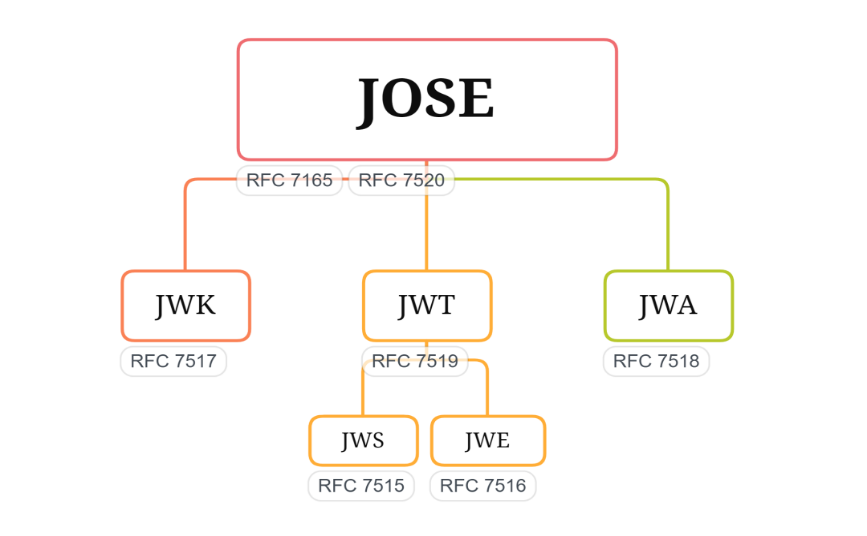
<https://mp.weixin.qq.com/s/1z1Q4NAjF_7-WBI0vH4Y0w>



传统服务端session：

* 需要从内存或数据库里存取 session 数据
* 扩展性差，对于分布式应用，需要实现 session 数据共享
* **JOSE 规范**
* **什么是 JOSE ，它和 JWT 之间又有什么关系呢。**
* JOSE 全称 JSON Object Signing and Encryption ( **RFC 7165[3]** , **RFC 7520[4]** )，它定义了一系列的标准，**用来规范网络传输过程中使用 JSON 的方式**，我们上面一直说的 JWT 其实是 JOSE 体系之一。



其中 JWT 又可分为 JWS 和 JWE 两种不同的实现，**我们大部分日常所使用的，所说的 JWT 其实应该属于 JWS 。**

## JWA 和 JWS 以及 JWK

JWA 的全称是 JSON Web Algorithms ( **RFC 7518[5]** ) ，字如其名， **JOSE 体系中涉及到的所有算法就是它来定义的**，比如通用算法有 Base64-URL 和 SHA，签名算法有 HMAC，RSA 和 Elliptic Curve（EC 椭圆曲线），本文不会深入到算法原理（我也不懂），只是想让你知道 JWA 是做什么的。我们上面的 JWT 例子中第一部分 Header 有个 alg 属性，其值是 HS256 ，也就是 HMAC + SHA256 算法。

说了那么多，好像都没有正式介绍过 JWS 。JWS 的全称是 JSON Web Signature ( **RFC 7515[6]** ) ，它的核心就是**签名**，保证数据未被篡改，而检查签名的过程就叫做**验证**。更通俗的理解，就是对应前面提到的 JWT 的第三部分 Signature ，所以我才会说我们日常所使用的 JWT 都是 JWS 。

通常在客户端-服务端模式中，JWS 使用 JWA 提供的 HS256 算法加上一个密钥即可，这种方式严格依赖密钥，但在分布式场景，可能多个服务都需要验证 JWT ，若要在每个服务里面都保存密钥，那么安全性将会大打折扣，要知道，密钥一旦泄露，任何人都可以随意伪造 JWT 。

解决办法就是使用非对称加密算法 RSA ，RSA 有两把钥匙，一把公钥，一把私钥，可以使用私钥签发（签名分发） JWT ，使用公钥验证 JWT ，公钥是所有人都可以获取到的。这样一来，就只有认证服务保存着私钥，进行签发，其他服务只能验证。

非对称加密算法的解析过程：

1. 读取头部算法信息，确认为非对称算法；
2. 读取载荷中的iss字段，请求JWK回来；
3. 根据JWK中的公钥，进行签名验证

## 另一种 JWT 的实现 ：JWE

我们说过，经过 Signature 签名后的 JWT 就是指的 JWS ，而 JWS 仅仅是对前两部分签名，保证无法篡改，但是其 Payload（载荷） 信息是暴露的（只是作了 base64UrlEncode 处理）。因此，使用 JWS 方式的 Payload 是不适合传递敏感数据的，JWT 的另一种实现 JWE 就是来解决这个问题的。

JWE 全称是 JSON Web Encryption ( **RFC 7516[8]** ) ，JWS 的 Payload 是 Base64Url 的明文，而 **JWE 的数据则是经过加密的。它可以使 JWT 更加安全。**

JWE 提供了两种方案：共享密钥方案和公钥/私钥方案。共享密钥方案的工作原理是让各方都知道一个密钥，大家都可以签名验证，这和 JWS 是一致的。而公钥/私钥方案的工作方式就不同了，在 JWS 中私钥对令牌进行**签名**，持有公钥的各方只能**验证**这些令牌；但在 JWE 中，持有私钥的一方是唯一可以**解密**令牌的一方，公钥持有者可以引入或交换新数据然后重新**加密**，因此，当使用公钥/私钥方案时，JWS 和 JWE 是互补的。

## 安全性考虑

不管怎样，JWT 多多少少还是存在一些安全性隐患的，**下面是平时开发过程的一些建议：**

1. 始终执行算法验证

签名算法的验证固定在后端，不以 JWT 里的算法为标准。假设每次验证 JWT ，验证算法都靠读取 Header 里面的 alg 属性来判断的话，攻击者只要签发一个 "alg: none" 的 JWT ，就可以绕过验证了。

1. 选择合适的算法

具体场景选择合适的算法，例如分布式场景下，建议选择 RS256 。

1. HMAC 算法的密钥安全

除了需要保证密钥不被泄露之外，密钥的强度也应该重视，防止遭到字典攻击。

1. 避免敏感信息保存在 JWT 中

JWS 方式下的 JWT 的 Payload 信息是公开的，不能将敏感信息保存在这里，如有需要，请使用 JWE 。

1. JWT 的有效时间尽量足够短

JWT 过期时间建议设置足够短，过期后重新使用 refresh\_token 刷新获取新的 token 。