.jongo.MongoCollection.find(java.lang.String)org.jongo.Find"
        },
        "param_index" : 1
      }
    },
    // ### CouchDB
    // #### jcouchdb
    // http://fforw.de/static/jcouchdb-javadoc/
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.jcouchdb\\.document\\.View\\.<init>\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.jcouchdb\\.document\\.View\\.<init>\\(.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.jcouchdb\\.document\\.View\\.(setMap|setReduce)\\(.*"
        },
        "param_index" : 1
      }
    },
    // #### couchdb4j
    // https://code.google.com/p/couchdb4j/
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.fourspaces\\.couchdb\\.AdHocView\\.<init>\\(java\\.lang\\.String\
).*"
        },
        "param_index" : 1
```

原文链接：coverity 部分规则提取

```
      }
    },
    // ### Couchbase
    // http://www.couchbase.com/autodocs/couchbase-java-client-1.2.0/index.html
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.couchbase\\.client\\.protocol\\.views\\.(Spatial)?ViewDesign\\.<ini
>\\(.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.couchbase\\.client\\.protocol\\.views\\.ViewDesign\\.<init>\\(.*"
        },
        "param_index" : 3
      }
    },
    // ### Neo4j
    // http://components.neo4j.org/neo4j/stable/apidocs/index.html
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.neo4j\\.cypher\\.javacompat\\.ExecutionEngine\\.(profile|execute)\
(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    }
  ]
 },
 // ## C# directives
  "type" : "Coverity analysis configuration",
  "format_version" : 4,
  "language" : "C#",
  "directives" : [
    // ## Sinks
    // ### Amazon SimpleDB
    // https://aws.amazon.com/simpledb/
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Amazon.SimpleDB.Model.SelectRequest::set_SelectExpression(System.Strin
)System.Void"
        },
        "param_index" : 1
      }
```

```
    },
    // ### Apache Cassandra
    // https://en.wikipedia.org/wiki/Apache_Cassandra
    // Variant: DataStax C# driver
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Cassandra.Data.CqlCommand::set_CommandText(System.String)System.Vo
d"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Cassandra.SimpleStatement::.ctor(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Cassandra.SimpleStatement::.ctor(System.String,System.Object[])System.Vo
d"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Cassandra.SimpleStatement::SetQueryString(System.String)Cassandra.Simp
eStatement"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "Cassandra\\.Session::(BeginExecute|Execute|Prepare|PrepareAsync)\\(Sys
em\\.String.*"
        },
        "param_index" : 1
      }
    },
    // Variant: Cassandra-Sharp
```

```
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          // XXX: Using a regex to avoid dependency on generic placeholder name
          "matching" : "CassandraSharp\\.ICqlCommand::Execute`1\\(System\\.String,System\\.
bject,CassandraSharp\\.PartitionKey\\)CassandraSharp\\.IQuery`1.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          // XXX: Using a regex to avoid dependency on generic placeholder name
          "matching" : "CassandraSharp\\.ICqlCommand::Prepare`1\\(System\\.String\\)Cassand
aSharp\\.IPreparedQuery`1.*"
        },
        "param_index" : 1
      }
    },
    // ### MongoDB
    // http://api.mongodb.org/csharp
    // XXX: I'm a little unsure of this technology, but it appears that Javascript code is execute

    //      on the server to evaluate queries.  If there are *other* uses of BsonJavaScript, there
ight
    //      be false positives here.
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "MongoDB.Bson.BsonJavaScript::.ctor(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "MongoDB.Bson.BsonJavaScript::Create(System.Object)MongoDB.Bson.Bson
avaScript"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "MongoDB.Bson.BsonJavaScriptWithScope::.ctor(System.String,MongoDB.B
on.BsonDocument)System.Void"
```

```
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "MongoDB.Bson.BsonJavaScriptWithScope::Create(System.Object)MongoDB
Bson.BsonJavaScriptWithScope"
        },
        "param_index" : 1
      }
    },
    // ### Redis
    // Variant: redis-sharp
    // https://github.com/migueldeicaza/redis-sharp
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "Redis::Send(Command|Expect|Get).*\\(System\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "Redis::SendData(Command|Expect).*\\(System\\.Byte\\[\\],System\\.Strin
.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Redis::StoreSetCommands(System.String,System.String[])System.Void"
        },
        "param_index" : 1
      }
    },
    // Variant: ServiceStack
    // https://github.com/ServiceStack/ServiceStack
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "ServiceStack\\.Redis\\.I?RedisClient::(ExecLua.*|ExecCachedLua|LoadLuaS
ript)\\(System\\.String.*"
        },
```

```
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "ServiceStack\\.Redis\\.I?RedisNativeClient::(Eval.*|ScriptLoad)\\(System\\
String.*"
        },
        "param_index" : 1
      }
    },
    // Variant: StackExchange
    // https://github.com/StackExchange/StackExchange.Redis
    {
      "sink_for_checker" : "NOSQL_QUERY_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "StackExchange.Redis.LuaScript::Prepare(System.String)StackExchange.Redis
LuaScript"
        },
        "param_index" : 1
      }
    }
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
 * Revision History
 * May 2017 - Bug 103331:
 *      Inital support for SAP's HANA XS classic (XSC).
 * 7/28/2017 - added PHP/Python support
////////////////////////////////////////////////////////////////////////////////
// Custom Dataflow Checker specification for NOSQL_QUERY_INJECTION_BUDA.
  "type" : "Coverity analysis configuration",
  "format_version" : 10,
  "language" : "any",
  "directives" : [
    {
--
    "sink_for_checker" : "NOSQL_QUERY_INJECTION",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "$where" } ],
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "find" } ],
          "read_from_object_of_type" : "MongoDbCollection"
        }
      }
    }
  },
  // 1b) Mongoose sinks.
  // sink: [type Module.mongoose.Model].$where(arg1)
  {
```

```
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$where" } ],
        "read_from_object_of_type" : "Module.mongoose.Model"
      }
    }
  }
},
// sink: [type Module.mongoose.Model].count(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "count" } ],
        "read_from_object_of_type" : "Module.mongoose.Model"
      }
    }
  }
},
// sink: [type Module.mongoose.Model].distinct(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "distinct" } ],
        "read_from_object_of_type" : "Module.mongoose.Model"
      }
    }
  }
},
// sink: [type Module.mongoose.Model].find(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "find" } ],
        "read_from_object_of_type" : "Module.mongoose.Model"
      }
    }
  }
},
// sink: [type Module.mongoose.Model].findOne(arg1.$where)
```

```
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOne" } ],
        "read_from_object_of_type" : "Module.mongoose.Model"
      }
    }
  }
},
// sink: [type Module.mongoose.Model].findOneAndRemove(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndRemove" } ],
        "read_from_object_of_type" : "Module.mongoose.Model"
      }
    }
  }
},
// sink: [type Module.mongoose.Model].findOneAndUpdate(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndUpdate" } ],
        "read_from_object_of_type" : "Module.mongoose.Model"
      }
    }
  }
},
// sink: [type Module.mongoose.Model].remove(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "remove" } ],
        "read_from_object_of_type" : "Module.mongoose.Model"
      }
    }
  }
```

原文链接：coverity 部分规则提取

```
},
// sink: [type Module.mongoose.Model].update(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "update" } ],
        "read_from_object_of_type" : "Module.mongoose.Model"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].$where(arg1)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$where" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].and(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "and" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].count(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "count" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
```

```
  }
},
// sink: [type Module.mongoose.Query].distinct(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "distinct" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].elemMatch(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "elemMatch" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].find(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "find" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].findOne(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOne" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
```

```
      }
    }
  }
},
// sink: [type Module.mongoose.Query].findOneAndRemove(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndRemove" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].findOneAndUpdate(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "findOneAndUpdate" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].merge(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "merge" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].nor(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
```

```
        "read" : [ { "property" : "nor" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].or(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "or" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].remove(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "remove" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// sink: [type Module.mongoose.Query].update(arg1.$where)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "$where" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "update" } ],
        "read_from_object_of_type" : "Module.mongoose.Query"
      }
    }
  }
},
// 2) HANA XSC sinks.
// sink: [type SAPHanaXSDSEntity].$delete(<x>)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
```

```
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "SAPHanaXSDSEntity",
        "read" : [ { "property" : "$delete" } ]
      },
    }
  }
},
// sink: [type SAPHanaXSDSEntity].$find(<x>)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "SAPHanaXSDSEntity",
        "read" : [ { "property" : "$find" } ]
      },
    }
  }
},
// sink: [type SAPHanaXSDSEntity].$findAll(<x>)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "SAPHanaXSDSEntity",
        "read" : [ { "property" : "$findAll" } ]
      },
    }
  }
},
// sink: [type SAPHanaXSDSEntity].$select(<x>)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "SAPHanaXSDSEntity",
        "read" : [ { "property" : "$select" } ]
      },
    }
  }
},
// sink: [type SAPHanaXSDSQuery].$matching(<x>)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
```

```
      "read_from_object_of_type" : "SAPHanaXSDSQuery",
      "read" : [ { "property" : "$matching" } ]
    }
   }
  }
},
// sink: [type SAPHanaXSDSQuery].$order(<x>)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "SAPHanaXSDSQuery",
        "read" : [ { "property" : "$order" } ]
      }
    }
  }
},
// sink: [type SAPHanaXSDSQuery].$where(<x>)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "SAPHanaXSDSQuery",
        "read" : [ { "property" : "$where" } ]
      }
    }
  }
},
// 3) HANA XSA sinks.
// sink: [type SAPXSA_CDS_Transaction].$get(entity, key, callback)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$get" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Transaction"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Transaction].$get(entity, key, callback)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
```

```
        "read" : [ { "property" : "$get" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Transaction"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Transaction].$find(entity, condition, callback)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$find" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Transaction"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Transaction].$find(entity, condition, callback)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$find" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Transaction"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Transaction].$getAll(refs, callback)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$getAll" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Transaction"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Transaction].$getAll(refs, callback)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
```

```
        "read" : [ { "property" : "$getAll" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Transaction"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Transaction].$findAll(refs, callback)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$findAll" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Transaction"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Transaction].$findAll(refs, callback)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$findAll" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Transaction"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Transaction].$delete(entity, condition, callback)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg2",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$delete" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Transaction"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Transaction].$delete(entity, condition, callback)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg2",
    "to_callsite" : {
      "call_on" : {
```

```
        "read" : [ { "property" : "$delete" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Transaction"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Query].$matching(template)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$matching" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Query"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Query].$matching(template)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$matching" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Query"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Query].$order(<x>)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "any_property" : true } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "$order" } ],
        "read_from_object_of_type" : "SAPXSA_CDS_Query"
      }
    }
  }
},
// sink: [type SAPXSA_CDS_Query].$order(<x>)
{
  "sink_for_checker" : "NOSQL_QUERY_INJECTION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
```

```
          "read" : [ { "property" : "$order" } ],
          "read_from_object_of_type" : "SAPXSA_CDS_Query"
        }
      }
    }
  },
  // sink: [type SAPXSA_CDS_Query].$where(cond)
  {
    "sink_for_checker" : "NOSQL_QUERY_INJECTION",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "$where" } ],
          "read_from_object_of_type" : "SAPXSA_CDS_Query"
        }
      }
    }
  },
  // sink: [type SAPXSA_CDS_Query].$where(cond)
  {
    "sink_for_checker" : "NOSQL_QUERY_INJECTION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "$where" } ],
          "read_from_object_of_type" : "SAPXSA_CDS_Query"
        }
      }
    }
  },
////////////////////////////////////////////////////////////////////////////
// End of directive list
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 7,
  "language" : "any",
  "directives" : [
    //
    // # OGNL Injection.
    //
--
      "sink_for_checker" : "OGNL_INJECTION",
      "sink" : {
        "methods" : {
          // http://struts.apache.org/release/2.3.x/xwork-core/apidocs/index.html
          "matching" : "com\\.opensymphony\\.(xwork2\\.ognl|xwork\\.util)\\.OgnlUtil\\.compile
\(.*"
        },
        "param_index" : 1
      }
    },
```

```
    {
      "sink_for_checker" : "OGNL_INJECTION",
      "sink" : {
        "methods" : {
          // Not tested...
          "matching" : "org\\.apache\\.commons\\.ognl\\.Ognl\\.(getValue|parseExpression)\\(.*"
        },
        "param_index" : 1
      }
    },
    // The TextProvider interface cannot be used for the below. Some implementations
    // like DefaultTextProvider are not susceptible to OGNL injection.
    // validator.DelegatingValidatorContext and its subclasses might will be TP if the class
    // passed in is an instance of ActionSupport.
    // The first param in the following three sinks is transposed to the second
    // param in their respective sinks below these. These are overridden usually
    // by other classes like ActionSupport.
    // If you're wondering why the regex seem to duplicate some class names...
    // xwork (WebWork) has ...xwork.validator.CompositeTextProvider
    // xwork2 (Struts2) has ...xwork2.CompositeTextProvider
    // http://struts.apache.org/release/2.3.x/xwork-core/apidocs/index.html
    {
      "sink_for_checker" : "OGNL_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "com\\.opensymphony\\.xwork2?\\.(TextProviderSupport|ActionSupport
CompositeTextProvider|validator\\.(CompositeTextProvider|DelegatingValidatorContext))\\.ge
Text\\(java\\.lang\\.String\\).*"
          }
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "OGNL_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "com\\.opensymphony\\.xwork2?\\.(TextProviderSupport|ActionSupport
CompositeTextProvider|validator\\.(CompositeTextProvider|DelegatingValidatorContext))\\.ge
Text\\(java\\.lang\\.String, java\\.lang\\.String\\[.*"
          }
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "OGNL_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "com\\.opensymphony\\.xwork2?\\.(TextProviderSupport|ActionSupport
CompositeTextProvider|validator\\.(CompositeTextProvider|DelegatingValidatorContext))\\.ge
```

```
Text\\(java\\.lang\\.String, java\\.util\\.List.*"
          }
        },
        "param_index" : 1
      }
    },
    // And these are the second param sinks for the above, just unlikely to
    // actually be hit upon
    {
      "sink_for_checker" : "OGNL_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "com\\.opensymphony\\.xwork2?\\.(TextProviderSupport|ActionSupport
CompositeTextProvider|validator\\.(CompositeTextProvider|DelegatingValidatorContext))\\.ge
Text\\(java\\.lang\\.String, java\\.lang\\.String\\).*"
          }
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "OGNL_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "com\\.opensymphony\\.xwork2?\\.(TextProviderSupport|ActionSupport
CompositeTextProvider|validator\\.(CompositeTextProvider|DelegatingValidatorContext))\\.ge
Text\\(java\\.lang\\.String, java\\.lang\\.String, java\\.lang\\.String\\[.*"
          }
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "OGNL_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "com\\.opensymphony\\.xwork2?\\.(TextProviderSupport|ActionSupport
CompositeTextProvider|validator\\.(CompositeTextProvider|DelegatingValidatorContext))\\.ge
Text\\(java\\.lang\\.String, java\\.lang\\.String, java\\.util\\.List.*"
          }
        },
        "param_index" : 2
      }
    },
    // Similar to the getText sinks, the second param is transposed to the forth
    // param to the sink below.
    {
      "sink_for_checker" : "OGNL_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.opensymphony\\.xwork2?\\.util\\.LocalizedTextUtil\\.findText\\(ja
```

```
a\\.lang\\.Class, java\\.lang\\.String, java\\.util\\.Locale\\).*"
          },
          "param_index" : 2
        }
      },
      {
        "sink_for_checker" : "OGNL_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "com\\.opensymphony\\.xwork2?\\.util\\.LocalizedTextUtil\\.findText\\(ja
a\\.lang\\.Class, java\\.lang\\.String, java\\.util\\.Locale, java\\.lang\\.String.*"
          },
          "param_index" : 4
        }
      },
      // Actual Struts2 OGNL evaluators. These are the true sinks, in addition to
      // the compilers above.
      {
        "sink_for_checker" : "OGNL_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "com\\.opensymphony\\.xwork2?\\.util\\.TextParseUtil\\.translateVariable
\\(java\\.lang\\.String.*"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "OGNL_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "com\\.opensymphony\\.xwork2?\\.util\\.TextParseUtil\\.translateVariable
\\(char, java\\.lang\\.String.*"
          },
          "param_index" : 2
        }
      },
      {
        "sink_for_checker" : "OGNL_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "com\\.opensymphony\\.xwork2?\\.util\\.TextParseUtil\\.translateVariable
\\(char\\[\\], java\\.lang\\.String.*"
          },
          "param_index" : 2
        }
      }
    ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
// -----
// Javascript sources and sinks
  "type" : "Coverity analysis configuration",
  "format_version" : 9,
  "language" : "javascript",
```

```
  "directives" : [
    //--------------------------
    //  Node.js
    //--------------------------
    // sink: [type: http.ServerResponse].writeHead(statusCode[,statusMessage][,headers])
    // to the field 'Location' of the headers parameter
--
        "sink_for_checker": "OPEN_REDIRECT",
        "sink": {
           "to_callsite": {
              "call_on": {
                 "read_from_object_of_type" : "http.ServerResponse",
                 "read": [ { "property": "writeHead" } ]
              },
              "when": {
                 "only_if_arg_index" : 2
              }
           },
           "input": "arg2",
           "path": [ { "property": "Location" } ]
        }
    },
    // sink: [type: http.ServerResponse].writeHead(statusCode[,statusMessage][,headers])
    // to the field 'Location' of the headers parameter
    {
        "sink_for_checker": "OPEN_REDIRECT",
        "sink": {
           "to_callsite": {
              "call_on": {
                 "read_from_object_of_type" : "http.ServerResponse",
                 "read": [ { "property": "writeHead" } ]
              },
              "when": {
                 "only_if_arg_index": 3
              }
           },
           "input": "arg3",
           "path": [ { "property": "Location" } ]
        }
    },
    // sink: [type: http.ServerResponse].setHeader('Location', 'redirect-url')
    {
        "sink_for_checker": "OPEN_REDIRECT",
        "sink": {
           "to_callsite": {
              "call_on": {
                 "read_from_object_of_type" : "http.ServerResponse",
                 "read": [ { "property": "setHeader" } ]
              },
              "when": {
                 "only_if_arg_index" : 1,
                  "regex_string" : "^Location$"
              }
           },
```

```
        "input": "arg2"
    }
},
// sink: [type: https.ServerResponse].writeHead(statusCode[,statusMessage][,headers])
// to the field 'Location' of the headers parameter
{
    "sink_for_checker": "OPEN_REDIRECT",
    "sink": {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "https.ServerResponse",
                "read": [ { "property": "writeHead" } ]
            },
            "when": {
                "only_if_arg_index" : 2
            }
        },
        "input": "arg2",
        "path": [ { "property": "Location" } ]
    }
},
// sink: [type: https.ServerResponse].writeHead(statusCode[,statusMessage][,headers])
// to the field 'Location' of the headers parameter
{
    "sink_for_checker": "OPEN_REDIRECT",
    "sink": {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "https.ServerResponse",
                "read": [ { "property": "writeHead" } ]
            },
            "when": {
                "only_if_arg_index": 3
            }
        },
        "input": "arg3",
        "path": [ { "property": "Location" } ]
    }
},
// sink: [type: https.ServerResponse].setHeader('Location', 'redirect-url')
{
    "sink_for_checker": "OPEN_REDIRECT",
    "sink": {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "https.ServerResponse",
                "read": [ { "property": "setHeader" } ]
            },
            "when": {
                "only_if_arg_index" : 1,
                 "regex_string" : "^Location$"
            }
        },
        "input": "arg2"
```

```
      }
    },
    //--------------------------
    //  Express
    //--------------------------
    // sink: [type ExpressResponse].redirect([status,]path)
    // redirect to the URL derived from the specified path
    {
      "sink_for_checker": "OPEN_REDIRECT",
      "sink": {
        "to_callsite": {
          "call_on": {
            "read_from_object_of_type" : "ExpressResponse",
            "read": [ { "property": "redirect" } ]
          }
        },
        "input": "last_arg"
      }
    },
    // sink: [type ExpressResponse].location(path)
    // sets the response Location HTTP header to the specified path
    {
      "sink_for_checker": "OPEN_REDIRECT",
      "sink": {
        "to_callsite": {
          "call_on": {
            "read_from_object_of_type" : "ExpressResponse",
            "read": [ { "property": "location" } ]
          }
        },
        "input": "arg1"
      }
    },
}, //Javascript
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 10,
    "language" : "any",
    "directives" : [
    // 0) Custom Dataflow Checker specification for OS_CMD_INJECTION_BUDA.
    {
      "dataflow_checker_name" : "OS_CMD_INJECTION",
      "dataflow_checker_internal_name" : "__OS_CMD_INJECTION_BUDA",
--
        "sink_for_checker" : "OS_CMD_INJECTION",
        "sink" : {
          "input" : "arg1",
          "to_callsite" : {
            "call_on" : {
              "path" : [ { "property" : "exec" } ],
              "read_from_js_require" : "child_process"
            }
          }
        }
```

```
},
// sink: require('child_process').exec(command, arg2.shell, callback)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg2",
        "path" : [ { "property" : "shell" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "exec" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').exec(command, arg2.env, callback)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg2",
        "path" : [ { "property" : "env" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "exec" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').execSync(arg1, options, callback)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execSync" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').execSync(command, arg2.shell, callback)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg2",
        "path" : [ { "property" : "shell" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execSync" } ],
                "read_from_js_require" : "child_process"
            }
        }
```

```
    }
},
// sink: require('child_process').execSync(command, arg2.env, callback)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg2",
        "path" : [ { "property" : "env" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execSync" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').execFile(arg1, args, options, callback)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFile" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').execFile(file, arg2.*, options, callback)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFile" } ],
                "read_from_js_require" : "child_process"
            }
        },
        "path" : [ { "any_property" : true } ]
    }
},
// sink: require('child_process').execFile(file, args, arg3.shell, callback)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "shell" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFile" } ],
                "read_from_js_require" : "child_process"
            }
```

```
        }
    }
},
// sink: require('child_process').execFile(file, args, arg3.env, callback)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "env" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFile" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').execFileSync(arg1, args, options)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFileSync" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').execFileSync(file, arg2.*, options)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFileSync" } ],
                "read_from_js_require" : "child_process"
            }
        },
        "path" : [ { "any_property" : true } ]
    }
},
// sink: require('child_process').execFileSync(file, args, arg3.shell)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "shell" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFileSync" } ],
                "read_from_js_require" : "child_process"
```

```
            }
        }
    }
},
// sink: require('child_process').execFileSync(file, args, arg3.env)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "env" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFileSync" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').fork(modulePath, arg2.*, arg3)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "fork" } ],
                "read_from_js_require" : "child_process"
            }
        },
        "path" : [ { "any_property" : true } ]
    }
},
// sink: require('child_process').fork(modulePath, args, arg3.shell)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "shell" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "fork" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').fork(modulePath, args, arg3.env)
{
    "sink_for_checker" : "OS_CMD_INJECTION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "env" } ],
        "to_callsite" : {
            "call_on" : {
```

```
            "path" : [ { "property" : "fork" } ],
            "read_from_js_require" : "child_process"
         }
      }
   }
},
// sink: require('child_process').spawn(arg1, args, options)
{
   "sink_for_checker" : "OS_CMD_INJECTION",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "path" : [ { "property" : "spawn" } ],
            "read_from_js_require" : "child_process"
         }
      }
   }
},
// sink: require('child_process').spawn(command, arg2.*, options)
{
   "sink_for_checker" : "OS_CMD_INJECTION",
   "sink" : {
      "input" : "arg2",
      "to_callsite" : {
         "call_on" : {
            "path" : [ { "property" : "spawn" } ],
            "read_from_js_require" : "child_process"
         }
      },
      "path" : [ { "any_property" : true } ]
   }
},
// sink: require('child_process').spawn(command, args, arg3.shell)
{
   "sink_for_checker" : "OS_CMD_INJECTION",
   "sink" : {
      "input" : "arg3",
      "path" : [ { "property" : "shell" } ],
      "to_callsite" : {
         "call_on" : {
            "path" : [ { "property" : "spawn" } ],
            "read_from_js_require" : "child_process"
         }
      }
   }
},
// sink: require('child_process').spawn(command, args, arg3.env)
{
   "sink_for_checker" : "OS_CMD_INJECTION",
   "sink" : {
      "input" : "arg3",
      "path" : [ { "property" : "env" } ],
      "to_callsite" : {
```

```
            "call_on" : {
               "path" : [ { "property" : "spawn" } ],
               "read_from_js_require" : "child_process"
            }
         }
      }
   },
   // sink: require('child_process').spawnSync(arg1, args, options)
   {
      "sink_for_checker" : "OS_CMD_INJECTION",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on" : {
               "path" : [ { "property" : "spawnSync" } ],
               "read_from_js_require" : "child_process"
            }
         }
      }
   },
   // sink: require('child_process').spawnSync(command, arg2.*, options)
   {
      "sink_for_checker" : "OS_CMD_INJECTION",
      "sink" : {
         "input" : "arg2",
         "to_callsite" : {
            "call_on" : {
               "path" : [ { "property" : "spawnSync" } ],
               "read_from_js_require" : "child_process"
            }
         },
         "path" : [ { "any_property" : true } ]
      }
   },
   // sink: require('child_process').spawnSync(command, args, arg3.shell)
   {
      "sink_for_checker" : "OS_CMD_INJECTION",
      "sink" : {
         "input" : "arg3",
         "path" : [ { "property" : "shell" } ],
         "to_callsite" : {
            "call_on" : {
               "path" : [ { "property" : "spawnSync" } ],
               "read_from_js_require" : "child_process"
            }
         }
      }
   },
   // sink: require('child_process').spawnSync(command, args, arg3.env)
   {
      "sink_for_checker" : "OS_CMD_INJECTION",
      "sink" : {
         "input" : "arg3",
         "path" : [ { "property" : "env" } ],
```

```
            "to_callsite" : {
               "call_on" : {
                  "path" : [ { "property" : "spawnSync" } ],
                  "read_from_js_require" : "child_process"
               }
            }
         }
      },
      // sink: require('cluster').setupMaster(arg1.exec)
      {
         "sink_for_checker" : "OS_CMD_INJECTION",
         "sink" : {
            "input" : "arg1",
            "path" : [ { "property" : "exec" } ],
            "to_callsite" : {
               "call_on" : {
                  "path" : [ { "property" : "setupMaster" } ],
                  "read_from_js_require" : "cluster"
               }
            }
         }
      },
      // sink: require('cluster').setupMaster(arg1.*)
      {
         "sink_for_checker" : "OS_CMD_INJECTION",
         "sink" : {
            "input" : "arg1",
            "path" : [ { "any_property" : true } ],
            "to_callsite" : {
               "call_on" : {
                  "path" : [ { "property" : "setupMaster" } ],
                  "read_from_js_require" : "cluster"
               }
            }
         }
      }
   ]
////////////////////////////////////////////////////////////////////////////
// End of directive list
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  // C# Directives
    "type" : "Coverity analysis configuration",
    "format_version" : 4,
    "language" : "C#",
    "directives" : [
--
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
         "matching" : "System\\.IO\\.DirectoryInfo::\\.ctor\\(System\\.String.*"
        },
        "param_index" : 1
      }
    },
```

原文链接：coverity 部分规则提取

```
// MoveTo(string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.DirectoryInfo::MoveTo\\(.*"
    },
    "param_index" : 1
  }
},
// C# System.IO.Directory
// ----------------------
// CreateDirectory(string)
// CreateDirectory(string, DirectorySecurity)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.Directory::CreateDirectory\\(.*"
    },
    "param_index" : 1
  }
},
// Delete(string)
// Delete(string, boolean)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.Directory::Delete\\(.*"
    },
    "param_index" : 1
  }
},
// Move(string, string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.Directory::Move\\(.*,.*"
    },
    "param_index" : 1
  }
},
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.Directory::Move\\(.*,.*"
    },
    "param_index" : 2
  }
},
// SetAccessControl(string, DirectorySecurity)
```

```
// SetCreationTime(string, DateTime)
// SetCreationTimeUtc(string, DateTime)
// SetCurrentDirectory(string)
// SetLastAccessTime(string, DateTime)
// SetLastAccessTimeUtc(string, DateTime)
// SetLastWriteTime(string, DateTime)
// SetLastWriteTimeUtc(string, DateTime)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.Directory::Set.*"
    },
    "param_index" : 1
  }
},
// C# System.IO.File
// -----------------
 // AppendAllLines(string, IEnumerable<string>)
// AppendAllLines(string, IEnumerable<string>, Encoding)
// AppendAllText(string, string)
// AppendAllText(string, string, Encoding)
// AppendText(string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Append.*"
    },
    "param_index" : 1
  }
},
// Copy(string, string)
// Copy(string, string, boolean)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Copy\\(.*,.*"
    },
    "param_index" : 1
  }
},
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Copy\\(.*,.*"
    },
    "param_index" : 2
  }
},
// Create(string)
// Create(string, int)
```

```
// Create(string, int, FileOptions)
// Create(string, int, FileOptions, FileSecurity)
// CreateText(string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Create.*"
    },
    "param_index" : 1
  }
},
// Delete(string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Delete\\(System\\.String\\).*"
    },
    "param_index" : 1
  }
},
// Move(string, string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Move\\(.*,.*"
    },
    "param_index" : 1
  }
},
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Move\\(.*,.*"
    },
    "param_index" : 2
  }
},
// Open(string, FileMode)
// Open(string, FileMode, FileAccess)
// Open(string, FileMode, FileAccess, FileShare)
// OpenRead(string)
// OpenText(string)
// OpenWrite(string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Open.*"
    },
    "param_index" : 1
```

```
  }
},
// ReadAllBytes(string)
// ReadAllLines(string)
// ReadAllLines(string, Encoding)
// ReadAllText(string)
// ReadAllText(string, Encoding)
// ReadLines(string)
// ReadLines(string, Encoding)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Read.*"
    },
    "param_index" : 1
  }
},
// Replace(string, string, string)
// Replace(string, string, string, boolean)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Replace\\(.*,.*,.*"
    },
    "param_index" : 1
  }
},
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Replace\\(.*,.*,.*"
    },
    "param_index" : 2
  }
},
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Replace\\(.*,.*,.*"
    },
    "param_index" : 3
  }
},
// SetAccessControl(string, FileSecurity)
// SetAccessControl(string, FileAttributes)
// SetCreationTime(string, DateTime)
// SetCreationTimeUtc(string, DateTime)
// SetLastAccessTime(string, DateTime)
// SetLastAccessTimeUtc(string, DateTime)
// SetLastWriteTime(string, DateTime)
```

```
// SetLastWriteTimeUtc(string, DateTime)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Set.*"
    },
    "param_index" : 1
  }
},
// WriteAllBytes(string, byte[])
// WriteAllLines(string, IEnumerable<string>)
// WriteAllLines(string, string[])
// WriteAllLines(string, IEnumerable<string>, Encoding)
// WriteAllLines(string, string[], Encoding)
// WriteAllText(string, string)
// WriteAllText(string, string, Encoding)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.File::Write.*"
    },
    "param_index" : 1
  }
},
// C# System.IO.FileInfo
// ---------------------
// FileInfo(string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.FileInfo::\\.ctor\\(System\\.String.*"
    },
    "param_index" : 1
  }
},
// MoveTo(string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.FileInfo::MoveTo\\(.*"
    },
    "param_index" : 1
  }
},
// Replace(string, string)
// Replace(string, string, bool)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
```

```
      "matching" : "System\\.IO\\.FileInfo::Replace\\(.*,.*"
    },
    "param_index" : 1
  }
},
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.FileInfo::Replace\\(.*,.*"
    },
    "param_index" : 2
  }
},
// C# System.IO.FileStream
// -----------------------
// FileStream(string, FileMode)
// FileStream(string, FileMode, FileAccess)
// FileStream(string, FileMode, FileAccess, FileShare)
// FileStream(string, FileMode, FileAccess, FileShare, Int32)
// FileStream(string, FileMode, FileAccess, FileShare, Int32, boolean)
// FileStream(string, FileMode, FileAccess, FileShare, Int32, FileOptions)
// FileStream(string, FileMode, FileSystemRights, FileShare, Int32, FileOptions)
// FileStream(string, FileMode, FileSystemRights, FileShare, Int32, FileOptions, FileSecurity)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.FileStream::\\.ctor\\(System\\.String.*"
    },
    "param_index" : 1
  }
},
// C# System.IO.Path
// -----------------
// Combine(string[])
// Combine(string, string)
// Combine(string, string, string)
// Combine(string, string, string, string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.Path::Combine\\(.*"
    },
    "param_index" : 1
  }
},
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.Path::Combine\\(.*,.*"
    },
```

```
    "param_index" : 2
   }
  },
  {
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
    "methods" : {
     "matching" : "System\\.IO\\.Path::Combine\\(.*,.*,.*"
    },
    "param_index" : 3
   }
  },
  {
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
    "methods" : {
     "matching" : "System\\.IO\\.Path::Combine\\(.*,.*,.*,.*"
    },
    "param_index" : 4
   }
  },
  // GetDirectoryName(string)
  {
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
    "methods" : {
     "matching" : "System\\.IO\\.Path::GetDirectoryName\\(System\\.String.*"
    },
    "param_index" : 1
   }
  },
  // GetFileName(string)
  {
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
    "methods" : {
     "matching" : "System\\.IO\\.Path::GetFileName\\(System\\.String.*"
    },
    "param_index" : 1
   }
  },
  // GetFileNameWithoutExtension(string)
  {
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
    "methods" : {
     "matching" : "System\\.IO\\.Path::GetFileNameWithoutExtension\\(System\\.String.*"
    },
    "param_index" : 1
   }
  },
  // GetFullPath(string)
  {
   "sink_for_checker" : "PATH_MANIPULATION",
```

```
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.Path::GetFullPath\\(System\\.String.*"
    },
    "param_index" : 1
  }
},
// GetPathRoot(string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.Path::GetPathRoot\\(System\\.String.*"
    },
    "param_index" : 1
  }
},
// C# System.IO.StreamReader
// -------------------------
// StreamReader(string)
// StreamReader(string, bool)
// StreamReader(string, Encoding)
// StreamReader(string, Encoding, bool)
// StreamReader(string, Encoding, bool, int)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.StreamReader::\\.ctor\\(System\\.String.*"
    },
    "param_index" : 1
  }
},
// C# System.IO.StreamWriter
// -------------------------
// StreamWriter(string)
// StreamWriter(string, bool)
// StreamWriter(string, bool, Encoding)
// StreamWriter(string, bool, Encoding, int)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "matching" : "System\\.IO\\.StreamWriter::\\.ctor\\(System\\.String.*"
    },
    "param_index" : 1
  }
},
// C# System.Web.HttpServerUtility
// -------------------------------
// MapPath(string)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
```

```
      "methods" : {
        "matching" : "System\\.Web\\.HttpServerUtility::MapPath.*"
      },
      "param_index" : 1
    }
  },
  // C# System.Web.UI.Page
  // ---------------------
  // MapPath(string)
  // MapPathSecure(string)
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "System\\.Web\\.UI\\.Page::MapPath.*"
      },
      "param_index" : 1
    }
  },
  // C# System.Web.UI.Control
  // ---------------------
  // OpenFile(string)
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "named" : "System.Web.UI.Control::OpenFile(System.String)System.IO.Stream"
      },
      "param_index" : 1
    }
  },
  // C# System.Web.UI.TemplateControl
  // ---------------------
  // LoadControl(string)
  // XXX: Could conceivably be a n UNRESTRICTED_DISPATCH sink instead.
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "named" : "System.Web.UI.TemplateControl::LoadControl(System.String)System.Web.U
.Control"
      },
      "param_index" : 1
    }
  },
  // LoadTemplate(string)
  // XXX: Could conceivably be a n UNRESTRICTED_DISPATCH sink instead.
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "named" : "System.Web.UI.TemplateControl::LoadTemplate(System.String)System.Web
UI.ITemplate"
      },
```

```
    "param_index" : 1
  }
},
// C# System.Web.UI.ServiceReference
// --------------------
// ServiceReference
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "named" : "System.Web.UI.ServiceReference::.ctor(System.String)System.Void"
    },
    "param_index" : 1
  }
},
// Path property
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "named" : "System.Web.UI.ServiceReference::set_Path(System.String)System.Void"
    },
    "param_index" : 1
  }
},
// C# System.Web.UI.ScriptReference
// --------------------
// ScriptReference
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "named" : "System.Web.UI.ScriptReference::.ctor(System.String)System.Void"
    },
    "param_index" : 1
  }
},
// Path property *** on base class ***
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "methods" : {
      "named" : "System.Web.UI.ScriptReferenceBase::set_Path(System.String)System.Void"
    },
    "param_index" : 1
  }
},
// ------------------------------------------------
// C# sanitizer directives
// ------------------------------------------------
// C# System.String
// --------------------
// IndexOf(char)
// IndexOf(string)
```

```
    // IndexOf(char, Int32)
    // IndexOf(string, Int32)
    // IndexOf(string, StringComparison)
    // IndexOf(char, Int32, Int32)
    // IndexOf(string, Int32, Int32)
    // IndexOf(string, Int32, StringComparison)
--
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.io\\.File\\.<init>\\(java\\.lang\\.String\\).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.io\\.File\\.<init>\\(java\\.lang\\.String, java\\.lang\\.String\\).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.io\\.File\\.<init>\\(java\\.lang\\.String, java\\.lang\\.String\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.io\\.File\\.<init>\\(java\\.io\\.File, java\\.lang\\.String\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.io\\.File\\.<init>\\(java\\.net\\.URI\\).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
```

```
      "methods" : {
        "matching" : "java\\.io\\.FileInputStream\\.<init>\\(java\\.lang\\.String\\).*"
      },
      "param_index" : 1
    }
  },
  // This should match both FileOutputStream(String name) and FileOutputStream(String
  // Updated for BZ 65242
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "java\\.io\\.FileOutputStream\\.<init>\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "java\\.io\\.FileReader\\.<init>\\(java\\.lang\\.String\\).*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "java\\.io\\.FileWriter\\.<init>\\(java\\.lang\\.String\\).*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "java\\.io\\.PrintStream\\.<init>\\(java\\.lang\\.String\\).*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "java\\.io\\.PrintStream\\.<init>\\(java\\.lang\\.String, java\\.lang\\.String\
).*"
      },
      "param_index" : 1
    }
  },
```

```
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.io\\.PrintWriter\\.<init>\\(java\\.lang\\.String\\).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.io\\.PrintWriter\\.<init>\\(java\\.lang\\.String, java\\.lang\\.String\\
.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.io\\.RandomAccessFile\\.<init>\\(java\\.lang\\.String, java\\.lang\\
String\\).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "all_params_of" : {
          "named" : "java.nio.file.FileSystem.getPath(java.lang.String, java.lang.String[])java.nio.fi
e.Path"
        }
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "all_params_of" : {
          "named" : "java.nio.file.Paths.get(java.lang.String, java.lang.String[])java.nio.file.Path"
        }
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "all_params_of" : {
          "named" : "java.nio.file.Path.resolve(java.lang.String)java.nio.file.Path"
        }
      }
    },
```

```
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "all_params_of" : {
          "named" : "java.nio.file.Path.resolveSibling(java.lang.String)java.nio.file.Path"
        }
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.nio\\.file\\.Files\\.createTempDirectory\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.nio\\.file\\.Files\\.createTempFile\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.nio\\.file\\.Files\\.createTempFile\\(java\\.nio\\.file\\.Path, java\\.la
g\\.String.*"
        },
        "param_index" : 2
      }
    },
    // This should match both JarFile(String name) and JarFile(String name, boolean verify)
    // This sink was added for BZ 65242
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.util\\.jar\\.JarFile\\.<init>\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    },
    // This should match both ZipFile(String name) and ZipFile(String name, java.nio.charset.C
arset cs)
    // This sink was added for BZ 65242
    {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
        "methods" : {
```

```
        "matching" : "java\\.util\\.zip\\.ZipFile\\.<init>\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  // ------------------------------------------------
  // Android sink directives
  // ------------------------------------------------
  // These methods accept only URI's with one of the following
  // scheme: file, content and android.resource.
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "android\\.content\\.ContentResolver\\.(openAssetFileDescriptor|openFil
Descriptor|openInputStream|openOutputStream|openTypedAssetFileDescriptor)\\(android\\.
et\\.Uri.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "android\\.content\\.ContentProviderClient\\.(openAssetFile|openFile|op
nTypedAssetFileDescriptor)\\(android\\.net\\.Uri.*"
      },
      "param_index" : 1
    }
  },
  // ContentProvide methods are not usually called directly in
  // user code, but accessed through a ContentResolver. However,
  // there are public facing classes that inheret from
  // ContentProvider.
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "android\\.content\\.ContentProvider\\.(openAssetFile|openFile|openTyp
dAssetFile)\\(android\\.net\\.Uri.*"
      },
      "param_index" : 1
    }
  },
  // Despite the fact that DocumentsProvider and FileProvider
  // inherit from ContentProvider, the methods are only considered
  // sinks if they are not overriden in the subclass
  // implementation so we check for them here.
  {
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "android\\.provider\\.DocumentsProvider\\.(openAssetFile|openFile|open
```

```
ypedAssetFile)\\(android\\.net\\.Uri.*"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "PATH_MANIPULATION",
        "sink" : {
          "methods" : {
            "matching" : "android\\.support\\.v4\\.content\\.FileProvider\\.(openAssetFile|openFile
openTypedAssetFile)\\(android\\.net\\.Uri.*"
          },
          "param_index" : 1
        }
      },
      // ------------------------------------------------
      // Java sanitizer directives
      // ------------------------------------------------
      {
        "sanitizer_for_checker" : "PATH_MANIPULATION",
        "sanitizer" : {
          "methods" : {
            "matching" : "org\\.apache\\.commons\\.io\\.FilenameUtils\\.getName\\(java\\.lang\\.S
ring\\).*"
          },
          "param_index" : 1
        }
      },
      {
--
          "sink_for_checker" : "PATH_MANIPULATION",
          "sink" : {
            "input" : "arg2",
            "path" : [ { "property" : "cwd" } ],
            "to_callsite" : {
              "call_on" : {
                "path" : [ { "property" : "exec" } ],
                "read_from_js_require" : "child_process"
              }
            }
          }
        },
        // sink: require('child_process').execSync(command, arg2.cwd)
        {
          "sink_for_checker" : "PATH_MANIPULATION",
          "sink" : {
            "input" : "arg2",
            "path" : [ { "property" : "cwd" } ],
            "to_callsite" : {
              "call_on" : {
                "path" : [ { "property" : "execSync" } ],
                "read_from_js_require" : "child_process"
              }
            }
```

```
    }
},
// sink: require('child_process').execFile(arg1)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFile" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').execFile(command, arg2.cwd, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "path" : [ { "property" : "cwd" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFile" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').execFile(command, args, arg3.cwd, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "cwd" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFile" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').execFileSync(arg1, args, options)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFileSync" } ],
                "read_from_js_require" : "child_process"
            }
        }
```

```
    }
},
// sink: require('child_process').execFileSync(file, arg2.cwd)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "path" : [ { "property" : "cwd" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFileSync" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').execFileSync(file, args, arg3.cwd)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "cwd" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "execFileSync" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').fork(arg1[, args][, options])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "fork" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('child_process').fork(modulePath, arg2.cwd)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "path" : [ { "property" : "cwd" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "fork" } ],
                "read_from_js_require" : "child_process"
            }
```

```
        }
    }
},
// sink: require('child_process').fork(modulePath, args, arg3.cwd)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "cwd" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "fork" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
/*
// XXX: BUG 87933
// sink: require('child_process').spawn(command, arg2[*])
{
"sink_for_checker" : "PATH_MANIPULATION",
"sink" : {
"input" : "this",
"path": [ { "any_property" : true } ],
"to_callsite" : {
"call_on" : {
"path" : [ { "property" : "spawn" } ],
"read_from_js_require" : "child_process"
}
}
}
},
 */
// sink: require('child_process').spawn(command, arg2.cwd)
{
    "sink_for_checker" : "PATH_MANIPULATION",
        "sink" : {
            "input" : "arg2",
            "path" : [ { "property" : "cwd" } ],
            "to_callsite" : {
                "call_on" : {
                    "path" : [ { "property" : "spawn" } ],
                    "read_from_js_require" : "child_process"
                }
            }
        }
},
// sink: require('child_process').spawn(command, args, arg3.cwd)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "cwd" } ],
```

```
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "spawn" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
/*
// XXX: BUG 87933
// sink: require('child_process').spawnSync(command, arg2[*])
{
"sink_for_checker" : "PATH_MANIPULATION",
"sink" : {
"input" : "this",
"path": [ { "any_property" : true} ],
"to_callsite" : {
"call_on" : {
"path" : [ { "property" : "spawnSync" } ],
"read_from_js_require" : "child_process"
}
}
}
},
 */
// sink: require('child_process').spawnSync(command, arg2.cwd)
{
    "sink_for_checker" : "PATH_MANIPULATION",
        "sink" : {
            "input" : "arg2",
            "path" : [ { "property" : "cwd" } ],
            "to_callsite" : {
                "call_on" : {
                    "path" : [ { "property" : "spawnSync" } ],
                    "read_from_js_require" : "child_process"
                }
            }
        }
},
// sink: require('child_process').spawnSync(command, args, arg3.cwd)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "property" : "cwd" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "spawnSync" } ],
                "read_from_js_require" : "child_process"
            }
        }
    }
},
// sink: require('cluster').setupMaster(arg1.exec)
```

```
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "path" : [ { "property" : "exec" } ],
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "setupMaster" } ],
                "read_from_js_require" : "cluster"
            }
        }
    }
},
// sink: require('fs').access(arg1[, mode], callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "access" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').accessSync(arg1[, mode])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "accessSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').appendFile(arg1, data[, options], callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "appendFile" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').appendFileSync(arg1, data[, options])
{
```

```
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "appendFileSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').chmod(arg1, mode, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "chmod" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').chmodSync(arg1, mode)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "chmodSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').chown(arg1, uid, gid, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "chown" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').chownSync(arg1, uid, gid)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
```

```
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "path" : [ { "property" : "chownSync" } ],
            "read_from_js_require" : "fs"
          }
        }
      }
},
// sink: require('fs').createReadStream(arg1[, options])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "path" : [ { "property" : "createReadStream" } ],
            "read_from_js_require" : "fs"
          }
        }
    }
},
// sink: require('fs').createWriteStream(arg1[, options])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "path" : [ { "property" : "createWriteStream" } ],
            "read_from_js_require" : "fs"
          }
        }
    }
},
// Only available on Mac OS X.
// sink: require('fs').lchmod(arg1, mode, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "path" : [ { "property" : "lchmod" } ],
            "read_from_js_require" : "fs"
          }
        }
    }
},
// Only available on Mac OS X.
// sink: require('fs').lchmodSync(arg1, mode)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
```

```
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "path" : [ { "property" : "lchmodSync" } ],
          "read_from_js_require" : "fs"
        }
      }
    }
},
// sink: require('fs').lchown(arg1, uid, gid, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "path" : [ { "property" : "lchown" } ],
          "read_from_js_require" : "fs"
        }
      }
    }
},
// sink: require('fs').lchownSync(arg1, uid, gid)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "path" : [ { "property" : "lchownSync" } ],
          "read_from_js_require" : "fs"
        }
      }
    }
},
// sink: require('fs').link(arg1, dstpath, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "path" : [ { "property" : "link" } ],
          "read_from_js_require" : "fs"
        }
      }
    }
},
// sink: require('fs').link(srcpath, arg2, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "input" : "arg2",
      "to_callsite" : {
```

```
            "call_on" : {
               "path" : [ { "property" : "link" } ],
               "read_from_js_require" : "fs"
            }
         }
      }
   },
   // sink: require('fs').linkSync(arg1, dstpath)
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on" : {
               "path" : [ { "property" : "linkSync" } ],
               "read_from_js_require" : "fs"
            }
         }
      }
   },
   // sink: require('fs').linkSync(srcpath, arg2)
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg2",
         "to_callsite" : {
            "call_on" : {
               "path" : [ { "property" : "linkSync" } ],
               "read_from_js_require" : "fs"
            }
         }
      }
   },
   // sink: require('fs').lstat(arg1, callback)
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on" : {
               "path" : [ { "property" : "lstat" } ],
               "read_from_js_require" : "fs"
            }
         }
      }
   },
   // sink: require('fs').lstatSync(arg1)
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on" : {
               "path" : [ { "property" : "lstatSync" } ],
```

```
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').mkdir(path[, mode], callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "mkdir" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').mkdirSync(path[, mode])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "mkdirSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').open(arg1, flags[, mode], callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "open" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').openSync(arg1, flags[, mode])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "openSync" } ],
                "read_from_js_require" : "fs"
            }
```

```
        }
    }
},
// sink: require('fs').readdir(arg1, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "readdir" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').readdirSync(arg1)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "readdirSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').readFile(arg1[, options], callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "readFile" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').readFileSync(arg1[, options])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "readFileSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
```

```
},
// sink: require('fs').readlink(arg1, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "readlink" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').readlinkSync(arg1)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "readlinkSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').realpath(arg1)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "realpath" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').realpathSync(arg1[, cache])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "realpathSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').rename(arg1, newPath, callback)
```

```
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "rename" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').rename(oldPath, arg2, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "rename" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').renameSync(arg1, newPath)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "renameSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').renameSync(oldPath, arg2)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "renameSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').rmdir(arg1, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
```

```
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "path" : [ { "property" : "rmdir" } ],
          "read_from_js_require" : "fs"
        }
      }
    }
},
// sink: require('fs').rmdirSync(arg1)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "path" : [ { "property" : "rmdirSync" } ],
          "read_from_js_require" : "fs"
        }
      }
    }
},
// sink: require('fs').stat(arg1, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "path" : [ { "property" : "stat" } ],
          "read_from_js_require" : "fs"
        }
      }
    }
},
// sink: require('fs').statSync(arg1)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "path" : [ { "property" : "statSync" } ],
          "read_from_js_require" : "fs"
        }
      }
    }
},
// sink: require('fs').symlink(arg1, path[, type], callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
      "input" : "arg1",
```

```
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "symlink" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').symlink(target, arg2[, type], callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "symlink" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').symlinkSync(arg1, path[, type])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "symlinkSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').symlinkSync(target, arg2[, type])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "symlinkSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').truncate(arg1, len, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
```

```
            "path" : [ { "property" : "truncate" } ],
            "read_from_js_require" : "fs"
         }
      }
   }
},
// sink: require('fs').truncateSync(arg1, len)
{
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "path" : [ { "property" : "truncateSync" } ],
            "read_from_js_require" : "fs"
         }
      }
   }
},
// sink: require('fs').unlink(arg1, callback)
{
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "path" : [ { "property" : "unlink" } ],
            "read_from_js_require" : "fs"
         }
      }
   }
},
// sink: require('fs').unlinkSync(arg1)
{
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "path" : [ { "property" : "unlinkSync" } ],
            "read_from_js_require" : "fs"
         }
      }
   }
},
// sink: require('fs').unwatchFile(arg1[, listener])
{
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "path" : [ { "property" : "unwatchFile" } ],
            "read_from_js_require" : "fs"
```

```
            }
        }
    }
},
// sink: require('fs').utimes(arg1, atime, mtime, callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "utimes" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').utimesSync(arg1, atime, mtime)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "utimesSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').watch(arg1[, options][, listener])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "watch" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').watchFile(arg1[, options][, listener])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "watchFile" } ],
                "read_from_js_require" : "fs"
            }
        }
```

```
    }
},
// sink: require('fs').writeFile(arg1, data[, options], callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "writeFile" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// sink: require('fs').writeFileSync(arg1, data[, options])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "writeFileSync" } ],
                "read_from_js_require" : "fs"
            }
        }
    }
},
// The "process" module may be loaded automatically, or via require().
// sink: process.chdir(arg1)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "Process",
                "read": [ { "property" : "chdir" } ]
            }
        }
    }
},
// 2) Express API sinks.
// sink: require('express').static(arg1, [options])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "express",
                "path" : [ { "property" : "static" } ]
            }
        }
```

```
    }
},
// sink: [type ExpressResponse].download(arg1 [, filename] [, fn])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "download" } ]
            },
        },
    }
},
// sink: [type ExpressResponse].render(arg1)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "render" } ]
            },
        },
    }
},
// sink: [type ExpressResponse].sendFile(arg1 [, options] [, fn])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "sendFile" } ]
            },
        },
    }
},
// sink: [type ExpressResponse].sendFile(path [, arg2.root] [, fn])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "sendFile" } ]
            },
        },
        "path" : [ { "property" : "root" } ]
    }
```

原文链接：coverity 部分规则提取

```
},
// 3) MongoDB API sinks.
//  sink: new [require('mongodb')].GridStore(db, arg2, mode, options)
{
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
      "input" : "arg2",
      "to_callsite" : {
         "new_on" : {
            "read_from_js_require" : "mongodb",
            "path" : [ { "property" : "GridStore" } ]
         },
      },
   }
},
//  sink: new [require('mongodb')].GridStore(db, [id], arg3, mode, options)
{
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
      "input" : "arg3",
      "to_callsite" : {
         "new_on" : {
            "read_from_js_require" : "mongodb",
            "path" : [ { "property" : "GridStore" } ]
         },
      },
   }
},
// sink: [type MongoDbGridStore].writeFile(arg1, callback)
{
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "MongoDbGridStore",
            "read" : [ { "property" : "writeFile" } ]
         },
      },
   }
},
// sink: [type MongoDbGridStoreClass].read(db, arg2[, length][, offset][, options], callback

{
   "sink_for_checker" : "PATH_MANIPULATION",
   "sink" : {
      "input" : "arg2",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "MongoDbGridStoreClass",
            "read" : [ { "property" : "read" } ]
         },
      },
   }
```

```
},
// sink: [type MongoDbGridStoreClass].readlines(db, arg2[, separator][, options], callback)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "MongoDbGridStoreClass",
                "read" : [ { "property" : "readlines" } ]
            },
        },
    }
},
// sink: [type MongoDbGridStoreClass].unlink(db, arg2[, options])
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "MongoDbGridStoreClass",
                "read" : [ { "property" : "unlink" } ]
            },
        },
    }
},
// 4) HANA XSC sinks.
// sink: [type SAPHanaStatic].import(arg1)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "import" } ],
        "read_from_object_of_type" : "SAPHanaStatic"
      },
      "when" : {
          "only_if_arg_index" : 1,
          "is_max_index" : true
      }
    }
  }
},
// sink: new [type SAPHanaSecurity].Store(arg1)
{
  "sink_for_checker" : "PATH_MANIPULATION",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "new_on" : {
        "read" : [ { "property" : "Store" } ],
        "read_from_object_of_type" : "SAPHanaSecurity"
```

```
      }
    }
  }
},
// 5) HANA XSA sinks.
// sink: new (require('sap-textbundle')).TextBundle(arg1)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "new_on" : {
                "read_from_js_require" : "sap-textbundle",
                "path" : [ { "property" : "TextBundle" } ]
            }
        }
    }
},
// sink: new (require('sap-textbundle')).TextBundle({path: arg1}, __)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "path": [ { "property": "path" }],
        "to_callsite" : {
            "new_on" : {
                "read_from_js_require" : "sap-textbundle",
                "path" : [ { "property" : "TextBundle" } ]
            }
        }
    }
},
// sink: new (require('sap-textbundle')).ResourceManager(arg1)
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "new_on" : {
                "read_from_js_require" : "sap-textbundle",
                "path" : [ { "property" : "ResourceManager" } ]
            }
        }
    }
},
// sink: require('sap-logging').createAppContext({logLocation: arg1})
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "path": [ { "property": "logLocation" }],
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "sap-logging",
```

```
                    "path" : [ { "property" : "createAppContext" } ]
                }
            }
        }
      },
      // sink: require('sap-logging').createAppContext({traceLocation: arg1})
      {
        "sink_for_checker" : "PATH_MANIPULATION",
        "sink" : {
          "input" : "arg1",
          "path": [ { "property": "traceLocation" }],
          "to_callsite" : {
            "call_on" : {
              "read_from_js_require" : "sap-logging",
              "path" : [ { "property" : "createAppContext" } ]
            }
          }
        }
      },
  ]
////////////////////////////////////////////////////////////////////////////
// End of directive list
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 7,
  "language" : "any",
  "directives" : [
   //
--
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.lang\\.String\\.(replace(All|First)|split|matches)\\(.*"
        },
        "param_index" : 1
      }
    },
    // ### Java regex
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.util\\.regex\\.Pattern\\.compile\\(.*"
        },
        "param_index" : 1
      }
    },
    // ### Apache Regex (Jakarta)
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.regexp\\.RE\\.<init>\\(.*"
        },
```

```
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "REGEX_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.apache\\.regexp\\.RECompiler\\.compile\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "REGEX_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.apache\\.regexp\\.REUtil\\.createRE\\(.*"
      },
      "param_index" : 1
    }
  },
  // ### Apache Oro Regex (Jakarta Oro)
  {
    "sink_for_checker" : "REGEX_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.apache\\.oro\\.text\\.perl\\.Perl5Util\\.match\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "REGEX_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.apache\\.oro\\.text\\.perl\\.Perl5Util\\.split\\(java\\.util\\.Collection,
java\\.lang\\.String, java\\.lang\\.String.*"
      },
      "param_index" : 2
    }
  },
  {
    "sink_for_checker" : "REGEX_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.apache\\.oro\\.text\\.regex\\.PatternMatcher\\.(contains|matches|
atchesPrefix)\\(.*"
      },
      "param_index" : 2
    }
  },
  {
    "sink_for_checker" : "REGEX_INJECTION",
    "sink" : {
```

```
        "methods" : {
          "matching" : "org\\.apache\\.oro\\.text\\.regex\\.Substitution\\.appendSubstitution\\(.
"
        },
        "param_index" : 6
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.oro\\.text\\.regex\\.Util\\.split\\(org\\.apache\\.oro\\.text\
.regex\\.PatternMatcher, org\\.apache\\.oro\\.text\\.regex\\.Pattern.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.oro\\.text\\.regex\\.Util\\.split\\(java\\.util\\.Collection, or
\\.apache\\.oro\\.text\\.regex\\.PatternMatcher, org\\.apache\\.oro\\.text\\.regex\\.Pattern.*"
        },
        "param_index" : 3
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.oro\\.text\\.regex\\.Util\\.substitute\\(org\\.apache\\.oro\\
text\\.regex\\.PatternMatcher, org\\.apache\\.oro\\.text\\.regex\\.Pattern.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.oro\\.text\\.regex\\.Util\\.substitute\\(java\\.lang\\.String
uffer, org\\.apache\\.oro\\.text\\.regex\\.PatternMatcher, org\\.apache\\.oro\\.text\\.regex\\.P
ttern.*"
        },
        "param_index" : 3
      }
    },
    // ### jregex
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "jregex\\.Pattern\\.(<init>|compile)\\(.*"
```

```
        },
        "param_index" : 1
      }
    },
    /// Sanitizeres
    {
      "sanitizer_for_checker" : "REGEX_INJECTION",
      "sanitizer" : {
        "return_value_of" : {
          "matching" : "java\\.util\\.Regex\\.Pattern\\.quote\\(java\\.lang\\.String.*"
        }
      }
    }
  ]
 },
  "type" : "Coverity analysis configuration",
  "format_version" : 5,
--
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::.ctor\\(System\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.RegexCompilationInfo::.ctor\\(Syst
m\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Web.UI.Design.WebControls.RegexEditorDialog::set_RegularExpress
on(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::IsMatch\\(System\\.String,S
stem\\.String\\).*"
        },
        "param_index" : 2
```

原文链接：coverity 部分规则提取

```
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::IsMatch\\(System\\.String,S
stem\\.String,System\\.Text\\.RegularExpressions\\.RegexOptions\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::IsMatch\\(System\\.String,S
stem\\.String,System\\.Text\\.RegularExpressions\\.RegexOptions,System\\.TimeSpan\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Match\\(System\\.String,Sys
em\\.String\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Match\\(System\\.String,Sys
em\\.String,System\\.Text\\.RegularExpressions\\.RegexOptions\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Match\\(System\\.String,Sys
em\\.String,System\\.Text\\.RegularExpressions\\.RegexOptions,System\\.TimeSpan\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
```

```
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Matches\\(System\\.String,
ystem\\.String\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Matches\\(System\\.String,
ystem\\.String,System\\.Text\\.RegularExpressions\\.RegexOptions\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Matches\\(System\\.String,
ystem\\.String,System\\.Text\\.RegularExpressions\\.RegexOptions,System\\.TimeSpan\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Replace\\(System\\.String,S
stem\\.String,System\\.Text\\.RegularExpressions\\.MatchEvaluator\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Replace\\(System\\.String,S
stem\\.String,System\\.Text\\.RegularExpressions\\.MatchEvaluator,System\\.Text\\.RegularEx
ressions\\.RegexOptions\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Replace\\(System\\.String,S
```

```
stem\\.String,System\\.Text\\.RegularExpressions\\.MatchEvaluator,System\\.Text\\.RegularExpressions\\.RegexOptions,System\\.TimeSpan\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Replace\\(System\\.String,System\\.String,System\\.String\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Replace\\(System\\.String,System\\.String,System\\.String,System\\.Text\\.RegularExpressions\\.RegexOptions\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Replace\\(System\\.String,System\\.String,System\\.String,System\\.Text\\.RegularExpressions\\.RegexOptions,System\\.TieSpan\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Split\\(System\\.String,System\\.String\\).*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Split\\(System\\.String,System\\.String,System\\.Text\\.RegularExpressions\\.RegexOptions\\).*"
        },
```

```
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Text\\.RegularExpressions\\.Regex::Split\\(System\\.String,Syst
m\\.String,System\\.Text\\.RegularExpressions\\.RegexOptions,System\\.TimeSpan\\).*"
        },
        "param_index" : 2
      }
    },
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
// 0) Custom Dataflow Checker specification for REGEX_INJECTION_BUDA.
// 1) Built-in REGEX_INJECTION_BUDA sinks.
// 2) jQuery sinks.
// 3) jQuery-UI sinks.
// 4) jQuery-ajax sinks.
  "type" : "Coverity analysis configuration",
  "format_version" : 8,
  "language" : "javascript",
  "directives" : [
    // 0) Custom Dataflow Checker specification for REGEX_INJECTION_BUDA.
    {
--
      "sink_for_checker" : "REGEX_INJECTION",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_path_off_global" : [ { "property" : "RegExp" } ]
          }
        }
      }
    }
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 7,
    "language" : "any",
    "directives" : [
      //
      // # Script injection checker
      // We'll use the same set of libraries as we have
      // for the XPath injection checker
      //
      // CWE ID: 95
--
        "sink_for_checker" : "SCRIPT_CODE_INJECTION",
        "sink" : {
          "methods" : {
            "matching" : "javax\\.script\\.ScriptEngine\\.eval\\(.*"
          },
```

```
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "javax\\.script\\.Compilable\\.compile\\(.*"
      },
      "param_index" : 1
    }
  },
  // ### Rhino
  {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.mozilla\\.javascript\\.Context\\.compile(String|Reader)\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "methods" : {
        "or" : [
          {
            "and" : [
              { "matching" : "org\\.mozilla\\.javascript\\.Context\\.compileReader\\(.*" },
              { "num_of_args" : 6 }
            ]
          },
          { "matching" : "org\\.mozilla\\.javascript\\.Context\\.evaluate(String|Reader)\\(.*" },
          { "matching" : "org\\.mozilla\\.javascript\\.Context\\.compileFunction\\(.*" }
        ]
      },
      "param_index" : 2
    }
  },
  // ### gwt
  // From the API documentation:
  //     CAUTION! This method calls the JavaScript eval() function, which can execute
  //     arbitrary script. DO NOT pass an untrusted string into this method.
--
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "methods" : {
        "or" : [
          { "matching" : "com\\.google\\.gwt\\.json\\.client\\.JSONParser\\.parseLenient\\(.*" },
          { "matching" : "com\\.google\\.gwt\\.json\\.client\\.JSONParser\\.parse\\(.*" }
        ]
      },
      "param_index" : 1
```

原文链接：coverity 部分规则提取

```
    }
  },
  // ### Jython
  {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.python\\.core\\.Py\\.(compile_flags|compile_command_flags|compi
e)\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.python\\.util\\.InteractiveConsole\\.push\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.python\\.util\\.PythonInterpreter\\.(compile|eval|exec)\\(.*"
      },
      "param_index" : 1
    }
  },
  // ### JRuby
  {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.jruby\\.Ruby\\.(evalScriptlet|executeScript)\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.jruby\\.embed\\.ScriptingContainer\\.(runScriptlet|parse)\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "methods" : {
```

```
          "matching" : "org\\.jruby\\.embed\\.jsr223\\.(JRubyCompiledScript|JRubyEngine)\\.eval
\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.jruby\\.embed\\.jsr223\\.JRubyEngine\\.compile\\(.*"
        },
        "param_index" : 1
      }
    },
    // ### Spring wrappers for Scripting environments
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.scripting\\.bsh\\.BshScriptUtils\\.createBshObje
t\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.scripting\\.bsh\\.BshScriptUtils\\.createBshObje
t\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.scripting\\.jruby\\.JRubyScriptUtils\\.createJRub
Object\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.scripting\\.support\\.StaticScriptSource\\.(<init
|setScript)\\(.*"
        },
        "param_index" : 1
```

```
      }
    }
  ]
 },
 // ### C# directives
  "type" : "Coverity analysis configuration",
  "format_version" : 4,
  "language" : "C#",
  "directives" : [
   // ## DLR Sinks
   {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
     "methods" : {
      "named" : "Microsoft.Scripting.Hosting.ScriptEngine::CreateScriptSourceFromString(Sy
stem.String)Microsoft.Scripting.Hosting.ScriptSource"
     },
     "param_index" : 1
    }
   },
   {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
     "methods" : {
      "named" : "Microsoft.Scripting.Hosting.ScriptEngine::CreateScriptSourceFromString(Sy
stem.String, System.String)Microsoft.Scripting.Hosting.ScriptSource"
     },
     "param_index" : 1
    }
   },
   {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
     "methods" : {
      "named" : "Microsoft.Scripting.Hosting.ScriptEngine::CreateScriptSourceFromString(Sy
stem.String, Microsoft.Scripting.SourceCodeKind)Microsoft.Scripting.Hosting.ScriptSource"
     },
     "param_index" : 1
    }
   },
   {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
     "methods" : {
      "named" : "Microsoft.Scripting.Hosting.ScriptEngine::CreateScriptSourceFromString(Sy
stem.String, System.String, Microsoft.Scripting.SourceCodeKind)Microsoft.Scripting.Hosting.Scr
ptSource"
     },
     "param_index" : 1
    }
   },
   // This uses a pattern because of the presence of a generic return type.
   // It is intended to match several sinks.
   {
```

```
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "Microsoft\\.Scripting\\.Hosting\\.ScriptEngine::Execute(`1|AndWrap)?\\(System.String.*"
        },
        "param_index" : 1
      }
    },
    // # Microsoft ClearScript
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Microsoft.ClearScript.ScriptEngine::Execute(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Microsoft.ClearScript.ScriptEngine::Execute(System.String, System.String)System.Void"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Microsoft.ClearScript.ScriptEngine::Execute(System.String, System.Boolean, System.String)System.Void"
        },
        "param_index" : 3
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Microsoft.ClearScript.ScriptEngine::ExecuteCommand(System.String)System String"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Microsoft.ClearScript.ScriptEngine::Evaluate(System.String)System.Object"
```

```
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Microsoft.ClearScript.ScriptEngine::Evaluate(System.String, System.String)S
stem.Object"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Microsoft.ClearScript.ScriptEngine::Evaluate(System.String, System.Boolean,
System.String)System.Object"
        },
        "param_index" : 3
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Microsoft.ClearScript.ScriptEngine::Invoke(System.String, System.Object[])S
stem.Object"
        },
        "param_index" : 1
      }
    },
    // # Javascript .NET
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Noesis.Javascript.JavascriptContext::Run(System.String)System.Object"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Noesis.Javascript.JavascriptContext::Run(System.String, System.String)Syst
m.Object"
        },
        "param_index" : 1
      }
    },
```

```
    // # V8 (Javascript)
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "V8.Net.V8Engine::Execute(System.String, System.String, System.Boolean)V8
Net.Handle"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "V8.Net.V8Engine::ConsoleExecute(System.String, System.String, System.Boo
ean)V8.Net.Handle"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "V8.Net.V8Engine::VerboseConsoleExecute(System.String, System.String, Sys
em.Boolean)V8.Net.Handle"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "V8.Net.V8Engine::Compile(System.String, System.String, System.Boolean)V8
Net.Handle"
        },
        "param_index" : 1
      }
    },
    // # Jint (Javascript)
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Jint.Engine::Execute(System.String)Jint.Engine"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
```

```
        "methods" : {
          "named" : "Jint.Engine::Execute(System.String, Jint.Parser.ParserOptions)Jint.Engine"
        },
        "param_index" : 1
      }
    },
    // # Chakra
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.JavaScriptContext::ParseScript(System.String, ChakraHost.Hosting.JavaScriptSourceContext, System.String)ChakraHost.Hosting.JavaScriptValue"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.JavaScriptContext::ParseScript(System.String, System.Byte[], ChakraHost.Hosting.JavaScriptSourceContext, System.String)ChakraHost.Hosting.JavaScriptValue"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.JavaScriptContext::ParseScript(System.String)ChakraHost.Hosting.JavaScriptValue"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.JavaScriptContext::ParseScript(System.String, System.Byte[])ChakraHost.Hosting.JavaScriptValue"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.JavaScriptContext::RunScript(System.String, ChakraHos.Hosting.JavaScriptSourceContext, System.String)ChakraHost.Hosting.JavaScriptValue"
```

```
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.JavaScriptContext::RunScript(System.String, System.By
e[], ChakraHost.Hosting.JavaScriptSourceContext, System.String)ChakraHost.Hosting.JavaScrip
Value"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.JavaScriptContext::RunScript(System.String)ChakraHos
.Hosting.JavaScriptValue"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.JavaScriptContext::RunScript(System.String, System.By
e[])ChakraHost.Hosting.JavaScriptValue"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.Native::JsParseScript(System.String, ChakraHost.Hosti
g.JavaScriptSourceContext, System.String, ChakraHost.Hosting.JavaScriptValue&)ChakraHost.
osting.JavaScriptErrorCode"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.Native::JsRunScript(System.String, ChakraHost.Hosting
JavaScriptSourceContext, System.String, ChakraHost.Hosting.JavaScriptValue&)ChakraHost.H
sting.JavaScriptErrorCode"
        },
```

```
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.Native::JsParseSerializedScript(System.String, System.Byte[], ChakraHost.Hosting.JavaScriptSourceContext, System.String, ChakraHost.Hosting.JavaScriptValue&)ChakraHost.Hosting.JavaScriptErrorCode"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SCRIPT_CODE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "ChakraHost.Hosting.Native::JsRunSerializedScript(System.String, System.Byte[], ChakraHost.Hosting.JavaScriptSourceContext, System.String, ChakraHost.Hosting.JavaScriptValue&)ChakraHost.Hosting.JavaScriptErrorCode"
        },
        "param_index" : 1
      }
    }
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
 * Revision History
 * May 2017 - Bug 103331:
 *      Inital support for SAP's HANA XS classic (XSC).
// 0) Custom Dataflow Checker specification for SCRIPT_CODE_INJECTION_BUDA.
// 1) JavaScript API sinks.
// 2) Node.js API sinks.
// 3) HANA XSC sinks.
// 4) Adding Python basic support.
// -----
// SCRIPT_CODE_INJECTION_BUDA definition
   "type" : "Coverity analysis configuration",
--
          "sink_for_checker" : "SCRIPT_CODE_INJECTION",
          "sink" : {
            "input" : "arg1",
            "to_callsite" : {
              "call_on" : {
                "read_path_off_global" : [ { "property" : "eval" } ]
              }
            }
          }
      },
      // sink: new Function(source)
      {
          "sink_for_checker" : "SCRIPT_CODE_INJECTION",
          "sink" : {
              "input" : "last_arg",
```

```
        "to_callsite" : {
          "new_on" : {
            "read_path_off_global" : [ { "property" : "Function" } ]
          }
        }
      }
},
// sink: new GeneratorFunction(source)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "new_on" : {
            "read_path_off_global" : [ { "property" : "GeneratorFunction" } ]
          }
        }
    }
},
// sink: setTimeout(source)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_path_off_global" : [ { "property" : "setTimeout" } ]
          }
        }
    }
},
// sink: setInterval(source)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_path_off_global" : [ { "property" : "setInterval" } ]
          }
        }
    }
},
// sink: setImmediate(source)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_path_off_global" : [ { "property" : "setImmediate" } ]
          }
        }
    }
```

```
},
// 2) Node.js API sinks.
// sink: new require('vm').Script(arg1...)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "Script" } ],
                "read_from_js_require" : "vm"
            }
        }
    }
},
// sink: new require('vm').createScript(arg1...)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "createScript" } ],
                "read_from_js_require" : "vm"
            }
        }
    }
},
// sink: new require('vm').runInDebugContext(arg1...)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "runInDebugContext" } ],
                "read_from_js_require" : "vm"
            }
        }
    }
},
// sink: new require('vm').runInContext(arg1...)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "runInContext" } ],
                "read_from_js_require" : "vm"
            }
        }
    }
},
```

```
// sink: new require('vm').runInNewContext(arg1...)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "runInNewContext" } ],
                "read_from_js_require" : "vm"
            }
        }
    }
},
// sink: new require('vm').runInThisContext(arg1...)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "path" : [ { "property" : "runInThisContext" } ],
                "read_from_js_require" : "vm"
            }
        }
    }
},
// sink: [type MongoDb] eval(arg1[, parameters][, options], callback)
{
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "MongoDb",
                "read" : [ { "property" : "eval" } ]
            }
        }
    }
},
// 3) HANA XSC sinks.
// sink: [type SAPHanaWebResponse].followUp({ functionName: <x> })
{
  "sink_for_checker" : "SCRIPT_CODE_INJECTION",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "functionName" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "followUp" } ],
        "read_from_object_of_type" : "SAPHanaWebResponse"
      }
    }
  }
},
```

```
    ]
}, // Javascript sinks
// -----
// Python sinks
// As the name suggests, this is a dummy sink so that we do not issue
// a warning that the checker has no sinks for Python. Some tests run
// with --xx-fail-on-builtin-directive-warnings and would fail even if
// the checker is not enabled.
// See the checker bug BZ 87612 for modeling the actual APIs and
--
            "sink_for_checker" : "SCRIPT_CODE_INJECTION",
            "sink" : {
                "input" : "arg1",
                "to_callsite" : {
                    "call_on" : {
                        "read_off_any" : [ { "property" : "dummy_sink" } ]
                    }
                }
            }
        },
    ],
}, // Python sinks
// -----
// PHP sinks
// See the checker bug BZ 87611 for enabling this sink and modeling
// other APIs.
    type : "Coverity analysis configuration",
    format_version : 10,
    language : "PHP",
    directives : [
        // sink: dummy_sink(source)
--
            "sink_for_checker" : "SCRIPT_CODE_INJECTION",
            "sink" : {
                "input" : "arg1",
                "to_callsite" : {
                    "call_on_php_function" : {
                        name : "dummy_sink"
                    }
                }
            }
        },
    ],
}, // PHP sinks
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
 * Revision History
 * May 2017 - Bug 91734: Initial batch of SENSITIVE_DATA_LEAK sources and sinks
// 1) JavaScript Sensitive data sources.
// 2) JavaScript Sensitive data sinks.
// 3) [temporary] PHP milestone 1 sources and sinks
// 4) [temporary] Python milestone 1 sources and sinks
// 5) Swift Sensitive data sinks.
//////////////////////////////////////////////////////////////////////////////
--
```

```
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "call_on" : {
             "read_from_object_of_type" : "Console",
             "read" : [ { "property" : "dir" } ]
          }
       }
    }
},
// sink: arg1.*:[type:Console].dir(obj)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
       "input" : "arg1",
       "path" : [ { "any_property" : true } ],
       "to_callsite" : {
          "call_on" : {
             "read_from_object_of_type" : "Console",
             "read" : [ { "property" : "dir" } ]
          }
       }
    }
},
/* Bug 104122: multiple levels of any_property not allowed
// sink: arg1.*.*:[type:Console].dir(obj)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
       "input" : "arg1",
       "path" : [ { "any_property" : true },
                { "any_property" : true } ],
       "to_callsite" : {
          "call_on" : {
             "read_from_object_of_type" : "Console",
             "read" : [ { "property" : "dir" } ]
          }
       }
    }
},
// sink: arg1.*.*.*:[type:Console].dir(obj)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
       "input" : "arg1",
       "path" : [ { "any_property" : true },
                { "any_property" : true },
                { "any_property" : true } ],
       "to_callsite" : {
```

```
            "call_on" : {
               "read_from_object_of_type" : "Console",
               "read" : [ { "property" : "dir" } ]
            }
         }
      }
},
*/
// sink: from_arg1:[type:Console].error(msg[, ...args])
// Bug 96339: Replace next 2 directives with "deep" sink once supported.
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "from_arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "Console",
            "read" : [ { "property" : "error" } ]
         }
      }
   }
},
// sink: from_arg1.*:[type:Console].error(msg[, ...args])
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "from_arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "Console",
            "read" : [ { "property" : "error" } ]
         }
      }
   }
},
// sink: from_arg1:[type:Console].info(msg[, ...args])
// Bug 96339: Replace next 2 directives with "deep" sink once supported.
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "from_arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "Console",
            "read" : [ { "property" : "info" } ]
         }
      }
   }
},
// sink: from_arg1.*:[type:Console].info(msg[, ...args])
```

```
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "from_arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "Console",
            "read" : [ { "property" : "info" } ]
         }
      }
   }
},
// sink: from_arg1:[type:Console].log(msg[, ...args])
// Bug 96339: Replace next 2 directives with "deep" sink once supported.
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "from_arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "Console",
            "read" : [ { "property" : "log" } ]
         }
      }
   }
},
// sink: from_arg1.*:[type:Console].log(msg[, ...args])
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "from_arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "Console",
            "read" : [ { "property" : "log" } ]
         }
      }
   }
},
// sink: arg1:[type:Console].timeEnd(label)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "Console",
            "read" : [ { "property" : "timeEnd" } ]
```

```
          }
        }
    }
},
// sink: from_arg1:[type:Console].trace(msg[, ...args])
// Bug 96339: Replace next 2 directives with "deep" sink once supported.
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
        "input" : "from_arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "Console",
                "read" : [ { "property" : "trace" } ]
            }
        }
    }
},
// sink: from_arg1.*:[type:Console].trace(msg[, ...args])
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
        "input" : "from_arg1",
        "path" : [ { "any_property" : true } ],
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "Console",
                "read" : [ { "property" : "trace" } ]
            }
        }
    }
},
// sink: from_arg1:[type:Console].warn(msg[, ...args])
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
        "input" : "from_arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "Console",
                "read" : [ { "property" : "warn" } ]
            }
        }
    }
},
// sink: from_arg1.*:[type:Console].warn(msg[, ...args])
// Bug 96339: Replace next 2 directives with "deep" sink once supported.
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
```

```
        "input" : "from_arg1",
        "path" : [ { "any_property" : true } ],
        "to_callsite" : {
          "call_on" : {
            "read_from_object_of_type" : "Console",
            "read" : [ { "property" : "warn" } ]
          }
        }
    }
},
// sink: arg1:require('dns').lookup(hostname)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_from_js_require" : "dns",
            "path" : [ { "property" : "lookup" } ]
          }
        }
    }
},
// sink: arg1:require('dns').lookupService(hostname)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_from_js_require" : "dns",
            "path" : [ { "property" : "lookupService" } ]
          }
        }
    }
},
// sink: arg1:require('dns').resolve(hostname)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_from_js_require" : "dns",
            "path" : [ { "property" : "resolve" } ]
          }
        }
    }
},
// sink: arg1:require('dns').resolve4(hostname)
{
```

```
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "call_on" : {
             "read_from_js_require" : "dns",
             "path" : [ { "property" : "resolve4" } ]
          }
       }
    }
},
// sink: arg1:require('dns').resolve6(hostname)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "call_on" : {
             "read_from_js_require" : "dns",
             "path" : [ { "property" : "resolve6" } ]
          }
       }
    }
},
// sink: arg1:require('dns').resolveCname(hostname)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "call_on" : {
             "read_from_js_require" : "dns",
             "path" : [ { "property" : "resolveCname" } ]
          }
       }
    }
},
// sink: arg1:require('dns').resolveMx(hostname)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "call_on" : {
             "read_from_js_require" : "dns",
             "path" : [ { "property" : "resolveMx" } ]
          }
       }
    }
},
```

```
// sink: arg1:require('dns').resolveNaptr(hostname)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "transit",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_js_require" : "dns",
            "path" : [ { "property" : "resolveNaptr" } ]
         }
      }
   }
},
// sink: arg1:require('dns').resolveNs(hostname)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "transit",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_js_require" : "dns",
            "path" : [ { "property" : "resolveNs" } ]
         }
      }
   }
},
// sink: arg1:require('dns').resolveSoa(hostname)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "transit",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_js_require" : "dns",
            "path" : [ { "property" : "resolveSoa" } ]
         }
      }
   }
},
// sink: arg1:require('dns').resolveSrv(hostname)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "transit",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_js_require" : "dns",
            "path" : [ { "property" : "resolveSrv" } ]
         }
      }
```

```
    }
},
// sink: arg1:require('dns').resolvePtr(hostname)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "dns",
                "path" : [ { "property" : "resolvePtr" } ]
            }
        }
    }
},
// sink: arg1:require('dns').resolveTxt(hostname)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "dns",
                "path" : [ { "property" : "resolveTxt" } ]
            }
        }
    }
},
// sink: arg2:require('fs').appendFile(file, data)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "filesystem",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "fs",
                "path" : [ { "property" : "appendFile" } ]
            }
        }
    }
},
// sink: arg2:require('fs').appendFileSync(file, data)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "filesystem",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "fs",
                "path" : [ { "property" : "appendFileSync" } ]
```

```
            }
        }
    }
},
// sink: arg2:require('fs').write(fd, data)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "filesystem",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "fs",
                "path" : [ { "property" : "write" } ]
            }
        }
    }
},
// sink: arg2:require('fs').writeSync(fd, data)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "filesystem",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "fs",
                "path" : [ { "property" : "writeSync" } ]
            }
        }
    }
},
// sink: arg2:require('fs').writeFile(file, data)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "filesystem",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "fs",
                "path" : [ { "property" : "writeFile" } ]
            }
        }
    }
},
// sink: arg2:require('fs').writeFileSync(file, data)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "filesystem",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
```

```
            "read_from_js_require" : "fs",
            "path" : [ { "property" : "writeFileSync" } ]
         }
      }
   }
},
// sink: arg1:[type:http.ClientRequest].end(data)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "transit",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "http.ClientRequest",
            "read" : [ { "property" : "end" } ]
         }
      }
   }
},
// sink: arg1:[type:http.ClientRequest].write(chunk)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "transit",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "http.ClientRequest",
            "read" : [ { "property" : "write" } ]
         }
      }
   }
},
// sink: arg1:[type:http.ClientRequest]._write(chunk)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "transit",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "http.ClientRequest",
            "read" : [ { "property" : "_write" } ]
         }
      }
   }
},
// sink: arg1[*]:[type:http.ClientRequest]._writev(chunks)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "transit",
   "sink" : {
      "input" : "arg1",
```

```
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ClientRequest",
          "read" : [ { "property" : "_writev" } ]
        }
      }
    }
},
// sink: arg1[*]:[type:http.ServerResponse].addTrailers(headers)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "addTrailers" } ]
        }
      }
    }
},
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "addTrailers" } ]
        }
      }
    }
},
// sink: arg1:[type:http.ServerResponse].end(data)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "end" } ]
        }
      }
    }
},
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
```

```
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "end" } ]
        }
      }
    }
},
// sink: arg1:[type:http.ServerResponse].setHeader(name, value)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "setHeader" } ]
        }
      }
    }
},
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "setHeader" } ]
        }
      }
    }
},
// sink: arg2:[type:http.ServerResponse].setHeader(name, value)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg2",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "setHeader" } ]
        }
      }
    }
},
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
```

```
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg2",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "setHeader" } ]
        }
      }
    }
},
// sink: arg1:[type:http.ServerResponse].write(chunk)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "write" } ]
        }
      }
    }
},
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "write" } ]
        }
      }
    }
},
// sink: arg1:[type:http.ServerResponse]._write(chunk)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "_write" } ]
        }
      }
    }
},
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
```

```
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "_write" } ]
        }
      }
    }
},
// sink: arg1[*]:[type:http.ServerResponse]._writev(chunks)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "_writev" } ]
        }
      }
    }
},
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "_writev" } ]
        }
      }
    }
},
// sink: arg2:[type:http.ServerResponse].writeHead(statusCode, statusMsg, headers)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg2",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "http.ServerResponse",
          "read" : [ { "property" : "writeHead" } ]
        }
      }
    }
},
```

```
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg2",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "http.ServerResponse",
            "read" : [ { "property" : "writeHead" } ]
         }
      }
   }
},
// sink: arg3[*]:[type:http.ServerResponse].writeHead(statusCode, statusMsg, headers)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "transit",
   "sink" : {
      "input" : "arg3",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "http.ServerResponse",
            "read" : [ { "property" : "writeHead" } ]
         }
      }
   }
},
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg3",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "http.ServerResponse",
            "read" : [ { "property" : "writeHead" } ]
         }
      }
   }
},
// sink: arg1[*]:[type:https.ServerResponse].addTrailers(headers)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "https.ServerResponse",
            "read" : [ { "property" : "addTrailers" } ]
         }
```

```
      }
    }
},
// sink: arg1:[type:https.ServerResponse].end(data)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "https.ServerResponse",
          "read" : [ { "property" : "end" } ]
        }
      }
    }
},
// sink: arg1:[type:https.ServerResponse].setHeader(name, value)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "https.ServerResponse",
          "read" : [ { "property" : "setHeader" } ]
        }
      }
    }
},
// sink: arg2:[type:https.ServerResponse].setHeader(name, value)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg2",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "https.ServerResponse",
          "read" : [ { "property" : "setHeader" } ]
        }
      }
    }
},
// sink: arg1:[type:https.ServerResponse].write(chunk)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "https.ServerResponse",
```

```
                "read" : [ { "property" : "write" } ]
            }
        }
    }
},
// sink: arg1:[type:https.ServerResponse]._write(chunk)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "https.ServerResponse",
                "read" : [ { "property" : "_write" } ]
            }
        }
    }
},
// sink: arg1[*]:[type:https.ServerResponse]._writev(chunks)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg1",
        "path" : [ { "any_property" : true } ],
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "https.ServerResponse",
                "read" : [ { "property" : "_writev" } ]
            }
        }
    }
},
// sink: arg2:[type:https.ServerResponse].writeHead(statusCode, statusMsg, headers)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "https.ServerResponse",
                "read" : [ { "property" : "writeHead" } ]
            }
        }
    }
},
// sink: arg3[*]:[type:https.ServerResponse].writeHead(statusCode, statusMsg, headers)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg3",
```

```
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "https.ServerResponse",
          "read" : [ { "property" : "writeHead" } ]
        }
      }
    }
},
// sink: arg1:[type:net.Socket].end(data)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "net.Socket",
          "read" : [ { "property" : "end" } ]
        }
      }
    }
},
// sink: arg1:[type:net.Socket].write(chunk)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "net.Socket",
          "read" : [ { "property" : "write" } ]
        }
      }
    }
},
// sink: arg1:[type:net.Socket]._write(chunk)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "net.Socket",
          "read" : [ { "property" : "_write" } ]
        }
      }
    }
},
// sink: arg1[*]:[type:net.Socket]._writev(chunks)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
```

```
    "sink_kind" : "transit",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "net.Socket",
          "read" : [ { "property" : "_writev" } ]
        }
      }
    }
},
// sink: arg1:require('readline').question(query)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "input" : "from_arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_js_require" : "readline",
          "path" : [ { "property" : "question" } ]
        }
      }
    }
},
// sink: arg1:require('readline').setPrompt(prompt)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_js_require" : "readline",
          "path" : [ { "property" : "setPrompt" } ]
        }
      }
    }
},
// sink: arg1:[type:tty.WriteStream].end(data)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "tty.WriteStream",
          "read" : [ { "property" : "end" } ]
        }
      }
    }
},
```

```
// sink: arg1:[type:tty.WriteStream].write(chunk)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "tty.WriteStream",
            "read" : [ { "property" : "write" } ]
         }
      }
   }
},
// sink: arg1:[type:tty.WriteStream]._write(chunk)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "tty.WriteStream",
            "read" : [ { "property" : "_write" } ]
         }
      }
   }
},
// sink: arg1[*]:[type:tty.WriteStream]._writev(chunks)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "tty.WriteStream",
            "read" : [ { "property" : "_writev" } ]
         }
      }
   }
},
// sink: arg1:[type:UDPSocket].send(msg)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "transit",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "UDPSocket",
            "read" : [ { "property" : "send" } ]
         }
```

```
      }
    }
},
// sink: arg2:require('util').deprecate(fn, msg)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "input" : "arg2",
      "to_callsite" : {
        "call_on" : {
          "read_from_js_require" : "util",
          "path" : [ { "property" : "deprecate" } ]
        }
      }
    }
},
// sink: arg1:require('util').debug(msg)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_js_require" : "util",
          "path" : [ { "property" : "debug" } ]
        }
      }
    }
},
// sink: from_arg1:require('util').error([msgs])
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "input" : "from_arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_js_require" : "util",
          "path" : [ { "property" : "error" } ]
        }
      }
    }
},
// sink: arg1:require('util').log(msg)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_js_require" : "util",
```

```
            "path" : [ { "property" : "log" } ]
         }
      }
   }
},
// sink: from_arg1:require('util').print([...msgs])
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "from_arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_js_require" : "util",
            "path" : [ { "property" : "print" } ]
         }
      }
   }
},
// sink: from_arg1:require('util').puts([...msgs])
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "input" : "from_arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_js_require" : "util",
            "path" : [ { "property" : "puts" } ]
         }
      }
   }
},
//--------------------------
// Express sinks
//--------------------------
//
// ExpressResponse methods derived from http.ServerResponse.
//
// sink: arg1[*]:[type:ExpressResponse].addTrailers(headers)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "ExpressResponse",
            "read" : [ { "property" : "addTrailers" } ]
         }
      }
   }
},
```

```
// sink: arg1:[type:ExpressResponse].end(data)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "ExpressResponse",
            "read" : [ { "property" : "end" } ]
         }
      }
   }
},
// sink: arg1:[type:ExpressResponse].setHeader(name, value)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "ExpressResponse",
            "read" : [ { "property" : "setHeader" } ]
         }
      }
   }
},
// sink: arg2:[type:ExpressResponse].setHeader(name, value)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg2",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "ExpressResponse",
            "read" : [ { "property" : "setHeader" } ]
         }
      }
   }
},
// sink: arg1:[type:ExpressResponse].write(chunk)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "ExpressResponse",
            "read" : [ { "property" : "write" } ]
         }
      }
```

```
    }
},
// sink: arg1:[type:ExpressResponse]._write(chunk)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "_write" } ]
            }
        }
    }
},
// sink: arg1[*]:[type:ExpressResponse]._writev(chunks)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg1",
        "path" : [ { "any_property" : true } ],
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "_writev" } ]
            }
        }
    }
},
// sink: arg2:[type:ExpressResponse].writeHead(statusCode, statusMsg, headers)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "writeHead" } ]
            }
        }
    }
},
// sink: arg3[*]:[type:ExpressResponse].writeHead(statusCode, statusMsg, headers)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg3",
        "path" : [ { "any_property" : true } ],
        "to_callsite" : {
            "call_on" : {
```

```
            "read_from_object_of_type" : "ExpressResponse",
            "read" : [ { "property" : "writeHead" } ]
          }
        }
    }
},
//
// Methods specific to ExpressResponse.
//
// sink: arg1:[type:ExpressResponse].send(msg)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "send" } ]
            }
        }
    }
},
// sink: arg1:[type:ExpressResponse].status(msg)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "status" } ]
            }
        }
    }
},
// sink: arg1:[type:ExpressResponse].sendStatus(msg)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "sendStatus" } ]
            }
        }
    }
},
// sink: arg1:[type:ExpressResponse].append(field, value)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
```

```
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "ExpressResponse",
          "read" : [ { "property" : "append" } ]
        }
      }
    }
},
// sink: arg2:[type:ExpressResponse].append(field, value)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg2",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "ExpressResponse",
          "read" : [ { "property" : "append" } ]
        }
      }
    }
},
// sink: arg1:[type:ExpressResponse].set(field, value)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "ExpressResponse",
          "read" : [ { "property" : "set" } ]
        }
      }
    }
},
// sink: arg2:[type:ExpressResponse].set(field, value)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
      "input" : "arg2",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "ExpressResponse",
          "read" : [ { "property" : "set" } ]
        }
      }
    }
},
// sink: arg1.*:[type:ExpressResponse].set(obj)
```

```
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "ExpressResponse",
            "read" : [ { "property" : "set" } ]
         }
      }
   }
},
// sink: arg1.*:[type:ExpressResponse].links(obj)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg1",
      "path" : [ { "any_property" : true } ],
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "ExpressResponse",
            "read" : [ { "property" : "links" } ]
         }
      }
   }
},
// sink: arg1:[type:ExpressResponse].type(type)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "ExpressResponse",
            "read" : [ { "property" : "type" } ]
         }
      }
   }
},
// sink: arg1:[type:ExpressResponse].vary(field)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "ui",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "ExpressResponse",
            "read" : [ { "property" : "vary" } ]
         }
```

```
        }
    }
},
// sink: arg1:[type:ExpressResponse].cookie(name, value)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "cookie",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "cookie" } ]
            }
        }
    }
},
// sink: arg2:[type:ExpressResponse].cookie(name, value)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "cookie",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "cookie" } ]
            }
        }
    }
},
// sink: arg1.*:[type:ExpressResponse].json(obj)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg1",
        "path" : [ { "any_property" : true } ],
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "ExpressResponse",
                "read" : [ { "property" : "json" } ]
            }
        }
    }
},
// sink: arg1.*:[type:ExpressResponse].jsonp(obj)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink" : {
        "input" : "arg1",
        "path" : [ { "any_property" : true } ],
        "to_callsite" : {
```

```
            "call_on" : {
               "read_from_object_of_type" : "ExpressResponse",
               "read" : [ { "property" : "jsonp" } ]
            }
         }
      }
   },
   // sink: arg1:[type:ExpressResponse].redirect(path)
   {
      "sink_for_checker" : "SENSITIVE_DATA_LEAK",
      "sink_kind" : "ui",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on" : {
               "read_from_object_of_type" : "ExpressResponse",
               "read" : [ { "property" : "redirect" } ]
            },
            "when" : {
               "only_if_arg_index" : 1,
               "is_max_index" : true
            }
         }
      }
   },
   // sink: arg2:[type:ExpressResponse].redirect(status, path)
   {
      "sink_for_checker" : "SENSITIVE_DATA_LEAK",
      "sink_kind" : "ui",
      "sink" : {
         "input" : "arg2",
         "to_callsite" : {
            "call_on" : {
               "read_from_object_of_type" : "ExpressResponse",
               "read" : [ { "property" : "redirect" } ]
            }
         }
      }
   },
   // sink: arg2.*:[type:ExpressResponse].render(view, locals)
   {
      "sink_for_checker" : "SENSITIVE_DATA_LEAK",
      "sink_kind" : "ui",
      "sink" : {
         "input" : "arg2",
         "path" : [ { "any_property" : true } ],
         "to_callsite" : {
            "call_on" : {
               "read_from_object_of_type" : "ExpressResponse",
               "read" : [ { "property" : "render" } ]
            }
         }
      }
   },
```

```
//--------------------------
//  Tedious sinks
//--------------------------
// sink: arg1:[type:TediousConnection].execSql(arg1)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "database",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read" : [ { "property" : "execSql" } ],
            "read_from_object_of_type" : "TediousConnection"
         }
      }
   }
},
// sink: arg1:[type:TediousConnection].execSqlBatch(arg1)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "database",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read" : [ { "property" : "execSqlBatch" } ],
            "read_from_object_of_type" : "TediousConnection"
         }
      }
   }
},
// sink: arg1:[type TediousConnection].callProcedure(arg1)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "database",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read" : [ { "property" : "callProcedure" } ],
            "read_from_object_of_type" : "TediousConnection"
         }
      }
   }
},
// sink: arg1:[type TediousConnection].execBulkLoad(arg1)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "database",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read" : [ { "property" : "execBulkLoad" } ],
```

```
            "read_from_object_of_type" : "TediousConnection"
         }
      }
   }
},
// sink: arg1:[type TediousConnection].prepare(arg1)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "database",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read" : [ { "property" : "prepare" } ],
            "read_from_object_of_type" : "TediousConnection"
         }
      }
   }
},
// sink: arg1:[type TediousConnection].execute(arg1, parameters)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "database",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read" : [ { "property" : "execute" } ],
            "read_from_object_of_type" : "TediousConnection"
         }
      }
   }
},
//---------------------------
//  MSSql sinks
//---------------------------
// sink: arg1:[type MSSqlConnection].query(arg1)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "database",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read" : [ { "property" : "query" } ],
            "read_from_object_of_type" : "MSSqlConnection"
         }
      }
   }
},
// sink: arg1:require('mssql').query(arg1)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "database",
```

```
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "path" : [ { "property" : "query" } ],
          "read_from_js_require" : "mssql"
        }
      }
    }
},
// sink: arg1:[type MSSqlRequest].query(arg1)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "query" } ],
          "read_from_object_of_type" : "MSSqlRequest"
        }
      }
    }
},
// sink: arg1:[type MSSqlRequest].execute(arg1)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "execute" } ],
          "read_from_object_of_type" : "MSSqlRequest"
        }
      }
    }
},
// sink: arg1:[type MSSqlRequest].input(arg1)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "input" } ],
          "read_from_object_of_type" : "MSSqlRequest"
        }
      }
    }
},
// sink: arg1:[type MSSqlRequest].output(arg1)
{
```

```
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "call_on" : {
             "read" : [ { "property" : "output" } ],
             "read_from_object_of_type" : "MSSqlRequest"
          }
       }
    }
},
// sink: arg1.columns:[type MSSqlRequest].bulk(arg1.columns)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
       "input" : "arg1",
       "path" : [ { "property" : "columns" } ],
       "to_callsite" : {
          "call_on" : {
             "read" : [ { "property" : "bulk" } ],
             "read_from_object_of_type" : "MSSqlRequest"
          }
       }
    }
},
// sink: arg1.rows:[type MSSqlRequest].bulk(arg1.rows)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
       "input" : "arg1",
       "path" : [ { "property" : "rows" } ],
       "to_callsite" : {
          "call_on" : {
             "read" : [ { "property" : "bulk" } ],
             "read_from_object_of_type" : "MSSqlRequest"
          }
       }
    }
},
// sink: arg1:new require('mssql').Table(arg1)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "new_on" : {
             "path" : [ { "property" : "Table" } ],
             "read_from_js_require" : "mssql"
          }
       }
```

```
    }
},
// sink: arg1:[type MSSqlPreparedStatement].prepare(arg1)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read" : [ { "property" : "prepare" } ],
                "read_from_object_of_type" : "MSSqlPreparedStatement"
            }
        }
    }
},
//---------------------------------
// SAP HANA XS sinks
//---------------------------------
// arg1:[type:SAPHanaTrace].debug(msg)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "SAPHanaTrace",
                "read" : [ { "property" : "debug" } ]
            }
        }
    }
},
// arg1:[type:SAPHanaTrace].error(msg)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "SAPHanaTrace",
                "read" : [ { "property" : "error" } ]
            }
        }
    }
},
// arg1:[type:SAPHanaTrace].fatal(msg)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
```

```
            "call_on" : {
               "read_from_object_of_type" : "SAPHanaTrace",
               "read" : [ { "property" : "fatal" } ]
            }
         }
      }
   },
   // arg1:[type:SAPHanaTrace].info(msg)
   {
      "sink_for_checker" : "SENSITIVE_DATA_LEAK",
      "sink_kind" : "logging",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on" : {
               "read_from_object_of_type" : "SAPHanaTrace",
               "read" : [ { "property" : "info" } ]
            }
         }
      }
   },
   // arg1:[type:SAPHanaTrace].warning(msg)
   {
      "sink_for_checker" : "SENSITIVE_DATA_LEAK",
      "sink_kind" : "logging",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on" : {
               "read_from_object_of_type" : "SAPHanaTrace",
               "read" : [ { "property" : "warning" } ]
            }
         }
      }
   },
   // arg1:[type SAPHanaNetHttpClient].request(req, dest)
   {
      "sink_for_checker" : "SENSITIVE_DATA_LEAK",
      "sink_kind" : "transit",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on" : {
               "read_from_object_of_type" : "SAPHanaNetHttpClient",
               "read" : [ { "property" : "request" } ]
            }
         }
      }
   },
   // arg1:[type SAPHanaNewSMTPConnection].send(mail)
   {
      "sink_for_checker" : "SENSITIVE_DATA_LEAK",
      "sink_kind" : "transit",
      "sink" : {
```

```
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaNetSMTPConnection",
          "read" : [ { "property" : "send" } ]
        }
      }
    }
},
// Same sinks as SQL Injection
// sink: [type SAPHanaDatabaseConn].prepareStatement(<x>)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "prepareStatement" } ],
          "read_from_object_of_type" : "SAPHanaDatabaseConn"
        }
      }
    }
},
// sink: [type SAPHanaDatabaseConn].prepareCall(<x>)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "prepareCall" } ],
          "read_from_object_of_type" : "SAPHanaDatabaseConn"
        }
      }
    }
},
// sink: [type SAPHanaHANADatabaseConn].executeQuery(<x>)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "executeQuery" } ],
          "read_from_object_of_type" : "SAPHanaHANADatabaseConn"
        }
      }
    }
},
// sink: [type SAPHanaHANADatabaseConn].executeUpdate(<x>)
{
```

```
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "call_on" : {
             "read" : [ { "property" : "executeUpdate" } ],
             "read_from_object_of_type" : "SAPHanaHANADatabaseConn"
          }
       }
    }
},
// Omitting the $.text.mining SQLI sinks, because those seem to be
// all about query parameter, not data storage.
// sink: [type SAPHanaSqlExecutor].callProcedure(<x>)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "call_on" : {
             "read" : [ { "property" : "callProcedure" } ],
             "read_from_object_of_type" : "SAPHanaSqlExecutor"
          }
       }
    }
},
// sink: [type SAPHanaSqlExecutor].execQuery(<x>)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "call_on" : {
             "read" : [ { "property" : "execQuery" } ],
             "read_from_object_of_type" : "SAPHanaSqlExecutor"
          }
       }
    }
},
// sink: [type SAPHanaSqlExecutor].execSingle(<x>)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
       "input" : "arg1",
       "to_callsite" : {
          "call_on" : {
             "read" : [ { "property" : "execSingle" } ],
             "read_from_object_of_type" : "SAPHanaSqlExecutor"
          }
       }
```

```
    }
},
// sink: [type SAPHanaSqlExecutor].execSingleIgnoreFailing(<x>)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "database",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on" : {
                "read" : [ { "property" : "execSingleIgnoreFailing" } ],
                "read_from_object_of_type" : "SAPHanaSqlExecutor"
            }
        }
    }
},
// sink: [type SAPXSA_XB_MESSAGING_CLIENT].publish(topic, qos, payload, done)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "SAPXSA_XB_MESSAGING_CLIENT",
                "read" : [ { "property" : "publish" } ]
            }
        },
        "input" : "arg1"
    }
},
// sink: [type SAPXSA_XB_MESSAGING_CLIENT].publish(topic, qos, payload, done)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "SAPXSA_XB_MESSAGING_CLIENT",
                "read" : [ { "property" : "publish" } ]
            }
        },
        "input" : "arg3"
    }
},
// sink: [type SAPXSA_XB_MESSAGING_CLIENT].forward(topic, qos, message, done)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "SAPXSA_XB_MESSAGING_CLIENT",
                "read" : [ { "property" : "forward" } ]
            }
```

```
      },
      "input" : "arg1"
    }
},
// sink: [type SAPXSA_XB_MESSAGING_CLIENT].forward(topic, qos, message, done)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "transit",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_XB_MESSAGING_CLIENT",
          "read" : [ { "property" : "forward" } ]
        }
      },
      "input" : "arg3"
    }
},
//
// require('sap-audit-logging') --> [type SAPXSA_AUDIT_LOGGING]
//
// sink: [type SAPXSA_AUDIT_LOGGING].create(objectID, objectName): arg1
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING",
          "read" : [ { "property" : "create" } ]
        }
      },
      "input" : "arg1"
    }
},
// sink: [type SAPXSA_AUDIT_LOGGING].create(objectID, objectName): arg2
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING",
          "read" : [ { "property" : "create" } ]
        }
      },
      "input" : "arg2"
    }
},
// sink: [type SAPXSA_AUDIT_LOGGING].read(objectID, objectName): arg1
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
```

```
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING",
          "read" : [ { "property" : "read" } ]
        }
      },
      "input" : "arg1"
    }
  },
  // sink: [type SAPXSA_AUDIT_LOGGING].read(objectID, objectName): arg2
  {
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING",
          "read" : [ { "property" : "read" } ]
        }
      },
      "input" : "arg2"
    }
  },
  // sink: [type SAPXSA_AUDIT_LOGGING].update(objectID, objectName): arg1
  {
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING",
          "read" : [ { "property" : "update" } ]
        }
      },
      "input" : "arg1"
    }
  },
  // sink: [type SAPXSA_AUDIT_LOGGING].update(objectID, objectName): arg2
  {
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING",
          "read" : [ { "property" : "update" } ]
        }
      },
      "input" : "arg2"
    }
  },
  // sink: [type SAPXSA_AUDIT_LOGGING].delete(objectID, objectName): arg1
  {
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
```

```
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING",
          "read" : [ { "property" : "delete" } ]
        }
      },
      "input" : "arg1"
    }
},
// sink: [type SAPXSA_AUDIT_LOGGING].delete(objectID, objectName): arg2
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING",
          "read" : [ { "property" : "delete" } ]
        }
      },
      "input" : "arg2"
    }
},
// sink: [type SAPXSA_AUDIT_LOGGING].securityMessage(args)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING",
          "read" : [ { "property" : "securityMessage" } ]
        }
      },
      "input" : "all_args"
    }
},
// sink: [type SAPXSA_AUDIT_LOGGING_MESSAGE].attribute(name, val1, val2):arg1
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING_MESSAGE",
          "read" : [ { "property" : "attribute" } ]
        }
      },
      "input" : "arg1"
    }
},
// sink: [type SAPXSA_AUDIT_LOGGING_MESSAGE].attribute(name, val1, val2):arg2
```

```
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING_MESSAGE",
            "read" : [ { "property" : "attribute" } ]
         }
      },
      "input" : "arg2"
   }
},
// sink: [type SAPXSA_AUDIT_LOGGING_MESSAGE].attribute(name, val1, val2):arg3
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING_MESSAGE",
            "read" : [ { "property" : "attribute" } ]
         }
      },
      "input" : "arg3"
   }
},
// sink: [type SAPXSA_AUDIT_LOGGING_MESSAGE].customAttribute(name, value):arg1
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING_MESSAGE",
            "read" : [ { "property" : "customAttribute" } ]
         }
      },
      "input" : "arg1"
   }
},
// sink: [type SAPXSA_AUDIT_LOGGING_MESSAGE].customAttribute(name, value):arg2
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING_MESSAGE",
            "read" : [ { "property" : "customAttribute" } ]
         }
      },
      "input" : "arg2"
   }
```

```
},
// sink: [type SAPXSA_AUDIT_LOGGING_MESSAGE].by(driverOfAction)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING_MESSAGE",
            "read" : [ { "property" : "by" } ]
         }
      },
      "input" : "arg1"
   }
},
// sink: [type SAPXSA_AUDIT_LOGGING_MESSAGE].category(category)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING_MESSAGE",
            "read" : [ { "property" : "category" } ]
         }
      },
      "input" : "arg1"
   }
},
// sink: [type SAPXSA_AUDIT_LOGGING_MESSAGE].accessChannel(accessChannel)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING_MESSAGE",
            "read" : [ { "property" : "accessChannel" } ]
         }
      },
      "input" : "arg1"
   }
},
// sink: [type SAPXSA_AUDIT_LOGGING_MESSAGE].externalIP(externalIP)
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "logging",
   "sink" : {
      "to_callsite" : {
         "call_on" : {
            "read_from_object_of_type" : "SAPXSA_AUDIT_LOGGING_MESSAGE",
            "read" : [ { "property" : "externalIP" } ]
         }
      },
```

```
      "input" : "arg1"
    }
},
//
// sap-logging module
//
// sink: [type ExpressRequest].loggingContext.getLogger().info(arg1)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "output" : "return",
          "path"   : [ { "property" : "info" } ],
          "from_callsite" : {
            "call_on" : {
              "read_from_object_of_type" : "ExpressRequest",
              "read" : [ { "property" : "loggingContext" }, { "property" : "getLogger" } ]
            },
          }
        }
      },
      "input" : "arg1"
    }
},
// sink: [type ExpressRequest].loggingContext.getLogger().warning(arg1)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "output" : "return",
          "path"   : [ { "property" : "warning" } ],
          "from_callsite" : {
            "call_on" : {
              "read_from_object_of_type" : "ExpressRequest",
              "read" : [ { "property" : "loggingContext" }, { "property" : "getLogger" } ]
            },
          }
        }
      },
      "input" : "arg1"
    }
},
// sink: [type ExpressRequest].loggingContext.getLogger().error(arg1)
{
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "output" : "return",
```

```
            "path"   : [ { "property" : "error" } ],
            "from_callsite" : {
               "call_on" : {
                  "read_from_object_of_type" : "ExpressRequest",
                  "read" : [ { "property" : "loggingContext" }, { "property" : "getLogger" } ]
               },
            }
         },
         "input" : "arg1"
      }
   },
   // sink: [type ExpressRequest].loggingContext.getLogger().fatal(arg1)
   {
      "sink_for_checker" : "SENSITIVE_DATA_LEAK",
      "sink_kind" : "logging",
      "sink" : {
         "to_callsite" : {
            "call_on" : {
               "output" : "return",
               "path"   : [ { "property" : "fatal" } ],
               "from_callsite" : {
                  "call_on" : {
                     "read_from_object_of_type" : "ExpressRequest",
                     "read" : [ { "property" : "loggingContext" }, { "property" : "getLogger" } ]
                  },
               }
            },
            "input" : "arg1"
      }
   },
   // sink: [type ExpressRequest].loggingContext.getTracer().info(arg1)
   {
      "sink_for_checker" : "SENSITIVE_DATA_LEAK",
      "sink_kind" : "logging",
      "sink" : {
         "to_callsite" : {
            "call_on" : {
               "output" : "return",
               "path"   : [ { "property" : "info" } ],
               "from_callsite" : {
                  "call_on" : {
                     "read_from_object_of_type" : "ExpressRequest",
                     "read" : [ { "property" : "loggingContext" }, { "property" : "getTracer" } ]
                  },
               }
            },
            "input" : "arg1"
      }
   },
   // sink: [type ExpressRequest].loggingContext.getTracer().entering(args)
   {
```

```
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "output" : "return",
          "path"   : [ { "property" : "entering" } ],
          "from_callsite" : {
            "call_on" : {
              "read_from_object_of_type" : "ExpressRequest",
              "read" : [ { "property" : "loggingContext" }, { "property" : "getTracer" } ]
            },
          }
        }
      },
      "input" : "all_args"
    }
  },
  // sink: [type ExpressRequest].loggingContext.getTracer().exiting(arg1)
  {
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "output" : "return",
          "path"   : [ { "property" : "exiting" } ],
          "from_callsite" : {
            "call_on" : {
              "read_from_object_of_type" : "ExpressRequest",
              "read" : [ { "property" : "loggingContext" }, { "property" : "getTracer" } ]
            },
          }
        }
      },
      "input" : "arg1"
    }
  },
  // sink: [type ExpressRequest].loggingContext.getTracer().throwing(arg1)
  {
    "sink_for_checker" : "SENSITIVE_DATA_LEAK",
    "sink_kind" : "logging",
    "sink" : {
      "to_callsite" : {
        "call_on" : {
          "output" : "return",
          "path"   : [ { "property" : "throwing" } ],
          "from_callsite" : {
            "call_on" : {
              "read_from_object_of_type" : "ExpressRequest",
              "read" : [ { "property" : "loggingContext" }, { "property" : "getTracer" } ]
            },
          }
        }
```

```
            },
            "input" : "arg1"
         }
      },
      // sink: [type ExpressRequest].loggingContext.getTracer().catching(arg1)
      {
         "sink_for_checker" : "SENSITIVE_DATA_LEAK",
         "sink_kind" : "logging",
         "sink" : {
            "to_callsite" : {
               "call_on" : {
                  "output" : "return",
                  "path"   : [ { "property" : "catching" } ],
                  "from_callsite" : {
                     "call_on" : {
                        "read_from_object_of_type" : "ExpressRequest",
                        "read" : [ { "property" : "loggingContext" }, { "property" : "getTracer" } ]
                     },
                  }
               }
            },
            "input" : "arg1"
         }
      },
   ]
////////////////////////////////////////////////////////////////////////////////
// 4) [temporary] Python milestone 1 sources and sinks.
--
         "sink_for_checker" : "SENSITIVE_DATA_LEAK",
         "sink_kind" : "logging",
         "sink" : {
            "write_off_any" : [ { "property" : "sink_logging" } ]
         }
      },
      // temporary milestone 1 sink: any.sink_logging_method( <arg1> )
      {
         "sink_for_checker" : "SENSITIVE_DATA_LEAK",
         "sink_kind" : "logging",
         "sink" : {
            "input" : "arg1",
            "to_callsite" : {
               "call_on" : {
                  "read_off_any" : [ { "property" : "sink_logging_method" } ]
               },
            }
         }
      },
      // temporary milestone 1 sink: any.sink_filesystem
      {
         "sink_for_checker" : "SENSITIVE_DATA_LEAK",
         "sink_kind" : "filesystem",
         "sink" : {
            "write_off_any" : [ { "property" : "sink_filesystem" } ]
         }
```

```
},
// temporary milestone 1 sink: any.sink_filesystem_method( <arg1> )
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "filesystem",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_off_any" : [ { "property" : "sink_filesystem_method" } ]
         },
      }
   }
},
// temporary milestone 1 sink: any.sink_database
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "database",
   "sink" : {
      "write_off_any" : [ { "property" : "sink_database" } ]
   }
},
// temporary milestone 1 sink: any.sink_database_method( <arg1> )
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "database",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_off_any" : [ { "property" : "sink_database_method" } ]
         },
      }
   }
},
// temporary milestone 1 sink: any.sink_cookie
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "cookie",
   "sink" : {
      "write_off_any" : [ { "property" : "sink_cookie" } ]
   }
},
// temporary milestone 1 sink: any.sink_cookie_method( <arg1> )
{
   "sink_for_checker" : "SENSITIVE_DATA_LEAK",
   "sink_kind" : "cookie",
   "sink" : {
      "input" : "arg1",
      "to_callsite" : {
         "call_on" : {
            "read_off_any" : [ { "property" : "sink_cookie_method" } ]
         },
      }
```

```
        }
      },
      // temporary milestone 1 sink: any.sink_transit
      {
        "sink_for_checker" : "SENSITIVE_DATA_LEAK",
        "sink_kind" : "transit",
        "sink" : {
          "write_off_any" : [ { "property" : "sink_transit" } ]
        }
      },
      // temporary milestone 1 sink: any.sink_transit_method( <arg1> )
      {
        "sink_for_checker" : "SENSITIVE_DATA_LEAK",
        "sink_kind" : "transit",
        "sink" : {
          "input" : "arg1",
          "to_callsite" : {
            "call_on" : {
              "read_off_any" : [ { "property" : "sink_transit_method" } ]
            },
          }
        }
      },
    ]
////////////////////////////////////////////////////////////////////////////
// 5) Swift sources and sinks.
    "type" : "Coverity analysis configuration",
    "format_version" : 10,
    "language" : "Swift",
    "directives" : [
      // -------------------- BEGIN SOURCES ---------------------------------------
      // token source: Accounts.ACAccountCredential oauthToken property getter
      {
--
      "sink_for_checker" : "SESSION_FIXATION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.apache\\.catalina\\.Session\\.setId\\(java\\.lang\\.String.*"
          }
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "SESSION_FIXATION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.apache\\.catalina\\.connector\\.Request\\.setRequestedSessionId
\(java\\.lang\\.String.*"
          }
        },
        "param_index" : 1
```

```
      }
     }
    ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
 * Revision History
 * May 2017 - Bug 103331:
 *     Inital support for SAP's HANA XS classic (XSC).
 * 7/28/2017 - Updated directive layout, added PHP.
 * 8/04/2017 - Added Python.
////////////////////////////////////////////////////////////////////////////
// JavaScript sinks are here for a historical reason.
// 1) Node.js API sinks.
// 2) HANA XSC sinks.
--
     "sink_for_checker" : "SQLI",
     "sink" : {
      "input" : "arg1",
      "to_callsite" : {
       "call_on" : {
        "read" : [ { "property" : "execSql" } ],
        "read_from_object_of_type" : "TediousConnection"
       }
      }
     }
    },
    // sink: [type TediousConnection].execSqlBatch(arg1)
    {
     "sink_for_checker" : "SQLI",
     "sink" : {
      "input" : "arg1",
      "to_callsite" : {
       "call_on" : {
        "read" : [ { "property" : "execSqlBatch" } ],
        "read_from_object_of_type" : "TediousConnection"
       }
      }
     }
    },
    // sink: [type TediousConnection].callProcedure(arg1)
    {
     "sink_for_checker" : "SQLI",
     "sink" : {
      "input" : "arg1",
      "to_callsite" : {
       "call_on" : {
        "read" : [ { "property" : "callProcedure" } ],
        "read_from_object_of_type" : "TediousConnection"
       }
      }
     }
    },
    // sink: [type TediousConnection].execBulkLoad(arg1)
    {
     "sink_for_checker" : "SQLI",
```

```
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "execBulkLoad" } ],
        "read_from_object_of_type" : "TediousConnection"
      }
    }
  }
},
// sink: [type TediousConnection].prepare(arg1)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "prepare" } ],
        "read_from_object_of_type" : "TediousConnection"
      }
    }
  }
},
// sink: [type TediousConnection].execute(arg1, parameters)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "execute" } ],
        "read_from_object_of_type" : "TediousConnection"
      }
    }
  }
},
// sink: [type MSSqlConnection].query(arg1)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "query" } ],
        "read_from_object_of_type" : "MSSqlConnection"
      }
    }
  }
},
// sink: request('mssql').query(arg1)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
```

```
    "to_callsite" : {
      "call_on" : {
        "path" : [ { "property" : "query" } ],
        "read_from_js_require" : "mssql"
      }
    }
  }
},
// sink: [type MSSqlRequest].query(arg1)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "query" } ],
        "read_from_object_of_type" : "MSSqlRequest"
      }
    }
  }
},
// sink: [type MSSqlRequest].execute(arg1)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "execute" } ],
        "read_from_object_of_type" : "MSSqlRequest"
      }
    }
  }
},
// sink: [type MSSqlRequest].input(arg1)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "input" } ],
        "read_from_object_of_type" : "MSSqlRequest"
      }
    }
  }
},
// sink: [type MSSqlRequest].output(arg1)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
```

```
        "read" : [ { "property" : "output" } ],
        "read_from_object_of_type" : "MSSqlRequest"
      }
    }
  }
},
// sink: [type MSSqlRequest].bulk(arg1.columns)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "columns" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "bulk" } ],
        "read_from_object_of_type" : "MSSqlRequest"
      }
    }
  }
},
// sink: [type MSSqlRequest].bulk(arg1.rows)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "rows" } ],
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "bulk" } ],
        "read_from_object_of_type" : "MSSqlRequest"
      }
    }
  }
},
// sink: new require('mssql').Table(arg1)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "new_on" : {
        "path" : [ { "property" : "Table" } ],
        "read_from_js_require" : "mssql"
      }
    }
  }
},
// sink: [type MSSqlPreparedStatement].prepare(arg1)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
```

```
        "read" : [ { "property" : "prepare" } ],
        "read_from_object_of_type" : "MSSqlPreparedStatement"
      }
    }
  }
},
// 2) HANA XSC sinks.
// sink: [type SAPHanaDatabaseConn].prepareStatement(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "prepareStatement" } ],
        "read_from_object_of_type" : "SAPHanaDatabaseConn"
      }
    }
  }
},
// sink: [type SAPHanaDatabaseConn].prepareCall(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "prepareCall" } ],
        "read_from_object_of_type" : "SAPHanaDatabaseConn"
      }
    }
  }
},
// sink: [type SAPHanaHANADatabaseConn].executeQuery(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "executeQuery" } ],
        "read_from_object_of_type" : "SAPHanaHANADatabaseConn"
      }
    }
  }
},
// sink: [type SAPHanaHANADatabaseConn].executeUpdate(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "executeUpdate" } ],
```

```
        "read_from_object_of_type" : "SAPHanaHANADatabaseConn"
      }
    }
  }
},
// sink: [type SAPHanaTextMiningSession].categorizeKNN({ inputDocumentSubquery: <x> }

{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "inputDocumentSubquery" } ],
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "categorizeKNN" } ]
      }
    }
  }
},
// sink: [type SAPHanaTextMiningSession].categorizeKNN({ inputDocumentCondition: <x> }

{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "inputDocumentCondition" } ],
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "categorizeKNN" } ]
      }
    }
  }
},
// sink: [type SAPHanaTextMiningSession].categorizeKNN({ documentRestriction: <x> })
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "path" : [ { "property" : "documentRestriction" } ],
    "to_callsite" : {
      "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "categorizeKNN" } ]
      }
    }
  }
},
// sink: [type SAPHanaTextMiningSession].categorizeKNN({ termTypeRestriction: <x> })
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
```

```
      "path" : [ { "property" : "termTypeRestriction" } ],
      "to_callsite" : {
       "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "categorizeKNN" } ]
       }
      }
     }
    },
    // sink: [type SAPHanaTextMiningSession].getRelatedDocuments({ inputDocumentSubquery
 <x> })
    {
     "sink_for_checker" : "SQLI",
     "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "inputDocumentSubquery" } ],
      "to_callsite" : {
       "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "getRelatedDocuments" } ]
       }
      }
     }
    },
    // sink: [type SAPHanaTextMiningSession].getRelatedDocuments({ inputDocumentConditio
: <x> })
    {
     "sink_for_checker" : "SQLI",
     "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "inputDocumentCondition" } ],
      "to_callsite" : {
       "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "getRelatedDocuments" } ]
       }
      }
     }
    },
    // sink: [type SAPHanaTextMiningSession].getRelatedDocuments({ documentRestriction: <
> })
    {
     "sink_for_checker" : "SQLI",
     "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "documentRestriction" } ],
      "to_callsite" : {
       "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "getRelatedDocuments" } ]
       }
      }
     }
    },
```

```
  // sink: [type SAPHanaTextMiningSession].getRelatedDocuments({ termTypeRestriction: <x
})
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "termTypeRestriction" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getRelatedDocuments" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getRelatedTerms({ inputDocumentSubquery: <x
})
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "inputDocumentSubquery" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getRelatedTerms" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getRelatedTerms({ inputDocumentCondition: <x
})
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "inputDocumentCondition" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getRelatedTerms" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getRelatedTerms({ documentRestriction: <x> })
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "documentRestriction" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
```

```
          "read" : [ { "property" : "getRelatedTerms" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getRelatedTerms({ termTypeRestriction: <x> })
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "termTypeRestriction" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getRelatedTerms" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getRelevantDocuments({ inputDocumentSubque
y: <x> })
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "inputDocumentSubquery" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getRelevantDocuments" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getRelevantDocuments({ inputDocumentConditi
n: <x> })
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "inputDocumentCondition" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getRelevantDocuments" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getRelevantDocuments({ documentRestriction: <
> })
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
```

```
      "input" : "arg1",
      "path" : [ { "property" : "documentRestriction" } ],
      "to_callsite" : {
       "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "getRelevantDocuments" } ]
       }
      }
     }
    },
    // sink: [type SAPHanaTextMiningSession].getRelevantDocuments({ termTypeRestriction: <
> })
    {
     "sink_for_checker" : "SQLI",
     "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "termTypeRestriction" } ],
      "to_callsite" : {
       "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "getRelevantDocuments" } ]
       }
      }
     }
    },
    // sink: [type SAPHanaTextMiningSession].getRelevantTerms({ inputDocumentSubquery: <
> })
    {
     "sink_for_checker" : "SQLI",
     "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "inputDocumentSubquery" } ],
      "to_callsite" : {
       "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "getRelevantTerms" } ]
       }
      }
     }
    },
    // sink: [type SAPHanaTextMiningSession].getRelevantTerms({ inputDocumentCondition: <
> })
    {
     "sink_for_checker" : "SQLI",
     "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "inputDocumentCondition" } ],
      "to_callsite" : {
       "call_on" : {
        "read_from_object_of_type" : "SAPHanaTextMiningSession",
        "read" : [ { "property" : "getRelevantTerms" } ]
       }
      }
     }
```

```
  },
  // sink: [type SAPHanaTextMiningSession].getRelevantTerms({ documentRestriction: <x> })
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "documentRestriction" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getRelevantTerms" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getRelevantTerms({ termTypeRestriction: <x> })
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "termTypeRestriction" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getRelevantTerms" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getSuggestedTerms({ inputDocumentSubquery:
x> })
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "inputDocumentSubquery" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getSuggestedTerms" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getSuggestedTerms({ inputDocumentCondition:
<x> })
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "inputDocumentCondition" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
```

```
          "read" : [ { "property" : "getSuggestedTerms" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getSuggestedTerms({ documentRestriction: <x>
)
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "documentRestriction" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getSuggestedTerms" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaTextMiningSession].getSuggestedTerms({ termTypeRestriction: <x> }

  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "path" : [ { "property" : "termTypeRestriction" } ],
      "to_callsite" : {
        "call_on" : {
          "read_from_object_of_type" : "SAPHanaTextMiningSession",
          "read" : [ { "property" : "getSuggestedTerms" } ]
        }
      }
    }
  },
  // sink: [type SAPHanaSqlExecutor].callProcedure(<x>)
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "callProcedure" } ],
          "read_from_object_of_type" : "SAPHanaSqlExecutor"
        }
      }
    }
  },
  // sink: [type SAPHanaSqlExecutor].execQuery(<x>)
  {
    "sink_for_checker" : "SQLI",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
```

```
      "call_on" : {
       "read" : [ { "property" : "execQuery" } ],
       "read_from_object_of_type" : "SAPHanaSqlExecutor"
      }
     }
    }
},
// sink: [type SAPHanaSqlExecutor].execSingle(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
       "read" : [ { "property" : "execSingle" } ],
       "read_from_object_of_type" : "SAPHanaSqlExecutor"
      }
     }
    }
},
// sink: [type SAPHanaSqlExecutor].execSingleIgnoreFailing(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
       "read" : [ { "property" : "execSingleIgnoreFailing" } ],
       "read_from_object_of_type" : "SAPHanaSqlExecutor"
      }
     }
    }
},
// 3) HANA XSA sinks.
// sink: [type SAPXSA_HDB_Client].exec(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
       "read" : [ { "property" : "exec" } ],
       "read_from_object_of_type" : "SAPXSA_HDB_Client"
      }
     }
    }
},
// sink: [type SAPXSA_HDB_Client].execute(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
```

```
        "read" : [ { "property" : "execute" } ],
        "read_from_object_of_type" : "SAPXSA_HDB_Client"
      }
    }
  }
},
// sink: [type SAPXSA_HDB_Client]._execute(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "_execute" } ],
        "read_from_object_of_type" : "SAPXSA_HDB_Client"
      }
    }
  }
},
// sink: [type SAPXSA_HDB_Client].prepare(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "prepare" } ],
        "read_from_object_of_type" : "SAPXSA_HDB_Client"
      }
    }
  }
},
// sink: [type SAPXSA_HDB_Client]._prepare(<x>)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
    "input" : "arg1",
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "_prepare" } ],
        "read_from_object_of_type" : "SAPXSA_HDB_Client"
      }
    }
  }
},
// Sequelize sinks
// sink: [Module.sequelize].query(arg1)
{
  "sink_for_checker" : "SQLI",
  "sink" : {
     "input" : "arg1",
     "to_callsite" : {
        "call_on" : {
           "read_from_object_of_type" : "Module.sequelize",
```

```
            "read" : [ { "property" : "query" } ]
          }
        }
      }
    },
    // sink: [Module.postgresql.client].query(arg1)
    {
      "sink_for_checker" : "SQLI",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_from_object_of_type" : "Module.postgresql.client",
            "read" : [ { "property" : "query" } ]
          }
        }
      }
    },
    // sink: [Module.mysql].query(arg1)
    {
      "sink_for_checker" : "SQLI",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_from_object_of_type" : "Module.mysql.connection",
            "read" : [ { "property" : "query" } ]
          }
        }
      }
    },
    // sink: [Module.sqlite.database].all(arg1)
    {
      "sink_for_checker" : "SQLI",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_from_object_of_type" : "Module.sqlite.database",
            "read" : [ { "property" : "all" } ]
          }
        }
      }
    },
    // sink: [Module.sqlite.database].each(arg1)
    {
      "sink_for_checker" : "SQLI",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_from_object_of_type" : "Module.sqlite.database",
            "read" : [ { "property" : "each" } ]
          }
```

原文链接：coverity 部分规则提取

```
        }
      }
    },
    // sink: [Module.sqlite.database].get(arg1)
    {
      "sink_for_checker" : "SQLI",
      "sink" : {
        "input" : "arg1",
        "to_callsite" : {
          "call_on" : {
            "read_from_object_of_type" : "Module.sqlite.database",
            "read" : [ { "property" : "get" } ]
          }
        }
      }
    },
////////////////////////////////////////////////////////////////////////////
// End of directive list
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 7,
    "language" : "any",
    "directives" : [
      //
      // # Unknown Language Injection
      //
--
      "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.antlr\\.runtime\\.ANTLRStringStream\\.<init>\\(.*"
        },
        "param_index" : 1
      }
    },
    // ### Antlr 4
    {
      "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.antlr\\.v4\\.runtime\\.ANTLRInputStream\\.<init>\\(java\\.io\\.Inpu
Stream.*"
        },
        "param_index" : 1
      }
    }
  ]
 },
 // ### C# directives
   "type" : "Coverity analysis configuration",
   "format_version" : 4,
   "language" : "C#",
   "directives" : [
    // ## Sinks
```

原文链接：coverity 部分规则提取

```
    // ### Antlr 3
    {
      "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Antlr.Runtime.ANTLRStringStream::.ctor(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Antlr.Runtime.ANTLRStringStream::.ctor(System.Char[], System.Int32)Syste
.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Antlr.Runtime.ANTLRStringStream::.ctor(System.String,System.String)Syste
.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Antlr.Runtime.ANTLRStringStream::.ctor(System.Char[], System.Int32, Syst
m.String)System.Void"
        },
        "param_index" : 1
      }
    },
    // ### Antlr 4
    {
      "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Antlr4.Runtime.AntlrInputStream::.ctor(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
      "sink" : {
        "methods" : {
```

```
        "named" : "Antlr4.Runtime.AntlrInputStream::.ctor(System.IO.Stream)System.Void"
       },
       "param_index" : 1
     }
    },
    {
     "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
     "sink" : {
       "methods" : {
        "named" : "Antlr4.Runtime.AntlrInputStream::.ctor(System.IO.TextReader)System.Void"
       },
       "param_index" : 1
     }
    },
    {
     "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
     "sink" : {
       "methods" : {
        "named" : "Antlr4.Runtime.AntlrInputStream::.ctor(System.Char[], System.Int32)System
Void"
       },
       "param_index" : 1
     }
    },
    {
     "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
     "sink" : {
       "methods" : {
        "named" : "Antlr4.Runtime.AntlrInputStream::.ctor(System.IO.Stream, System.Int32)Sys
em.Void"
       },
       "param_index" : 1
     }
    },
    {
     "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
     "sink" : {
       "methods" : {
        "named" : "Antlr4.Runtime.AntlrInputStream::.ctor(System.IO.TextReader, System.Int32
System.Void"
       },
       "param_index" : 1
     }
    },
    {
     "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
     "sink" : {
       "methods" : {
        "named" : "Antlr4.Runtime.AntlrInputStream::.ctor(System.IO.Stream, System.Int32, Sys
em.Int32)System.Void"
       },
       "param_index" : 1
     }
    },
```

```
    {
      "sink_for_checker" : "UNKNOWN_LANGUAGE_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "Antlr4.Runtime.AntlrInputStream::.ctor(System.IO.TextReader, System.Int32,
System.Int32)System.Void"
        },
        "param_index" : 1
      }
    }
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 7,
  "language" : "any",
  "directives" : [
    //
    // # Unrestricted Dispatch
    //
    // CWE ID: 73
    {
      "dataflow_checker_name" : "UNRESTRICTED_DISPATCH",
      "languages" : {
--
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "javax\\.servlet\\.Servlet(Context|Request)\\.getRequestDispatcher\\(.*"
          }
        },
        "param_index" : 1
      }
    },
    /*
    // implements interface from above
    {
      "sink_for_checker": "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods": {
          "matching": "javax\\.servlet\\.(ServletRequestWrapper|http\\.HttpServletRequestWrap
er)\\.getRequestDispatcher\\(.*"
        },
        "param_index": 1
      }
    },
    */
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "matching" : "javax\\.servlet\\.AsyncContext\\.dispatch\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
```

```
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "matching" : "javax\\.servlet\\.AsyncContext\\.dispatch\\(javax\\.servlet\\.ServletContex
, java\\.lang\\.String.*"
        },
        "param_index" : 2
      }
    },
    // ## Sanitizers
    // Any sanitizer that escapes both / and \ are valid. This prevents
    // path traversal issues enough. URLEncoder returns the sanitzied value
    // it does not sanitize its argument.
    {
      "sanitizer_for_checker" : "UNRESTRICTED_DISPATCH",
      "sanitizer" : {
        "return_value_of" : {
          "matching" : "java\\.net\\.URLEncoder\\.encode\\(.*"
        },
      }
    }
  ]
--
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.UI.Page::set_MasterPageFile(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.ViewResultBase::set_ViewName(System.String)System.Vo
d"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.ViewResult::set_MasterName(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
```

```
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.Controller::View(System.String)System.Web.Mvc.ViewRes
lt"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.Controller::View(System.String,System.String)System.Web
Mvc.ViewResult"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.Controller::View(System.String,System.String)System.Web
Mvc.ViewResult"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.Controller::View(System.String,System.Object)System.We
.Mvc.ViewResult"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.Controller::View(System.String,System.String,System.Obje
t)System.Web.Mvc.ViewResult"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.Controller::View(System.String,System.String,System.Obje
```

```
t)System.Web.Mvc.ViewResult"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.Controller::PartialView(System.String)System.Web.Mvc.Pa
tialViewResult"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.Controller::PartialView(System.String,System.Object)Syst
m.Web.Mvc.PartialViewResult"
        },
        "param_index" : 1
      }
    },
    // XXX: The FilePathResult sinks could just as easily be PATH_MANIPULATION defects,
    //    but I've decided to include them here because they can be exploited in a
    //    very specific (as opposed to unknown general) way.
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.Controller::File(System.String,System.String)System.Web
Mvc.FilePathResult"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.Controller::File(System.String,System.String,System.String
System.Web.Mvc.FilePathResult"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.Mvc.FilePathResult::.ctor(System.String, System.String)System.
oid"
```

```
        },
        "param_index" : 1
      }
    },
    // Transfer methods in HttpServerUtility
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtility::Transfer(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtility::Transfer(System.String,System.Boolean)Syst
m.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtility::TransferRequest(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtility::TransferRequest(System.String,System.Bool
an)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtility::TransferRequest(System.String,System.Bool
an,System.String,System.Collections.Specialized.NameValueCollection,System.Boolean)System
Void"
        },
        "param_index" : 1
      }
    },
```

```
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtility::TransferRequest(System.String,System.Bool
an,System.String,System.Collections.Specialized.NameValueCollection)System.Void"
        },
        "param_index" : 1
      }
    },
    // Transfer methods in HttpServerUtilityBase
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityBase::Transfer(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityBase::Transfer(System.String,System.Boolean)
ystem.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityBase::TransferRequest(System.String)System.
oid"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityBase::TransferRequest(System.String,System.
oolean)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
```

```
          "named" : "System.Web.HttpServerUtilityBase::TransferRequest(System.String,System.
oolean,System.String,System.Collections.Specialized.NameValueCollection,System.Boolean)Sy
tem.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityBase::TransferRequest(System.String,System.
oolean,System.String,System.Collections.Specialized.NameValueCollection)System.Void"
        },
        "param_index" : 1
      }
    },
    // Transfer methods in HttpServerUtilityWrapper
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityWrapper::Transfer(System.String)System.Void

        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityWrapper::Transfer(System.String,System.Bool
an)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityWrapper::TransferRequest(System.String)Syst
m.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityWrapper::TransferRequest(System.String,Syst
m.Boolean)System.Void"
```

```
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityWrapper::TransferRequest(System.String,Syst
m.Boolean,System.String,System.Collections.Specialized.NameValueCollection,System.Boolean
System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNRESTRICTED_DISPATCH",
      "sink" : {
        "methods" : {
          "named" : "System.Web.HttpServerUtilityWrapper::TransferRequest(System.String,Syst
m.Boolean,System.String,System.Collections.Specialized.NameValueCollection)System.Void"
        },
        "param_index" : 1
      }
    },
    // ## Sanitizers
    {
      "sanitizer_for_checker" : "UNRESTRICTED_DISPATCH",
      "sanitizer" : {
        "methods" : {
          "named" : "System.Web.HttpUtility::UrlEncode(System.Byte[])System.String"
        },
        "param_index" : 1
      }
    },
    {
      "sanitizer_for_checker" : "UNRESTRICTED_DISPATCH",
      "sanitizer" : {
--
    sink_for_checker : "UNRESTRICTED_MESSAGE_TARGET",
    sink : {
      "input": "arg2",
      "to_callsite": {
        "call_on": {
          "read_from_object_of_type" : "Window",
          "read" : [ { "property" : "postMessage" } ]
        }
      }
    }
  }
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 10,
  "language" : "any",
```

```
    "directives" : [
      {
        "dataflow_checker_name" : "UNSAFE_DESERIALIZATION",
        "dataflow_checker_internal_name" : "__UNSAFE_DESERIALIZATION_BUDA",
        "languages" : {
            "PHP" : "Webapp-Security-Preview",
--
      "sink_for_checker" : "UNSAFE_DESERIALIZATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.beans\\.XMLDecoder\\.<init>\\((org\\.xml\\.sax\\.InputSource|java
\.io\\.InputStream).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNSAFE_DESERIALIZATION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.thoughtworks\\.xstream\\.XStream\\.fromXML\\(.*"
        },
        "param_index" : 1
      }
    },
    // Other sinks
    {
      "sink_for_checker" : "UNSAFE_DESERIALIZATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.io\\.ObjectInputStream\\.<init>\\(java\\.io\\.InputStream\\).*"
        },
        "param_index" : 1
      }
    }
    // Commented out for now
    // These are the true sinks, in that they call a private readObject0
    // method that is the sink. However, to correctly analyze these
    // ObjectInputStream would need to be properly modeled. For now, tainted
    // data going into the ctor is a reasonable defect.
    /*
    {
      "sink_for_checker" : "UNSAFE_DESERIALIZATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.io\\.ObjectInputStream\\.readObject\\(\\)"
        },
        "param_index" : 0
      }
    },
    {
      "sink_for_checker" : "UNSAFE_DESERIALIZATION",
      "sink" : {
        "methods" : {
```

```
        "matching" : "java\\.io\\.ObjectInputStream\\.readFields\\(\\)"
       },
       "param_index" : 0
     }
    },
    {
     "sink_for_checker" : "UNSAFE_DESERIALIZATION",
     "sink" : {
       "methods" : {
        "matching" : "java\\.io\\.ObjectInputStream\\.readObjectOverride\\(\\)"
       },
       "param_index" : 0
     }
    },
    {
     "sink_for_checker" : "UNSAFE_DESERIALIZATION",
     "sink" : {
       "methods" : {
        "matching" : "java\\.io\\.ObjectInputStream\\.readUnshared\\(\\)"
       },
       "param_index" : 0
     }
    },
    {
     "sink_for_checker" : "UNSAFE_DESERIALIZATION",
     "sink" : {
       "methods" : {
        "matching" : "java\\.io\\.ObjectInputStream\\.resolveObject\\(java\\.lang\\.Object\\)"
       },
       "param_index" : 0
     }
    },
    {
     "sink_for_checker" : "UNSAFE_DESERIALIZATION",
     "sink" : {
       "methods" : {
        "matching" : "java\\.io\\.ObjectInputStream\\.defaultReadObject\\(\\)"
       },
       "param_index" : 0
     }
    },
    {
     "sink_for_checker" : "UNSAFE_DESERIALIZATION",
     "sink" : {
       "methods" : {
        "matching" : "java\\.io\\.ObjectInputStream\\.readArray\\(\\)"
       },
       "param_index" : 0
     }
    }
    */
  ]
},
// ------------------------------------------------
```

```
  // C# Directives
  // ---------------------------------------------
    "type" : "Coverity analysis configuration",
    "format_version" : 5,
    "language" : "C#",
    "directives" : [
      // XXX: This checker is really really narrow. It only cares about
      // XXX: BinaryFormatter. There are plenty of other issues with, say
      // XXX: DataContractSerializer, that expose the same issues.
--
        "sink_for_checker" : "UNSAFE_DESERIALIZATION",
        "sink" : {
          "methods" : {
            "or" : [
              { "named" : "System.Runtime.Serialization.Formatters.Binary.BinaryFormatter::Deserial
ze(System.IO.Stream)System.Object" },
              { "named" : "System.Runtime.Serialization.Formatters.Binary.BinaryFormatter::Deserial
ze(System.IO.Stream,System.Runtime.Remoting.Messaging.HeaderHandler)System.Object" },
              { "named" : "System.Runtime.Serialization.Formatters.Binary.BinaryFormatter::Deserial
zeMethodResponse(System.IO.Stream,System.Runtime.Remoting.Messaging.HeaderHandler,
ystem.Runtime.Remoting.Messaging.IMethodCallMessage)System.Object" },
              { "named" : "System.Runtime.Serialization.Formatters.Binary.BinaryFormatter::Unsafe
eserialize(System.IO.Stream,System.Runtime.Remoting.Messaging.HeaderHandler)System.Obj
ct" },
              { "named" : "System.Runtime.Serialization.Formatters.Binary.BinaryFormatter::Unsafe
eserializeMethodResponse(System.IO.Stream,System.Runtime.Remoting.Messaging.HeaderH
ndler,System.Runtime.Remoting.Messaging.IMethodCallMessage)System.Object" }
            ]
          },
          "param_index" : 1
        }
      }
    ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 7,
    "language" : "any",
    "directives" : [
      //
--
        "sink_for_checker" : "UNSAFE_JNI",
        "sink" : {
          "methods" : {
            "matching" : "java\\.lang\\.(System|Runtime)\\.loadLibrary\\(.*"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "UNSAFE_JNI",
        "sink" : {
          "methods" : {
            "matching" : "java\\.lang\\.(System|Runtime)\\.load\\(.*"
          },
```

```
        "param_index" : 1
      }
     }
    ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 7,
    "language" : "any",
    "directives" : [
     //
     // # Unsafe Reflection
     //
     // CWE ID: 470
     {
      "dataflow_checker_name" : "UNSAFE_REFLECTION",
      "languages" : {
--
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
       "methods" : {
        "matching" : "java\\.lang\\.Class\\.(forName|getMethod|getDeclaredMethod|getDeclar
dField|getField)\\(.*"
       },
       "param_index" : 1
      }
     },
     {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
       "methods" : {
        "overrides" : {
         "matching" : "java\\.lang\\.ClassLoader\\.(defineClass|findClass|loadClass)\\(java\\.lan
\\.String.*"
        }
       },
       "param_index" : 1
      }
     },
     /*
     // Note: ELContext is not trackable nor modeled, The below are commented out.
     {
      "sink_for_checker": "UNSAFE_REFLECTION",
      "sink" : {
       "param_index": 1,
       "methods": { "matching" : "javax\\.el\\.BeanELResolver\\.BeanProperties\\.getBean
      }
     },
     // shouldn't be able to arbitrarily control destination variable
     {
      "sink_for_checker": "UNSAFE_REFLECTION",
      "sink" : {
       "param_index": 2,
       "methods" : {
        "overrides": { "matching" : "javax\\.el\\.ValueExpression\\.setValue\\(.*,.*"
```

```
    }
  }
},
// shouldn't be able to arbirtarily control parameter values passed to a method
{
  "sink_for_checker": "UNSAFE_REFLECTION",
  "sink" : {
    "param_index": 2,
    "methods": {
      "overrides": { "matching": "javax\\.el\\.MethodExpression\\.invoke\\(.*,.*" }
    }
  }
},
// shouldn't be able to arbitrarily control the base / target object
{
  "sink_for_checker": "UNSAFE_REFLECTION",
  "sink" : {
    "param_index": 1,
    "methods": { "matching" : "javax\\.(servlet\\.jsp\\.el\\.ImplicitObjectELResolve
  }
},
*/
// shouldn't be able to arbitrarily control the index, key, or property of the
// base object
{
  "sink_for_checker" : "UNSAFE_REFLECTION",
  "sink" : {
    "methods" : {
      "overrides" : {
        "matching" : "javax\\.el\\.ELResolver\\.(get|set)Value\\(.*,.*"
      }
    },
    "param_index" : 2
  }
},
// shouldn't be able to arbitrarily control destination variable
{
  "sink_for_checker" : "UNSAFE_REFLECTION",
  "sink" : {
    "methods" : {
      "overrides" : {
        "matching" : "javax\\.el\\.ELResolver\\.setValue\\(.*,.*,.*"
      }
    },
    "param_index" : 3
  }
},
// ### Spring Utils
// http://docs.spring.io/spring-framework/docs/3.1.0.RELEASE/javadoc-api/
// abstract
{
  "sink_for_checker" : "UNSAFE_REFLECTION",
  "sink" : {
    "methods" : {
```

```
          "overrides" : {
            "matching" : "org\\.springframework\\.util\\.ReflectionUtils\\.(findField|findMethod)\\
.*,.*"
          }
        },
        "param_index" : 2
      }
    },
    // ### Apache commons lang
    // http://commons.apache.org/proper/commons-lang/apidocs/index.html
    {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.commons\\.(lang3\\.reflect|beanutils)\\.MethodUtils\\.(ge
AccessibleMethod|getMatchingAccessibleMethod|invokeExactMethod|invokeExactStaticMeth
d|invokeMethod|invokeStaticMethod)\\(.*,.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.commons\\.lang3\\.reflect\\.FieldUtils\\.(get|read|write)(
eclared)?(Static)?Field\\(.*,.*"
        },
        "param_index" : 2
      }
    },
    // ### Apache commons beanutils
    // http://commons.apache.org/proper/commons-beanutils/javadocs/v1.8.3/apidocs/index
html
    {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.apache\\.commons\\.beanutils\\.(BeanUtils|BeanUtilsBean|Proper
yUtils|PropertyUtilsBean|LocaleBeanUtils|LocaleBeanUtilsBean)\\.(getArrayProperty|getIndexe
Property|getMappedProperty|getNestedProperty|getSimpleProperty)\\(.*,.*"
          }
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.apache\\.commons\\.beanutils\\.(BeanUtils|BeanUtilsBean|Proper
yUtils|PropertyUtilsBean|LocaleBeanUtils|LocaleBeanUtilsBean)\\.setProperty\\(.*,.*"
```

```
          }
        },
        "param_index" : 2
      }
    },
    // Apache JEXL 1
    {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.apache\\.commons\\.jexl\\.util\\.AbstractExecutor\\.execute\\(jav
\\.lang\\.Object\\).*"
          }
        },
        "param_index" : 1
      }
    },
    // ### FEST Reflection
    {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.fest\\.reflect\\.core\\.Reflection\\.(field|method|property|staticField
staticInnerClass|staticMethod|type)\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.fest\\.reflect\\.field\\.(FieldName\\.beginFieldAccess|StaticFieldNa
e\\.beginStaticFieldAccess)\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.fest\\.reflect\\.beanproperty\\.PropertyName\\.startPropertyAccess
\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.fest\\.reflect\\.innerclass\\.StaticInnerClassName\\.startStaticInnerC
```

```
assAccess\\(.*"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "UNSAFE_REFLECTION",
        "sink" : {
          "methods" : {
            "matching" : "org\\.fest\\.reflect\\.method\\.StaticMethodName\\.startStaticMethodAc
ess\\(.*"
          },
          "param_index" : 1
        }
      },
      {
        "sink_for_checker" : "UNSAFE_REFLECTION",
        "sink" : {
          "methods" : {
            "matching" : "org\\.fest\\.reflect\\.type\\.Type\\.newType\\(.*"
          },
          "param_index" : 1
        }
      },
       // JAVA 8 reflection
      {
        "sink_for_checker" : "UNSAFE_REFLECTION",
        "sink" : {
          "methods" : {
            "matching" : "java\\.lang\\.invoke\\.MethodHandles\\.Lookup\\.(findVirtual|findGetter|f
ndSetter|findSpecial|findStatic|findStaticGetter|findStaticSetter|findVirtual)\\(.*"
          },
          "param_index" : 2
        }
      },
      // Android specific reflection
      {
        "sink_for_checker" : "UNSAFE_REFLECTION",
        "sink" : {
          "methods" : {
            "matching" : "android\\.content\\.ComponentName\\.<init>\\((java\\.lang\\.String|and
oid\\.content\\.Context), java\\.lang\\.String.*"
          },
          "param_index" : 2
        }
      },
      {
        "sink_for_checker" : "UNSAFE_REFLECTION",
        "sink" : {
          "methods" : {
            "matching" : "android\\.content\\.Intent\\.setClassName\\((java\\.lang\\.String|android
\.content\\.Context), java\\.lang\\.String.*"
          },
          "param_index" : 2
```

```
      }
    },
    {
      "sink_for_checker" : "UNSAFE_REFLECTION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.app\\.Instrumentation\\.(newActivity|newApplication)\\(java\\.
ang\\.ClassLoader, java\\.lang\\.String.*"
        },
        "param_index" : 2
      }
    },
    // ## Sanitizers
    // Kludges borrowed from path-manipulation. They're here to satisfy
    // the remediation advice on safe comparisons.
    {
      "sanitizer_for_checker" : "UNSAFE_REFLECTION",
      "sanitizer" : {
        "methods" : {
          "matching" : "java\\.lang\\.String\\.equals\\(.*"
        },
        "param_index" : 0
      }
    }
  ]
--
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "java\\.net\\.URL\\.(openStream|getContent)\\(.*"
        },
        "param_index" : 0
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.content\\.Intent\\.<init>\\(java\\.lang\\.String, android\\.net\\
Uri.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.net\\.Network\\.openConnection\\(java\\.net\\.URL.*"
        },
        "param_index" : 1
      }
    },
    {
```

```
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.webkit\\.WebView\\.(loadDataWithBaseURL|loadUrl|postUrl)\\(
ava\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.app\\.DownloadManager\\$Request\\.<init>\\(android\\.net\
.Uri.*"
        },
        "param_index" : 1
      }
    },
    // ### ContentResolver
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.content\\.ContentResolver\\.acquireContentProviderClient\\((
ndroid\\.net\\.Uri|java\\.lang\\.String).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.content\\.ContentResolver\\.acquireUnstableContentProviderC
ient\\((android\\.net\\.Uri|java\\.lang\\.String).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.content\\.ContentResolver\\.acquireUnstableContentProviderC
ient\\((android\\.net\\.Uri|java\\.lang\\.String).*"
        },
        "param_index" : 1
      }
    },
    // The file related methods: openAssetFileDescriptor,
    // openFileDescriptor, openInputStream, openOutputStream and
    // openTypedAssetFileDescriptor are sinks for PATH_MANIPULATION
    // since they only accept file, content and android.resource Uri
```

```
    // schemes.
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.content\\.ContentResolver\\.(bulkInsert|call|delete|insert|query
update)\\(android\\.net\\.Uri.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.content\\.ContentResolver\\.applyBatch\\(java\\.lang\\.String.*

        },
        "param_index" : 1
      }
    },
    // ### apache.http
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.http\\.client\\.methods\\.HttpGet<init>\\((java\\.lang\\.St
ing.*|java\\.net\\.URI)"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.http\\.client\\.methods\\.HttpPost<init>\\((java\\.lang\\.S
ring.*|java\\.net\\.URI)"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.http\\.client\\.methods\\.HttpPut<init>\\((java\\.lang\\.St
ing.*|java\\.net\\.URI)"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
```

```
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.http\\.message\\.BasicHttpRequest<init>\\(java\\.lang\\.
tring, java\\.lang\\.String.*"
        },
        "param_index" : 2
      }
    },
    // ### okhttp
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.squareup\\.okhttp\\.Request\\.Builder\\.url\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.squareup\\.okhttp\\.okUrlFactory\\.open\\(java\\.net\\.URL.*"
        },
        "param_index" : 1
      }
    },
    // ### volley
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.android\\.volley\\.Request\\.open\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "URL_MANIPULATION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.android\\.volley\\.Request\\.open\\(int, java\\.lang\\.String.*"
        },
        "param_index" : 2
      }
    },
   // ## Sanitizers
    {
      "sanitizer_for_checker" : "URL_MANIPULATION",
      "sanitizer" : {
        "return_value_of" : {
          "matching" : "android\\.net\\.Uri\\.getEncodedFragment\\(.*"
        }
      }
```

原文链接：coverity 部分规则提取

```
    },
    {
      "sanitizer_for_checker" : "URL_MANIPULATION",
      "sanitizer" : {
        "return_value_of" : {
--
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.taglibs\\.standard\\.tag\\.(el|rt)\\.xml\\.(ParseTag|Transfo
mTag)\\.setXml\\(.*"
        },
        "param_index" : 1
      }
    },
    // ### Java API for XML Processing (JAXP)
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "javax\\.xml\\.parsers\\.DocumentBuilder\\.parse\\((java\\.io\\.InputStrea
|org\\.xml\\.sax\\.InputSource).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "javax\\.xml\\.parsers\\.SAXParser\\.parse\\((java\\.io\\.InputStream|org\\.
ml\\.sax\\.InputSource).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "javax\\.xml\\.stream\\.XML(Input|Output)Factory\\.createXML(Event|Stre
m)Reader\\((java\\.io\\.(Input|Output)Stream|org\\.xml\\.sax\\.InputSource).*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "javax\\.xml\\.bind\\.(JAXB|Unmarshaller)\\.unmarshal\\(.*"
        },
        "param_index" : 1
      }
```

原文链接：coverity 部分规则提取

```
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "javax\\.xml\\.bind\\.helpers\\.AbstractUnmarshallerImpl\\.unmarshal\\(.*

        },
        "param_index" : 1
      }
    },
    // ### XPath
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "javax\\.xml\\.xpath\\.XPathExpression\\.evaluate\\(org\\.xml\\.sax\\.InpuSource.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.xml\\.sax\\.Parser\\.parse\\(org\\.xml\\.sax\\.InputSource.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.xml\\.sax\\.InputSource\\.(<init>|setByteStream)\\(java\\.io\\.Inputtream.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.xml\\.sax\\.XMLReader\\.parse\\(java\\.io\\.InputStream.*"
          }
        },
        "param_index" : 1
      }
    },
    // ### Spring Web services
    {
```

```
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.ws\\.WebServiceMessageFactory\\.createWebS
rviceMessage\\(java\\.io\\.InputStream.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.xml\\.transform\\.StringSource\\.(<init>|setInp
tStream)\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.springframework\\.ws\\.soap\\.saaj\\.support\\.SaajXmlReader\\.pa
se\\(java\\.io\\.InputStream.*"
        },
        "param_index" : 1
      }
    },
    // ### Xalan (Apache XML)
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.xml\\.utils\\.DOM2Helper\\.parse\\(org\\.xml\\.sax\\.Inpu
Source.*"
        },
        "param_index" : 1
      }
    },
    // ### JXpath (Apache commons)
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.commons\\.jxpath\\.xml\\.DocumentContainer\\.parseX
L\\(java\\.io\\.InputStream.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
```

```
        "methods" : {
          "matching" : "org\\.apache\\.commons\\.jxpath\\.xml\\.(DocumentContainer|DOMPar
er|JDOMParser|XMLParser2)\\.parseXML\\(java\\.io\\.InputStream.*"
        },
        "param_index" : 1
      }
    },
    // ### dom4j
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.dom4j\\.DocumentHelper\\.parseText\\(.*"
        },
        "param_index" : 1
      }
    },
    // ### JDOM 1 AND 2
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.jdom2\\.input\\.SAXBuilder\\.build\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.jdom2\\.input\\.sax\\.SAX(Builder)?Engine\\.build\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.jdom2\\.input\\.stax\\.DTDParser\\.parse\\(.*"
        },
        "param_index" : 1
      }
    },
    // ### XOM
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "nu\\.xom\\.Builder\\.build\\(.*"
        },
        "param_index" : 1
      }
```

原文链接：coverity 部分规则提取

```
    },
    // ### Android sinks
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "android\\.util\\.Xml\\.parse\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.xmlpull\\.v1\\.XmlPullParser\\.setInput\\(.*"
        },
        "param_index" : 1
      }
    },
  ]
 },
 // C# Directives
  "type" : "Coverity analysis configuration",
  "format_version" : 4,
  "language" : "C#",
  "directives" : [
    // ## Sinks
    // ### LINQ (C#)
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XElement::Parse(System.String)System.Xml.Linq.XElement"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XElement::Parse(System.String,System.Xml.Linq.LoadOpti
ns)System.Xml.Linq.XElement"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XElement::Load(System.IO.Stream)System.Xml.Linq.XElem
nt"
```

```
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XElement::Load(System.IO.Stream,System.Xml.Linq.LoadO
tions)System.Xml.Linq.XElement"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XElement::Load(System.IO.TextReader)System.Xml.Linq.XE
ement"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XElement::Load(System.IO.TextReader,System.Xml.Linq.Lo
dOptions)System.Xml.Linq.XElement"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XDocument::Parse(System.String)System.Xml.Linq.XDocu
ent"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XDocument::Parse(System.String,System.Xml.Linq.LoadOpt
ons)System.Xml.Linq.XDocument"
        },
        "param_index" : 1
      }
    },
```

```
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XDocument::Load(System.IO.Stream)System.Xml.Linq.XDo
ument"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XDocument::Load(System.IO.Stream,System.Xml.Linq.Loa
Options)System.Xml.Linq.XDocument"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XDocument::Load(System.IO.TextReader)System.Xml.Linq.
Document"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.Linq.XDocument::Load(System.IO.TextReader,System.Xml.Linq.
oadOptions)System.Xml.Linq.XDocument"
        },
        "param_index" : 1
      }
    },
    // ### System.Xml readers (C#)
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlReader::Create(System.IO.Stream)System.Xml.XmlReader"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
```

```
          "named" : "System.Xml.XmlReader::Create(System.IO.Stream,System.Xml.XmlReaderSe
tings)System.Xml.XmlReader"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlReader::Create(System.IO.Stream,System.Xml.XmlReaderSe
tings,System.String)System.Xml.XmlReader"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlReader::Create(System.IO.Stream,System.Xml.XmlReaderSe
tings,System.Xml.XmlParserContext)System.Xml.XmlReader"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlReader::Create(System.IO.TextReader)System.Xml.XmlRead
r"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlReader::Create(System.IO.TextReader,System.Xml.XmlRead
rSettings)System.Xml.XmlReader"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlReader::Create(System.IO.TextReader,System.Xml.XmlRead
rSettings,System.String)System.Xml.XmlReader"
        },
        "param_index" : 1
```

```
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlReader::Create(System.IO.TextReader,System.Xml.XmlRead
rSettings,System.Xml.XmlParserContext)System.Xml.XmlReader"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlTextReader::.ctor(System.IO.Stream)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlTextReader::.ctor(System.IO.Stream,System.Xml.XmlNameT
ble)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlTextReader::.ctor(System.IO.Stream,System.Xml.XmlNodeT
pe,System.Xml.XmlParserContext)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlTextReader::.ctor(System.String,System.Xml.XmlNodeType,
ystem.Xml.XmlParserContext)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
```

```
        "methods" : {
          "named" : "System.Xml.XmlTextReader::.ctor(System.String,System.IO.Stream)System.
oid"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlTextReader::.ctor(System.String,System.IO.Stream,System.
ml.XmlNameTable)System.Void"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlTextReader::.ctor(System.String,System.IO.TextReader)Syst
m.Void"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlTextReader::.ctor(System.String,System.IO.TextReader,Syst
m.Xml.XmlNameTable)System.Void"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlTextReader::.ctor(System.IO.TextReader)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlTextReader::.ctor(System.IO.TextReader,System.Xml.XmlN
meTable)System.Void"
        },
        "param_index" : 1
```

```
      }
    },
    // TODO: There are also some methods on other XmlReader sub-classes.
    // ### System.Xml DOM (C#)
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlDocument::Load(System.IO.Stream)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlDocument::Load(System.IO.TextReader)System.Void"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XML_INJECTION",
      "sink" : {
        "methods" : {
          "named" : "System.Xml.XmlDocument::LoadXml(System.String)System.Void"
        },
        "param_index" : 1
      }
    },
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 7,
  "language" : "any",
  "directives" : [
    //
    // # XPath injection checker
    //
    // Note: http://java-source.net/open-source/xml-parsers lists
    //       25 different open source XML libraries. This checker
    //       understands most of them, but not all of them (mostly
    //       based on how widespread their use is)
--
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            //interface
            "matching" : "javax\\.xml\\.xpath\\.XPath\\.(compile|evaluate)\\(.*"
          }
        },
        "param_index" : 1
```

```
      }
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            //interface
            "matching" : "javax\\.xml\\.xpath\\.XPathExpression\\.evaluate\\(.*"
          }
        },
        "param_index" : 1
      }
    },
    // #### JAXP JVM internal implementations.
    // These are a mess because Oracle uses the Apache implementations for the
    // XPath APIs. However, these implementations aren't extending interfaces
    // so they're duplicated in essence.
    // Many are commented out below because the source / bytecode just passses
    // a string onto another method.
    // Search this JSON for "com\\.sun\\.org\\.apache" for other sinks. They're
    // sprinkled under the appropriate Apache section.
    // http://grepcode.com/file/repository.grepcode.com/java/root/jdk/openjdk/6-b14/com/
un/org/apache/xpath/internal/XPath.java#XPath.%3Cinit%3E%28com.sun.org.apache.xpath.in
ernal.Expression%29
    /*
--
      "sink_for_checker": "XPATH_INJECTION",
      "sink" : {
        "methods": {
          "matching": "^com\\.sun\\.org\\.apache\\.xpath\\.internal\\.(Cached)?XPathAPI\\.(eval|
electNodeIterator|selectNodeList|selectSingleNode)\\("
        },
        "param_index": 2
      }
    },
    // passing param 1 as the sink's param 1
    {
      "sink_for_checker": "XPATH_INJECTION",
      "sink" : {
        "methods": {
          "matching": "^com\\.sun\\.org\\.apache\\.xpath\\.internal\\.jaxp\\.XPathImpl\\.(compi
e|evaluate)\\("
        },
        "param_index": 1
      }
    },
    // passing param 1 as the sink's param 1
    {
      "sink_for_checker": "XPATH_INJECTION",
      "sink" : {
        "methods": {
          "matching": "^com\\.sun\\.org\\.apache\\.xpath\\.internal\\.domapi\\.XPathEvaluator
mpl\\.(createExpression|evaluate)\\("
```

```
        },
        "param_index": 1
      }
    },
    // Calls com.sun.org.apache.xpath.internal.dtm.DTMManager(java\\.lang\\.String, .*),
    // passing the String parameter directly.
    {
      "sink_for_checker": "XPATH_INJECTION",
      "sink" : {
        "methods": {
          "matching": "^com\\.sun\\.org\\.apache\\.xpath\\.internal\\.XPathContext\\.createDT
Iterator\\(java\\.lang\\.(String|Object), .*"
        },
        "param_index": 1
      }
    },
    */
    // ### Xalan (Apache XML)
    // https://xml.apache.org/xalan-j/apidocs/index.html
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "(com\\.sun\\.org\\.apache\\.xpath\\.internal|org\\.apache\\.xpath)\\.XPa
h\\.<init>\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            // Abstract class
            "matching" : "(com\\.sun\\.org\\.apache\\.xml\\.internal|org\\.apache\\.xml)\\.dtm\\.
TMManager\\.createDTMIterator\\(java\\.lang\\.String.*"
          }
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            // interface
            "matching" : "(com\\.sun\\.org\\.apache\\.xpath\\.internal|org\\.apache\\.xpath)\\.XP
thFactory\\.create\\(java\\.lang\\.String.*"
          }
        },
        "param_index" : 1
      }
```

原文链接：coverity 部分规则提取

```
    },
    /*
    // Calls org.apache.xpath.XPath.<init>(), passing param 2 as the sink's param 1
    {
      "sink_for_checker": "XPATH_INJECTION",
      "sink" : {
        "methods": {
          "matching": "^org\\.apache\\.xpath\\.(Cached)?XPathAPI\\.(eval|selectNodeIterator
        },
        "param_index": 2
      }
    },
    */
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "(com\\.sun\\.org\\.apache\\.xpath\\.internal|org\\.apache\\.xpath)\\.com
iler\\.XPathParser\\.(initMatchPattern|initXPath)\\(.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "org\\.w3c\\.dom\\.xpath\\.XPathEvaluator\\.(createExpression|evaluate)
\(.*"
          }
        },
        "param_index" : 1
      }
    },
    // ### JXpath (Apache commons)
    // Note that we do not support the `JXPathServletContexts` as we would
    // need to parse the XPath query ourselves to extract the variables
    // dereferenced from the different scopes (page, request, session, application)
    // An example from the doc shows that using this servlet, we can do something
    // like this in xpath:
    // ```
    //  $request/myvar
    // ```
    // to get access to `myvar` from the request attributes map.
    // http://commons.apache.org/proper/commons-jxpath/javadocs/api-1.2/
--
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            // abstract class
            "matching" : "org\\.apache\\.commons\\.jxpath\\.JXPathContext\\.(compile|createPat
(AndSetValue)?|getPointer|getValue|iterate(Pointers)?|remove(All|Path)|select(Single)?Node(s)?
```

```
setValue)\\(.*"
          }
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            // interface
            "matching" : "org\\.apache\\.commons\\.jxpath\\.CompiledExpression\\.getPointer\\(
*"
          }
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.commons\\.jxpath\\.ri\\.JXPathCompiledExpression\\.<ini
>\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    },
    // ### XMLDB-Org
    // http://xmldb-org.sourceforge.net/xapi/api/org/xmldb/api/modules/XPathQueryService
html
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.xmldb\\.api\\.modules\\.XPathQueryService\\.query\\(.*"
        },
        "param_index" : 1
      }
    },
    // ### Jaxen
    // http://jaxen.codehaus.org/apidocs/
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            // base class extends a lot
            "matching" : "^org\\.jaxen\\.BaseXPath\\.<init>\\(java\\.lang\\.String.*"
          }
        },
        "param_index" : 1
      }
```

原文链接：coverity 部分规则提取

```
},
{
  "sink_for_checker" : "XPATH_INJECTION",
  "sink" : {
    "methods" : {
      "overrides" : {
        // interface
        "matching" : "org\\.jaxen\\.saxpath\\.XPathReader\\.parse\\(java\\.lang\\.String.*"
      }
    },
    "param_index" : 1
  }
},
/*
{
  "sink_for_checker": "XPATH_INJECTION",
  "sink" : {
    "methods": {
      "matching": "^org\\.jaxen\\.dom4j\\.Dom4jXPath\\.<init>\\(java\\.lang\\.String.*"
    },
    "param_index": 1
  }
},
{
  "sink_for_checker": "XPATH_INJECTION",
  "sink" : {
    "methods": {
      "matching": "^org\\.jaxen\\.dom\\.DOMXPath\\.<init>\\(java\\.lang\\.String.*"
    },
    "param_index": 1
  }
},
{
  "sink_for_checker": "XPATH_INJECTION",
  "sink" : {
    "methods": {
      "matching": "^org\\.jaxen\\.javabean\\.JavaBeanXPath\\.<init>\\(java\\.lang\\.String.*"
    },
    "param_index": 1
  }
},
{
  "sink_for_checker": "XPATH_INJECTION",
  "sink" : {
    "methods": {
      "matching": "^org\\.jaxen\\.jdom\\.JDOMXPath\\.<init>\\(java\\.lang\\.String.*"
    },
    "param_index": 1
  }
},
{
  "sink_for_checker": "XPATH_INJECTION",
  "sink" : {
    "methods": {
```

```
        "matching": "^org\\.jaxen\\.xom\\.XOMXPath\\.<init>\\(java\\.lang\\.String.*"
      },
      "param_index": 1
    }
  },
  */
  // ### dom4j
  // http://dom4j.sourceforge.net/dom4j-1.6.1/apidocs/
  {
    "sink_for_checker" : "XPATH_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.dom4j\\.xpath\\.DefaultXPath\\.<init>\\(java\\.lang\\.String\\).*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "XPATH_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.dom4j\\.xpath\\.XPathPattern\\.<init>\\(java\\.lang\\.String\\).*"
      },
      "param_index" : 1
    }
  },
  // ### Saxon
  // http://www.saxonica.com/html/documentation/javadoc/index.html
  {
    "sink_for_checker" : "XPATH_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "net\\.sf\\.saxon\\.s9api\\.XPathCompiler\\.(compile|evaluate(Single)?)\\(j
va\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "XPATH_INJECTION",
    "sink" : {
      "methods" : {
        "overrides" : {
          "matching" : "net\\.sf\\.saxon\\.(s)?xpath\\.XPathEvaluator\\.createExpression\\(java\\.
ang\\.String.*"
        }
      },
      "param_index" : 1
    }
  },
  /*
  // The below class implements javax.xml.xpath.XPath, so its compile and
  //  evaluate methods aren't modeled
  {
```

```
      "sink_for_checker": "XPATH_INJECTION",
      "sink" : {
        "methods": {
          "matching": "^net\\.sf\\.saxon\\.xpath\\.(XPathEvaluator|XPathExpressionImpl)\\.(comp
le|createExpression|evaluate)\\(java\\.lang\\.String.*"
        },
        "param_index": 1
      }
    },
    */
    // ### JXP
    // http://www.japisoft.com/jxpath/javadoc/index.html
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.japisoft\\.xpath\\.XPath\\.setXPathExpression\\(java\\.lang\\.String
*"
        },
        "param_index" : 1
      }
    },
    // ### Resin
    // http://javadoc4.caucho.com/index.html?com/caucho/xpath/package-summary.html
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.caucho\\.xpath\\.XPath\\.(eval(Boolean|Number|Object|String)|par
e(Expr|Match|Select)|find|select)\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    },
    // ### JDOM 1
    // http://www.jdom.org/docs/apidocs.1.1/
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.jdom\\.xpath\\.XPath\\.newInstance\\(java\\.lang\\.String.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.jdom\\.xpath\\.XPath\\.(selectNodes|selectSingleNode)\\(.*"
        },
        "param_index" : 2
      }
    },
```

原文链接：coverity 部分规则提取

```
// ### JDOM 2
// http://www.jdom.org/docs/apidocs/index.html
{
  "sink_for_checker" : "XPATH_INJECTION",
  "sink" : {
    "methods" : {
      "overrides" : {
        // subclassed
        "matching" : "org\\.jdom2\\.xpath\\.XPath\\.newInstance\\(java\\.lang\\.String.*"
      }
    },
    "param_index" : 1
  }
},
{
  "sink_for_checker" : "XPATH_INJECTION",
  "sink" : {
    "methods" : {
      "overrides" : {
        // subclassed
        "matching" : "org\\.jdom2\\.xpath\\.XPath\\.(selectNodes|selectSingleNode)\\(.*"
      }
    },
    "param_index" : 2
  }
},
{
  "sink_for_checker" : "XPATH_INJECTION",
  "sink" : {
    "methods" : {
      "overrides" : {
        // subclassed
        "matching" : "org\\.jdom2\\.xpath\\.XPathFactory\\.compile\\(.*"
      }
    },
    "param_index" : 1
  }
},
{
  "sink_for_checker" : "XPATH_INJECTION",
  "sink" : {
    "methods" : {
      "matching" : "org\\.jdom2\\.xpath\\.XPathBuilder\\.<init>\\(.*"
    },
    "param_index" : 1
  }
},
// ### XOM
// http://www.xom.nu/apidocs/index.html?nu/xom/Nodes.html
{
  "sink_for_checker" : "XPATH_INJECTION",
  "sink" : {
    "methods" : {
      "overrides" : {
```

```
            // subclassed
            "matching" : "nu\\.xom\\.Node\\.query\\(java\\.lang\\.String.*"
          }
        },
        "param_index" : 1
      }
    },
    // ### Xindice (Apache attic, not active since 2011)
    // http://xml.apache.org/xindice/1.1/api/index.html
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "org\\.apache\\.xindice\\.core\\.query\\.XPathQueryResolver\\.(compileQ
ery|query)\\(.*"
        },
        "param_index" : 2
      }
    },
    // ### VTD-XML
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "com\\.ximpleware\\.AutoPilot(Huge)?\\.selectXPath\\(.*"
        },
        "param_index" : 1
      }
    }
  ]
},
  "type" : "Coverity analysis configuration",
  "format_version" : 4,
  "language" : "C#",
  "directives" : [
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Web\\.UI\\.XPathBinder::(Eval|Select)\\(.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "overrides" : {
            "matching" : "System\\.Web\\.UI\\.PageTheme::(XPath|Eval|XPathSelect)\\(.*"
          }
        },
        "param_index" : 1
      }
```

原文链接：coverity 部分规则提取

```
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Xml\\.XPath\\.Extensions::(XPathEvaluate|XPathSelectElement|
PathSelectElements)\\(.*"
        },
        "param_index" : 2
      }
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Xml\\.XPath\\.XPathNavigator::(Compile|Evaluate|SelectSingle
ode|Select)\\(.*"
        },
        "param_index" : 1
      }
    },
    {
      "sink_for_checker" : "XPATH_INJECTION",
      "sink" : {
        "methods" : {
          "matching" : "System\\.Xml\\.XPath\\.XPathExpression::(Compile)\\(.*"
        },
        "param_index" : 1
      }
    }
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
 * Revision History
 * May 2017 - Bug 103331:
 *     Inital support for SAP's HANA XS classic (XSC).
 * 7/28/2017 - Updated directive layout (Bug 105495)
////////////////////////////////////////////////////////////////////////////////
// Custom Dataflow Checker specification for XSS_BUDA.
  "type" : "Coverity analysis configuration",
  "format_version" : 10,
  "language" : "any",
  "directives" : [
    // 0) Custom Dataflow Checker specification for XSS_BUDA.
--
    "sink_for_checker" : "XSS",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "end" } ],
          "read_from_object_of_type" : "http.ServerResponse"
        }
      },
    }
```

```
},
// sink: [type http.ServerResponse].end([arg1.*] [, encoding])
{
  "sink_for_checker" : "XSS",
  "sink" : {
     "input" : "arg1",
     "to_callsite" : {
        "call_on" : {
           "read" : [ { "property" : "end" } ],
           "read_from_object_of_type" : "http.ServerResponse"
        },
     },
     "path" : [ { "any_property" : true } ]
  }
},
// sink: [type http.ServerResponse].write(arg1[, encoding][, callback])
{
  "sink_for_checker" : "XSS",
  "sink" : {
     "input" : "arg1",
     "to_callsite" : {
        "call_on" : {
           "read" : [ { "property" : "write" } ],
           "read_from_object_of_type" : "http.ServerResponse"
        }
     },
  }
},
// sink: [type http.ServerResponse].write(arg1.*[, encoding][, callback])
{
  "sink_for_checker" : "XSS",
  "sink" : {
     "input" : "arg1",
     "to_callsite" : {
        "call_on" : {
           "read" : [ { "property" : "write" } ],
           "read_from_object_of_type" : "http.ServerResponse"
        }
     },
     "path" : [ { "any_property" : true } ]
  }
},
// sink: [type https.ServerResponse].end([arg1][, encoding][, callback])
{
  "sink_for_checker" : "XSS",
  "sink" : {
     "input" : "arg1",
     "to_callsite" : {
        "call_on" : {
           "read" : [ { "property" : "end" } ],
           "read_from_object_of_type" : "https.ServerResponse"
        }
     },
  }
```

原文链接：coverity 部分规则提取

```
},
// sink: [type https.ServerResponse].end([arg1.*] [, encoding])
{
  "sink_for_checker" : "XSS",
  "sink" : {
     "input" : "arg1",
     "to_callsite" : {
        "call_on" : {
           "read" : [ { "property" : "end" } ],
           "read_from_object_of_type" : "https.ServerResponse"
        },
     },
     "path" : [ { "any_property" : true } ]
  }
},
// sink: [type https.ServerResponse].write(arg1[, encoding][, callback])
{
  "sink_for_checker" : "XSS",
  "sink" : {
     "input" : "arg1",
     "to_callsite" : {
        "call_on" : {
           "read" : [ { "property" : "write" } ],
           "read_from_object_of_type" : "https.ServerResponse"
        }
     },
  }
},
// sink: [type https.ServerResponse].write(arg1.*[, encoding][, callback])
{
  "sink_for_checker" : "XSS",
  "sink" : {
     "input" : "arg1",
     "to_callsite" : {
        "call_on" : {
           "read" : [ { "property" : "write" } ],
           "read_from_object_of_type" : "https.ServerResponse"
        }
     },
     "path" : [ { "any_property" : true } ]
  }
},
// The ExpressResponse object represents the HTTP response that an Express app sends
// when it gets an HTTP request.
// A few of its methods are sinks for XSS defects.
// Note: ExpressResponse is a type name that we use to refers to the
// Response object in Express API.
// sink: [type ExpressResponse].send([arg1])
{
  "sink_for_checker" : "XSS",
  "sink" : {
     "input" : "arg1",
     "to_callsite" : {
        "call_on" : {
```

```
            "read" : [ { "property" : "send" } ],
            "read_from_object_of_type" : "ExpressResponse"
          },
          "when" : {
            "only_if_arg_index" : 1,
            "is_max_index" : true
          }
        },
    }
},
// sink: [type ExpressResponse].send([arg1.*])
{
  "sink_for_checker" : "XSS",
  "sink" : {
      "input" : "arg1",
      "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "send" } ],
            "read_from_object_of_type" : "ExpressResponse"
          },
          "when" : {
            "only_if_arg_index" : 1,
            "is_max_index" : true
          }
      },
      "path" : [ { "any_property" : true } ]
  }
},
// sink: [type ExpressResponse].end([arg1] [, encoding])
{
  "sink_for_checker" : "XSS",
  "sink" : {
      "input" : "arg1",
      "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "end" } ],
            "read_from_object_of_type" : "ExpressResponse"
          },
      },
  }
},
// sink: [type ExpressResponse].end([arg1.*] [, encoding])
{
  "sink_for_checker" : "XSS",
  "sink" : {
      "input" : "arg1",
      "to_callsite" : {
          "call_on" : {
            "read" : [ { "property" : "end" } ],
            "read_from_object_of_type" : "ExpressResponse"
          },
      },
      "path" : [ { "any_property" : true } ]
  }
```

```
  },
  // sink: [type ExpressResponse].write(arg1[, encoding][, callback])
  {
    "sink_for_checker" : "XSS",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "write" } ],
          "read_from_object_of_type" : "ExpressResponse"
        }
      },
    }
  },
  // sink: [type ExpressResponse].write(arg1.*[, encoding][, callback])
  {
    "sink_for_checker" : "XSS",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on" : {
          "read" : [ { "property" : "write" } ],
          "read_from_object_of_type" : "ExpressResponse"
        }
      },
      "path" : [ { "any_property" : true } ]
    }
  },
  // The following methods of ServerResponse and ExpressResponse classes are not sinks (B
g 89597).
  // http.ServerResponse.addTrailers
  // http.ServerResponse.removeHeader
  // http.ServerResponse.setHeader(*)
  // http.ServerResponse.writeHead
  // https.ServerResponse.addTrailers
  // https.ServerResponse.removeHeader
  // https.ServerResponse.setHeader(*)
  // https.ServerResponse.writeHead
--
    "sink_for_checker" : "XSS",
    "sink" : {
     "input" : "arg1",
     "to_callsite" : {
      "call_on" : {
       "read" : [ { "property" : "setBody" } ],
       "read_from_object_of_type" : "SAPHanaWebResponse"
      },
     }
    }
  },
  // sink: [type SAPHanaWebResponse].setBody(arg1.*)
  {
    "sink_for_checker" : "XSS",
    "sink" : {
```

```
      "input" : "arg1",
      "to_callsite" : {
       "call_on" : {
        "read" : [ { "property" : "setBody" } ],
        "read_from_object_of_type" : "SAPHanaWebResponse"
       },
      },
      "path" : [ { "any_property" : true } ]
    }
   },
////////////////////////////////////////////////////////////////////////////
// End of directive list
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 7,
  "language" : "Java",
  "directives" : [
   //
   // # Regex injection checker
--
    "sink_for_checker" : "BSON_INJECTION",
    "sink" : {
     "methods" : {
      "overrides" : {
       "matching" : "org\\.bson\\.BSONDecoder\\.decode\\(.*"
      }
     },
     "param_index" : 1
    }
   },
   {
    "sink_for_checker" : "BSON_INJECTION",
    "sink" : {
     "methods" : {
      "matching" : "org\\.bson\\.(BSONDecoder|BasicBSONDecoder|DefaultDBDecoder|LazyB
ONDecoder|LazyDBDecoder|LazyWriteableDBDecoder|NewBSONDecoder)\\.decode\\(.*"
     },
     "param_index" : 1
    }
   },
   //
   // # JSON injection checker
   // Libraries listed on json.org. Modeled only
   // the most important ones (e.g., maintained ones)
   //
   // CWE ID: 20 (default injection)
   {
    "dataflow_checker_name" : "JSON_INJECTION",
    "languages" : {
     "Java" : "Webapp-Security-Explicit"
    },
    "taint_kinds" : [
     "servlet",
--
```

原文链接：coverity 部分规则提取

```
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.json\\.(JSONArray|JSONTokener)\\.<init>\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  // ### org.json.me
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.json\\.me\\.(JSONArray|JSONTokener)\\.<init>\\(java\\.lang\\.String
*"
      },
      "param_index" : 1
    }
  },
  // ### Jackson JSON Processor
  // Note that there are several implementations, hence
  // the different package names:
  //  - com.fasterxml: Jackson 2.x
  //  - org.jcodehaus: Jackson 1.x
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "(com\\.fasterxml|org\\.codehaus)\\.jackson\\.core\\.JsonFactory\\.createJs
nParser\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "(com\\.fasterxml|org\\.codehaus)\\.jackson\\.map\\.(ObjectMapper|Objec
Codec)\\.(readValue|readTree)\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "net\\.sf\\.json\\.groovy\\.JsonSlurper\\.parse\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  {
```

原文链接：coverity 部分规则提取

```
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "net\\.sf\\.json\\.util\\.JSONTokener\\.<init>\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "net\\.sf\\.json\\.JSONSerializer\\.toJSON\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "com\\.google\\.gson\\.JsonParser\\.parse\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "com\\.google\\.gson\\.JsonStreamParser\\.<init>\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  // ### JSON-io
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "com\\.cedarsoftware\\.util\\.io\\.JsonReader\\.(toJava|toMaps|jsonToJava|
sonToMaps)\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  // ### jjson
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "cde\\.grobmeier\\.jjson\\.convert\\.JSONDecoder\\.<init>\\(java\\.lang\\.
tring.*"
      },
```

```
      "param_index" : 1
    }
  },
  // ### jonij
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "cc\\.plural\\.jsonij\\.JSON(Parser)?\\.parse(Value)?\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "cc\\.plural\\.jsonij\\.StringJSONReader\\.<init>\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "cc\\.plural\\.jsonij\\.marshal\\.JSONDocumentMarshaler\\.marshalJSOND
cument\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  // ### json-simple
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.json\\.simple\\.JSONValue\\.parse(WithException)?\\(java\\.lang\\.St
ing.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.json\\.simple\\.parser\\.JSONParser\\.parse\\(java\\.lang\\.String.*"
      },
      "param_index" : 1
    }
  },
  // ### json-smart
  {
```

```
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
     "methods" : {
      "matching" : "net\\.minidev\\.json\\.JSONNavi\\.<init>\\(java\\.lang\\.String.*"
     },
     "param_index" : 1
    }
   },
   {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
     "methods" : {
      "matching" : "net\\.minidev\\.json\\.JSONValue\\.(parse(KeepingOrder|WithException|St
ict)?|compress|uncompress)\\(.*"
     },
     "param_index" : 1
    }
   },
   {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
     "methods" : {
      "matching" : "net\\.minidev\\.json\\.parser\\.JSONParser(Base|ByteArray|InputSteam|Re
der|String)?\\.parse\\(.*"
     },
     "param_index" : 1
    }
   },
   // ### MongoDB JSON
   {
    "sink_for_checker" : "JSON_INJECTION",
    "sink" : {
     "methods" : {
      "matching" : "com\\.mongodb\\.util\\.JSON\\.parse\\(java\\.lang\\.String.*"
     },
     "param_index" : 1
    }
   },
   //
   // # SOAP message manipulation checker
   //
   // CWE ID: 91
   {
    "dataflow_checker_name" : "SOAP_MESSAGE_MANIPULATION",
    "languages" : {
     "Java" : "Webapp-Security-Explicit"
    },
    "taint_kinds" : [
     "servlet",
     "network",
     "database"
--
    "sink_for_checker" : "SOAP_MESSAGE_MANIPULATION",
    "sink" : {
```

```
      "methods" : {
        "matching" : "javax\\.xml\\.soap\\.MessageFactory\\.createMessage\\(.*"
      },
      "param_index" : 2
    }
  },
  // ### Apache Axis 1 & 2
  {
    "sink_for_checker" : "SOAP_MESSAGE_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.apache\\.axis\\.soap\\.MessageFactoryImpl\\.createMessage\\(.*"
      },
      "param_index" : 2
    }
  },
  {
    "sink_for_checker" : "SOAP_MESSAGE_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.apache\\.axis2\\.saaj\\.MessageFactoryImpl\\.createMessage\\(.*"
      },
      "param_index" : 2
    }
  },
  // ### Spring Web Services
  {
    "sink_for_checker" : "SOAP_MESSAGE_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.springframework\\.ws\\.soap\\.SoapMessageFactory\\.createWebSe
viceMessage\\(.*"
      },
      "param_index" : 1
    }
  },
  {
    "sink_for_checker" : "SOAP_MESSAGE_MANIPULATION",
    "sink" : {
      "methods" : {
        "matching" : "org\\.springframework\\.ws\\.soap\\.(axiom\\.AxiomSoapMessageFactory
saaj\\.SaajSoapMessageFactory)\\.createWebServiceMessage\\(.*"
      },
      "param_index" : 1
    }
  }
  // TODO: need to inspect more frameworks for SOAP
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 4,
  "language" : "cpp",
  "directives" : [
    {
      "antecedent_checker" : "SECURE_CODING",
```

```
      "dc_checker_name" : "DC.WEAK_CRYPTO",
      "disposition" : "ED_SECURITY"
    },
    {
      "covlstr_defect_message" : "{CovLStrv2{{t{{0} should not be used for security related appl
cations, as linear congruential algorithms are too easy to break.}{\"\"}}}}",
--
    sink_for_checker : "UNCHECKED_ORIGIN",
    sink : {
       "write_to_object_of_type" : "StrictWindow",
       "write" : [ { "property" : "onmessage" } ]
    }
  },
  // sink: [type StrictWindow].addEventListener( "message", <source> )
  {
    sink_for_checker : "UNCHECKED_ORIGIN",
    sink : {
      "input": "arg2",
      "to_callsite": {
         "call_on": {
            "read_from_object_of_type" : "StrictWindow",
            "read" : [ { "property" : "addEventListener" } ]
         },
         "when" : {
            "only_if_arg_index" : 1,
            "equals_string" : "message"
         }
      }
    }
  },
  // sink: [type StrictWindow].attachEvent( "onmessage", <source> )
  {
    sink_for_checker : "UNCHECKED_ORIGIN",
    sink : {
      "input": "arg2",
      "to_callsite": {
         "call_on": {
            "read_from_object_of_type" : "StrictWindow",
            "read" : [ { "property" : "attachEvent" } ]
         },
         "when" : {
            "only_if_arg_index" : 1,
            "equals_string" : "onmessage"
         }
      }
    }
  },
  ////////////////////////////////////////////////////////////////////////
  // jQuery support
  // type: jQuery([type StrictWindow]) -> JQueryWindow
  // THIS IS A MODEL. NOT IMPLEMENTED YET.
  /*{
     "type" : "JQueryWindow",
--
```

```
    sink_for_checker : "UNCHECKED_ORIGIN",
    sink : {
      "input": "arg2",
      "to_callsite": {
        "call_on": {
          "read_from_object_of_type" : "JQuery",
          "read" : [ { "property" : "on" } ]
        },
        "when" : {
          "only_if_arg_index" : 1,
          "iregex_string" : "(.*[^a-zA-Z0-9])?(on)?message([^a-zA-Z0-9].*)?"
          // Explanation of regex above:
          // Requires "message" or "onmessage" at the begining of string
          // or non-alpha-numeric character before it.
          // Also, requires the end of string or non-alpha-numeric character
          // after that.
          // By allowing any non-alpha-numeric character, user code
          // may use arbitrary unicode spaces.
        }
      }
    }
--
    sink_for_checker : "UNCHECKED_ORIGIN",
    sink : {
      "input": "arg2",
      "to_callsite": {
        "call_on": {
          "read_from_object_of_type" : "JQuery",
          "read" : [ { "property" : "bind" } ]
        },
        "when" : {
          "only_if_arg_index" : 1,
          "iregex_string" : "(.*[^a-zA-Z0-9])?(on)?message([^a-zA-Z0-9].*)?"
          // See above for regex explanation.
        }
      }
    }
  }
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
// -----
// JavaScript sinks
  "type" : "Coverity analysis configuration",
  "format_version" : 8,
--
      sink_for_checker: "HARDCODED_CREDENTIALS",
      sink: {
        "input": "arg2",
        "to_callsite": {
          "call_on": {
            "read_from_object_of_type": "Module.crypto",
            "read": [{ "property": "createCipher" }]
          }
        }
      }
```

原文链接：coverity 部分规则提取

```
},
// crypto sink: crypto.createCipheriv(algorithm, key, iv)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "input": "arg2",
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type": "Module.crypto",
                "read": [{ "property": "createCipheriv" }]
            }
        }
    }
},
// crypto sink: crypto.createDecipher(algorithm, password)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "input": "arg2",
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type": "Module.crypto",
                "read": [{ "property": "createDecipher" }]
            }
        }
    }
},
// crypto sink: crypto.createDecipheriv(algorithm, key, iv)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "input": "arg2",
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type": "Module.crypto",
                "read": [{ "property": "createDecipheriv" }]
            }
        }
    }
},
// crypto sink: crypto.createHmac(algorithm, key)
{
    sink_for_checker : "HARDCODED_CREDENTIALS",
    sink : {
        "input": "arg2",
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.crypto",
                "read" : [ { "property" : "createHmac" } ]
            }
        }
    }
},
// crypto sink: crypto.pbkdf2(password, salt, iterations, keylen, digest, callback)
```

```
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "input": "arg1",
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type": "Module.crypto",
                "read": [{ "property": "pbkdf2" }]
            }
        }
    }
},
// crypto sink: crypto.pbkdf2Sync(password, salt, iterations, keylen, digest)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "input": "arg1",
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type": "Module.crypto",
                "read": [{ "property": "pbkdf2Sync" }]
            }
        }
    }
},
// crypto sink: crypto.privateDecrypt(private_key, buffer)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "input": "arg1",
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type": "Module.crypto",
                "read": [{ "property": "privateDecrypt" }]
            }
        }
    }
},
// crypto sink: crypto.privateEncrypt(private_key, buffer)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "input": "arg1",
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type": "Module.crypto",
                "read": [{ "property": "privateEncrypt" }]
            }
        }
    }
},
//--------------------------
//  http
//--------------------------
```

```
// password sink: http.request(options[, callback])
// options may include: {'auth': 'user:password', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.http",
                "read": [ { "property": "request" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "auth" } ]
    }
},
// password sink: http.request(options[, callback])
// options may include: {'headers': {'authorization': auth, ...}, ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.http",
                "read": [ { "property": "request" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "headers" }, { "property": "authorization" } ]
    }
},
// password sink: http.request(options[, callback])
// options may include: {'headers': {'proxy-authorization': auth, ...}, ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.http",
                "read": [ { "property": "request" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "headers" }, { "property": "proxy-authorization" } ]
    }
},
//--------------------------
//  https
//--------------------------
// password sink: https.request(options, callback)
// options may include: {'auth': 'user:password', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
```

```
            "call_on": {
                "read_from_object_of_type" : "Module.https",
                "read": [ { "property": "request" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "auth" } ]
    }
},
// password sink: https.request(options, callback)
// options may include: {'headers': {'authorization': auth, ...}, ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.https",
                "read": [ { "property": "request" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "headers" }, { "property": "authorization" } ]
    }
},
// password sink: https.request(options, callback)
// options may include: {'headers': {'proxy-authorization': auth, ...}, ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.https",
                "read": [ { "property": "request" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "headers" }, { "property": "proxy-authorization" } ]
    }
},
// password sink: https.request(options, callback)
// options may include: {'pfx': 'certificate, private key and CA certificates', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.https",
                "read": [ { "property": "request" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "pfx" } ]
    }
},
```

```
    // password sink: https.request(options, callback)
    // options may include: {'key': 'private-key', ...}
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_object_of_type" : "Module.https",
                    "read": [ { "property": "request" } ]
                },
            },
            "input": "arg1",
            "path": [ { "property": "key" } ]
        }
    },
    // password sink: https.request(options, callback)
    // options may include: {'passphrase': 'string of passphrase', ...}
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_object_of_type" : "Module.https",
                    "read": [ { "property": "request" } ]
                },
            },
            "input": "arg1",
            "path": [ { "property": "passphrase" } ]
        }
    },
    // password sink: https.request(options, callback)
    // options may include: {'ca': 'a string, buffer or array of strings or buffers of trusted certific
tes', ...}
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_object_of_type" : "Module.https",
                    "read": [ { "property": "request" } ]
                },
            },
            "input": "arg1",
            "path": [ { "property": "ca" } ]
        }
    },
    //---------------------------
    //  tls
    //---------------------------
    // password sink: tls.connect(options[, callback])
    // options may include: {'pfx': 'a string or buffer containing private key and certificate', ...}
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
```

```
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "pfx" } ]
    }
},
// password sink: tls.connect(path[, options, callback])
// options may include: {'pfx': 'a string or buffer containing private key and certificate', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg2",
        "path": [ { "property": "pfx" } ]
    }
},
// password sink: tls.connect(port[, host][, options][, callback])
// options may include: {'pfx': 'a string or buffer containing private key and certificate', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg3",
        "path": [ { "property": "pfx" } ]
    }
},
// password sink: tls.connect(options[, callback])
// options may include: {'key': 'a string or buffer containing private key', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "key" } ]
    }
```

```
},
// password sink: tls.connect(path[, options][, callback])
// options may include: {'key': 'a string or buffer containing private key', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg2",
        "path": [ { "property": "key" } ]
    }
},
// password sink: tls.connect(port[, host][, options][, callback])
// options may include: {'key': 'a string or buffer containing private key', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg3",
        "path": [ { "property": "key" } ]
    }
},
// password sink: tls.connect(options[, callback])
// options may include: {'passphrase': 'a string of passphrase for the private key or pfx', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "passphrase" } ]
    }
},
// password sink: tls.connect(path[, options][, callback])
// options may include: {'passphrase': 'a string of passphrase for the private key or pfx', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
```

```
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg2",
        "path": [ { "property": "passphrase" } ]
    }
},
// password sink: tls.connect(port[, host][, options][, callback])
// options may include: {'passphrase': 'a string of passphrase for the private key or pfx', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg3",
        "path": [ { "property": "passphrase" } ]
    }
},
// password sink: tls.connect(options[, callback])
// options may include: {'cert': 'a string or buffer containing the certificate key', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "cert" } ]
    }
},
// password sink: tls.connect(path[, options][, callback])
// options may include: {'cert': 'a string or buffer containing the certificate key', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg2",
        "path": [ { "property": "cert" } ]
    }
},
// password sink: tls.connect(port[, host][, options][, callback])
// options may include: {'cert': 'a string or buffer containing the certificate key', ...}
```

```
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg3",
        "path": [ { "property": "cert" } ]
    }
},
// password sink: tls.connect(options[, callback])
// options may include: {'ca': 'a string or list of strings of trusted certificates', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "ca" } ]
    }
},
// password sink: tls.connect(path[, options][, callback])
// options may include: {'ca': 'a string or list of strings of trusted certificates', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg2",
        "path": [ { "property": "ca" } ]
    }
},
// password sink: tls.connect(port[, host][, options][, callback])
// options may include: {'ca': 'a string or list of strings of trusted certificates', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "connect" } ]
            },
        },
```

```
        "input": "arg3",
        "path": [ { "property": "ca" } ]
    }
},
// password sink: tls.createServer([options][, secureConnectionListener])
// options may include: {'pfx': 'a string or buffer containing private key and certificate', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "createServer" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "pfx" } ]
    }
},
// password sink: tls.createServer([options][, secureConnectionListener])
// options may include: {'key': 'a string or buffer containing private key', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "createServer" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "key" } ]
    }
},
// password sink: tls.createServer([options][, secureConnectionListener])
// options may include: {'passphrase': 'a string of passphrase for the private key or pfx', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "createServer" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "passphrase" } ]
    }
},
// password sink: tls.createServer([options][, secureConnectionListener])
// options may include: {'cert': 'a string or buffer containing the certificate key', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
```

```
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "createServer" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "cert" } ]
    }
},
// password sink: tls.createServer([options][, secureConnectionListener])
// options may include: {'ca': 'a string or list of strings of trusted certificates', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "createServer" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "ca" } ]
    }
},
// password sink: tls.createSecureContext(options)
// options may include: {'pfx': 'a string or buffer containing private key and certificate', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "createSecureContext" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "pfx" } ]
    }
},
// password sink: tls.createSecureContext(options)
// options may include: {'key': 'a string or buffer containing private key', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.tls",
                "read": [ { "property": "createSecureContext" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "key" } ]
    }
```

原文链接：coverity 部分规则提取

```
    },
    // password sink: tls.createSecureContext(options)
    // options may include: {'passphrase': 'a string of passphrase for the private key or pfx', ...}
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_object_of_type" : "Module.tls",
                    "read": [ { "property": "createSecureContext" } ]
                },
            },
            "input": "arg1",
            "path": [ { "property": "passphrase" } ]
        }
    },
    // password sink: tls.createSecureContext(options)
    // options may include: {'cert': 'a string or buffer containing the certificate key', ...}
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_object_of_type" : "Module.tls",
                    "read": [ { "property": "createSecureContext" } ]
                },
            },
            "input": "arg1",
            "path": [ { "property": "cert" } ]
        }
    },
    // password sink: tls.createSecureContext(options)
    // options may include: {'ca': 'a string or list of strings of trusted certificates', ...}
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_object_of_type" : "Module.tls",
                    "read": [ { "property": "createSecureContext" } ]
                },
            },
            "input": "arg1",
            "path": [ { "property": "ca" } ]
        }
    },
    //----------------------------
    //  mongoose (database module)
    //----------------------------
    // password sink: mongoose.createConnection([uri], [options], [options.config], [options.co
fig.autoIndex])
    // uri may include password, such as mongoose.createConnection('mongodb://[username:
assword@]host[:port][[/database][?options]]');
    {
```

```
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_object_of_type" : "Module.mongoose",
                    "read": [ { "property": "createConnection" } ]
                },
                "when" : {
                    "only_if_arg_index" : 1,
                    "regex_string" : "^mongodb://.*:.*@.*"
                }
            },
            "input": "arg1"
        }
    },
    // password sink: mongoose.createConnection([uri], [options], [options.config], [options.co
fig.autoIndex])
    // options may include: {'pass': 'password for authentication', ...}
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_object_of_type" : "Module.mongoose",
                    "read": [ { "property": "createConnection" } ]
                },
            },
            "input": "arg2",
            "path": [ { "property": "pass" } ]
        }
    },
    // password sink: mongoose.createConnection(connection_string, database, port, options)
    // options may include: {'pass': 'password for authentication', ...}
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_object_of_type" : "Module.mongoose",
                    "read": [ { "property": "createConnection" } ]
                },
            },
            "input": "arg4",
            "path": [ { "property": "pass" } ]
        }
    },
    // password sink: mongoose.connect(uri(s), [options], [callback])
    // uri(s) may include password, such as mongoose.connect('mongodb://[username:passwo
d@]host[:port][[/database][?options]]');
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
```

```
                "read_from_object_of_type" : "Module.mongoose",
                "read": [ { "property": "connect" } ]
            },
            "when" : {
                "only_if_arg_index" : 1,
                "regex_string" : "^mongodb://.*:.*@.*"
            }
        },
        "input": "arg1"
    }
  },
  // password sink: mongoose.connect(uri(s), [options], [callback])
  // options may include password: {'pass': 'password for authentication (if not specified in ur
)', ...}
  {
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
      "to_callsite": {
        "call_on": {
          "read_from_object_of_type" : "Module.mongoose",
          "read": [ { "property": "connect" } ]
        },
      },
      "input": "arg2",
      "path": [ { "property": "pass" } ]
    }
  },
  // password sink: [type Module.mongoose.Type.Connection].open(connection_string, [data
ase], [port], [options], [callback])
  // options may include: {'pass': 'password for authentication', ...}
  // when [database] and [port] are not provided, then [options] will be the second argument
  {
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
      "to_callsite": {
        "call_on": {
          "read_from_object_of_type" : "Module.mongoose.Type.Connection",
          "read": [ { "property": "open" } ]
        },
      },
      "input": "arg2",
      "path": [ { "property": "pass" } ]
    }
  },
  // password sink: [type Module.mongoose.Type.Connection].open(connection_string, [data
ase], [port], [options], [callback])
  // options may include: {'pass': 'password for authentication', ...}
  // when either [database] and [port] is provided, then [options] will be the third argument
  {
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
      "to_callsite": {
        "call_on": {
          "read_from_object_of_type" : "Module.mongoose.Type.Connection",
```

```
                "read": [ { "property": "open" } ]
            },
        },
        "input": "arg3",
        "path": [ { "property": "pass" } ]
    }
},
// password sink: [type Module.mongoose.Type.Connection].open(connection_string, [data
ase], [port], [options], [callback])
// options may include: {'pass': 'password for authentication', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.mongoose.Type.Connection",
                "read": [ { "property": "open" } ]
            },
        },
        "input": "arg4",
        "path": [ { "property": "pass" } ]
    }
},
// password sink: [type Module.mongoose.Type.Connection].openSet(uris, [database], [opti
ns], [callback])
// uris may include password, such as openSet('mongodb://[username:password@]host[:po
t][[/database][?options]]');
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.mongoose.Type.Connection",
                "read": [ { "property": "openSet" } ]
            },
            "when" : {
                "only_if_arg_index" : 1,
                "regex_string" : "^mongodb://.*:.*@.*"
            }
        },
        "input": "arg1"
    }
},
// password sink: [type Module.mongoose.Type.Connection].openSet(uris, [database], [opti
ns], [callback])
// options may include: {'pass': 'password for authentication', ...}
// when [database] is not provided, then [options] will be the second argument
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.mongoose.Type.Connection",
                "read": [ { "property": "openSet" } ]
```

```
            },
        },
        "input": "arg2",
        "path": [ { "property": "pass" } ]
    }
},
// password sink: [type Module.mongoose.Type.Connection].openSet(uris, [database], [opti
ns], [callback])
// options may include: {'pass': 'password for authentication', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "Module.mongoose.Type.Connection",
                "read": [ { "property": "openSet" } ]
            },
        },
        "input": "arg3",
        "path": [ { "property": "pass" } ]
    }
},
//----------------------------------
//  Node.js MongoDB (database module)
//----------------------------------
// password sink: [type MongoClientClass].connect(url, options, callback)
// url may include password: connect('mongodb://[username:password@]host[:port][[/data
ase][?options]]');
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "MongoClientClass",
                "read": [ { "property": "connect" } ]
            },
            "when" : {
                "only_if_arg_index" : 1,
                "regex_string" : "^mongodb://.*:.*@.*"
            }
        },
        "input": "arg1"
    }
},
// password sink: [type MongoClientClass].connect(url, options, callback)
// options: sslPass (SSL Certificate pass phrase)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "MongoClientClass",
                "read": [ { "property": "connect" } ]
            },
```

```
        },
        "input": "arg2",
        "path": [ { "property": "sslPass" } ]
    }
},
// password sink: [type MongoClientClass].connect(url, options, callback)
// options: sslKey (SSL private key)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "MongoClientClass",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg2",
        "path": [ { "property": "sslKey" } ]
    }
},
// password sink: [type MongoClientClass].connect(url, options, callback)
// options: sslCert (SSL certificate)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "MongoClientClass",
                "read": [ { "property": "connect" } ]
            },
        },
        "input": "arg2",
        "path": [ { "property": "sslCert" } ]
    }
},
// password sink: [type MongoDb].authenticate(username, password, options, callback)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_object_of_type" : "MongoDb",
                "read": [ { "property": "authenticate" } ]
            },
        },
        "input": "arg2"
    }
},
// password sink: [type MongoDb].addUser(username, password, options, callback)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
```

```
            "read_from_object_of_type" : "MongoDb",
            "read": [ { "property": "addUser" } ]
        },
    },
    "input": "arg2"
  }
},
// password sink: new require('mongodb').Mongos(servers, options)
// options:
//  - sslKey
//  - sslPass
{
  sink_for_checker: "HARDCODED_CREDENTIALS",
  sink: {
    "to_callsite" : {
      "new_on" : {
        "read_from_js_require" : "mongodb",
        "path" : [ { "property" : "Mongos" } ]
      }
    },
    "input": "last_arg",
    "path": [ { "property": "sslKey" } ]
  }
},
// password sink: new require('mongodb').Mongos(servers, options)
// options:
//  - sslKey
//  - sslPass
{
  sink_for_checker: "HARDCODED_CREDENTIALS",
  sink: {
    "to_callsite" : {
      "new_on" : {
        "read_from_js_require" : "mongodb",
        "path" : [ { "property" : "Mongos" } ]
      }
    },
    "input": "last_arg",
    "path": [ { "property": "sslPass" } ]
  }
},
// password sink: new require('mongodb').ReplSetServers(servers, options)
// options:
//  - sslKey
//  - sslPass
{
  sink_for_checker: "HARDCODED_CREDENTIALS",
  sink: {
    "to_callsite" : {
      "new_on" : {
        "read_from_js_require" : "mongodb",
        "path" : [ { "property" : "ReplSetServers" } ]
      }
    },
```

```
        "input": "last_arg",
        "path": [ { "property": "sslKey" } ]
    }
},
// password sink: new require('mongodb').ReplSetServers(servers, options)
// options:
//  - sslKey
//  - sslPass
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite" : {
            "new_on" : {
                "read_from_js_require" : "mongodb",
                "path" : [ { "property" : "ReplSetServers" } ]
            }
        },
        "input": "last_arg",
        "path": [ { "property": "sslPass" } ]
    }
},
// password sink: new require('mongodb').Server(host, port, options)
// options:
//  - sslKey
//  - sslPass
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite" : {
            "new_on" : {
                "read_from_js_require" : "mongodb",
                "path" : [ { "property" : "Server" } ]
            }
        },
        "input": "last_arg",
        "path": [ { "property": "sslKey" } ]
    }
},
// password sink: new require('mongodb').Server(host, port, options)
// options:
//  - sslKey
//  - sslPass
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite" : {
            "new_on" : {
                "read_from_js_require" : "mongodb",
                "path" : [ { "property" : "Server" } ]
            }
        },
        "input": "last_arg",
        "path": [ { "property": "sslPass" } ]
    }
```

```
    },
    //----------------------------
    //  Sequelize (database module)
    //----------------------------
    // password sink: new require('sequelize')(database, [username=null], [password=null], [opt
ons={}])
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite" : {
                "new_on" : {
                    "read_from_js_require" : "sequelize"
                }
            },
            "input": "arg3"
        }
    },
    //----------------------------
    //  MySql (database module)
    //----------------------------
    // password sink: require('mysql').createConnection(options)
    // options:
    //  - password
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_js_require" : "mysql",
                    "path" : [ { "property" : "createConnection" } ]
                }
            },
            "input": "arg1",
            "path": [ { "property": "password" } ]
        }
    },
    // password sink: require('mysql').createPool(options)
    // options:
    //  - password
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_js_require" : "mysql",
                    "path" : [ { "property" : "createPool" } ]
                }
            },
            "input": "arg1",
            "path": [ { "property": "password" } ]
        }
    },
    //----------------------------
    //  Knex.js (database module)
```

```
//--------------------------
// password sink: require('knex')(options)
// options:
//  - connection: {password : ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_js_require" : "knex"
            },
        },
        "input": "arg1",
        "path": [ { "property": "password" } ]
    }
},
//---------------------------------
//  Passport (authentication module)
//---------------------------------
// password sink: new require('passport-facebook').Strategy(options)
// options:
//  - clientSecret: FACEBOOK_APP_SECRET
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite" : {
            "new_on" : {
                "read_from_js_require" : "passport-facebook",
                "path" : [ { "property" : "Strategy" } ]
            }
        },
        "input": "arg1",
        "path": [ { "property": "clientSecret" } ]
    }
},
// password sink: new require('passport-twitter').Strategy(options)
// options:
//  - consumerKey: TWITTER_CONSUMER_KEY
//  - consumerSecret: TWITTER_CONSUMER_SECRET
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite" : {
            "new_on" : {
                "read_from_js_require" : "passport-twitter",
                "path" : [ { "property" : "Strategy" } ]
            }
        },
        "input": "arg1",
        "path": [ { "property": "consumerKey" } ]
    }
},
// password sink: new require('passport-twitter').Strategy(options)
// options:
```

```
//  - consumerKey: TWITTER_CONSUMER_KEY
//  - consumerSecret: TWITTER_CONSUMER_SECRET
{
   sink_for_checker: "HARDCODED_CREDENTIALS",
   sink: {
      "to_callsite" : {
         "new_on" : {
            "read_from_js_require" : "passport-twitter",
            "path" : [ { "property" : "Strategy" } ]
         }
      },
      "input": "arg1",
      "path": [ { "property": "consumerSecret" } ]
   }
},
// password sink: new require('passport-google-oauth').OAuthStrategy(options)
// options:
//  - consumerKey   : GOOGLE_CONSUMER_KEY
//  - consumerSecret: GOOGLE_CONSUMER_SECRET (passport-google-oauth1)
{
   sink_for_checker: "HARDCODED_CREDENTIALS",
   sink: {
      "to_callsite" : {
         "new_on" : {
            "read_from_js_require" : "passport-google-oauth",
            "path" : [ { "property" : "OAuthStrategy" } ]
         }
      },
      "input": "arg1",
      "path": [ { "property": "consumerKey" } ]
   }
},
// password sink: new require('passport-google-oauth').OAuthStrategy(options)
// options:
//  - consumerKey   : GOOGLE_CONSUMER_KEY
//  - consumerSecret: GOOGLE_CONSUMER_SECRET (passport-google-oauth1)
{
   sink_for_checker: "HARDCODED_CREDENTIALS",
   sink: {
      "to_callsite" : {
         "new_on" : {
            "read_from_js_require" : "passport-google-oauth",
            "path" : [ { "property" : "OAuthStrategy" } ]
         }
      },
      "input": "arg1",
      "path": [ { "property": "consumerSecret" } ]
   }
},
// password sink: new require('passport-google-oauth20').Strategy(options)
// options:
//  - clientSecret: GOOGLE_CLIENT_SECRET (passport-google-oauth20)
{
   sink_for_checker: "HARDCODED_CREDENTIALS",
```

```
    sink: {
        "to_callsite" : {
            "new_on" : {
                "read_from_js_require" : "passport-google-oauth20",
                "path" : [ { "property" : "Strategy" } ]
            }
        },
        "input": "arg1",
        "path": [ { "property": "clientSecret" } ]
    }
},
//-------------------------------------
//  sink models from Codiscope guidance
//-------------------------------------
// password sink: require('express-session')({'secret': ...})
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_js_require" : "express-session"
            },
        },
        "input": "arg1",
        "path": [ { "property": "secret" } ]
    }
},
// password sink: require('cookie-session')({'secret': ...})
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_js_require" : "cookie-session"
            },
        },
        "input": "arg1",
        "path": [ { "property": "secret" } ]
    }
},
// password sink: require('client-sessions')({'secret': ...})
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_js_require" : "client-sessions"
            },
        },
        "input": "arg1",
        "path": [ { "property": "secret" } ]
    }
},
// password sink: require('client-sessions')({'encryptionKey': new Buffer(key_string)})
```

```
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_js_require" : "client-sessions"
                },
            },
            "input": "arg1",
            "path": [ { "property": "encryptionKey" } ]
        }
    },
    // password sink: require('client-sessions')({'signatureKey': new Buffer(key_string)})
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on": {
                    "read_from_js_require" : "client-sessions"
                },
            },
            "input": "arg1",
            "path": [ { "property": "signatureKey" } ]
        }
    },
    // password sink: [type ExpressRequest].file('file-name').upload(config, ...)
    // config options:
    // -key
    {
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on" : {
                    "from_callsite" : {
                        "call_on": {
                            "read_from_object_of_type" : "ExpressRequest",
                            "read" : [ { "property": "file" } ]
                        }
                    },
                    "output" : "return",
                    "path" : [ { "property": "upload" } ]
                },
            },
            "input": "arg1",
            "path": [ { "property": "key" } ]
        }
    },
    // password sink: [type ExpressRequest].file('file-name').upload(config, ...)
    // config options:
    // -secret
--
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
```

```
            "call_on" : {
               "from_callsite" : {
                  "call_on": {
                     "read_from_object_of_type" : "ExpressRequest",
                     "read" : [ { "property": "file" } ]
                  }
               },
               "output" : "return",
               "path" : [ { "property": "upload" } ]
            },
         },
         "input": "arg1",
         "path": [ { "property": "secret" } ]
      }
   },
   // password sink: [type ExpressRequest].file('file-name').upload(config, ...)
   // config options:
   // -password
--
      sink_for_checker: "HARDCODED_CREDENTIALS",
      sink: {
         "to_callsite": {
            "call_on" : {
               "from_callsite" : {
                  "call_on": {
                     "read_from_object_of_type" : "ExpressRequest",
                     "read" : [ { "property": "file" } ]
                  }
               },
               "output" : "return",
               "path" : [ { "property": "upload" } ]
            },
         },
         "input": "arg1",
         "path": [ { "property": "password" } ]
      }
   },
   // password sink: [type ExpressRequest].file('file-name').upload(config, ...)
   // config options:
   // -uri (it's an option of the skipper-gridfs filesystem adapter)
--
      sink_for_checker: "HARDCODED_CREDENTIALS",
      sink: {
         "to_callsite": {
            "call_on" : {
               "from_callsite" : {
                  "call_on": {
                     "read_from_object_of_type" : "ExpressRequest",
                     "read" : [ { "property": "file" } ]
                  }
               },
               "output" : "return",
               "path" : [ { "property": "upload" } ]
            },
```

```
            },
            "input": "arg1",
            "path": [ { "property": "uri" } ]
        }
    },
    // password sink: [type ExpressRequest].file('file-name').upload(config, ...)
    // config options:
    // - connection
--
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on" : {
                    "from_callsite" : {
                        "call_on": {
                            "read_from_object_of_type" : "ExpressRequest",
                            "read" : [ { "property": "file" } ]
                        }
                    },
                    "output" : "return",
                    "path" : [ { "property": "upload" } ]
                },
            },
            "input": "arg1",
            "path": [ { "property": "connection" } ]
        }
    },
    // password sink: [type ExpressRequest].file('file-name').upload(config, ...)
    // config options:
    // - connection
--
        sink_for_checker: "HARDCODED_CREDENTIALS",
        sink: {
            "to_callsite": {
                "call_on" : {
                    "from_callsite" : {
                        "call_on": {
                            "read_from_object_of_type" : "ExpressRequest",
                            "read" : [ { "property": "file" } ]
                        }
                    },
                    "output" : "return",
                    "path" : [ { "property": "upload" } ]
                },
            },
            "input": "arg1",
            "path": [ { "property": "connection" }, { "property": "password" } ]
        }
    },
    // password sink: [type ExpressRequest].file('file-name').upload(config, ...)
    // config options:
    // - credentials: {password: ...} (it's an option of the skipper-openstack/skipper-postgreql fil
system adapter)
--
```

```
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on" : {
                "from_callsite" : {
                    "call_on": {
                        "read_from_object_of_type" : "ExpressRequest",
                        "read" : [ { "property": "file" } ]
                    }
                },
                "output" : "return",
                "path" : [ { "property": "upload" } ]
            },
        },
        "input": "arg1",
        "path": [ { "property": "credentials" }, { "property": "password" } ]
    }
},
// password sink: require('skipper-gridfs')({'password': ...})
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite": {
            "call_on": {
                "read_from_js_require" : "skipper-gridfs"
            },
        },
        "input": "arg1",
        "path": [ { "property": "password" } ]
    }
},
//---------------------------
//  HANA XSC sinks.
//---------------------------
// crypto sink: [type SAPHanaSecurityCrypto].md5(data[, key])
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "SAPHanaSecurityCrypto",
                "read": [ { "property" : "md5" } ]
            },
            "when" : {
                "only_if_arg_index" : 2,
                "is_max_index" : true
            }
        }
    }
},
// crypto sink: [type SAPHanaSecurityCrypto].sha1(data[, key])
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
```

```
    sink: {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type": "SAPHanaSecurityCrypto",
                "read": [ { "property" : "sha1" } ]
            },
            "when" : {
                "only_if_arg_index" : 2,
                "is_max_index" : true
            }
        }
    }
  },
  // crypto sink: [type SAPHanaSecurityCrypto].sha256(data[, key])
  {
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "input" : "arg2",
        "to_callsite" : {
            "call_on" : {
                "read_from_object_of_type" : "SAPHanaSecurityCrypto",
                "read": [ { "property" : "sha256" } ]
            },
            "when" : {
                "only_if_arg_index" : 2,
                "is_max_index" : true
            }
        }
    }
  },
  //--------------------------
  //  HANA XSA sinks.
  //--------------------------
  //
  // node-hdb: require("hdb") -> [type SAPXSA_HDB]
  //
--
    "sink_for_checker" : "HARDCODED_CREDENTIALS",
    "sink" : {
      "to_callsite" : {
        "new_on" : {
          "read_from_object_of_type" : "SAPXSA_HDB",
          "read" : [ { "property" : "Client" } ]
        }
      },
      "input" : "arg1",
      "path"  : [ { "property" : "password" } ]
    }
  },
  // sink: new [type SAPXSA_HDB].Client(options)
  // through options 'key' field, e.g., {key: 'client-key', ...}
  // it's used to establish an encrypted database connection
  {
```

```
  "sink_for_checker" : "HARDCODED_CREDENTIALS",
  "sink" : {
    "to_callsite" : {
      "new_on" : {
        "read_from_object_of_type" : "SAPXSA_HDB",
        "read" : [ { "property" : "Client" } ]
      }
    },
    "input" : "arg1",
    "path"  : [ { "property" : "key" } ]
  }
},
// sink: [type SAPXSA_HDB].createClient(options)
// through options 'password' field, e.g., {user: 'user', password: 'secret', ...}
{
  "sink_for_checker" : "HARDCODED_CREDENTIALS",
  "sink" : {
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "createClient" } ],
        "read_from_object_of_type" : "SAPXSA_HDB"
      }
    },
    "input" : "arg1",
    "path"  : [ { "property" : "password" } ]
  }
},
// sink: [type SAPXSA_HDB].createClient(options)
// through options 'key' field, e.g., {key: 'client-key', ...}
// it's used to establish an encrypted database connection
{
  "sink_for_checker" : "HARDCODED_CREDENTIALS",
  "sink" : {
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "createClient" } ],
        "read_from_object_of_type" : "SAPXSA_HDB"
      }
    },
    "input" : "arg1",
    "path"  : [ { "property" : "key" } ]
  }
},
// sink: [type SAPXSA_HDB_Client].connect([options,] cb)
// through options 'password' field, e.g., {user: 'user', password: 'secret', ...}
// the user and password specified in the options will override the defaults of the client
{
  "sink_for_checker" : "HARDCODED_CREDENTIALS",
  "sink" : {
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "connect" } ],
        "read_from_object_of_type" : "SAPXSA_HDB_Client"
      }
```

```
    },
    "input" : "arg1",
    "path"  : [ { "property" : "password"  } ]
  }
},
// sink: [type SAPXSA_HDB_Client].connect([options,] cb)
// through options 'key' field, e.g., {key: 'client-key', ...}
// it's used to establish an encrypted database connection
{
  "sink_for_checker" : "HARDCODED_CREDENTIALS",
  "sink" : {
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "connect" } ],
        "read_from_object_of_type" : "SAPXSA_HDB_Client"
      }
    },
    "input" : "arg1",
    "path"  : [ { "property" : "key"  } ]
  }
},
//
// require('sap-hdbext') --> [type SAPXSA_HDBEXT]
//
// sink: [type SAPXSA_HDBEXT].createConnection(options, callback)
// through options 'password' field, e.g., {password: 'password', ...}
{
  "sink_for_checker" : "HARDCODED_CREDENTIALS",
  "sink" : {
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "createConnection" } ],
        "read_from_object_of_type" : "SAPXSA_HDBEXT"
      }
    },
    "input" : "arg1",
    "path"  : [ { "property" : "password"  } ]
  }
},
// sink: [type SAPXSA_HDBEXT].createPool(hanaService, poolConfig)
// through options 'password' field, e.g., {password: 'password', ...}
{
  "sink_for_checker" : "HARDCODED_CREDENTIALS",
  "sink" : {
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "createPool" } ],
        "read_from_object_of_type" : "SAPXSA_HDBEXT"
      }
    },
    "input" : "arg1",
    "path"  : [ { "property" : "password"  } ]
  }
},
```

```
//
// require('sap-hdb-connection') --> [type SAPXSA_HDB_CONNECTION]
//
// sink: [type SAPXSA_HDB_CONNECTION].createConnection(options, callback)
// through options 'password' field, e.g., {password: 'password', ...}
{
  "sink_for_checker" : "HARDCODED_CREDENTIALS",
  "sink" : {
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "createConnection" } ],
        "read_from_object_of_type" : "SAPXSA_HDB_CONNECTION"
      }
    },
    "input" : "arg1",
    "path"  : [ { "property" : "password"  } ]
  }
},
//
// require('sap-xb-messaging') --> [type SAPXSA_XB_MESSAGING]
//
// sink: [type SAPXSA_XB_MESSAGING].createClient(options)
// through options 'password' field, e.g., {password: 'password', ...}
{
  "sink_for_checker" : "HARDCODED_CREDENTIALS",
  "sink" : {
    "to_callsite" : {
      "call_on" : {
        "read" : [ { "property" : "createClient" } ],
        "read_from_object_of_type" : "SAPXSA_XB_MESSAGING"
      }
    },
    "input" : "arg1",
    "path"  : [ { "property" : "password"  } ]
  }
},
// sink: new [type SAPXSA_XB_MESSAGING].Client(options)
// through options 'password' field, e.g., {password: 'password', ...}
{
  "sink_for_checker" : "HARDCODED_CREDENTIALS",
  "sink" : {
    "to_callsite" : {
      "new_on" : {
        "read_from_object_of_type" : "SAPXSA_XB_MESSAGING",
        "read" : [ { "property" : "Client" } ]
      }
    },
    "input" : "arg1",
    "path"  : [ { "property" : "password" } ]
  }
},
//
// module sap-jobs-client
//
```

```
// sink: new require('sap-jobs-client').Scheduler(options)
// through options 'password' field, e.g., {password: 'password', ...}
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite" : {
            "new_on" : {
                "read_from_js_require" : "sap-jobs-client",
                "path" : [ { "property" : "Scheduler" } ]
            }
        },
        "input": "arg1",
        "path": [ { "property": "password" } ]
    }
},
//
// module sap-jobs-client
//
// sink: require('sap-xssec').createSecurityContext(token, config, cb)
{
    sink_for_checker: "HARDCODED_CREDENTIALS",
    sink: {
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "sap-xssec",
                "path" : [ { "property" : "createSecurityContext" } ]
            }
        },
        "input": "arg1"
    }
},
// bz105234: function express.session
// sink: require('express').session({secret: 'secret', ...})
{
    "sink_for_checker" : "HARDCODED_CREDENTIALS",
    "sink" : {
        "to_callsite" : {
            "call_on" : {
                "read_from_js_require" : "express",
                "path" : [ { "property" : "session" } ]
            }
        },
        "input" : "arg1",
        "path"  : [ { "property" : "secret"  } ]
    }
},
// bz107872: AngularJS crypto
// sink: $cryptoProvider.setCryptographyKey('key')
{
    "sink_for_checker" : "HARDCODED_CREDENTIALS",
    "sink" : {
        "to_callsite" : {
            "call_on" : {
                "from_mangled_function": ".*__coverity_angjs__.service_map.\\$cryptoProvider.se
```

```
CryptographyKey$",
                }
            },
            "input" : "arg1",
        }
    },
    // sink (AngularJS): $crypto.decrypt(encrypted, 'key')
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function": ".*__coverity_angjs__.service_map.\\$crypto.decrypt$",
                }
            },
            "input" : "arg2",
        }
    },
    // sink (AngularJS): $crypto.encrypt(decrypted, 'key')
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function": ".*__coverity_angjs__.service_map.\\$crypto.encrypt$",
                }
            },
            "input" : "arg2",
        }
    },
    // sink (AngularJS): cfCryptoHttpInterceptor.base64key
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "write" : [ { "property" : "base64key" } ],
            "write_to_object_of_type" : "CfCryptoHttpInterceptor"
        }
    },
}, // javascript
// -----
// PYTHON sinks
  "type" : "Coverity analysis configuration",
  "format_version" : 10,
  "language" : "Python",
  "directives" : [
    // temporary milestone 1 HARDCODED_CREDENTIALS sink: any.hardcodedCredentialsSink
ethod
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "input" : "arg1",
            "to_callsite" : {
                "call_on" : {
                    "read_off_any" : [ { "property" : "hardcodedCredentialsPythonSinkMethod" } ]
```

```
            },
          }
        }
    },
}, // PYTHON
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
// -----
// Swift sinks
  "type" : "Coverity analysis configuration",
  "format_version" : 10,
  "language" : "Swift",
  "directives" : [
    // password sink: Foundation.URLCredential ctor
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
          "to_callsite" : {
            "call_on" : {
              "from_mangled_function" : "Foundation.URLCredential.init\\(user:Swift.String, pas
word:Swift.String, persistence:Foundation.URLCredential.Persistence\\)Foundation.URLCredent
al"
            }
          },
          "input" : "arg2"
        }
    },
    // token sink: Accounts.ACAccountCredential ctor
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
          "to_callsite" : {
            "call_on" : {
              "from_mangled_function" : "Accounts.ACAccountCredential.init\\(oAuthToken:Swi
t.ImplicitlyUnwrappedOptional`1<Swift.String>, tokenSecret:Swift.ImplicitlyUnwrappedOption
l`1<Swift.String>\\)Swift.ImplicitlyUnwrappedOptional`1<Accounts.ACAccountCredential>"
            }
          },
          "input" : "arg1"
        }
    },
    // token sink: Accounts.ACAccountCredential ctor
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
          "to_callsite" : {
            "call_on" : {
              "from_mangled_function" : "Accounts.ACAccountCredential.init\\(oAuthToken:Swi
t.ImplicitlyUnwrappedOptional`1<Swift.String>, tokenSecret:Swift.ImplicitlyUnwrappedOption
l`1<Swift.String>\\)Swift.ImplicitlyUnwrappedOptional`1<Accounts.ACAccountCredential>"
            }
          },
          "input" : "arg2"
        }
    },
```

```
    // token sink: Accounts.ACAccountCredential ctor
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Accounts.ACAccountCredential.init\\(oAuth2Token:S
ift.ImplicitlyUnwrappedOptional`1<Swift.String>, refreshToken:Swift.ImplicitlyUnwrappedOpti
nal`1<Swift.String>, expiryDate:Swift.ImplicitlyUnwrappedOptional`1<Foundation.Date>\\)Swi
t.ImplicitlyUnwrappedOptional`1<Accounts.ACAccountCredential>",
                }
            },
            "input" : "arg1"
        }
    },
    // token sink: Accounts.ACAccountCredential ctor
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Accounts.ACAccountCredential.init\\(oAuth2Token:S
ift.ImplicitlyUnwrappedOptional`1<Swift.String>, refreshToken:Swift.ImplicitlyUnwrappedOpti
nal`1<Swift.String>, expiryDate:Swift.ImplicitlyUnwrappedOptional`1<Foundation.Date>\\)Swi
t.ImplicitlyUnwrappedOptional`1<Accounts.ACAccountCredential>",
                }
            },
            "input" : "arg2"
        }
    },
    // token sink: Accounts.ACAccountCredential oauthToken property setter
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Accounts.ACAccountCredential.\\$set_oauthToken\\(_
Swift.ImplicitlyUnwrappedOptional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: Foundation.NSURLComponents percentEncodedPassword property setter
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Foundation.NSURLComponents.\\$set_percentEncod
dPassword\\(_:Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
```

```
        }
    },
    // password sink: Foundation.NSURLComponents password property setter
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Foundation.NSURLComponents.\\$set_password\\(_:
wift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: Foundation.URLComponents percentEncodedPassword property setter
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Foundation.URLComponents.\\$set_percentEncoded
assword\\(_:Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: Foundation.URLComponents password property setter
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Foundation.URLComponents.\\$set_password\\(_:Swi
t.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: CFNetwork.CFHTTPMessageApplyCredentials
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "CFNetwork.CFHTTPMessageApplyCredentials\\(_:CF
etwork.CFHTTPMessage, _:CFNetwork.CFHTTPAuthentication, _:Swift.Optional`1<CoreFoundat
on.CFString>, _:Swift.Optional`1<CoreFoundation.CFString>, _:Swift.Optional`1<Swift.Unsafe
utablePointer`1<CoreFoundation.CFStreamError>>\\)Bool"
                }
            },
            "input" : "arg4"
```

```
        }
    },
    // password sink: CFNetwork.CFHTTPMessageAddAuthentication
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "CFNetwork.CFHTTPMessageAddAuthentication\\(_:C
Network.CFHTTPMessage, _:Swift.Optional`1<CFNetwork.CFHTTPMessage>, _:CoreFoundation
CFString, _:CoreFoundation.CFString, _:Swift.Optional`1<CoreFoundation.CFString>, _:Bool\\)
ool"
                }
            },
            "input" : "arg4"
        }
    },
    // password sink: CGPDFDocument.unlockWithPassword
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "CoreGraphics.CGPDFDocument.unlockWithPassword
\(_:Swift.UnsafePointer`1<Int8>\\)Bool"
                }
            },
            "input" : "arg2"
        }
    },
    // password sink: NEProxyServer password property
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "NetworkExtension.NEProxyServer.\\$set_password\\(_
Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: NEHotspotEAPSettings password property
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "NetworkExtension.NEHotspotEAPSettings.\\$set_pas
word\\(_:Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
```

```
      }
   },
   // password sink: NEVPNProtocol identityDataPassword property
   {
      "sink_for_checker" : "HARDCODED_CREDENTIALS",
      "sink" : {
         "to_callsite" : {
            "call_on" : {
               "from_mangled_function" : "NetworkExtension.NEVPNProtocol.\\$set_identityDat
Password\\(_:Swift.Optional`1<Swift.String>\\)Void"
            }
         },
         "input" : "arg1"
      }
   },
   // password sink: NEVPNProtocol passwordReference property
   {
      "sink_for_checker" : "HARDCODED_CREDENTIALS",
      "sink" : {
         "to_callsite" : {
            "call_on" : {
               "from_mangled_function" : "NetworkExtension.NEVPNProtocol.\\$set_passwordR
ference\\(_:Swift.Optional`1<Foundation.Data>\\)Void"
            }
         },
         "input" : "arg1"
      }
   },
   // password sink: NEFilterProviderConfiguration passwordReference property
   {
      "sink_for_checker" : "HARDCODED_CREDENTIALS",
      "sink" : {
         "to_callsite" : {
            "call_on" : {
               "from_mangled_function" : "NetworkExtension.NEFilterProviderConfiguration.\\$
et_passwordReference\\(_:Swift.Optional`1<Foundation.Data>\\)Void"
            }
         },
         "input" : "arg1"
      }
   },
   // password sink: Security.SecAddSharedWebCredential
   {
      "sink_for_checker" : "HARDCODED_CREDENTIALS",
      "sink" : {
         "to_callsite" : {
            "call_on" : {
               "from_mangled_function" : "Security.SecAddSharedWebCredential\\(_:CoreFound
tion.CFString, _:CoreFoundation.CFString, _:Swift.Optional`1<CoreFoundation.CFString>, _:\\(_
Swift.Optional`1<CoreFoundation.CFError>\\)Void\\)Void"
            }
         },
         "input" : "arg3"
      }
```

```
    },
    // password sink: CKFetchWebAuthTokenOperation ctor
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "CloudKit.CKFetchWebAuthTokenOperation.init\\(api
oken:Swift.String\\)CloudKit.CKFetchWebAuthTokenOperation"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: CKFetchWebAuthTokenOperation apiToken property setter
    {
        "sink_for_checker" : "HARDCODED_CREDENTIALS",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "CloudKit.CKFetchWebAuthTokenOperation.\\$set_AP
Token\\(_:Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
}, // Swift
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    {
        "type" : "Coverity analysis configuration",
        "format_version" : 7,
        "language" : "javascript",
        "directives" : [
            // There already is an iframe directive in
            // analysis/checkers/directives/js/javascript-client.json
            {
                dataflow_through_callsite: {
--
        "sink_for_checker": "INSECURE_SALT",
        "sink":{
            "input": "arg2",
            "to_callsite": {
                "call_on": {
                    "read_from_object_of_type": "Module.bcrypt",
                    "read" : [ { "property" : "hash" } ]
                }
            }
        }
    },
    {
        "sink_for_checker": "INSECURE_SALT",
        "sink":{
            "input": "arg2",
```

```
            "to_callsite": {
               "call_on": {
                  "read_from_js_require": "bcrypt",
                  "path": [{ "property": "hashSync" }],
               }
            }
         }
      },
      {
         "sink_for_checker": "INSECURE_SALT",
         "sink": {
            "input": "arg2",
            "to_callsite": {
               "call_on": {
                  "read_from_object_of_type": "Module.crypto",
                  "read" : [ { "property" : "pbkdf2" } ]
               },
            },
         }
      },
      ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
// 1) MongoDb actions requiring authorization checks.
// 2) Mongoose actions requiring authorization checks.
// 3) Sequelize actions requiring authorization checks.
// 4) Bookshelf actions requiring authorization checks.
// 5) orm actions requiring authorization checks.
// 6) HANA XSC actions requiring authorization checks.
// 7) Acl actions requiring authorization checks.
// 8) Tedious actions requiring authorization checks.
--
         "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
         "sink" : {
            "to_callsite" : {
               "call_on" : {
                  "from_mangled_function" : "Foundation.URLCredential.init\\(user:Swift.String, pas
word:Swift.String, persistence:Foundation.URLCredential.Persistence\\)Foundation.URLCredent
al"
               }
            },
            "input" : "arg2"
         }
      },
      // token sink: Accounts.ACAccountCredential ctor
      {
         "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
         "sink" : {
            "to_callsite" : {
               "call_on" : {
                  "from_mangled_function" : "Accounts.ACAccountCredential.init\\(oAuthToken:Swi
t.ImplicitlyUnwrappedOptional`1<Swift.String>, tokenSecret:Swift.ImplicitlyUnwrappedOption
l`1<Swift.String>\\)Swift.ImplicitlyUnwrappedOptional`1<Accounts.ACAccountCredential>"
               }
            },
```

```
            "input" : "arg1"
         }
      },
      // token sink: Accounts.ACAccountCredential ctor
      {
         "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
         "sink" : {
            "to_callsite" : {
               "call_on" : {
                  "from_mangled_function" : "Accounts.ACAccountCredential.init\\(oAuthToken:Swi
t.ImplicitlyUnwrappedOptional`1<Swift.String>, tokenSecret:Swift.ImplicitlyUnwrappedOption
l`1<Swift.String>\\)Swift.ImplicitlyUnwrappedOptional`1<Accounts.ACAccountCredential>"
               }
            },
            "input" : "arg2"
         }
      },
      // token sink: Accounts.ACAccountCredential ctor
      {
         "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
         "sink" : {
            "to_callsite" : {
               "call_on" : {
                  "from_mangled_function" : "Accounts.ACAccountCredential.init\\(oAuth2Token:S
ift.ImplicitlyUnwrappedOptional`1<Swift.String>, refreshToken:Swift.ImplicitlyUnwrappedOpti
nal`1<Swift.String>, expiryDate:Swift.ImplicitlyUnwrappedOptional`1<Foundation.Date>\\)Swi
t.ImplicitlyUnwrappedOptional`1<Accounts.ACAccountCredential>",
               }
            },
            "input" : "arg1"
         }
      },
      // token sink: Accounts.ACAccountCredential ctor
      {
         "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
         "sink" : {
            "to_callsite" : {
               "call_on" : {
                  "from_mangled_function" : "Accounts.ACAccountCredential.init\\(oAuth2Token:S
ift.ImplicitlyUnwrappedOptional`1<Swift.String>, refreshToken:Swift.ImplicitlyUnwrappedOpti
nal`1<Swift.String>, expiryDate:Swift.ImplicitlyUnwrappedOptional`1<Foundation.Date>\\)Swi
t.ImplicitlyUnwrappedOptional`1<Accounts.ACAccountCredential>",
               }
            },
            "input" : "arg2"
         }
      },
      // token sink: Accounts.ACAccountCredential oauthToken property setter
      {
         "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
         "sink" : {
            "to_callsite" : {
               "call_on" : {
                  "from_mangled_function" : "Accounts.ACAccountCredential.\\$set_oauthToken\\(_
```

```
Swift.ImplicitlyUnwrappedOptional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: Foundation.NSURLComponents percentEncodedPassword property setter
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Foundation.NSURLComponents.\\$set_percentEncod
dPassword\\(_:Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: Foundation.NSURLComponents password property setter
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Foundation.NSURLComponents.\\$set_password\\(_:
wift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: Foundation.URLComponents percentEncodedPassword property setter
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Foundation.URLComponents.\\$set_percentEncoded
assword\\(_:Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: Foundation.URLComponents password property setter
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Foundation.URLComponents.\\$set_password\\(_:Swi
t.Optional`1<Swift.String>\\)Void"
                }
```

```
            },
            "input" : "arg1"
        }
    },
    // password sink: CFNetwork.CFHTTPMessageApplyCredentials
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "CFNetwork.CFHTTPMessageApplyCredentials\\(_:CF
etwork.CFHTTPMessage, _:CFNetwork.CFHTTPAuthentication, _:Swift.Optional`1<CoreFoundat
on.CFString>, _:Swift.Optional`1<CoreFoundation.CFString>, _:Swift.Optional`1<Swift.Unsafe
utablePointer`1<CoreFoundation.CFStreamError>>\\)Bool"
                }
            },
            "input" : "arg4"
        }
    },
    // password sink: CFNetwork.CFHTTPMessageAddAuthentication
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "CFNetwork.CFHTTPMessageAddAuthentication\\(_:C
Network.CFHTTPMessage, _:Swift.Optional`1<CFNetwork.CFHTTPMessage>, _:CoreFoundation
CFString, _:CoreFoundation.CFString, _:Swift.Optional`1<CoreFoundation.CFString>, _:Bool\\)
ool"
                }
            },
            "input" : "arg4"
        }
    },
    // password sink: CGPDFDocument.unlockWithPassword
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "CoreGraphics.CGPDFDocument.unlockWithPassword
\(_:Swift.UnsafePointer`1<Int8>\\)Bool"
                }
            },
            "input" : "arg2"
        }
    },
    // password sink: NEProxyServer password property
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "NetworkExtension.NEProxyServer.\\$set_password\\(_
```

```
Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: NEHotspotEAPSettings password property
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "NetworkExtension.NEHotspotEAPSettings.\\$set_pas
word\\(_:Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: NEVPNProtocol identityDataPassword property
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "NetworkExtension.NEVPNProtocol.\\$set_identityDat
Password\\(_:Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: NEVPNProtocol passwordReference property
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "NetworkExtension.NEVPNProtocol.\\$set_passwordR
ference\\(_:Swift.Optional`1<Foundation.Data>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: NEFilterProviderConfiguration passwordReference property
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "NetworkExtension.NEFilterProviderConfiguration.\\$
et_passwordReference\\(_:Swift.Optional`1<Foundation.Data>\\)Void"
                }
```

```
            },
            "input" : "arg1"
        }
    },
    // password sink: Security.SecAddSharedWebCredential
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "Security.SecAddSharedWebCredential\\(_:CoreFound
tion.CFString, _:CoreFoundation.CFString, _:Swift.Optional`1<CoreFoundation.CFString>, _:\\(_
Swift.Optional`1<CoreFoundation.CFError>\\)Void\\)Void"
                }
            },
            "input" : "arg3"
        }
    },
    // password sink: CKFetchWebAuthTokenOperation ctor
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "CloudKit.CKFetchWebAuthTokenOperation.init\\(api
oken:Swift.String\\)CloudKit.CKFetchWebAuthTokenOperation"
                }
            },
            "input" : "arg1"
        }
    },
    // password sink: CKFetchWebAuthTokenOperation apiToken property setter
    {
        "sink_for_checker" : "UNENCRYPTED_SENSITIVE_DATA",
        "sink" : {
            "to_callsite" : {
                "call_on" : {
                    "from_mangled_function" : "CloudKit.CKFetchWebAuthTokenOperation.\\$set_AP
Token\\(_:Swift.Optional`1<Swift.String>\\)Void"
                }
            },
            "input" : "arg1"
        }
    },
}, // Swift
typ.getKind() == AuthzCheckType::ANNOTATION_KIND
analysis/checkers/security/checkers/missing-authz/authz-common.cpp
Unexpected null pointer annotation
org.springframework.security.access.prepost.PreAuthorize
Unexpected null pointer callee
org.springframework.security.access.prepost.PostAuthorize
org.springframework.security.access.annotation.Secured
javax.annotation.security.RolesAllowed
System.Web.Http.AuthorizeAttribute
```

```
System.Web.Mvc.AuthorizeAttribute
--
            "sink_for_checker": "XSS",
            "sink": {
                "to_callsite": {
                    "call_on_php_function": {
                        "name": "echo"
                    }
                },
                "input": "all_args"
            }
        },
        // sink: print( [arg1] )
        {
            "sink_for_checker": "XSS",
            "sink": {
                "to_callsite": {
                    "call_on_php_function": {
                        "name": "print"
                    }
                },
                "input": "arg1"
            }
        },
        // sink: printf( ... [all_arg] ... )
        {
            "sink_for_checker": "XSS",
            "sink": {
                "to_callsite": {
                    "call_on_php_function": {
                        "name": "printf"
                    }
                },
                "input": "from_arg1"
            }
        },
        // sink: vprintf( [arg1], ... )
        {
            "sink_for_checker": "XSS",
            "sink": {
                "to_callsite": {
                    "call_on_php_function": {
                        "name": "vprintf"
                    }
                },
                "input": "arg1"
            }
        },
        // sink: vprintf( ..., [arg2][*] )
        {
            "sink_for_checker": "XSS",
            "sink": {
                "to_callsite": {
                    "call_on_php_function": {
```

```
            "name": "vprintf"
         }
      },
      "input": "arg2",
      "path": [
         {
            "any_property": true
         }
      ]
   }
},
// sink: exit( [arg1] )  // exit prints arg1 if it is a string
{
   "sink_for_checker": "XSS",
   "sink": {
      "to_callsite": {
         "call_on_php_function": {
            "name": "exit"
         }
      },
      "input": "arg1",
   }
},
// sink: die( [arg1] )  // alias of "exit"
{
   "sink_for_checker": "XSS",
   "sink": {
      "to_callsite": {
         "call_on_php_function": {
            "name": "die"
         }
      },
      "input": "arg1",
   }
},
// sink: trigger_error( [arg1], ... )
{
   "sink_for_checker": "XSS",
   "sink": {
      "to_callsite": {
         "call_on_php_function": {
            "name": "trigger_error"
         }
      },
      "input": "arg1"
   }
},
// sink: user_error( [arg1], ... )  // alias of trigger_error
{
   "sink_for_checker": "XSS",
   "sink": {
      "to_callsite": {
         "call_on_php_function": {
            "name": "user_error"
```

```
                }
            },
            "input": "arg1"
        }
    },
////////////////////////////////////////////////////////////////////////////
// SENSITIVE_DATA_LEAK Sinks
    // sink: exit( [arg1] )  // exit prints arg1 if it is a string
    {
        "sink_for_checker": "SENSITIVE_DATA_LEAK",
        "sink_kind" : "ui",
        "sink": {
            "to_callsite": {
                "call_on_php_function": {
                    "name": "exit"
                }
            },
            "input": "arg1",
        }
    },
    // sink: die( [arg1] )  // alias of "exit"
    {
        "sink_for_checker": "SENSITIVE_DATA_LEAK",
        "sink_kind" : "ui",
        "sink": {
            "to_callsite": {
                "call_on_php_function": {
                    "name": "die"
                }
            },
            "input": "arg1",
        }
    },
    // sink: trigger_error( [arg1], ... )
    {
        "sink_for_checker": "SENSITIVE_DATA_LEAK",
        "sink_kind" : "ui",
        "sink": {
            "to_callsite": {
                "call_on_php_function": {
                    "name": "trigger_error"
                }
            },
            "input": "arg1"
        }
    },
    // sink: user_error( [arg1], ... )  // alias of trigger_error
    {
        "sink_for_checker": "SENSITIVE_DATA_LEAK",
        "sink_kind" : "ui",
        "sink": {
            "to_callsite": {
                "call_on_php_function": {
                    "name": "user_error"
```

```
                }
            },
            "input": "arg1"
        }
    },
////////////////////////////////////////////////////////////////////////////
// Sources of sensitive data
// Exception::getTrace()
// Exception::getTraceAsString()
// Throwable::getTrace()
// Throwable::getTraceAsString()
// debug_backtrace()
// TODO error_reporting()
    {
        taint_kind : "exception",
--
        sink_for_checker : "SQLI",
        sink : {
            input : "arg1",
            to_callsite : {
                call_on_php_instance_method : {
                    name : "query",
                    receiver_object_is_class : {
                        class_name : "mysqli",
                        namespace: ""
                    },
                },
            }
        }
    },
    {
        sink_for_checker : "SQLI",
        sink : {
            input : "arg1",
            to_callsite : {
                call_on_php_instance_method : {
                    name : "multi_query",
                    receiver_object_is_class : {
                        class_name : "mysqli",
                        namespace: ""
                    },
                },
            }
        }
    },
    {
        sink_for_checker : "SQLI",
        sink : {
            input : "arg1",
            to_callsite : {
                call_on_php_instance_method : {
                    name : "real_query",
                    receiver_object_is_class : {
                        class_name : "mysqli",
```

```
                    namespace: ""
                },
            },
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_instance_method : {
                name : "prepare",
                receiver_object_is_class : {
                    class_name : "mysqli",
                    namespace: ""
                },
            },
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_instance_method : {
                name : "send_query",
                receiver_object_is_class : {
                    class_name : "mysqli",
                    namespace: ""
                },
            },
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_instance_method : {
                name : "prepare",
                receiver_object_is_class : {
                    class_name : "mysqli_stmt",
                    namespace: ""
                },
            },
        }
    }
},
// mysqli_query ( mysqli $link , string $query )
// mysqli_multi_query ( mysqli $link , string $query )
// mysqli_real_query ( mysqli $link , string $query )
```

```
// mysqli_prepare ( mysqli $link , string $query )
// mysqli_send_query ( mysqli $link , string $query )
// mysqli_stmt_prepare ( mysqli_stmt $stmt , string $query )
        {
            sink_for_checker : "SQLI",
            sink : {
                input : "arg2",
                to_callsite : {
                    call_on_php_function : {
                        name : "mysqli_query"
                    }
                }
            }
        },
        {
            sink_for_checker : "SQLI",
            sink : {
                input : "arg2",
                to_callsite : {
                    call_on_php_function : {
                        name : "mysqli_multi_query"
                    }
                }
            }
        },
        {
            sink_for_checker : "SQLI",
            sink : {
                input : "arg2",
                to_callsite : {
                    call_on_php_function : {
                        name : "mysqli_real_query"
                    }
                }
            }
        },
        {
            sink_for_checker : "SQLI",
            sink : {
                input : "arg2",
                to_callsite : {
                    call_on_php_function : {
                        name : "mysqli_prepare"
                    }
                }
            }
        },
        {
            sink_for_checker : "SQLI",
            sink : {
                input : "arg2",
                to_callsite : {
                    call_on_php_function : {
                        name : "mysqli_send_query"
```

```
                }
            }
        }
    },
    {
        sink_for_checker : "SQLI",
        sink : {
            input : "arg2",
            to_callsite : {
                call_on_php_function : {
                    name : "mysqli_stmt_prepare"
                }
            }
        }
    },
// Password Sinks
// --------------
// TODO Add sinks for UNENCRYPTED_SENSITIVE_DATA
// TODO mysqli::__construct([ string $host [, string $username [, string $passwd
//                  [, string $dbname [, int $port [, string $socket]]]]] )
// mysqli::real_connect ([ string $host [, string $username [, string $passwd
//                  [, string $dbname [, int $port [, string $socket [, int $flags ]]]]]] )
// mysqli::change_user ( string $user , string $password , string $database )
    {
        sink_for_checker : "HARDCODED_CREDENTIALS",
        sink_kind : "hardcoded_credential_passwd",
        sink : {
            input : "arg3",
            to_callsite : {
                call_on_php_instance_method : {
                    name : "real_connect",
                    receiver_object_is_class : {
                        class_name : "mysqli",
                        namespace: ""
                    },
                },
            }
        }
    },
    {
        sink_for_checker : "HARDCODED_CREDENTIALS",
        sink_kind : "hardcoded_credential_passwd",
        sink : {
            input : "arg2",
            to_callsite : {
                call_on_php_instance_method : {
                    name : "change_user",
                    receiver_object_is_class : {
                        class_name : "mysqli",
                        namespace: ""
                    },
                },
            }
        }
```

```
        },
// mysqli_connect([ string $host [, string $username [, string $passwd
//                  [, string $dbname [, int $port [, string $socket]]]]]] )
// mysqli_real_connect ( mysqli $link [, string $host [, string $username [, string $passwd
//                  [, string $dbname [, int $port [, string $socket [, int $flags ]]]]]] )
// mysqli_change_user ( mysqli $link , string $user , string $password , string $database )
        {
            sink_for_checker : "HARDCODED_CREDENTIALS",
            sink_kind : "hardcoded_credential_passwd",
            sink : {
                input : "arg3",
                to_callsite : {
                    call_on_php_function : {
                        name : "mysqli_connect"
                    },
                }
            }
        },
        {
            sink_for_checker : "HARDCODED_CREDENTIALS",
            sink_kind : "hardcoded_credential_passwd",
            sink : {
                input : "arg4",
                to_callsite : {
                    call_on_php_function : {
                        name : "mysqli_real_connect"
                    },
                }
            }
        },
        {
            sink_for_checker : "HARDCODED_CREDENTIALS",
            sink_kind : "hardcoded_credential_passwd",
            sink : {
                input : "arg3",
                to_callsite : {
                    call_on_php_function : {
                        name : "mysqli_change_user"
                    },
                }
            }
        },
// Escapers
// --------
// TODO Dzin: For now, the FP-avoiding approach is to not model any dataflow.
// If we start modeling escapers, add models for the following.
// mysqli::real_escape_string ( string $escapestr )
// mysqli_real_escape_string ( mysqli $link , string $escapestr )
// escape_string ( string $escapestr )
// mysqli::mysqli_escape_string ( mysqli $link , string $escapestr )
    ]
////////////////////////////////////////////////////////////////////////////
--
            sink_for_checker : "SQLI",
```

```
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_function : {
                name : "pg_query_params"
            },
            when : {
                only_if_arg_index : 2,
                is_max_index : true
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg2",
        to_callsite : {
            call_on_php_function : {
                name : "pg_query_params"
            },
            when : {
                only_if_arg_index : 3,
                is_max_index : true
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg2",
        to_callsite : {
            call_on_php_function : {
                name : "pg_prepare"
            },
            when : {
                only_if_arg_index : 2,
                is_max_index : true
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg3",
        to_callsite : {
            call_on_php_function : {
                name : "pg_prepare"
            },
            when : {
                only_if_arg_index : 3,
                is_max_index : true
```

```
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_function : {
                name : "pg_query"
            },
            when : {
                only_if_arg_index : 1,
                is_max_index : true
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg2",
        to_callsite : {
            call_on_php_function : {
                name : "pg_query"
            },
            when : {
                only_if_arg_index : 2,
                is_max_index : true
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg3",
        to_callsite : {
            call_on_php_function : {
                name : "pg_send_prepare"
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg2",
        to_callsite : {
            call_on_php_function : {
                name : "pg_send_query_params"
            }
        }
```

原文链接：coverity 部分规则提取

```
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg2",
        to_callsite : {
            call_on_php_function : {
                name : "pg_send_query"
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg2",
        to_callsite : {
            call_on_php_function : {
                name : "pg_select"
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg3",
        to_callsite : {
            call_on_php_function : {
                name : "pg_select"
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg2",
        to_callsite : {
            call_on_php_function : {
                name : "pg_delete"
            }
        }
    }
},
{
    sink_for_checker : "SQLI",
    sink : {
        input : "arg3",
        to_callsite : {
            call_on_php_function : {
                name : "pg_delete"
            }
```

```
            }
        }
    },
    {
        sink_for_checker : "SQLI",
        sink : {
            input : "arg2",
            to_callsite : {
                call_on_php_function : {
                    name : "pg_insert"
                }
            }
        }
    },
    {
        sink_for_checker : "SQLI",
        sink : {
            input : "arg3",
            to_callsite : {
                call_on_php_function : {
                    name : "pg_insert"
                }
            }
        }
    },
// Connection String Sinks
// ----------------------
// TODO Add sinks for UNENCRYPTED_SENSITIVE_DATA
// XXX connection_string includes password
    {
        sink_for_checker : "HARDCODED_CREDENTIALS",
        sink_kind : "hardcoded_credential_connection_string",
        sink : {
            input : "arg1",
            to_callsite : {
                call_on_php_function : {
                    name : "pg_connect"
                },
            }
        }
    },
    {
        sink_for_checker : "HARDCODED_CREDENTIALS",
        sink_kind : "hardcoded_credential_connection_string",
        sink : {
            input : "arg1",
            to_callsite : {
                call_on_php_function : {
                    name : "pg_pconnect"
                },
            }
        }
    },
// Directory traversal sinks
```

```
// -------------------------
// TODO pg_lo_import ([ resource $connection ], string $pathname [, mixed $object_id ] )
// pg_lo_export ([ resource $connection ], int $oid , string $pathname )
      {
         sink_for_checker : "PATH_MANIPULATION",
         sink : {
            input : "arg2",
            to_callsite : {
               call_on_php_function : {
                  name : "pg_lo_export"
               },
               when : {
                  only_if_arg_index : 2,
                  is_max_index : true
               }
            }
         }
      },
      {
         sink_for_checker : "PATH_MANIPULATION",
         sink : {
            input : "arg3",
            to_callsite : {
               call_on_php_function : {
                  name : "pg_lo_export"
               },
               when : {
                  only_if_arg_index : 3,
                  is_max_index : true
               }
            }
         }
      },
// Escapers
// --------
// TODO Dzin: For now, the FP-avoiding approach is to not model any dataflow.
// If we start modeling escapers, add models for the following.
// pg_escape_string ([ resource $connection ], string $data )
// pg_escape_identifier ([ resource $connection ], string $data )
// pg_escape_literal ([ resource $connection ], string $data )
--
         sink_for_checker : "SQLI",
         sink : {
            input : "arg1",
            to_callsite : {
               call_on_php_instance_method : {
                  name : "prepare",
                  receiver_object_is_class : {
                     class_name : "PDO",
                     namespace: ""
                  },
               },
            }
         }
```

```
        },
        {
            sink_for_checker : "SQLI",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_instance_method : {
                        name : "exec",
                        receiver_object_is_class : {
                            class_name : "PDO",
                            namespace: ""
                        },
                    },
                }
            }
        },
        {
            sink_for_checker : "SQLI",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_instance_method : {
                        name : "query",
                        receiver_object_is_class : {
                            class_name : "PDO",
                            namespace: ""
                        },
                    },
                }
            }
        },
    ]
////////////////////////////////////////////////////////////////////////////
// End of directive list
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
////////////////////////////////////////////////////////////////////////////
// Definitions for the IBM-DB2 plugin
    "type" : "Coverity analysis configuration",
--
            sink_for_checker : "HARDCODED_CREDENTIALS",
            sink_kind : "hardcoded_credential_passwd",
            sink : {
                input : "arg3",
                to_callsite : {
                    call_on_php_function : {
                        name : "db2_connect"
                    },
                }
            }
        },
        {
            sink_for_checker : "HARDCODED_CREDENTIALS",
            sink_kind : "hardcoded_credential_passwd",
            sink : {
```

```
            input : "arg3",
            to_callsite : {
                call_on_php_function : {
                    name : "db2_pconnect"
                },
            }
        }
    },
// SQL Injection Sinks
// -------------------
// db2_exec ( resource $connection , string $statement [, array $options ] )
// db2_prepare ( resource $connection , string $statement [, array $options ] )
    {
        sink_for_checker : "SQLI",
        sink : {
            input : "arg2",
            to_callsite : {
                call_on_php_function : {
                    name : "db2_exec"
                },
            }
        }
    },
    {
        sink_for_checker : "SQLI",
        sink : {
            input : "arg2",
            to_callsite : {
                call_on_php_function : {
                    name : "db2_prepare"
                },
            }
        }
    },
  ]
////////////////////////////////////////////////////////////////////////////
// End of directive list
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
////////////////////////////////////////////////////////////////////////////
// Definitions for the ODBC plugin
  "type" : "Coverity analysis configuration",
  "format_version" : 10,
  "language" : "PHP",
  "directives" : [
// Database Sources
--
        sink_for_checker : "HARDCODED_CREDENTIALS",
        sink_kind : "hardcoded_credential_passwd",
        sink : {
            input : "arg3",
            to_callsite : {
                call_on_php_function : {
                    name : "odbc_connect"
                },
```

```
            }
        }
    },
    {
        sink_for_checker : "HARDCODED_CREDENTIALS",
        sink_kind : "hardcoded_credential_passwd",
        sink : {
            input : "arg3",
            to_callsite : {
                call_on_php_function : {
                    name : "odbc_pconnect"
                },
            }
        }
    },
// SQL Injection Sinks
// -------------------
// odbc_exec ( resource $connection_id , string $query_string [, int $flags ] )
// odbc_do ( resource $connection_id , string $query_string [, int $flags ] )
// odbc_prepare ( resource $connection_id , string $query_string )
    {
        sink_for_checker : "SQLI",
        sink : {
            input : "arg2",
            to_callsite : {
                call_on_php_function : {
                    name : "odbc_exec"
                }
            }
        }
    },
    {
        sink_for_checker : "SQLI",
        sink : {
            input : "arg2",
            to_callsite : {
                call_on_php_function : {
                    name : "odbc_do"
                }
            }
        }
    },
    {
        sink_for_checker : "SQLI",
        sink : {
            input : "arg2",
            to_callsite : {
                call_on_php_function : {
                    name : "odbc_prepare"
                }
            }
        }
    },
]
```

原文链接：coverity 部分规则提取

```
////////////////////////////////////////////////////////////////////////////
// End of directive list
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
////////////////////////////////////////////////////////////////////////////
// Definitions for the DBA plugin
    "type" : "Coverity analysis configuration",
    "format_version" : 10,
    "language" : "PHP",
    "directives" : [
// Database Sources
--
            sink_for_checker : "PATH_MANIPULATION",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_function : {
                        name : "dba_popen"
                    },
                }
            }
        },
        {
            sink_for_checker : "PATH_MANIPULATION",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_function : {
                        name : "dba_open"
                    },
                }
            }
        },
    ]
////////////////////////////////////////////////////////////////////////////
// End of directive list
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 10,
    "language" : "PHP",
    "directives" : [
////////////////////////////////////////////////////////////////////////////
// Sources of tainted data
        //-----------------------------------------------------------------
--
            sink_for_checker: "HEADER_INJECTION",
            sink: {
                to_callsite: {
                    call_on_php_function: {
                        namespace: "\\", name: "header"
                    },
                },
                input: "arg1"
            }
        },
```

```
        // source: headers_list()[*]
        // Not working - "path" for call's return value
        /*{
            taint_kind: "network",
            tainted_data: {
                from_callsite: {
                    call_on_php_function: {
                        namespace: "\\", name: "headers_list"
                    },
                },
                output: "return",
--
            sink_for_checker: "SENSITIVE_DATA_LEAK",
            sink_kind: "logging",
            sink: {
                to_callsite: {
                    call_on_php_function: {
                        namespace: "\\", name: "openlog"
                    },
                },
                input: "arg1"
            }
        },
        // sink: setcookie( /*taint*/name, [, ...] )
        // Note: COOKIE_INJECTION not enabled for napa.
        {
            sink_for_checker: "COOKIE_INJECTION",
            sink: {
                to_callsite: {
                    call_on_php_function: {
                        namespace: "\\", name: "setcookie"
                    },
                },
                input: "arg1"
            }
        },
        // sink: setcookie( name, /*taint*/value, [, ...] )
        // Note: COOKIE_INJECTION not enabled for napa.
        {
            sink_for_checker: "COOKIE_INJECTION",
            sink: {
                to_callsite: {
                    call_on_php_function: {
                        namespace: "\\", name: "setcookie"
                    },
                },
                input: "arg2"
            }
        },
        // sink: setrawcookie( /*taint*/name, [, ...] )
        // Note: COOKIE_INJECTION not enabled for napa.
        {
            sink_for_checker: "COOKIE_INJECTION",
            sink: {
```

```
            to_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "setrawcookie"
                },
            },
            input: "arg1"
        }
    },
    // sink: setrawcookie( name, /*taint*/value, [, ...] )
    // Note: COOKIE_INJECTION not enabled for napa.
    {
        sink_for_checker: "COOKIE_INJECTION",
        sink: {
            to_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "setrawcookie"
                },
            },
            input: "arg2"
        }
    },
    // source: socket_get_status(...)["uri"]
    // Not working - "path" for call's return value
    /*{
        taint_kind: "network",
        tainted_data: {
            from_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "socket_get_status"
                },
            },
            output: "return",
--
        sink_for_checker: "SENSITIVE_DATA_LEAK",
        sink_kind: "logging",
        sink: {
            to_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "syslog"
                },
            },
            input: "arg2"
        }
    },
]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
type : "Coverity analysis configuration",
format_version : 10,
language : "php",
directives : [
    // source: output[*] from exec( command, /*tainted array*/&output, ... )
    // Not working due to Bug 108860.
    /*{
        taint_kind: "console",
```

```
--
        sink_for_checker: "OS_CMD_INJECTION",
        sink: {
            to_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "exec"
                },
            },
            input: "arg1",
        }
    },
    // sink: passthru( /*sink*/command, ... )
    {
        sink_for_checker: "OS_CMD_INJECTION",
        sink: {
            to_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "passthru"
                },
            },
            input: "arg1",
        }
    },
    // sink: proc_open( /*sink*/command ... )
    {
        sink_for_checker: "OS_CMD_INJECTION",
        sink: {
            to_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "proc_open"
                },
            },
            input: "arg1",
        }
    },
    // sink: proc_open( cmd, descriptorspec, pipes, cwd, /*sink*/env, ... )
    // This sink is for future checker OS_CMD_ENV_INJECTION (Bug 47272).
    // Also, it's not working due to Bug 108860.
    /*{
        sink_for_checker: "OS_CMD_ENV_INJECTION",
        sink: {
            to_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "proc_open"
                },
            },
            input: "arg5",
            path: [ { any_element: true } ]
        }
    },*/
    // source: shell_exec( command )
    {
        taint_kind: "console",
        tainted_data: {
```

```
            from_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "shell_exec"
                },
            },
            output: "return"
--
            sink_for_checker: "OS_CMD_INJECTION",
            sink: {
                to_callsite: {
                    call_on_php_function: {
                        namespace: "\\", name: "shell_exec"
                    },
                },
                input: "arg1",
            }
        },
        // source: system( command )
        {
            taint_kind: "console",
            tainted_data: {
                from_callsite: {
                    call_on_php_function: {
                        namespace: "\\", name: "system"
                    },
                },
                output: "return"
            }
--
            sink_for_checker: "OS_CMD_INJECTION",
            sink: {
                to_callsite: {
                    call_on_php_function: {
                        namespace: "\\", name: "system"
                    },
                },
                input: "arg1",
            }
        },
    ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
// PHP Guidance: Plugin: Mail
// https://codiscope.atlassian.net/wiki/spaces/GUID/pages/127426205/Plugin+Mail
    type : "Coverity analysis configuration",
    format_version : 10,
    language : "php",
    directives : [
        // see also: Bug 111825 - new PHP checker: sendmail injection for
        // calling \mail\mail() with tainted arguments
        // signature of \mail\mail():
--
            sink_for_checker: "SENSITIVE_DATA_LEAK",
            sink_kind : "transit",
            sink: {
```

```
            to_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "mail"
                },
            },
            input: "arg2"
        }
    },
    // sink: mail( to, subject, /*sink*/message, ... )
    {
        sink_for_checker: "SENSITIVE_DATA_LEAK",
        sink_kind : "transit",
        sink: {
            to_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "mail"
                },
            },
            input: "arg3"
        }
    },
    // TODO: sink: mail( to, subject, message, /*sink*/additional_headers, additional_paramet
rs )
    // This injects email headers (mail injection).
    // But, we don't have a relevant checker, yet.
    // sink: mail( to, subject, message, additional_headers, /*sink*/additional_parameters )
    {
        sink_for_checker: "OS_CMD_INJECTION",
        sink: {
            to_callsite: {
                call_on_php_function: {
                    namespace: "\\", name: "mail"
                },
            },
            input: "arg5"
        }
    },
]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    type : "Coverity analysis configuration",
    format_version : 10,
    language : "php",
    directives : [
        //----------------------------------------------------------------------
        // Methods
        // source: new MongoDB\Driver\Manager(...)
        // - Sensitive connection details like username/password.
        {
--
            "sink_for_checker" : "HARDCODED_CREDENTIALS",
            "sink" : {
                "input" : "arg1",
                "to_callsite" : {
                    "call_on_php_function" : {
```

```
                "name" : "password_hash",
                "namespace" : ""
            }
        }
    }
},
// source: arg1, password_needs_rehash
// boolean password_needs_rehash(string $hash ,integer $algo [, array $options ])
{
    "taint_kind" : "hash",
    "tainted_data" : {
        "from_callsite" : {
            "call_on_php_function" : {
                "name" : "password_needs_rehash",
                "namespace" : ""
            },
--
    "sink_for_checker" : "HARDCODED_CREDENTIALS",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_php_function" : {
                "name" : "password_needs_rehash",
                "namespace" : ""
            }
        }
    }
},
// source: arg1, password_verify
// boolean password_verify(string $password ,string $hash)
{
    "taint_kind" : "password",
    "tainted_data" : {
        "from_callsite" : {
            "call_on_php_function" : {
                "name" : "password_verify",
                "namespace" : ""
            },
--
    "sink_for_checker" : "HARDCODED_CREDENTIALS",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_php_function" : {
                "name" : "password_verify",
                "namespace" : ""
            }
        }
    }
},
// Sink: arg2
// boolean password_verify(string $password ,string $hash)
{
    "sink_for_checker" : "HARDCODED_CREDENTIALS",
```

```
            "sink" : {
               "input" : "arg2",
               "to_callsite" : {
                  "call_on_php_function" : {
                     "name" : "password_verify",
                     "namespace" : ""
                  }
               }
            }
         }
      ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
      "type" : "Coverity analysis configuration",
      "format_version" : 10,
      "language" : "php",
      "directives" : [
         // source: return, random_bytes
         // string random_bytes(int $length)
         {
            "taint_kind" : "secure_random",
--
            "sink_for_checker" : "HARDCODED_CREDENTIALS",
            "sink" : {
               "input" : "arg3",
               "to_callsite" : {
                  "call_on_php_function" : {
                     "name" : "ldap_bind",
                     "namespace" : ""
                  }
               }
            }
         },
         // source: return, ldap_first_attribute
         // string ldap_first_attribute(resource $link_identifier, resource $result_entry_identifier)
         {
            "taint_kind" : "configuration",
            "tainted_data" : {
               "from_callsite" : {
                  "call_on_php_function" : {
                     "name" : "ldap_first_attribute",
                     "namespace" : ""
                  },
--
            "sink_for_checker" : "HARDCODED_CREDENTIALS",
            "sink" : {
               "input" : "arg3",
               "to_callsite" : {
                  "call_on_php_function" : {
                     "name" : "ldap_sasl_bind",
                     "namespace" : ""
                  }
               }
            }
         }
```

```
    ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    type : "Coverity analysis configuration",
    format_version : 10,
    language : "php",
    directives : [
   ////////////////////////////////////////////////////////////////////////////
   // SENSITIVE_DATA_LEAK Sinks
       // sink: echo(*)
       {
          "sink_for_checker": "SENSITIVE_DATA_LEAK",
          "sink_kind" : "ui",
          "sink": {
             "to_callsite": {
                "call_on_php_function": {
                   "name": "echo"
                }
             },
             "input": "all_args"
          }
       },
       // sink: print( [arg1] )
       {
          "sink_for_checker": "SENSITIVE_DATA_LEAK",
          "sink_kind" : "ui",
          "sink": {
             "to_callsite": {
                "call_on_php_function": {
                   "name": "print"
                }
             },
             "input": "arg1"
          }
       },
       // sink: printf( ... [all_arg] ... )
       {
          "sink_for_checker": "SENSITIVE_DATA_LEAK",
          "sink_kind" : "ui",
          "sink": {
             "to_callsite": {
                "call_on_php_function": {
                   "name": "printf"
                }
             },
             "input": "from_arg1"
          }
       },
       // sink: vprintf( [arg1], ... )
       {
          "sink_for_checker": "SENSITIVE_DATA_LEAK",
          "sink_kind" : "ui",
          "sink": {
             "to_callsite": {
                "call_on_php_function": {
```

```
                "name": "vprintf"
            }
        },
        "input": "arg1"
    }
},
// sink: vprintf( ..., [arg2][*] )
{
    "sink_for_checker": "SENSITIVE_DATA_LEAK",
    "sink_kind" : "ui",
    "sink": {
        "to_callsite": {
            "call_on_php_function": {
                "name": "vprintf"
            }
        },
        "input": "arg2",
        "path": [
            {
                "any_property": true
            }
        ]
    }
},
]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    type : "Coverity analysis configuration",
    format_version : 10,
    language : "php",
--
        "sink_for_checker" : "SCRIPT_CODE_INJECTION",
        "sink" : {
            "input" : "arg1",
            "to_callsite" : {
                "call_on_php_function" : {
                    "name" : "eval",
                    "namespace" : ""
                }
            }
        }
    },
]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
////////////////////////////////////////////////////////////////////////////
// Definitions for the DBA plugin
    "type" : "Coverity analysis configuration",
    "format_version" : 10,
    "language" : "PHP",
    "directives" : [
// UNSAFE_DESERIALIZATION sinks
// -------------------------
--
        sink_for_checker : "UNSAFE_DESERIALIZATION",
        sink : {
```

```
            input : "arg1",
            to_callsite : {
                call_on_php_function : {
                    name : "unserialize"
                },
            }
        }
    },
]
////////////////////////////////////////////////////////////////////////////
// End of directive list
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 10,
    "language" : "php",
    "directives" : [
    // TODO:
    // ASYNC method modeling of following functions
    // 1.
--
        "sink_for_checker" : "PATH_MANIPULATION",
        "sink" : {
            "input" : "arg1",
            "to_callsite" : {
                "call_on_php_function" : {
                    "name" : "eio_chmod",
                    "namespace" : ""
                },
            }
        }
    },
    {
        "sink_for_checker" : "PATH_MANIPULATION",
        "sink" : {
            "input" : "arg1",
            "to_callsite" : {
                "call_on_php_function" : {
                    "name" : "eio_chown",
                    "namespace" : ""
                },
            }
        }
    },
    {
        "sink_for_checker" : "PATH_MANIPULATION",
        "sink" : {
            "input" : "arg1",
            "to_callsite" : {
                "call_on_php_function" : {
                    "name" : "eio_link",
                    "namespace" : ""
                },
            }
        }
```

```
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
            "call_on_php_function" : {
                "name" : "eio_link",
                "namespace" : ""
            },
        }
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_php_function" : {
                "name" : "eio_lstat",
                "namespace" : ""
            },
        }
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_php_function" : {
                "name" : "eio_mkdir",
                "namespace" : ""
            },
        }
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_php_function" : {
                "name" : "eio_mknod",
                "namespace" : ""
            },
        }
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
```

```
            "call_on_php_function" : {
               "name" : "eio_open",
               "namespace" : ""
            },
         }
      }
   },
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on_php_function" : {
               "name" : "eio_readdir",
               "namespace" : ""
            },
         }
      }
   },
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on_php_function" : {
               "name" : "eio_readlink",
               "namespace" : ""
            },
         }
      }
   },
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on_php_function" : {
               "name" : "eio_rename",
               "namespace" : ""
            },
         }
      }
   },
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg2",
         "to_callsite" : {
            "call_on_php_function" : {
               "name" : "eio_rename",
               "namespace" : ""
            },
         }
      }
```

```
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_php_function" : {
                "name" : "eio_rmdir",
                "namespace" : ""
            },
        }
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_php_function" : {
                "name" : "eio_stat",
                "namespace" : ""
            },
        }
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_php_function" : {
                "name" : "eio_statvfs",
                "namespace" : ""
            },
        }
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_php_function" : {
                "name" : "eio_symlink",
                "namespace" : ""
            },
        }
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "input" : "arg2",
        "to_callsite" : {
```

```
            "call_on_php_function" : {
               "name" : "eio_symlink",
               "namespace" : ""
            },
         }
      }
   },
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on_php_function" : {
               "name" : "eio_truncate",
               "namespace" : ""
            },
         }
      }
   },
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on_php_function" : {
               "name" : "eio_unlink",
               "namespace" : ""
            },
         }
      }
   },
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "input" : "arg1",
         "to_callsite" : {
            "call_on_php_function" : {
               "name" : "eio_utime",
               "namespace" : ""
            },
         }
      }
   },
   // TODO
   // Bug 110040 - New checker needed to model Authorization bypass.
   // Directive page = https://codiscope.atlassian.net/wiki/spaces/GUID/pages/127354145/Plugin+EIO
   // 1.
   // resource eio_chown ( string $path , int $uid [, int $gid = -1 [, int $pri = EIO_PRI_DEFAULT , callable $callback = NULL [, mixed $data = NULL ]]] )
   //                                    ^^^   and   ^^^ <- cannot be in user's control
   // 2.
   // resource eio_fchown ( mixed $fd , int $uid [, int $gid = -1 [, int $pri = EIO_PRI_DEFAULT [, callable $callback = NULL [, mixed $data = NULL ]]] )
```

```
  //                              ^^^   and   ^^^ <- cannot be in user's control
  // 3.
--
      "sink_for_checker": "SENSITIVE_DATA_LEAK",
      "sink_kind" : "ui",
       "sink": {
          "to_callsite": {
             "call_on_php_function": {
                "name": "print_r"
             },
             "when" : {
                "only_if_arg_index" : 1,
                "is_last_arg" : true
             }
          },
          "input": "arg1"
       }
    },
   {
      "sink_for_checker": "XSS",
      "sink": {
          "to_callsite": {
             "call_on_php_function": {
                "name": "print_r"
             },
             "when" : {
                "only_if_arg_index" : 1,
                "is_last_arg" : true
             }
          },
          "input": "arg1"
       }
    },
    {
       "dataflow_through_callsite" : {
          "call_on_php_function": {
             "name": "print_r"
          },
          "when" : {
             "only_if_arg_index" : 2,
--
       sink_for_checker : "PATH_MANIPULATION",
       sink : {
          input : "arg3",
          to_callsite : {
             call_on_php_constructor : {
                class_name : "FilesystemCache",
                namespace: "Symfony\\Component\\Cache\\Simple"
             },
          }
       }
    },
    {
       sink_for_checker : "PATH_MANIPULATION",
```

```
        sink : {
            input : "arg3",
            to_callsite : {
                call_on_php_constructor : {
                    class_name : "PhpFilesCache",
                    namespace: "Symfony\\Component\\Cache\\Simple"
                },
            }
        }
    },
    {
        sink_for_checker : "PATH_MANIPULATION",
        sink : {
            input : "arg3",
            to_callsite : {
                call_on_php_constructor : {
                    class_name : "FilesystemAdapter",
                    namespace: "Symfony\\Component\\Cache\\Adapter"
                },
            }
        }
    },
    {
        sink_for_checker : "HARDCODED_CREDENTIALS",
        sink_kind : "hardcoded_credential_passwd",
        sink : {
            input : "arg1",
            to_callsite : {
                call_on_php_constructor : {
                    class_name : "PdoCache",
                    namespace: "Symfony\\Component\\Cache\\Simple"
                },
            }
        }
    },
    /*** TODO once we support call_on_php_static_method ***/
    // SENSITIVE_DATA_LEAK logging sink:
    // static CacheItem::log(LoggerInterface $logger = null, $message, $context = array())
    // HARDCODED_CREDENTIALS sinks if the constant string actually
    // contains a password (can update HARDCODED_CREDENTIALS to
    // use a regex to check for this).
    // static Redis|Client createConnection($dsn, array $options = array())
    // static ApcuAdapter::createConnection($dsn, array $options = array())
    // static DoctrineAdapter::createConnection($dsn, array $options = array())
--
    //     sink_for_checker : "OPEN_REDIRECT",
    //     sink : {
    //         input : "arg1",
    //         to_callsite : {
    //             call_on_php_constructor : {
    //                 class_name : "RedirectResponse",
    //                 namespace: "Symfony\\Component\\HttpFoundation"
    //             },
    //         }
```

```
        //     }
        // },
        // {
        //     sink_for_checker : "OPEN_REDIRECT",
        //     sink : {
        //         input : "arg1",
        //         to_callsite : {
        //             call_on_php_instance_method : {
        //                 name : "setTargetUrl",
        //                 receiver_object_is_class : {
        //                     class_name : "RedirectResponse",
        //                     namespace: "Symfony\\Component\\HttpFoundation"
        //                 }
        //             },
        //         }
        //     }
        // },
////////////// XSS
        {
            sink_for_checker : "XSS",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_instance_method : {
                        name : "setCallback",
                        receiver_object_is_class : {
                            class_name : "JsonResponse",
                            namespace: "Symfony\\Component\\HttpFoundation"
                        }
                    },
                }
            }
        },
        {
            sink_for_checker : "XSS",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_instance_method : {
                        name : "setContent",
                        receiver_object_is_class : {
                            class_name : "StreamedResponse",
                            namespace: "Symfony\\Component\\HttpFoundation"
                        }
                    },
                }
            }
        },
        // The guidance material proposes treating the callback as
        // SCRIPT_CODE_INJECTION sinks as well but mentions that they
        // are more likely to show up as XSS than arbitrary script
        // code injection. For this not adding SCRIPT_CODE_INJECTION
        // sinks. See:
        // https://codiscope.atlassian.net/wiki/spaces/GUID/pages/128811494/Plugin+HttpFoun
```

原文链接：coverity 部分规则提取

```
ation
        {
            sink_for_checker : "XSS",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_instance_method : {
                        name : "setCallback",
                        receiver_object_is_class : {
                            class_name : "StreamedResponse",
                            namespace: "Symfony\\Component\\HttpFoundation"
                        }
                    },
                }
            }
        },
        {
            sink_for_checker : "XSS",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_constructor : {
                        class_name : "Response",
                        namespace: "Symfony\\Component\\HttpFoundation"
                    },
                }
            }
        },
        {
            sink_for_checker : "XSS",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_constructor : {
                        class_name : "StreamedResponse",
                        namespace: "Symfony\\Component\\HttpFoundation"
                    },
                }
            }
        },
 ////////////// HEADER_INJECTION
        // The following sinks using any_element do not work (see BZ 111700).
        // BinaryFileResponse
        {
            sink_for_checker : "HEADER_INJECTION",
            sink : {
                input : "arg3",
                path: [ { any_element: true } ],
                to_callsite : {
                    call_on_php_constructor : {
                        class_name : "BinaryFileResponse",
                        namespace: "Symfony\\Component\\HttpFoundation"
                    },
                }
```

```
    }
},
// JsonResponse
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg3",
        path: [ { any_element: true } ],
        to_callsite : {
            call_on_php_constructor : {
                class_name : "JsonResponse",
                namespace: "Symfony\\Component\\HttpFoundation"
            },
        }
    }
},
// setVary can take either an array or a string so adding two
// sink directives, one that sinks the argument and one that
// sinks any_element.
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg1",
        path: [ { any_element: true } ],
        to_callsite : {
            call_on_php_instance_method : {
                name : "setVary",
                receiver_object_is_class : {
                    class_name : "JsonResponse",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_instance_method : {
                name : "setVary",
                receiver_object_is_class : {
                    class_name : "JsonResponse",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg1",
```

```
        to_callsite : {
            call_on_php_instance_method : {
                name : "setEtag",
                receiver_object_is_class : {
                    class_name : "JsonResponse",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
// RedirectResponse
        {
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg3",
        path: [ { any_element: true } ],
        to_callsite : {
            call_on_php_constructor : {
                class_name : "RedirectResponse",
                namespace: "Symfony\\Component\\HttpFoundation"
            },
        }
    }
},
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg1",
        path: [ { any_element: true } ],
        to_callsite : {
            call_on_php_instance_method : {
                name : "setVary",
                receiver_object_is_class : {
                    class_name : "RedirectResponse",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_instance_method : {
                name : "setVary",
                receiver_object_is_class : {
                    class_name : "RedirectResponse",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
```

```
    }
},
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_instance_method : {
                name : "setEtag",
                receiver_object_is_class : {
                    class_name : "RedirectResponse",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
// Response
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg3",
        path: [ { any_element: true } ],
        to_callsite : {
            call_on_php_constructor : {
                class_name : "Response",
                namespace: "Symfony\\Component\\HttpFoundation"
            },
        }
    }
},
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg1",
        path: [ { any_element: true } ],
        to_callsite : {
            call_on_php_instance_method : {
                name : "setVary",
                receiver_object_is_class : {
                    class_name : "Response",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_instance_method : {
                name : "setVary",
```

```
                receiver_object_is_class : {
                    class_name : "Response",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_instance_method : {
                name : "setEtag",
                receiver_object_is_class : {
                    class_name : "Response",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
// StreamedResponse
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg3",
        path: [ { any_element: true } ],
        to_callsite : {
            call_on_php_constructor : {
                class_name : "StreamedResponse",
                namespace: "Symfony\\Component\\HttpFoundation"
            },
        }
    }
},
{
    sink_for_checker : "HEADER_INJECTION",
    sink : {
        input : "arg1",
        path: [ { any_element: true } ],
        to_callsite : {
            call_on_php_instance_method : {
                name : "setVary",
                receiver_object_is_class : {
                    class_name : "StreamedResponse",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
{
```

```
            sink_for_checker : "HEADER_INJECTION",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_instance_method : {
                        name : "setVary",
                        receiver_object_is_class : {
                            class_name : "StreamedResponse",
                            namespace: "Symfony\\Component\\HttpFoundation"
                        }
                    },
                }
            }
        },
        {
            sink_for_checker : "HEADER_INJECTION",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_instance_method : {
                        name : "setEtag",
                        receiver_object_is_class : {
                            class_name : "StreamedResponse",
                            namespace: "Symfony\\Component\\HttpFoundation"
                        }
                    },
                }
            }
        },
////////////// PATH_MANIPULATION
        // BinaryFileResponse
        {
            sink_for_checker : "PATH_MANIPULATION",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_constructor : {
                        class_name : "BinaryFileResponse",
                        namespace: "Symfony\\Component\\HttpFoundation"
                    },
                }
            }
        },
        {
            sink_for_checker : "PATH_MANIPULATION",
            sink : {
                input : "arg1",
                to_callsite : {
                    call_on_php_instance_method : {
                        name : "setFile",
                        receiver_object_is_class : {
                            class_name : "BinaryFileResponse",
                            namespace: "Symfony\\Component\\HttpFoundation"
                        }
```

```
            },
        }
    }
},
// File
{
    sink_for_checker : "PATH_MANIPULATION",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_constructor : {
                class_name : "File",
                namespace: "Symfony\\Component\\HttpFoundation"
            },
        }
    }
},
{
    sink_for_checker : "PATH_MANIPULATION",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_instance_method : {
                name : "move",
                receiver_object_is_class : {
                    class_name : "File",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
{
    sink_for_checker : "PATH_MANIPULATION",
    sink : {
        input : "arg2",
        to_callsite : {
            call_on_php_instance_method : {
                name : "move",
                receiver_object_is_class : {
                    class_name : "File",
                    namespace: "Symfony\\Component\\HttpFoundation"
                }
            },
        }
    }
},
// FileStream
{
    sink_for_checker : "PATH_MANIPULATION",
    sink : {
        input : "arg1",
        to_callsite : {
            call_on_php_constructor : {
```

```
            class_name : "FileStream",
            namespace: "Symfony\\Component\\HttpFoundation"
        },
    }
  }
},
{
  sink_for_checker : "PATH_MANIPULATION",
  sink : {
    input : "arg1",
    to_callsite : {
      call_on_php_instance_method : {
        name : "move",
        receiver_object_is_class : {
          class_name : "FileStream",
          namespace: "Symfony\\Component\\HttpFoundation"
        }
      },
    }
  }
},
{
  sink_for_checker : "PATH_MANIPULATION",
  sink : {
    input : "arg2",
    to_callsite : {
      call_on_php_instance_method : {
        name : "move",
        receiver_object_is_class : {
          class_name : "FileStream",
          namespace: "Symfony\\Component\\HttpFoundation"
        }
      },
    }
  }
},
// UploadedFile
{
  sink_for_checker : "PATH_MANIPULATION",
  sink : {
    input : "arg1",
    to_callsite : {
      call_on_php_constructor : {
        class_name : "UploadedFile",
        namespace: "Symfony\\Component\\HttpFoundation"
      },
    }
  }
},
{
  sink_for_checker : "PATH_MANIPULATION",
  sink : {
    input : "arg1",
    to_callsite : {
```

```
                call_on_php_instance_method : {
                    name : "move",
                    receiver_object_is_class : {
                        class_name : "UploadedFile",
                        namespace: "Symfony\\Component\\HttpFoundation"
                    }
                },
            }
        }
    },
    {
        sink_for_checker : "PATH_MANIPULATION",
        sink : {
            input : "arg2",
            to_callsite : {
                call_on_php_instance_method : {
                    name : "move",
                    receiver_object_is_class : {
                        class_name : "UploadedFile",
                        namespace: "Symfony\\Component\\HttpFoundation"
                    }
                },
            }
        }
    },
    // TODO: file system sources as well as related dataflow:
    // File::__construct(string $path, bool $checkPath = true)
    // FileStream::__construct(string $path, bool $checkPath = true)
    // UploadedFile::__construct(string $path, string $originalName, string|null $mimeType =
null,
    // TODO: when we support static methods:
    // static Response RedirectResponse::create($url = '', int $status = 302, array $headers =
rray())
    // static Response BinaryFileResponse::create(SplFileInfo|string $file = null, int $status =
00, array $headers = array(), bool $public = true, null|string $contentDisposition = null, bool
autoEtag = false, bool $autoLastModified = true)
--
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on_python_function" : "eval",
        "module" : "__builtin__",
      }
    }
  },
  {
    "sink_for_checker" : "SCRIPT_CODE_INJECTION",
    "sink" : {
      "input" : "arg1",
      "to_callsite" : {
        "call_on_python_function" : "compile",
        "module" : "__builtin__",
      }
```

```
      }
    },
    {
       "sink_for_checker": "SENSITIVE_DATA_LEAK",
       "sink_kind" : "logging",
       "sink": {
          "to_callsite": {
             "call_on_python_function": "print",
             "module": "__builtin__",
          },
          "input": "all_args",
       }
    },
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 10,
  "language" : "Python",
  "directives" : [
  {
     // "sink_for_checker" : "SCRIPT_CODE_INJECTION"
     "taint_kind": "environment",
     "tainted_data": {
        "read_from_python_module": "os",
        "path": [ { "element": "environ" }]
     },
     "is_deep_taint" : true
  },
  {
     "taint_kind": "environment",
     "tainted_data": {
        "from_callsite": {
           "call_on_python_function": "getenv",
           "module": "os",
        },
        "output": "return"
     }
  },
  {
     "taint_kind": "platform",
     "tainted_data": {
--
     "sink_for_checker": "OS_CMD_INJECTION",
     "sink": {
        "to_callsite": {
           "call_on_python_function": "execl",
           "module": "os",
        },
        "input": "all_args",
     }
  },
  {
     "sink_for_checker": "OS_CMD_INJECTION",
     "sink": {
```

原文链接：coverity 部分规则提取

```
        "to_callsite": {
            "call_on_python_function": "execlp",
            "module": "os",
        },
        "input": "all_args",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "execle",
            "module": "os",
        },
        "input": "all_args",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "execlpe",
            "module": "os",
        },
        "input": "all_args",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "execv",
            "module": "os",
        },
        "input": "all_args",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "execve",
            "module": "os",
        },
        "input": "all_args",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "execvp",
            "module": "os",
        },
```

```
        "input": "all_args",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "execvpe",
            "module": "os",
        },
        "input": "all_args",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "spawnl",
            "module": "os",
        },
        "input": "from_arg2",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "spawnle",
            "module": "os",
        },
        "input": "from_arg2",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "spawnlp",
            "module": "os",
        },
        "input": "from_arg2",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "spawnlpe",
            "module": "os",
        },
        "input": "from_arg2",
    }
},
{
```

```
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "spawnv",
            "module": "os",
        },
        "input": "from_arg2",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "spawnve",
            "module": "os",
        },
        "input": "from_arg2",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "spawnvp",
            "module": "os",
        },
        "input": "from_arg2",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "spawnvpe",
            "module": "os",
        },
        "input": "from_arg2",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "system",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker": "OS_CMD_INJECTION",
    "sink": {
        "to_callsite": {
            "call_on_python_function": "startfile",
```

```
            "module": "os",
        },
        "input": "all_args",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "chdir",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "open",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "chroot",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "chmod",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "chown",
            "module": "os",
        },
        "input": "arg1",
    }
```


原文链接：coverity 部分规则提取

```
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "lchmod",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "lchown",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "link",
            "module": "os",
        },
        "input": "all_args",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "listdir",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "lstat",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
```

```
        "to_callsite" : {
            "call_on_python_function": "mkfifo",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "mknod",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "mkdir",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "makedirs",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "remove",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "removedirs",
            "module": "os",
        },
```

```
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "rename",
            "module": "os",
        },
        "input": "all_args",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "renames",
            "module": "os",
        },
        "input": "all_args",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "rmdir",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "stat",
            "module": "os",
        },
        "input": "arg1",
    }
},
{
    "sink_for_checker" : "PATH_MANIPULATION",
    "sink" : {
        "to_callsite" : {
            "call_on_python_function": "symlink",
            "module": "os",
        },
        "input": "all_args",
    }
},
{
```

```
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "to_callsite" : {
            "call_on_python_function": "unlink",
            "module": "os",
         },
         "input": "arg1",
      }
   },
   {
      "sink_for_checker" : "PATH_MANIPULATION",
      "sink" : {
         "to_callsite" : {
            "call_on_python_function": "tempnam",
            "module": "os",
         },
         "input": "arg1",
      }
   },
   // TODO
   // Bug 110040 - New checker needed to model Authorization bypass.
   // Directive page = https://codiscope.atlassian.net/wiki/spaces/GUID/pages/125927595/Pl
gin+os
   // TODO
   // Bug 110041 - New checker needed to capture defects which can cause DoS.
   // Directive page - https://codiscope.atlassian.net/wiki/spaces/GUID/pages/125927595/Plu
in+os
   // TODO
   // Bug 110042 - New checker needed for File injection.
   // Directive page - https://codiscope.atlassian.net/wiki/spaces/GUID/pages/125927595/Plu
in+os
   {
      "dataflow_through_callsite" : {
         "call_on_python_function": "getenv",
--
     "sink_for_checker" : "OS_CMD_INJECTION",
     "sink" : {
       "to_callsite" : {
         "call_on_python_function": "call",
         "module": "subprocess",
       },
       "input": "arg1",
     }
   },
   {
     "sink_for_checker" : "OS_CMD_INJECTION",
     "sink" : {
       "to_callsite" : {
         "call_on_python_function": "check_call",
         "module": "subprocess",
       },
       "input": "arg1",
     }
   },
```

```
    {
      "sink_for_checker" : "OS_CMD_INJECTION",
      "sink" : {
        "to_callsite" : {
          "call_on_python_function": "check_output",
          "module": "subprocess",
        },
        "input": "arg1",
      }
    },
    {
      "sink_for_checker" : "OS_CMD_INJECTION",
      "sink" : {
        "to_callsite" : {
          "call_on_python_function": "Popen",
          "module": "subprocess",
        },
        "input": "arg1",
      }
    },
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 10,
    "language" : "Python",
    "directives" : [
      {
        "sink_for_checker" : "UNSAFE_DESERIALIZATION",
        "sink" : {
          "input" : "arg1",
          "to_callsite" : {
            "call_on_python_function" : "loads",
            "module" : "pickle",
          }
        }
      },
      {
        "sink_for_checker" : "UNSAFE_DESERIALIZATION",
        "sink" : {
          "input" : "arg1",
          "to_callsite" : {
            "call_on_python_function" : "loads",
            "module" : "cPickle",
          }
        }
      },
    ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 10,
    "language" : "Python",
    "directives" : [
      {
        "sink_for_checker" : "UNSAFE_DESERIALIZATION",
        "sink" : {
```

```
        "input" : "arg1",
        "to_callsite" : {
          "call_on_python_function" : "load",
          "module" : "yaml",
        }
      }
    },
  ]
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 10,
  "language" : "Python",
  "directives" : [
   //****************** types
   // [class django.http.HttpRequest].session -> [type django.contrib.sessions.backends.base.
essionBase]
   {
     "class" : {
       "class_name": "SessionBase",
       "module": "django.contrib.sessions.backends.base",
--
      "sink_for_checker": "XSS",
      "sink": {
        "to_callsite" : {
          "call_on_python_class": {
            "class_name": "HttpResponse",
            "module": "django.http",
          }
        },
        "input": "pos:1:keyword:content"
      }
  },
  // sink: [type HttpResponse](content=/*taint*/, content_type=/*taint*/)
  {
      "sink_for_checker": "XSS",
      "sink": {
        "to_callsite" : {
          "call_on_python_class": {
            "class_name": "HttpResponse",
            "module": "django.http",
          }
        },
        "input": "pos:2:keyword:content_type"
      }
  },
  // sink: [type HttpResponse].write(/*taint*/)
  {
      "sink_for_checker": "XSS",
      "sink": {
        "to_callsite" : {
          "call_on_python_method" : "write",
          "class": {
            "class_name": "HttpResponse",
            "module": "django.http",
```

```
            }
        },
        "input": "arg2",
    }
},
// sink: [type HttpResponse].set_cookie(/*taint key*/, value = /*taint value*/, ...)
{
    "sink_for_checker": "HEADER_INJECTION",
    "sink": {
        "to_callsite" : {
            "call_on_python_method" : "set_cookie",
            "class": {
                "class_name": "HttpResponse",
                "module": "django.http",
            }
        },
        "input": "arg2",
    }
},
// sink: [type HttpResponse].set_cookie(/*taint key*/, value = /*taint value*/, ...)
{
    "sink_for_checker": "HEADER_INJECTION",
    "sink": {
        "to_callsite" : {
            "call_on_python_method" : "set_cookie",
            "class": {
                "class_name": "HttpResponse",
                "module": "django.http",
            }
        },
        "input": "pos:2:keyword:value",
    }
},
// sink: [type HttpResponse].set_signed_cookie(/*taint key*/, /*taint value*/, salt = /*taint sa
t*/, ...)
{
    "sink_for_checker": "HEADER_INJECTION",
    "sink": {
        "to_callsite" : {
            "call_on_python_method" : "set_signed_cookie",
            "class": {
                "class_name": "HttpResponse",
                "module": "django.http",
            }
        },
        "input": "arg2",
    }
},
// sink: [type HttpResponse].set_signed_cookie(/*taint key*/, /*taint value*/, salt = /*taint sa
t*/, ...)
{
    "sink_for_checker": "HEADER_INJECTION",
    "sink": {
        "to_callsite" : {
```

```
            "call_on_python_method" : "set_signed_cookie",
            "class": {
                "class_name": "HttpResponse",
                "module": "django.http",
            }
        },
        "input": "arg3",
    }
},
// sink: [type HttpResponse].set_signed_cookie(/*taint key*/, /*taint value*/, salt = /*taint sa
t*/, ...)
{
    "sink_for_checker": "HEADER_INJECTION",
    "sink": {
        "to_callsite" : {
            "call_on_python_method" : "set_signed_cookie",
            "class": {
                "class_name": "HttpResponse",
                "module": "django.http",
            }
        },
        "input": "pos:3:keyword:salt",
    }
},
// sink: [type HttpResponse]['header'] = /* taint */
{
    "sink_for_checker": "HEADER_INJECTION",
    "sink": {
        "write_to_object_of_class": {
            "class_name": "HttpResponse",
            "module": "django.http",
        },
        "path" : [ { "property" : "header"} ]
    }
},
// sink: JsonResponse(/*taint*/)
{
    "sink_for_checker": "XSS",
    "sink": {
        "to_callsite": {
            "call_on_python_class": {
                "class_name": "JsonResponse",
                "module": "django.http",
            }
        },
        "input": "arg1",
    }
},
// sink: [type Signer].sign(/*taint*/)
{
    "sink_for_checker": "XSS",
    "sink": {
        "to_callsite" : {
            "call_on_python_method" : "sign",
```

```
            "class": {
                "class_name": "Signer",
                "module": "django.core.signing",
            }
        },
        "input": "arg2",
    }
},
//****************** Signed Response Values (SENSITIVE_SINKS)
// sink: [type HttpRequest].get_signed_cookie(key, default=RAISE_ERROR, salt=/*taint*/, m
x_age=None)
{
    "sink_for_checker": "XSS",
    "sink": {
        "to_callsite" : {
            "call_on_python_method" : "get_signed_cookie",
            "class": {
                "class_name": "HttpRequest",
                "module": "django.http",
            }
        },
        "input": "pos:3:keyword:salt",
    }
},
// sink: [type SessionBase].set_expiry(/*taint*/)
{
    "sink_for_checker": "SENSITIVE_DATA_LEAK",
    "sink_kind": "ui",
    "sink": {
        "to_callsite" : {
            "call_on_python_method" : "set_expiry",
            "class": {
                "class_name": "SessionBase",
                "module": "django.contrib.sessions.backends.base",
            }
        },
        "input": "arg2",
    }
},
// sink: [type SessionStore].set_expiry(/*taint*/)
{
    "sink_for_checker": "SENSITIVE_DATA_LEAK",
    "sink_kind": "ui",
    "sink": {
        "to_callsite" : {
            "call_on_python_method" : "set_expiry",
            "class": {
                "class_name": "SessionStore",
                "module": "django.contrib.sessions.backends.db",
            },
        },
        "input": "arg2",
    }
},
```

```
//****************** OPEN_REDIRECT sinks
// sink: django.shortcuts.redirect(/*taint*/ to, permanent=False, *args, **kwargs)
{
    "sink_for_checker" : "OPEN_REDIRECT",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_python_function": "redirect",
            "module": "django.shortcuts",
        },
    }
},
// sink: class HttpResponseRedirect(/*taint*/)
{
    "sink_for_checker" : "OPEN_REDIRECT",
    "sink" : {
        "input" : "arg1",
        "to_callsite" : {
            "call_on_python_class": {
                "class_name": "HttpResponseRedirect",
                "module": "django.http",
            }
        },
    }
},
// sink: class HttpResponsePermanentRedirect(/*taint*/)
{
    "sink_for_checker": "OPEN_REDIRECT",
    "sink": {
        "to_callsite": {
            "call_on_python_class": {
                "class_name": "HttpResponsePermanentRedirect",
                "module": "django.http",
            }
        },
        "input": "arg1",
    }
},
//****************** Passthroughs
// pass-through: [type SessionBase].get(key, default_value) -> return default_value
{
    "dataflow_through_callsite": {
        "call_on": {
            "read_from_object_of_class": {
                "class_name": "SessionBase",
                "module": "django.contrib.sessions.backends.base",
            },
            "path" : [ { "property" : "get" } ]
--
  sink_for_checker : "XSS",
  sink : {
    to_callsite : { module: "flask", call_on_python_function: "make_response", },
    input : "pos:1:keyword:response_body",
  }
```

原文链接：coverity 部分规则提取

```
},
{
  sink_for_checker : "XSS",
  sink : {
    to_callsite : { call_on_python_class: { module: "flask.wrappers",
                                            class_name: "Response" } },
    input : "arg1",
  }
},
{
  sink_for_checker : "XSS",
  sink : {
    to_callsite: { class: { module: "flask.wrappers",
                            class_name: "Response" },
                   call_on_python_method: "set_data", },
    input : "arg2",
  }
},
{
  sink_for_checker : "XSS",
  sink : {
    to_callsite : { call_on_python_class: { module: "werkzeug.wrappers",
                                            class_name: "BaseResponse" } },
    input : "arg1",
  }
},
{
  sink_for_checker : "XSS",
  sink : {
    to_callsite: { class: { module: "werkzeug.wrappers",
                            class_name: "BaseResponse" },
                   call_on_python_method: "set_data", },
    input : "arg2",
  }
},
{
  sink_for_checker : "XSS",
  sink : {
    to_callsite : { call_on_python_class: { module: "werkzeug.wrappers",
                                            class_name: "Response" } },
    input : "arg1",
  }
},
{
  sink_for_checker : "XSS",
  sink : {
    to_callsite: { class: { module: "werkzeug.wrappers",
                            class_name: "Response" },
                   call_on_python_method: "set_data", },
    input : "arg2",
  }
},
////////////////////////////////////////////////////////////////////
// PATH_MANIPULATION sinks
```

```
    // werkzeug.utils.find_modules(import_path, include_packages=False, recursive=False)
    {
      sink_for_checker : "PATH_MANIPULATION",
      sink : {
        to_callsite : { module: "werkzeug.utils", call_on_python_function: "find_modules" },
        input : "pos:1:keyword:import_path",
      }
    },
    ////////////////////////////////////////////////////////////////
    // SENSITIVE_DATA_LEAK sinks
    // flask.wrappers.Response.set_cookie(key, value='', max_age=None, expires=None, path='/
, domain=None, secure=False, httponly=False)
    // werkzeug.wrappers.BaseResponse.set_cookie(key, value='', max_age=None, expires=No
e, path='/', domain=None, secure=False, httponly=False)
    // werkzeug.wrappers.Response.set_cookie(key, value='', max_age=None, expires=None, p
th='/', domain=None, secure=False, httponly=False)
    {
      sink_for_checker : "SENSITIVE_DATA_LEAK",
      sink_kind : "cookie",
      sink : {
        to_callsite: { class: { module: "flask.wrappers",
                                class_name: "Response" },
                       call_on_python_method: "set_cookie", },
        input : "arg2", // XXX BZ 111784 "pos:2:keyword:key",
      }
    },
    {
      sink_for_checker : "SENSITIVE_DATA_LEAK",
      sink_kind : "cookie",
      sink : {
        to_callsite: { class: { module: "flask.wrappers",
                                class_name: "Response" },
                       call_on_python_method: "set_cookie", },
        input : "keyword:value", // XXX BZ 111784 "pos:3:keyword:value",
      }
    },
    {
      sink_for_checker : "SENSITIVE_DATA_LEAK",
      sink_kind : "cookie",
      sink : {
        to_callsite: { class: { module: "flask.wrappers",
                                class_name: "Response" },
                       call_on_python_method: "set_cookie", },
        input : "keyword:path", // XXX BZ 111784 "pos:6:keyword:path",
      }
    },
    {
      sink_for_checker : "SENSITIVE_DATA_LEAK",
      sink_kind : "cookie",
      sink : {
        to_callsite: { class: { module: "werkzeug.wrappers",
                                class_name: "BaseResponse" },
                       call_on_python_method: "set_cookie", },
        input : "arg2", // XXX BZ 111784 "pos:2:keyword:key",
```

```
  }
},
{
  sink_for_checker : "SENSITIVE_DATA_LEAK",
  sink_kind : "cookie",
  sink : {
    to_callsite: { class: { module: "werkzeug.wrappers",
                            class_name: "BaseResponse" },
                   call_on_python_method: "set_cookie", },
    input : "keyword:value", // XXX BZ 111784 "pos:3:keyword:value",
  }
},
{
  sink_for_checker : "SENSITIVE_DATA_LEAK",
  sink_kind : "cookie",
  sink : {
    to_callsite: { class: { module: "werkzeug.wrappers",
                            class_name: "BaseResponse" },
                   call_on_python_method: "set_cookie", },
    input : "keyword:path", // XXX BZ 111784 "pos:6:keyword:path",
  }
},
{
  sink_for_checker : "SENSITIVE_DATA_LEAK",
  sink_kind : "cookie",
  sink : {
    to_callsite: { class: { module: "werkzeug.wrappers",
                            class_name: "Response" },
                   call_on_python_method: "set_cookie", },
    input : "arg2", // XXX BZ 111784 "pos:2:keyword:key",
  }
},
{
  sink_for_checker : "SENSITIVE_DATA_LEAK",
  sink_kind : "cookie",
  sink : {
    to_callsite: { class: { module: "werkzeug.wrappers",
                            class_name: "Response" },
                   call_on_python_method: "set_cookie", },
    input : "keyword:value", // XXX BZ 111784 "pos:3:keyword:value",
  }
},
{
  sink_for_checker : "SENSITIVE_DATA_LEAK",
  sink_kind : "cookie",
  sink : {
    to_callsite: { class: { module: "werkzeug.wrappers",
                            class_name: "Response" },
                   call_on_python_method: "set_cookie", },
    input : "keyword:path", // XXX BZ 111784 "pos:6:keyword:path",
  }
},
////////////////////////////////////////////////////////////////
// HARDCODED_CREDENTIALS sinks
```

```
    // XXX Add unencrypted sensitive data sinks
    // werkzeug.contrib.securecookie.SecureCookie(data=None, secret_key=None, new=True)
    {
      sink_for_checker : "HARDCODED_CREDENTIALS",
      sink_kind : "hardcoded_credential_crypto",
      sink : {
        to_callsite : { call_on_python_class: { module: "werkzeug.contrib.securecookie",
                                                class_name: "SecureCookie" } },
        input : "pos:2:keyword:secret_key",
      }
    },
    // werkzeug.security.generate_password_hash(password, method=
pbkdf2:sha256
, salt_length=8)
    // werkzeug.security.check_password_hash(pwhash, password)
    // werkzeug.security.pbkdf2_hex(data, salt, iterations=50000, keylen=None, hashfunc=Non
)
    // werkzeug.security.pbkdf2_bin(data, salt, iterations=50000, keylen=None, hashfunc=None

    {
      sink_for_checker : "HARDCODED_CREDENTIALS",
      sink_kind : "hardcoded_credential_passwd",
      sink : {
        to_callsite: { module: "werkzeug.security",
                       call_on_python_function: "generate_password_hash", },
        input : "pos:1:keyword:password",
      }
    },
    {
      sink_for_checker : "HARDCODED_CREDENTIALS",
      sink_kind : "hardcoded_credential_passwd",
      sink : {
        to_callsite: { module: "werkzeug.security",
                       call_on_python_function: "check_password_hash", },
        input : "pos:2:keyword:password",
      }
    },
    {
      sink_for_checker : "HARDCODED_CREDENTIALS",
      sink_kind : "hardcoded_credential_passwd",
      sink : {
        to_callsite: { module: "werkzeug.security",
                       call_on_python_function: "pbkdf2_hex", },
        input : "pos:1:keyword:data",
      }
    },
    {
      sink_for_checker : "HARDCODED_CREDENTIALS",
      sink_kind : "hardcoded_credential_passwd",
      sink : {
        to_callsite: { module: "werkzeug.security",
                       call_on_python_function: "pbkdf2_bin", },
        input : "pos:1:keyword:data",
      }
```

```
    },
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
  "type" : "Coverity analysis configuration",
  "format_version" : 10,
  "language" : "Python",
  "directives" : [
    ////////////////////////////////////////////////////////////////////
    // CouchDB
    {
      sink_for_checker : "NOSQL_QUERY_INJECTION",
      sink : {
        to_callsite : { call_on_python_class: { module: "couchdb.design",
                                                class_name: "ViewDefinition" } },
        input : "arg3",
      }
    },
    {
      sink_for_checker : "NOSQL_QUERY_INJECTION",
      sink : {
        to_callsite : { call_on_python_class: { module: "couchdb.design",
                                                class_name: "ViewDefinition" } },
        input : "pos:4:keyword:reduce_fun",
      }
    },
// Copyright (c) 2017 Synopsys, Inc. All rights reserved worldwide.
    "type" : "Coverity analysis configuration",
    "format_version" : 10,
    "language" : "Swift",
    "directives" : [
    {
        "text_checker_name" : "CONFIG.ATS_INSECURE",
        // Checker description:
        //
        // Looking for Info.plist files where <key>NSAppTransportSecurity<key> tag is and chec
king for various
        // subtags and values since certain setting may allow for
        // insecure connections not being blocked by ATS, allowing for potential MITM attacks.
        //
        // A defect example:
```

sqi注入函数

[> Hibernate

Hibernate native queries

Hibernate criteria

JPA native queries

JPA criteria

JDBC prepared statements

PostgreSQL JDBC

MySQL JDBC

Oracle JDBC

MS SQLServer JDBC

SQLite JDBC

jQuery

System.Data.IDbCommand

System.Data.EntityClient

System.Data.Linq

System.Data.Odbc

System.Data.OleDb

System.Data.OracleClient

System.Data.SqlClient

MySql.Data.MySqlClient

System.Data.SQLite

System.Web.UI.WebControls

Android

--

^(org|net).hibernate.(.)Criteri(a|n)(.)$

^(org|net).hibernate.(.*)createQuery$

^(org|net).hibernate.(.)(Query|Session)(.)$

^(org|net).hibernate.(.)SQL(.)$

^javax.persistence.(.*)createQuery$

^javax.persistence.(.*)createNativeQuery$

^javax.persistence.Criteria(.*)$

^org.springframework.jdbc.
.*)$

^org.springframework.orm.

原文链接：coverity 部分规则提取

ibernate3.(.*)$

^org.springframework.orm.
pa.(.*)$

^org.springframework.orm.
pa.JpaTemplate.(.*)$

^java.lang.Runtime.exec(
ava.lang.String(,|))(.*)$

^java.lang.Runtime.exec(
ava.lang.String[](.*)$

^java.lang.ProcessBuilder(.*)$

^java.lang.ProcessBuilder.(
ava.util.List)java.lang.
rocessBuilder$

^java.lang.ProcessBuilder.command(
ava.util.List)java.lang.
rocessBuilder$

^java.lang.ProcessBuilder.(
ava.lang.String[])java.
ang.ProcessBuilder$

^java.lang.ProcessBuilder.command(
ava.lang.String[])java.
ang.ProcessBuilder$

^System.Data.IDbCommand(.*)$

^System.Data.EntityClient.
.*)$

^System.Data.OracleClient.
.*)$

^System.Data.SqlClient.(
*)$

^System.Web.UI.WebControls.
.*)$

^Microsoft.Practices.EnterpriseLibrary.
ata(.*)$

^MySql.Data.MySqlClient(.*)$

^android.(content|provider|support.v4.
ontent).(.*)$

^android.database.sqlite.

QLiteDatabase(.*)$

^android.database.sqlite.
QLiteQueryBuilder(.*)$

^org.apache.commons.lang(\d+)?(
*)

--

^(org|net).hibernate.(.*)$

^javax.persistence.(.*)$

^java.lang.Runtime(.*)$

^System.Data.Linq.(.*)$

^System.Data.Odbc.(.*)$

^System.Data.OleDb.(.*)$

^System.Data.SQLite.(.*)$

^android.(.*)$](http://)

原文链接：coverity 部分规则提取