TLS 的基本过程

* 客户端发送一个 ClientHello 消息到服务器端，消息中同时包含了它的 Transport Layer Security (TLS) 版本，可用的加密算法和压缩算法。
* 服务器端向客户端返回一个 ServerHello 消息，消息中包含了服务器端的 TLS 版本，服务器所选择的加密和压缩算法，以及数字证书认证机构（Certificate Authority，缩写 CA）签发的服务器公开证书，证书中包含了公钥。客户端会使用这个公钥加密接下来的握手过程，直到协商生成一个新的对称密钥。证书中还包含了该证书所应用的域名范围（Common Name，简称 CN），用于客户端验证身份。
* 客户端根据自己的信任 CA 列表，验证服务器端的证书是否可信。如果认为可信（具体的验证过程在下一节讲解），客户端会生成一串伪随机数，使用服务器的公钥加密它。这串随机数会被用于生成新的对称密钥
* 服务器端使用自己的私钥解密上面提到的随机数，然后使用这串随机数生成自己的对称主密钥
* 客户端发送一个 Finished 消息给服务器端，使用对称密钥加密这次通讯的一个散列值
* 服务器端生成自己的 hash 值，然后解密客户端发送来的信息，检查这两个值是否对应。如果对应，就向客户端发送一个 Finished 消息，也使用协商好的对称密钥加密
* 从现在开始，接下来整个 TLS 会话都使用对称秘钥进行加密，传输应用层（HTTP）内容

TLS 的完整过程需要三个算法（协议），密钥交互算法，对称加密算法，和消息认证算法

我们以 Github 网站使用的 TLS 为例，使用浏览器可以看到它使用的加密为 TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256。其中密钥交互算法是 ECDHE\_RSA，对称加密算法是 AES\_128\_GCM，消息认证（MAC）算法为 SHA256。

**TLS 证书机制**





