链滴

JWT 入门教程

作者：opsxdev

原文链接：https://ld246.com/article/1607357497717

来源网站：链滴

JWT

JSON Web Token 入门教程 http://www.ruanyifeng.com/blog/2018/07/json_web_token-tutoria.html

什么是JWT?

Json web token (JWT), 是为了在网络应用环境间传递声明而执行的一种基于JSON的开放标准（(RFC 519).该token被设计为紧凑且安全的，特别适用于分布式站点的单点登录（SSO）场景。 JWT的声明般被用来在身份提供者和服务提供者间传递被认证的用户身份信息，以便于从资源服务器获取资源，可以增加一些额外的其它业务逻辑所必须的声明信息，该token也可直接被用于认证，也可被加密。

Cookie和Session模式的问题

- Session: 每个用户经过我们的应用认证之后，我们的应用都要在服务端做一次记录，以方便用户下请求的鉴别，通常而言session都是保存在内存中，而随着认证用户的增多，服务端的开销会明显增。
- 扩展性: 用户认证之后，服务端做认证记录，如果认证的记录被保存在内存中的话，这意味着用户次请求还必须要请求在这台服务器上,这样才能拿到授权的资源，这样在分布式的应用上，相应的限制负载均衡器的能力。这也意味着限制了应用的扩展能力。
- CSRF: 因为是基于cookie来进行用户识别的, cookie如果被截获，用户就会很容易受到跨站请求伪的攻击。

基于Token的认证模式

基于Token的认证流程大致上是这样的:

- 用户使用用户名密码来请求服务器
- 服务器进行验证用户的信息
- 服务器通过验证发送给用户一个token

原文链接：JWT 入门教程

- 客户端存储token，并在每次请求时附送上这个token值
- 服务端验证token值，并返回数据

JWT介绍

JWT由三部分组成：　HEADER　PAYLOAD　SIGNATURE

第一部分我们称它为头部（header),第二部分我们称其为载荷（payload, 类似于飞机上承载的物品)
第三部分是签证（signature).

JWT优缺点

无状态和可扩展性：Tokens存储在客户端。完全无状态，可扩展。我们的负载均衡器可以将用户传到任意服务器，因为在任何地方都没有状态或会话信息。

安全：　Token不是Cookie。（The token, not a cookie.）每次请求的时候Token都会被发送。

无法作废已颁布的令牌。所有的认证信息都在JWT中，由于在服务端没有状态，即使你知道了某个JW被盗取了，你也没有办法将其作废。在JWT过期之前（你绝对应该设置过期时间），你无能为力

2.用户校验中间件

```
const (
    ContextUserIDKey = "userID"
    )
    var (
    ErrorUserNotLogin = errors.New("当前用户未登录")
    )
    // JWTAuthMiddleware 基于JWT的认证中间件
    func JWTAuthMiddleware() func(c *gin.Context) {
    return func(c *gin.Context) {
    // 客户端携带Token有三种方方式 1.放在请求头 2.放在请求体 3.放在URI
    // 这□假设Token放在Header的中
    // 这□的具体实现方式要依据你的实际业务情况决定
    authHeader := c.Request.Header.Get("Auth")
    if authHeader == "" {
    ResponseErrorWithMsg(c, CodeInvalidToken, "请求头缺少Auth Token")
    c.Abort()
    return
    }
    mc, err := utils.ParseToken(authHeader)
    if err != nil {
    ResponseError(c, CodeInvalidToken)
    c.Abort()
    return
    }
    // 将当前请求的username信息保存到请求的上下文文c上
    c.Set(ContextUserIDKey, mc.UserID)
    c.Next() // 后续的处理□函数可以用用过c.Get("userID")来获取当前请求的用用户信息
    }
    }
    func GetCurrentUserID(c *gin.Context) (userID uint64, err error) {
    _userID, ok := c.Get(ContextUserIDKey)
```

原文链接：JWT 入门教程

```
    if !ok {
    err = ErrorUserNotLogin
    return
    }
    userID, ok = _userID.(uint64)
    if !ok {
        err = ErrorUserNotLogin
return
}
return
}
```