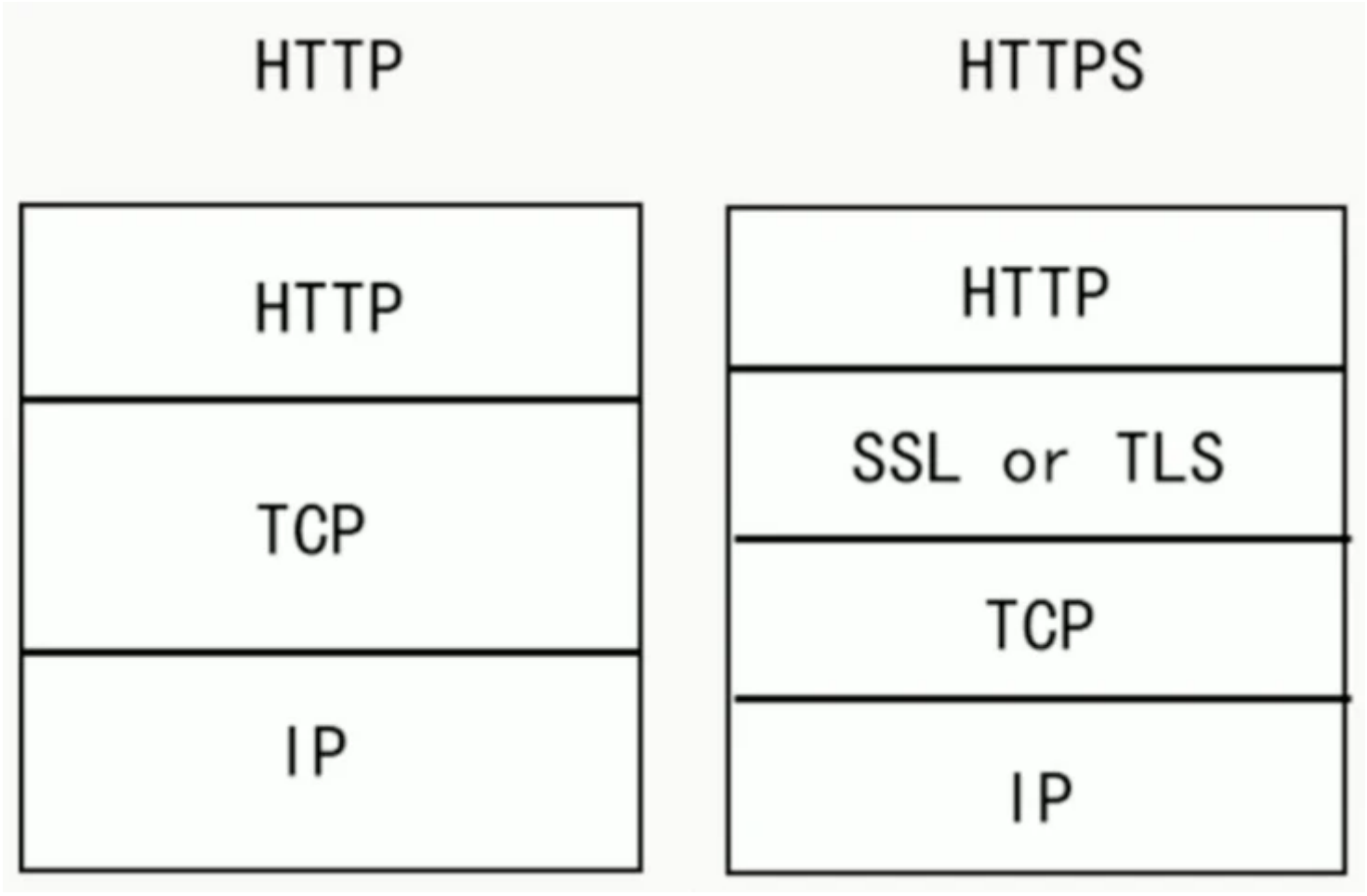
HTTPS详解二：SSL / TLS 工作原理和详细握手过程

前言

在上篇文章HTTPS详解一中，我已经为大家介绍了 HTTPS 的详细原理和通信流程，但总感觉少了点什么，应该是少了对安全层的针对性介绍，那么这篇文章就算是对**HTTPS 详解一**的补充吧。还记得这张图吧。



HTTPS 和 HTTP的区别

显然，HTTPS 相比 HTTP最大的不同就是多了一层 SSL (Secure Sockets Layer 安全套接层)或 TLS (Transport Layer Security 安全传输层协议)。有了这个安全层，就确保了互联网上通信双方的通信安全，那么这个安全层是怎么工作的，SSL / TLS 握手过程又是怎样的呢？本文将对这些问题一一解答。

1、SSL / TLS 以及 SSL / TLS 握手的概念

SSL 和 TLS 协议可以为通信双方提供识别和认证通道，从而保证通信的机密性和数据完整性。TLS 协议是从Netscape SSL 3.0协议演变而来的，不过这两种协议并不兼容，SSL 已经逐渐被 TLS 取代，所以下文就以 TLS 指代安全层。 TLS 握手是启动 HTTPS 通信的过程，类似于 TCP 建立连接时的三次握手。 在 TLS 握手的过程中，通信双方交换消息以相互验证，相互确认，并确立它们所要使用的加密算法以及会话密钥 (用于对称加密的密钥)。可以说，TLS 握手是 HTTPS 通信的基础部分。

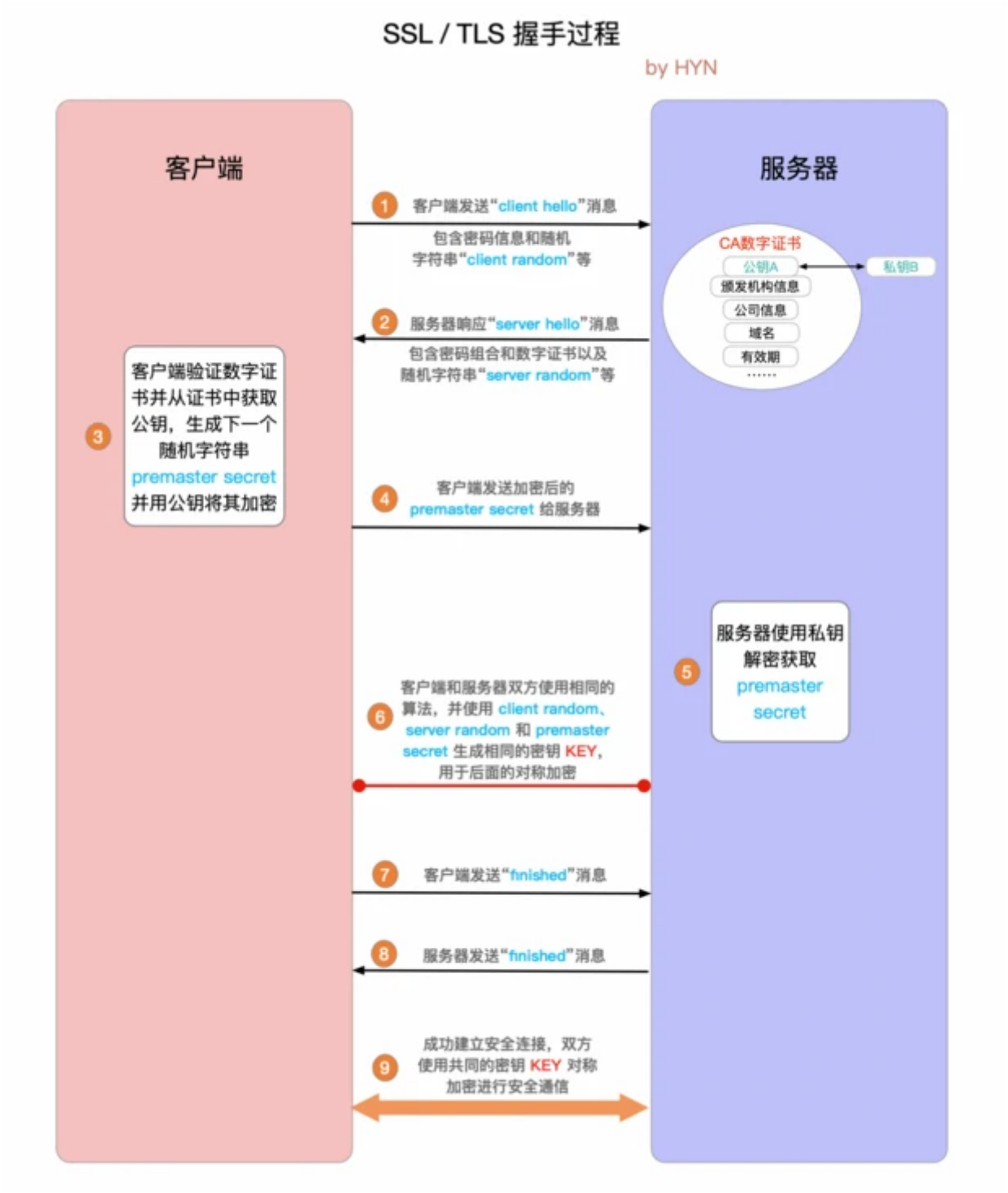
2、TLS 握手过程中发生了什么

我们已经知道 TLS 握手的目的是建立安全连接，那么通信双方在这个过程中究竟干了什么呢？下面就是答案：

- 商定双方通信所使用的的 TLS 版本 (例如 TLS1.0, 1.2, 1.3等等)；
- 确定双方所要使用的密码组合；
- 客户端通过服务器的公钥和数字证书 (上篇文章已有介绍)上的数字签名验证服务端的身份；
- 生成会话密钥，该密钥将用于握手结束后的对称加密。

3、TLS 握手详细过程

下面来看 TLS 握手的详细过程 (**注**：此图与**HTTPS详解一**中的 **HTTPS** 原理图的流程大致相同，不同的是此图把重点放在了**TLS**握手的相关概念上)：



SSL / TLS 握手详细过程

1. **"client hello"消息：**客户端通过发送"client hello"消息向服务器发起握手请求，该消息包含了客户端所支持的 TLS 版本和密码组合以供服务器进行选择，还有一个"client random"随机字符串。
2. **"server hello"消息：**服务器发送"server hello"消息对客户端进行回应，该消息包含了数字证书，服务器选择的密码组合和"server random"随机字符串。
3. **验证：**客户端对服务器发来的证书进行验证，确保对方的合法身份，验证过程可以细化为以下几个步骤：
    1. 检查数字签名
    2. 验证证书链 (这个概念下面会进行说明)
    3. 检查证书的有效期
    4. 检查证书的撤回状态 (撤回代表证书已失效)
4. **"premaster secret"字符串：**客户端向服务器发送另一个随机字符串"premaster secret (预主密钥)"，这个字符串是经过服务器的公钥加密过的，只有对应的私钥才能解密。
5. **使用私钥：**服务器使用私钥解密"premaster secret"。
6. **生成共享密钥**：客户端和服务器均使用 client random，server random 和 premaster secret，并通过相同的算法生成相同的共享密钥 **KEY**。
7. **客户端就绪：**客户端发送经过共享密钥 **KEY**加密过的"finished"信号。
8. **服务器就绪：**服务器发送经过共享密钥 **KEY**加密过的"finished"信号。
9. **达成安全通信：**握手完成，双方使用对称加密进行安全通信。

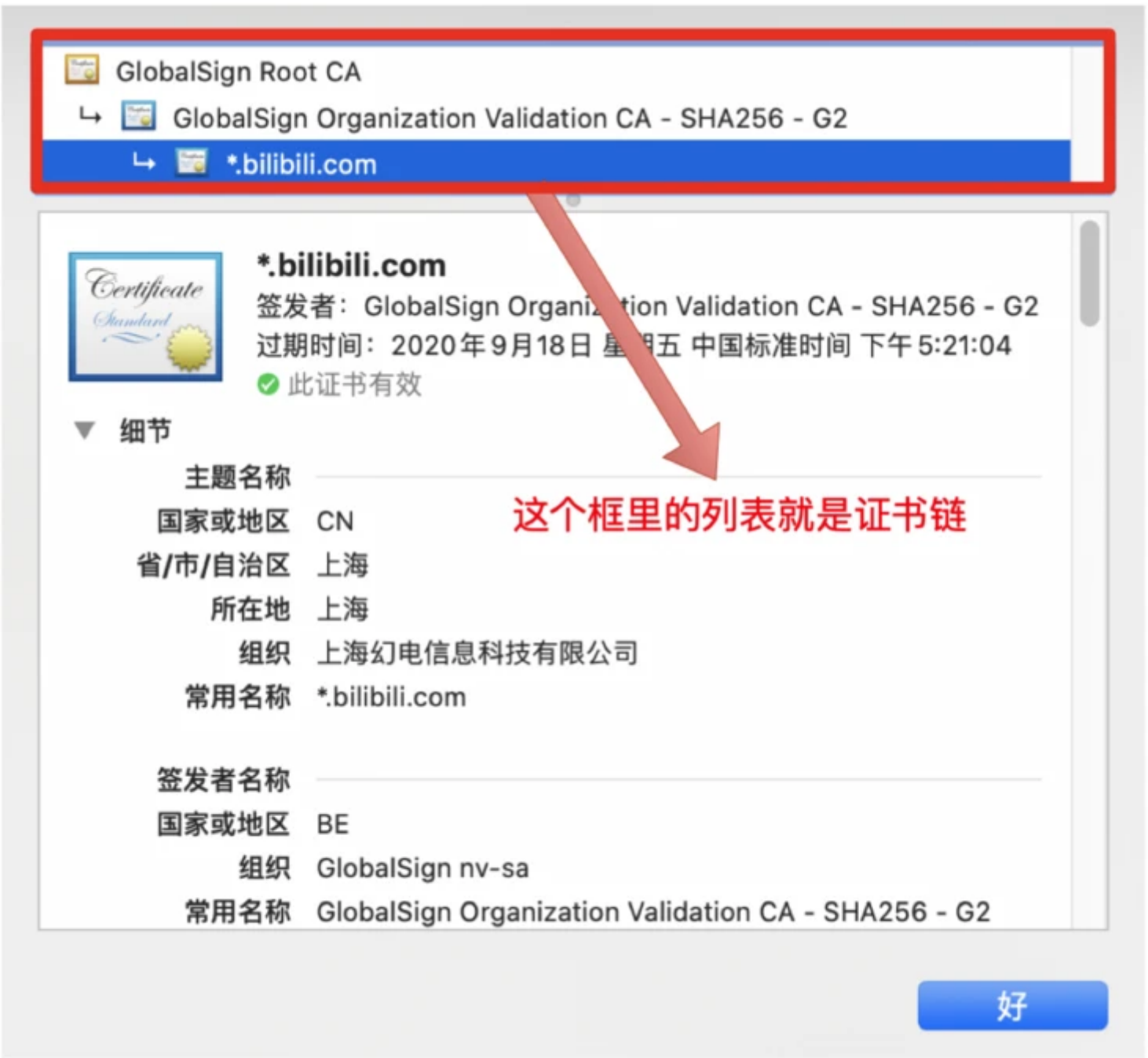
4、TLS 握手过程中的一些重要概念

1. **数字证书 (digital certificate)：**在非对称加密通信过程中，服务器需要将公钥发送给客户端，在这一过程中，公钥很可能会被第三方拦截并替换，然后这个第三方就可以冒充服务器与客户端进行通信，这就是传说中的“中间人攻击”(man in the middle attack)。解决此问题的方法是通过受信任的第三方交换公钥，具体做法就是服务器不直接向客户端发送公钥，而是要求受信任的第三方，也就是证书认证机构 (Certificate Authority, 简称 CA)将公钥合并到数字证书中，然后服务器会把公钥连同证书一起发送给客户端，私钥则由服务器自己保存以确保安全。数字证书一般包含以下内容：
    1. 证书所有者的公钥
    2. 证书所有者的专有名称
    3. 证书颁发机构的专有名称

4. 证书的有效起始日期
5. 证书的过期日期
6. 证书数据格式的版本号
7. 序列号，这是证书颁发机构为该证书分配的唯一标识符

   … …

2. **数字签名 (digital signature)：**这个概念很好理解，其实跟人的手写签名类似，是为了确保数据发送者的合法身份，也可以确保数据内容未遭到篡改，保证数据完整性。与手写签名不同的是，数字签名会随着文本数据的变化而变化。具体到数字证书的应用场景，数字签名的生成和验证流程如下：
   1. 服务器对证书内容进行信息摘要计算 (常用算法有 SHA-256等)，得到摘要信息，再用私钥把摘要信息加密，就得到了数字签名
   2. 服务器把数字证书连同数字签名一起发送给客户端
   3. 客户端用公钥解密数字签名，得到摘要信息
   4. 客户端用相同的信息摘要算法重新计算证书摘要信息，然后对这两个摘要信息进行比对，如果相同，则说明证书未被篡改，否则证书验证失败
3. **证书链 (certificate chain)：**证书链，也称为证书路径，是用于认证实体合法身份的证书列表，具体到 HTTPS 通信中，就是为了验证服务器的合法身份。之所以使用证书链，是为了保证根证书 (root CA certificate)的安全，中间层可以看做根证书的代理，起到了缓冲的作用，如下图所示，这里还以 B 站证书为例：



证书链

证书链从根证书开始，并且证书链中的每一级证书所标识的实体都要为其下一级证书签名，而根证书自身则由证书颁发机构签名。客户端在验证证书链时，必须对链中所有证书的数字签名进行验证，直到达到根证书为止。

4.**密码规范和密码组合 (CipherSpecs 和 CipherSuites)：**通信双方在安全连接中所使用的算法必须符合密码安全协议的规定，CipherSpecs 和 CipherSuites 正好定义了合法的密码算法组合。CipherSpecs 用于认证加密算法和信息摘要算法的组合，通信双方必须同意这个密码规范才能进行通信。而 CipherSuites 则定义了 SSL / TLS 安全连接中所使用的加密算法的组合，该组合包含三种不同的算法：

1. 握手期间所使用的的密钥交换和认证算法 (最常用的是 RSA 算法)
2. 加密算法 (用于握手完成后的对称加密，常用的有 AES、3DES等)
3. 信息摘要算法 (常用的有 SHA-256、SHA-1 和 MD5 等)

4、总结

本文对 SSL / TLS 握手过程进行了详细的解析，也对握手过程中的几个重要概念进行了补充说明。想继续深入了解的小伙伴可以去参考一下 CloudFlare 和 IBM 的官网，这两个网站也是本文内容的主要来源。多逛逛英文网站，不仅可以涨姿势，获取权威信息，还能顺带提升一波英语水平，一举两得，岂不美哉，希望小伙伴们也能尽快养成这样的好习惯。HTTPS 详解部分到此结束，咱们下篇文章再见。

IamHYN

Java开发工程师，前端亦有所涉猎。热衷于技术分享，立志于输出干货...

1.2k 声望 **925** 粉丝

关注作者