讲师介绍

Hash　　QQ：805921455

从事Java软件研发近十年。

前新浪支付核心成员、

咪咕视讯(中国移动)项目经理、

对分布式架构、高性能编程有深入的研究。

明天，你一定会感谢今天奋力拼搏的自己

构建基于Nginx的WEB安全体系，让你的数据更安全

分布式高并发—负载均衡

目录

课程安排

01

数据加密技术简述

从网络协议安全到加密算法，细数关于秘密的事情

02

Https如何保证数据传输安全

关于Https你知道多少？

03

制作你的第一个SSL证书

把数字证书的底全都给抖出来

04

Nginx中实现高性能Https

Https有性能消耗，该怎么调整？

目录

数据加密技术简述

Web应用安全

曾经有个男人不会修自己的电脑，后来的事情大家都知道了

历史中的网络安全事故，都让很多人和公司都站上了舆论风口。

雅虎、百度、网易邮箱都曾经出现过账户泄露的安全事故。

2017年4月，洲际酒店超过1000家旗下酒店遭遇支付卡信息泄露的问题。

2018年8月28，华住旗下多家连锁酒店开房信息的数据在暗网出售。

有人曾因数据泄露婚事泡汤，有人被迫改姓

Facebook因数据泄露，股价大跌

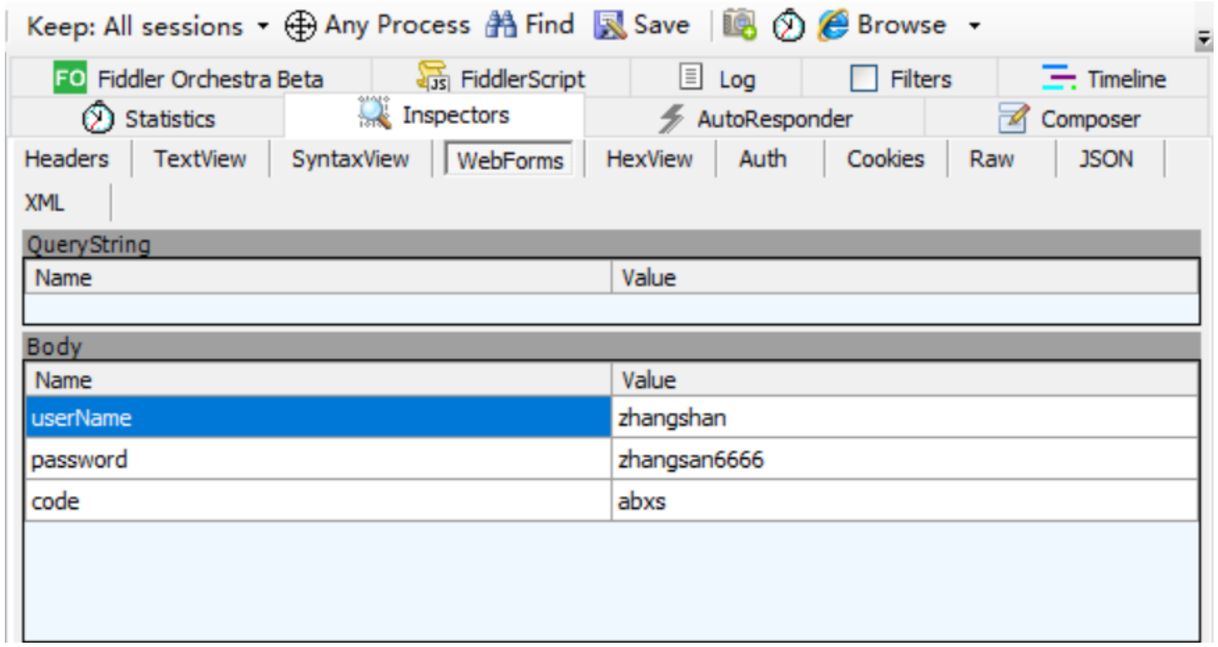
不会打造安全体系的开发者，与不会修自己电脑的男人，没有什么区别！

Http安全如何？

HTTP

HTTP协议，即超文本传输协议(Hypertext transfer protocol)。是一种详细规定了浏览器和万维网(WWW = World Wide Web)服务器之间互相通信的规则，通过因特网传送万维网文档的数据传送协议。

明文传输，容易泄密，容易被拦截



如何解决Http的这个安全问题？——加密技术

摘要算法

摘要算法

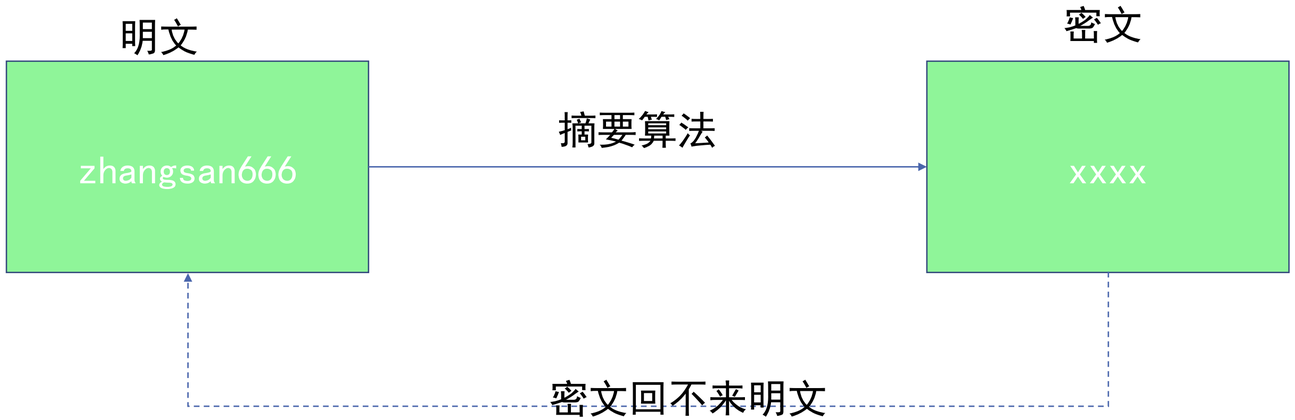
消息摘要算法用来验证数据的完整性，分为三类

MD(Message Digest)：消息摘要

SHA(Secure Hash Algorithm)：安全散列

MAC(Message Authentication Code)：消息认证码

最常见的登录密码加密处理，MD5加密，MD5属于消息摘要类。



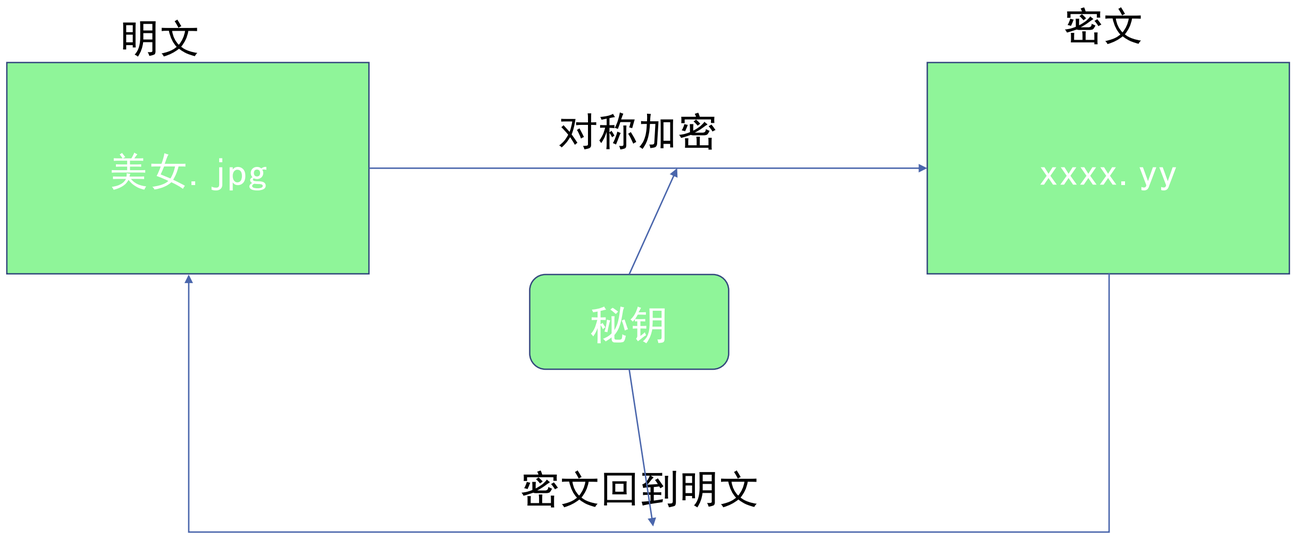
可以用来保证内容不被篡改：密码加密、文件校验码

对称加密

对称加密

DES(Data Encryption Standard，数据加密算法)

PBE(Password-based encryption，基于密码验证)



陈老师当年要是懂对称加密，多好啊！

网络上的加密算法—非对称加密

非对称加密

RSA(算法的名字以发明者的名字命名：Ron Rivest, AdiShamir 和Leonard Adleman)

DH(Diffie-Hellman算法，密钥一致协议)

DSA(Digital Signature Algorithm，数字签名)

ECC(Elliptic Curves Cryptography，椭圆曲线密码编码学)

过程相同，客户端公钥加密，服务端私钥解密

用在不可信的环境下

Https协议

SSL/STL

SSL(Secure Socket Layer 安全套接层)是基于HTTPS下的一个协议加密层。HTTP在传输数据时使用的是明文，是不安全的，为了解决这一隐患网景公司推出了SSL安全套接字协议层，SSL是基于HTTP之下TCP之上的一个协议层，是基于HTTP标准并对TCP传输数据时进行加密，所以HPPTS是HTTP+SSL/TCP的简称。TLS(Transport Layer Security)即安全传输层协议，保障应用程序之间通信的安全性。SSL 3.0版本改名叫TLS，所以TLS是从SSl转过来的。

HTTPS

HTTPS（全称：Hyper Text Transfer Protocol over Secure Socket Layer超文本传输安全协议），是以安全为目标的HTTP通道，简单讲是HTTP的安全版。HTTP下加入SSL层，HTTPS的安全基础是SSL，因此加密的详细内容就需要SSL。

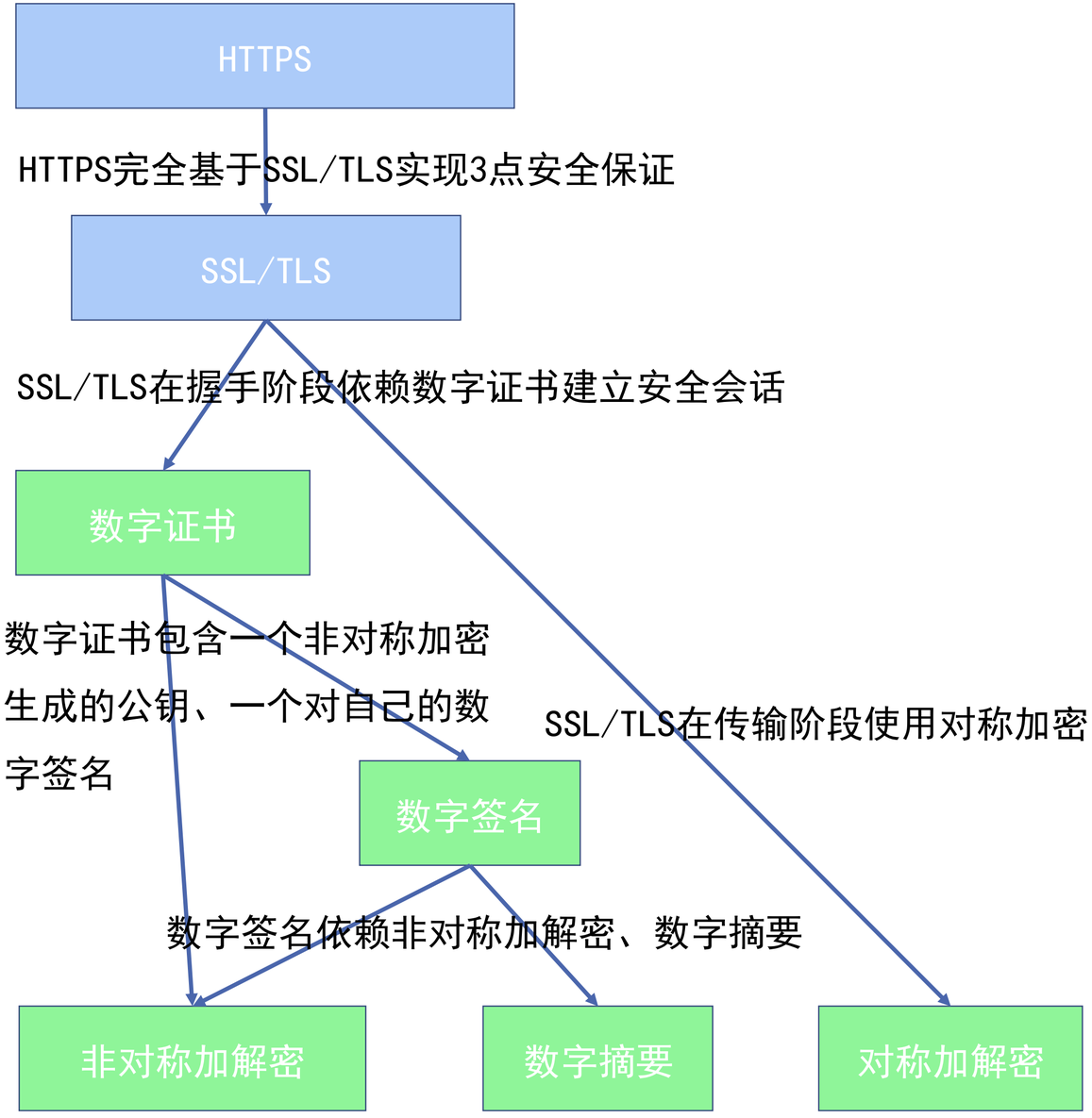
Https协议架构组成

安全保证：

1. **身份认证**，认证用户或服务器，确保数据发送到正确的客户机或服务器

2. **内容加密**，加密数据以防止数据中途被窃取

3. **数据完整性**，维护数据的完整性，确保数据在传输过程中不被篡改



Https与Http的区别

1. https协议需要到CA申请数字证书。

2. http是超文本传输协议，信息是明文传输；https 则是具有安全性的ssl加密传输协议。

3. http和https使用的是完全不同的连接方式，用的端口也不一样，前者是80，后者是443。

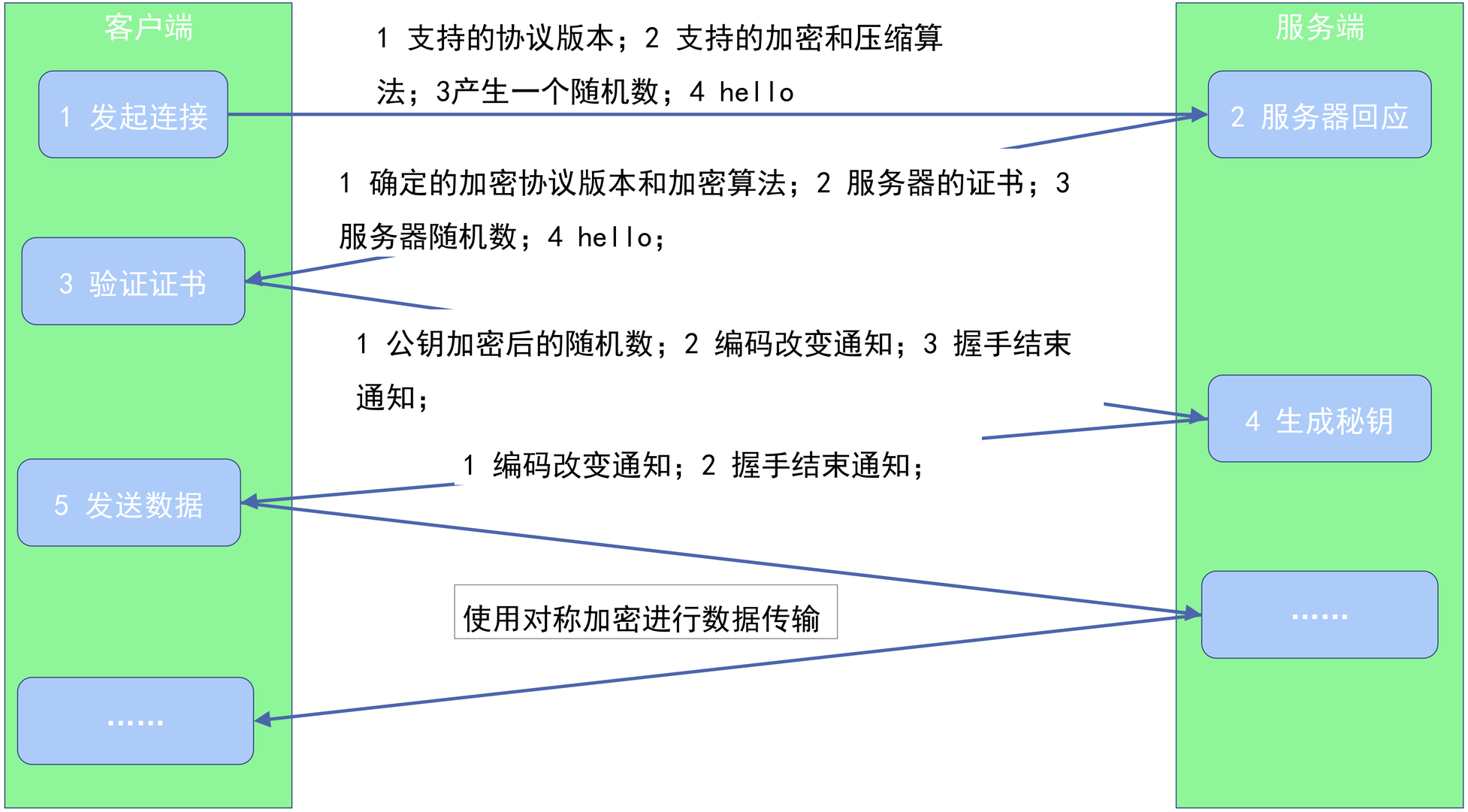
4. http的连接很简单，是无状态的；HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议，比http协议安全

HTTPS对数据进行加解密，决定了它比http慢

目录

Https如何保证数据传输安全

SSL/TLS的握手过程—秘钥协商



Https协议安全保证的原因

注意：

SSL协议在握手阶段使用的是非对称加密，在传输阶段使用的是对称加密，也就是说在SSL上传送的数据是使用对称密钥加密的！因为非对称加密的速度缓慢，耗费资源。

其实当客户端和主机使用非对称加密方式建立连接后，客户端和主机已经决定好了在传输过程使用的对称加密算法和关键的对称加密密钥，由于这个过程本身是安全可靠的，也即对称加密密钥是不可能被窃取盗用的，因此，保证了在传输过程中对数据进行对称加密也是安全可靠的，因为除了客户端和主机之外，不可能有第三方窃取并解密出对称加密密钥！

如果有人窃听通信，他可以知道双方选择的加密方法，以及三个随机数中的两个。整个通话的安全，只取决于第三个随机数能不能被破解。

https实际就是在TCP层与http层之间加入了SSL/TLS来为上层的安全保驾护航，主要用到对称加密、非对称加密、证书，等技术进行客户端与服务器的数据加密传输，最终达到保证整个通信的安全性。

目录

制作你的第一个SSL证书

CA和数字证书

怎么能确定客户端所得到的公钥一定是从目标主机那里发布的，而且没有被篡改过呢？

需要有一个权威的值得信赖的第三方机构CA(一般是由政府审核并授权的机构)来统一对外发放主机机构的公钥，只要请求方这种机构获取公钥，就避免了上述问题的发生。

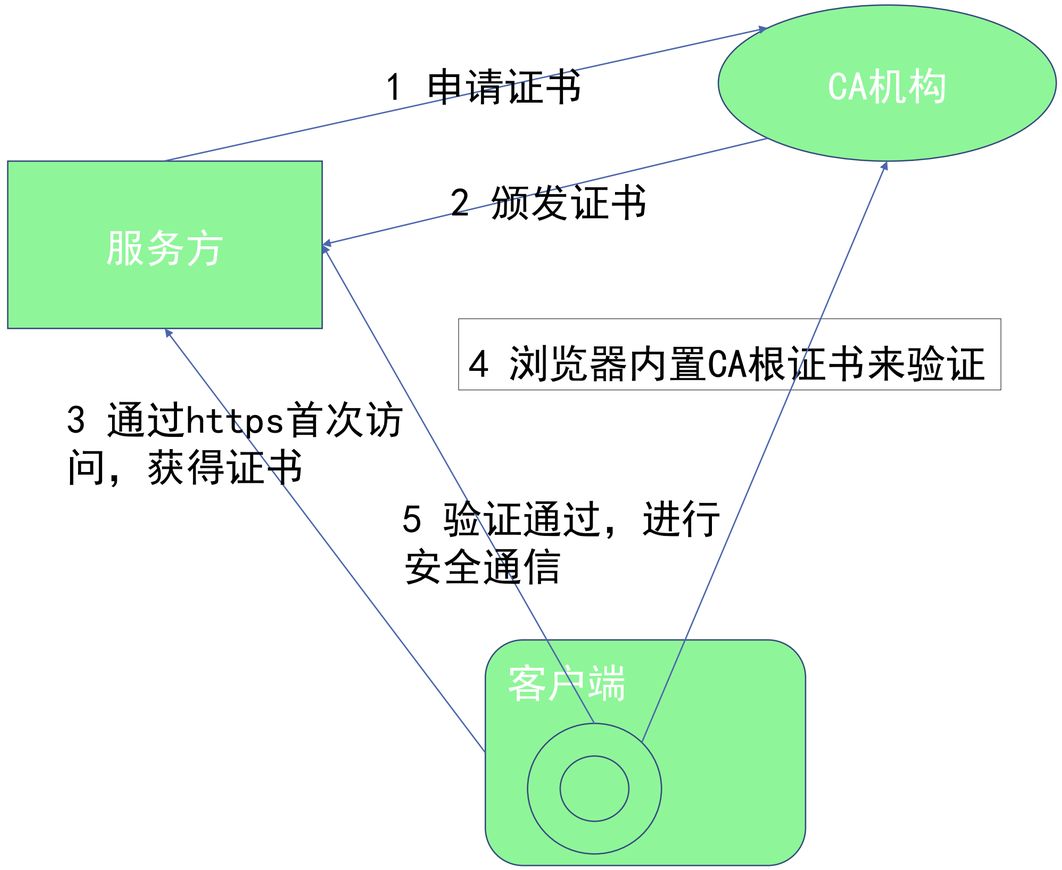
数字证书的颁发过程

用户首先产生自己的密钥对，并将公共密钥及部分个人身份信息传送给认证中心。认证中心在核实身份后，将执行一些必要的步骤，以确信请求确实由用户发送而来，然后，认证中心将发给用户一个数字证书，该证书内包含用户的个人信息和他的公钥信息，同时还附有认证中心的签名信息(根证书私钥签名)。用户就可以使用自己的数字证书进行相关的各种活动。数字证书由独立的证书发行机构发布，数字证书各不相同，每种证书可提供不同级别的可信度。

浏览器CA认证流程

浏览器默认都会内置CA根证书，其中根证书包含了CA的公钥

1. 证书颁发的机构是伪造的：浏览器不认识，直接认为是危险证书
2. 证书颁发的机构是确实存在的，于是根据CA名，找到对应内置的CA根证书、CA的公钥。用CA的公钥，对伪造的证书的摘要进行解密，发现解不了，认为是危险证书。
3. 对于篡改的证书，使用CA的公钥对数字签名进行解密得到摘要A，然后再根据签名的Hash算法计算出证书的摘要B，对比A与B，若相等则正常，若不相等则是被篡改过的。
4. 证书可在其过期前被吊销，通常情况是该证书的私钥已经失密。较新的浏览器如Chrome、Firefox、Opera和Internet Explorer都实现了在线证书状态协议（OCSP）以排除这种情形：浏览器将网站提供的证书的序列号通过OCSP发送给证书颁发机构，后者会告诉浏览器证书是否还是有效的。

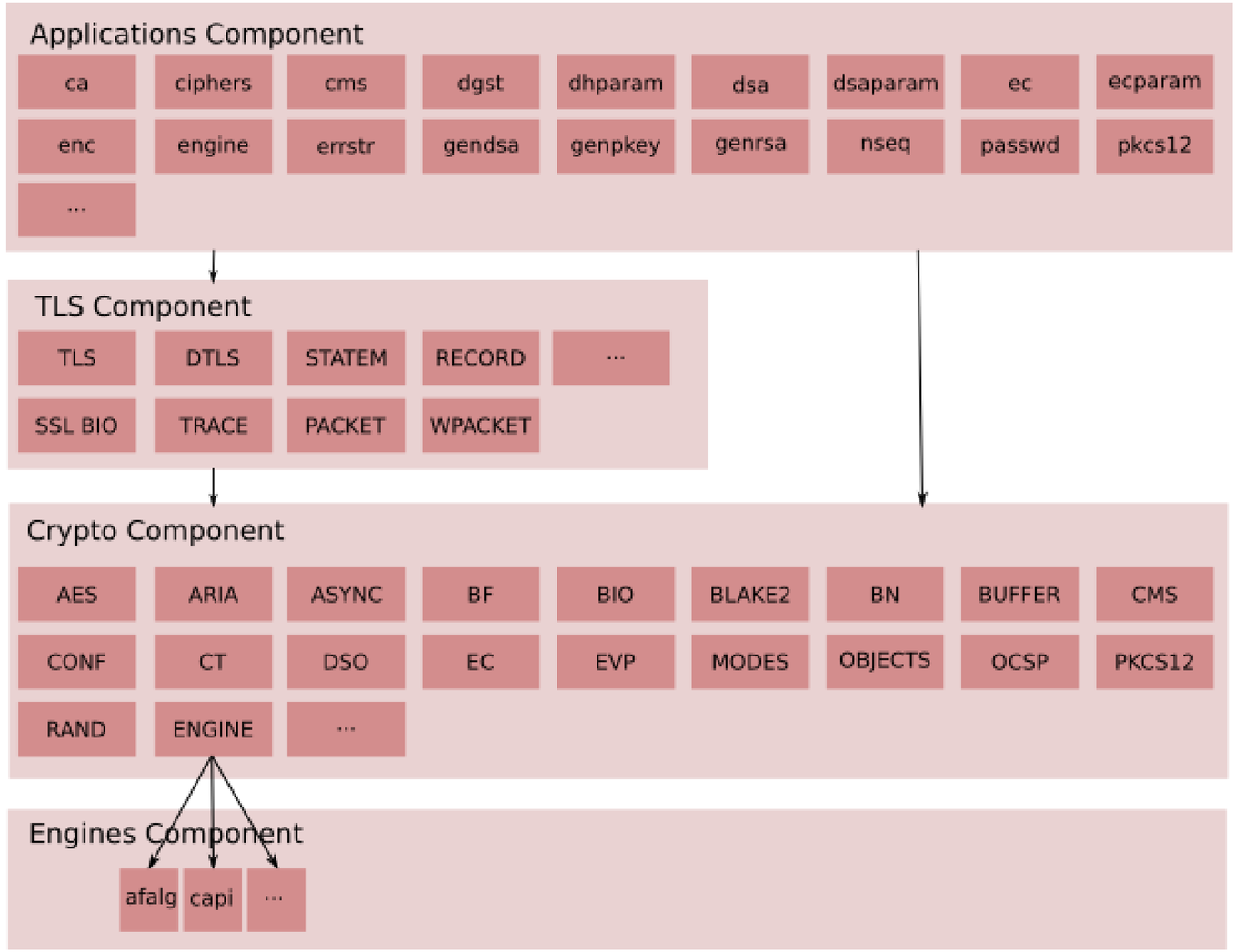


1、2点是对伪造证书进行的，3是对于篡改后的证书验证，4是对于过期失效的验证。

OpenSSL

OpenSSL是个多功能命令行工具、他可以实现加密解密、甚至还可以当CA来用、可以让你创建证书、吊销证书。

OpenSSL由应用、TLS、加密、引擎四个组件层组成。加密组件通过引擎组件来扩展，TLS依赖加密组件，应用组件则依赖TLS和加密组件。



更多资料参考官网：https://www.openssl.org/

动手制作证书

SSL 证书就是遵守 SSL协议，由受信任的数字证书颁发机构CA，在验证服务器身份后颁发，具有服务器身份验证和数据传输加密功能的文件，当前大多数的ssl证书是收费的。

免费证书

也可以到阿里云、腾讯云申请1年的免费的简单证书。

下面，我们将通过OpenSSL自建一个CA服务，自己给自己的服务办法数字证书，学习整个证书签发过程

具体操作参考《 制作第一张自己的SSL证书.md》手册

目录

Nginx中实现高性能Https

Nginx中配置HTTPS

```
server {
    listen              443 ssl;
    server_name         www.example.com;
    ssl_certificate     www.example.com.crt;     # 发送到客户端具有PEM格式的公共证书
    ssl_certificate_key www.example.com.key;     # 具有PEM格式的服务器端私钥，注意访问权限设置
    ssl_protocols       TLSv1 TLSv1.1 TLSv1.2;   # ssl支持的协议
    ssl_ciphers         HIGH:!aNULL:!MD5;   # 以OpenSS L库所理解的格式指定
    ...
}
```

Https服务的优化

优化方法

SSL操作消耗额外的CPU资源。在多处理器系统上，应该运行多个工作进程，而不小于可用CPU核的数量。CPU密集型操作是SSL握手。有两种方法可以使每个客户端的这些操作的数量最小化：

第一个是通过启用Keepalive连接来通过一个连接发送多个请求。

第二个是重用SSL会话参数，以避免用于并行和后续连接的SSL握手。

Nginx中为了减少处理器负载，建议

1. 将工作进程的数量设置为等于处理器的数目，worker_processes
2. 启用维持生命的连接，keepalive_timeout
3. 启用共享会话缓存，禁用内置会话缓存，ssl_session_cache shared:SSL:10m;

   off，禁用、

   none，客户端可重用服务端不缓存

   builtin，在openssl中重构缓存，`builtin`[:`size`]

   shared，在worker进程间共享的缓存，`shared`:`name`:`size`
4. 增加会话生存期(默认为5分钟)，ssl_session_timeout

总结

Web安全的重要性

细数历史安全事件

加密技术

摘要算法、对称加密、非对称加密

Https

定义、组成、SSL/TLS、与http的区别

Https如何保证数据安全

SSL/TLS握手过程、保证安全的原因

证书

配置Https服务

谢谢观看