TLS/SSL 协议详解 (30) SSL中的RSA、DHE、ECDHE、ECDH流程与区别

置顶 2017年09月19日 08:50:16 阅读数：8126 更多

本文是对前面章节关于非对称算法在SSL中运用的总结和细化，但也可以作为详解SSL中RSA、ECDHE非对称加密算法来看。

在不安全信道上构建安全信道，这是SSL的核心，所谓安全包括身份认证、数据完整性、数据加密性。而非对称算法在SSL中的运用就是为了协商一个密钥，密钥的目的就是为了后续数据能够被加密，而加密密钥有且只有通信双方知道。

通常网络上传输的数据一般都被认为是可见的。端对端传输的数据，不仅经过交换机、路由器，还经过各种DPI、IPS、WAF等审计安全设备，甚至可能经过负载均衡等反向代理设备，只要在任何一个环节抓包，都可以轻松获取网络上传输的数据。所以如果A和B需要加密通信，即通信的内容需要使用有且只有A和B知道的“密钥”加密，那么必然需要传输这个“密钥”，也就是说“密钥”本身需要在不安全传输的信道传输，如果简单的传输“密钥”，那么这个“密钥”就不再保密，任何第三方都能获取“密钥”，即任何第三方都能解密A和B发出来的密文数据。

非对称算法就是为了解决**“密钥”传输**（A和B共享）的问题。

1：RSA密钥交换算法

详细原理请参考我的这篇博客http://blog.csdn.net/mrpre/article/details/52609087

本篇不讲解具体原理，而是讲解交互过程。

RSA的核心涉及公钥私钥的概念

（1）：**使用公钥加密的数据只有私钥能解密**

（2）：**使用私钥加密的数据只有公钥能解密**

我们构建这么一种场景，服务器配置有公钥+私钥，客户端是离散的。

RSA算法流程文字描述如下：

（1）：**任意客户端对服务器发起请求，服务器首先发回复自己的公钥到客户端（公钥明文传输）。**

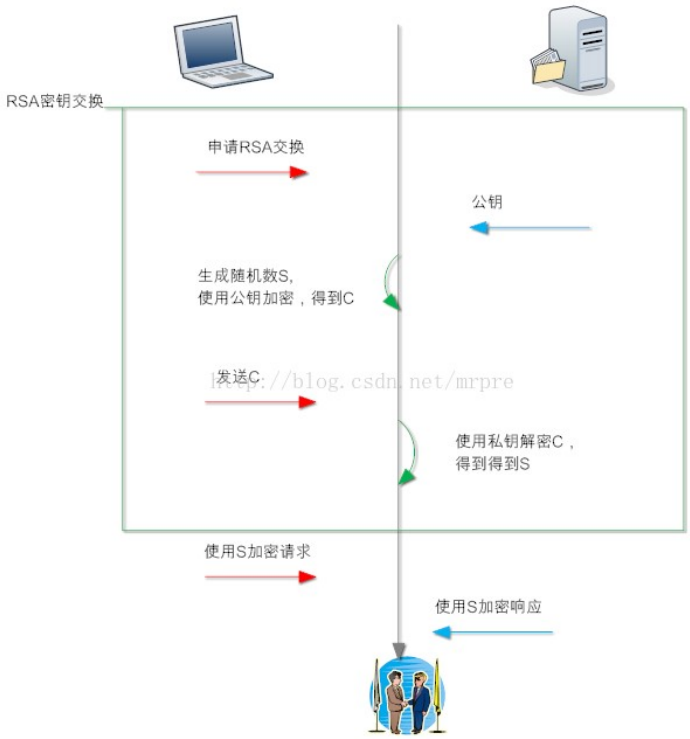
（2）：客户端使用随机数算法，生成一个密钥S，使用收到的公钥进行 加密，生成C，把C发送到服务器。

（3）：服务器收到C，使用公钥对应的私钥进行解密，得到S。

（4）：上述交换步骤后，客户端和服务器都得到了S，S为密钥（预主密钥）。

我们来看看上述过程中，为何第三方无法得到S。首先第一步后，客户端有公钥，服务器有公钥和私钥。由于公钥是明文传输的，所以可以假设第三方也有公钥。

第二步后，客户端发送C，服务器能够使用自己的私钥进行解密，而第三方只有公钥，无法解密。即第三方无法计算得到S。



上述中，服务器发送的公钥在SSL中是通过certificate报文发送的，certificate中的包含了公钥。C是通过Client key exchange报文发送的。

其实，在实际SSL实际设计中，S其实并没有直接被当成密钥加密，这里为了描述原理，省去了对S后续进行KDF等操作，并不影响实际理解RSA。

RSA有一个问题，就是如果私钥泄漏，即私钥被第三方知道，那么第三方就能从C中解密得到S，即只要保存所有的A和B的报文，等到私钥被泄漏的那一天，或者有办法快从C中计算S的方法出现（量子计算机分解大素数），那么A和B就没有什么私密性可言了。

这就是所谓的前向不安全，私钥参与了密钥交换，安全性取决于私钥是否安全保存。

有网友问了这么一个问题：为何客户端不也安装一个公钥私钥，然后客户端和服务器交互的时候，各自传送给对方公钥，然后各自拿对方的公钥加密数据发送给对方，然后各自拿私钥解密收到的数据?

先不说性能，我们看RSA加解密算法，若要加密m，那么需要计算

m^e mod n

如果m > n，我们记作m = n + k

那么原式子 (n + k)^e mod n

多项式展开，除了最后一项k^e ，其余的每一项都有n，故mod n后，

k^e mod n

换句话说，如果m大于n，那么其加密的结果和k的结果时一样的，这就有二义性了，所以RSA本身就不允许m>n的情况出现。所以拿来直接加密数据时不可取的。

2：DHE密钥交换算法

详细原理请参考我的这篇博客http://blog.csdn.net/mrpre/article/details/52608867

本篇不讲解具体原理，而是讲解交互过程。

DHE算法流程文字描述如下：

（1）：客户端计算一个随机值Xa，使用Xa作为指数，即计算Pa = q^Xa mod p，其中q和p是全世界公认的一对值。客户端把Pa发送至服务器，Xa作为自己私钥，仅且自己知道。

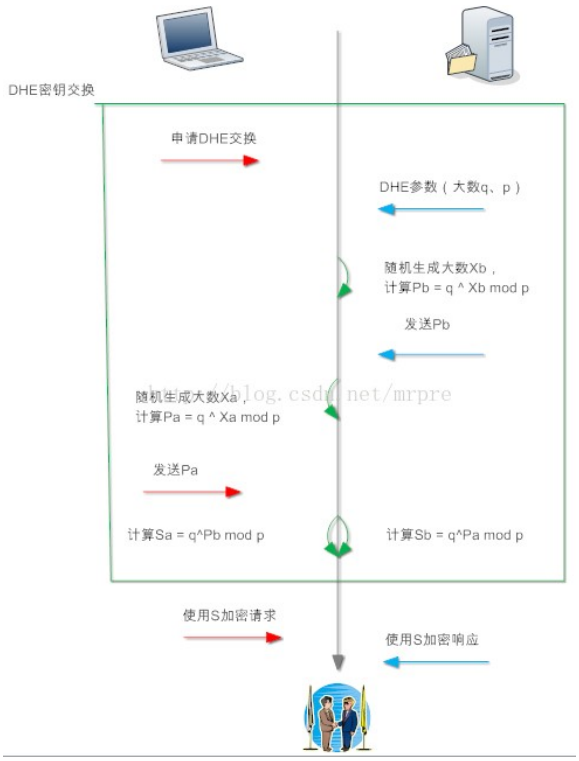
（2）：服务器和客户端计算流程一样，生成一个随机值Xb，使用Xb作为指数，计算

Pb = q^Xb mod p，将结果Pb发送至客户端，Xb仅自己保存。

（3）：客户端收到Pb后计算Sa = Pb ^Xa mod p；服务器收到Pa后计算Sb = Pa^Xb mod p

（4）：算法保证了Sa = Sb = S，故密钥交换成功，S为密钥（预主密钥）。

DHE密钥交换握手流程图



上述途中，Sa和Sb得到的结果是相同的，即记为S。

上述密钥交换流程中，和RSA密钥交换有较大不同，DHE密钥交换时，服务器私钥没有参与进来。也就是说，私钥即使泄漏，也不会导致会话加密密钥S被第三方解密。

实际使用过程中，私钥的功能被削弱到用来身份认证（上图中没有画出）。

上图中DHE参数和Pb都是通过server key exchange发送给客户端，Pa通过client key exchange发送给服务器。server key exchange的结尾处需要使用服务器私钥对该报文本身进行签名，以表明自己拥有私钥（图中为了表明私钥没有参与密钥计算，没有画出，但不影响理解DHE算法）。

3：ECDHE密钥交换算法

详细原理请参考我的这几片篇博客

http://blog.csdn.net/mrpre/article/details/72850486

http://blog.csdn.net/mrpre/article/details/72850598

http://blog.csdn.net/mrpre/article/details/72850644

本篇不讲解具体原理，而是讲解交互过程。

只要理解DHE密钥交换原理，那么理解ECDHE密钥交换原理其实并不难（如果不想深究的话）。

ECDHE的运算是把DHE中模幂运算替换成了点乘运算，速度更快，可逆更难。

ECDHE算法流程文字描述如下：

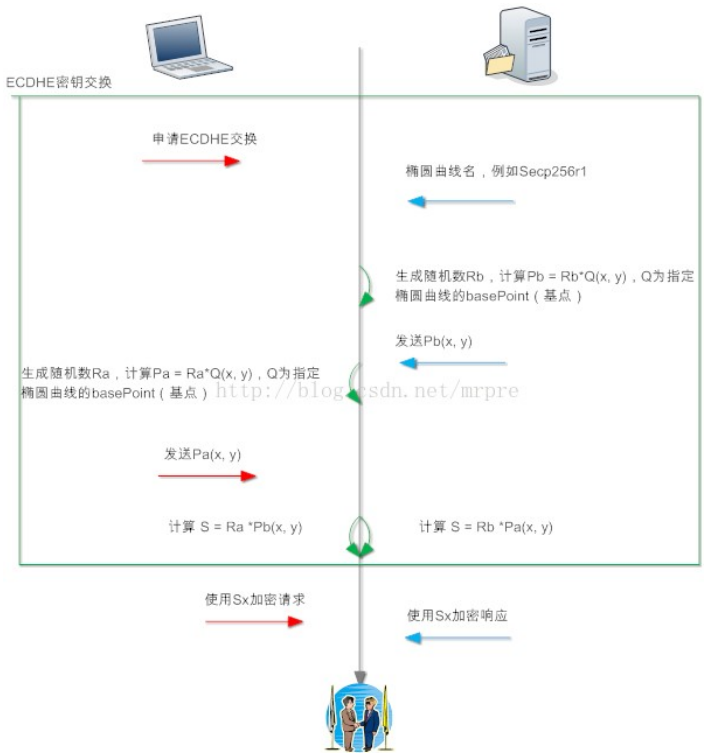
（1）：客户端随机生成随机值Ra，计算Pa(x, y) = Ra * Q(x, y)，Q(x, y)为全世界公认的某个椭圆曲线算法的基点。将Pa(x, y)发送至服务器。

（2）：服务器随机生成随机值Rb，计算Pb(x,y) - Rb * Q(x, y)。将Pb(x, y)发送至客户端。

（3）：客户端计算Sa(x, y) = Ra * Pb(x, y)；服务器计算Sb(x, y) = Rb *Pa(x, y)

（4）：算法保证了Sa = Sb = S，提取其中的S的x向量作为密钥（预主密钥）。

ECDHE密钥交换握手流程图



SSL协议中，上图中椭圆曲线名和Pb通过server key exchange报文发送；Pa通过client key exchange报文发送。

4：ECDHE与ECDH算法的区别

字面少了一个E，E代表了“临时”，即在握手流程中，作为服务器端，ECDH少了一步计算Pb的过程，Pb用证书中的公钥代替，而证书对应的私钥就是Xb。由此可见，使用ECDH密钥交换算法，服务器必须采用ECC证书；服务器不发送server key exchange报文，因为发送certificate报文时，证书本身就包含了Rb信息。

5：ECDHE与RSA的区别

ECDHE（DHE）算法属于DH类密钥交换算法， 私钥不参与密钥的协商，故即使私钥泄漏，客户端和服务器之间加密的报文都无法被解密，这叫 前向安全（forward secrity）。由于ECDHE每条会话都重新计算一个密钥（Ra、Rb），故一条会话被解密后，其他会话仍旧安全。

然而，ECDH算法服务器端的私钥是固定的，即证书的私钥作为Rb，故ECDH不被认为前向安全，因为私钥泄漏相当于Rb泄漏，Rb泄漏，导致会话密钥可被第三方计算。