加密传输中的NPN和ALPN

自从HTTP从1.1升级到了2，一切都变得不同了。虽然HTTP2没有强制说必须使用加密协议进行传输，但是业界的标准包括各大流行的浏览器都只支持HTTPS情况下的HTTP2协议。

那么怎么在HTTPS之中加入HTTP2协议的支持呢？今天本文将会跟大家聊一下SSL/TLS协议的扩展NPN和ALPN。

SSL/TLS协议

SSL(Secure Socket Layer)：安全套接层。

TLS(Transport Layer Security)：传输层安全是在SSL3.0基础上设计的协议，实际上相当于SSL的后续版本。



TLS主要分为两层，底层的是TLS记录协议，主要负责使用对称密码对消息进行加密。

上层的是TLS握手协议，主要分为握手协议，密码规格变更协议和应用数据协议4个部分。

其中最重要的就是握手协议，通过客户端和服务器端的交互，和共享一些必要信息，从而生成共享密钥和交互证书。



接下来我们一步步的介绍每一步的含义：

1、client hello

客户端向服务器端发送一个client hello的消息，包含下面内容：

可用版本号

当前时间

客户端随机数

会话ID

可用的密码套件清单

可用的压缩方式清单

我们之前提到了TLS其实是一套加密框架，其中的有些组件其实是可以替换的，这里可用版本号，可用的密码套件清单，可用的压缩方式清单就是向服务器询问对方支持哪些服务。

客户端随机数是一个由客户端生成的随机数，用来生成对称密钥。

2、server hello

服务器端收到client hello消息后，会向客户端返回一个server hello消息，包含如下内容：

使用的版本号

当前时间

服务器随机数

会话ID

使用的密码套件

使用的压缩方式

使用的版本号，使用的密码套件，使用的压缩方式是对步骤1的回答。

服务器随机数是一个由服务器端生成的随机数，用来生成对称密钥。

3、可选步骤:certificate

服务器端发送自己的证书清单，因为证书可能是层级结构的，所以处理服务器自己的证书之外，还需要发送为服务器签名的证书。 客户端将会对服务器端的证书进行验证。如果是以匿名的方式通信则不需要证书。

4、可选步骤:ServerKeyExchange

如果第三步的证书信息不足，则可以发送ServerKeyExchange用来构建加密通道。

ServerKeyExchange的内容可能包含两种形式：

如果选择的是RSA协议，那么传递的就是RSA构建公钥密码的参数（E，N）。我们回想一下RSA中构建公钥的公式：密文=明文E mod N密文=明文^E\ mod\ N密文=明文E mod N， 只要知道了E和N，那么就知道了RSA的公钥，这里传递的就是E，N两个数字。

如果选择的是Diff-Hellman密钥交换协议，那么传递的就是密钥交换的参数。

5、可选步骤:CertificateRequest

如果是在一个受限访问的环境，比如fabric中，服务器端也需要向客户端索要证书。 如果并不需要客户端认证，则不需要此步骤。

1. server hello done 服务器端发送server hello done的消息告诉客户端自己的消息结束了。

7、可选步骤:Certificate

对步骤5的回应，客户端发送客户端证书给服务器

8、ClientKeyExchange

还是分两种情况：

如果是公钥或者RSA模式情况下，客户端将根据客户端生成的随机数和服务器端生成的随机数，生成预备主密码，通过该公钥进行加密，返送给服务器端。

如果使用的是Diff-Hellman密钥交换协议，则客户端会发送自己这一方要生成Diff-Hellman密钥而需要公开的值。这样服务器端可以根据这个公开值计算出预备主密码。

9、可选步骤:CertificateVerify

客户端向服务器端证明自己是客户端证书的持有者。

10、ChangeCipherSpec(准备切换密码)

ChangeCipherSpec是密码规格变更协议的消息，表示后面的消息将会以前面协商过的密钥进行加密。

11、finished(握手协议结束)

客户端告诉服务器端握手协议结束了。

12、ChangeCipherSpec(准备切换密码)

服务器端告诉客户端自己要切换密码了。

13、finished(握手协议结束)

服务器端告诉客户端，握手协议结束了。

14、切换到应用数据协议

这之后服务器和客户端就是以加密的方式进行沟通了。

NPN和ALPN

上面我们介绍SSL/TLS协议的时候，最后一步是切换到应用数据协议，那么客户端是怎么和服务器端讨论协商具体使用哪种应用数据协议呢？是使用HTTP1.1？还是HTTP2？还是SPDY呢？

这里就要用到TLS扩展协议了。而NPN(Next Protocol Negotiation) 和 ALPN (Application Layer Protocol Negotiation) 就是两个TLS的扩展协议。

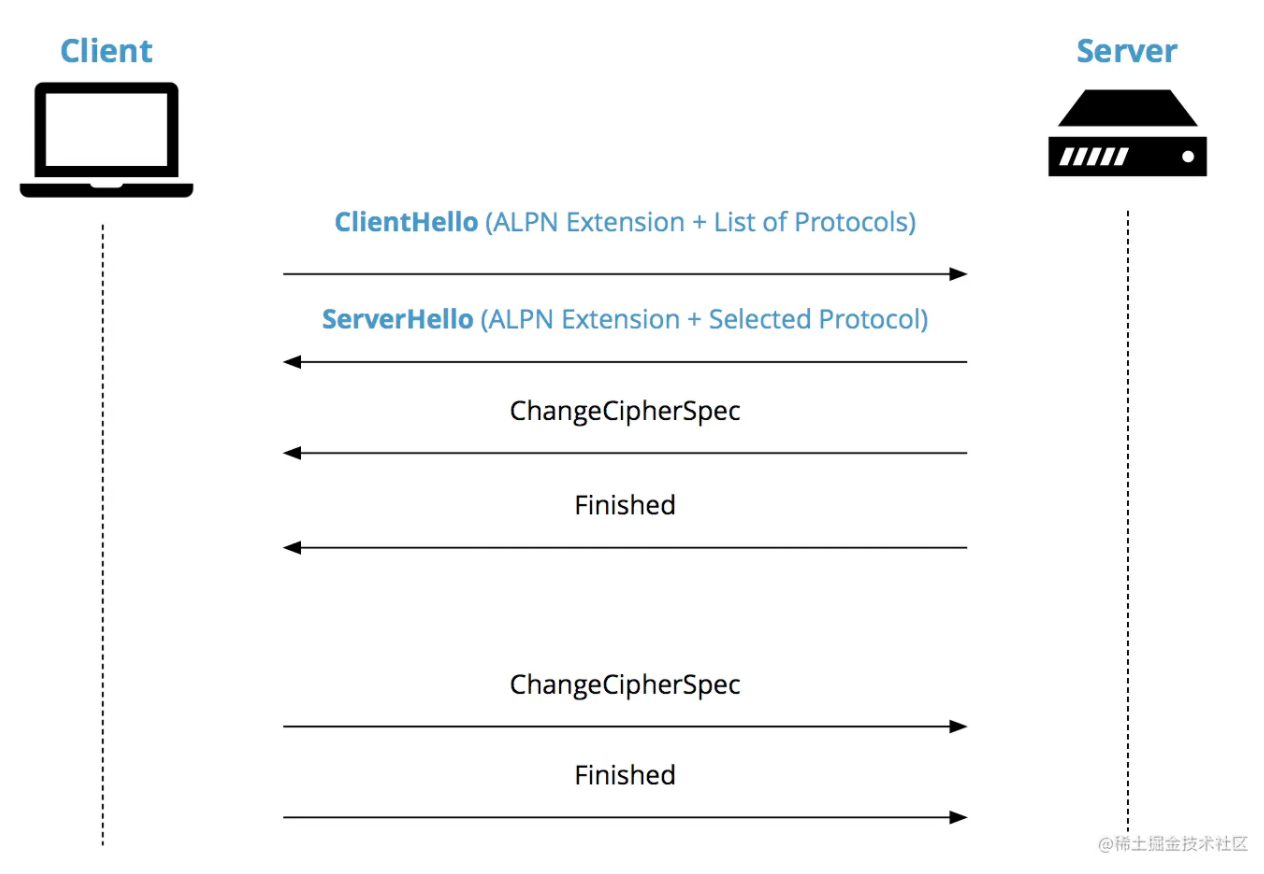
他们主要用在TLS中，用来协商客户端和服务器端到底应该使用什么应用数据协议进行沟通。

其中NPN是SPDY使用的扩展，而ALPN是HTTP2使用的扩展。

他们两个有什么区别呢？

相较于NPN来说，ALPN在client hello消息中已经列出了客户端支持的应用层协议，服务器端只需要从中选择出它支持的协议即可。比NPN少了一个交互的步骤，所以ALPN是推荐的协议。

下面是具体的交互流程图：



下面以ALPN为例，讲解下具体的交互流程，首先是客户端发送”Client Hello“消息：

Handshake Type: Client Hello (1)

Length: 141

Version: TLS 1.2 (0x0303)

Random: dd67b5943e5efd0740519f38071008b59efbd68ab3114587...

Session ID Length: 0

Cipher Suites Length: 10

Cipher Suites (5 suites)

Compression Methods Length: 1

Compression Methods (1 method)

Extensions Length: 90

[other extensions omitted]

Extension: application\_layer\_protocol\_negotiation (len=14)

Type: application\_layer\_protocol\_negotiation (16)

Length: 14

ALPN Extension Length: 12

ALPN Protocol

ALPN string length: 2

ALPN Next Protocol: h2

ALPN string length: 8

ALPN Next Protocol: http/1.1

可以看到在client hello消息中的Extension字段中，使用了ALPN，并且列出了可以选择使用的两种ALPN Protocol：h2和http/1.1。

对应的“server hello” 消息会选择出具体使用的ALPN protocol如下：

Handshake Type: Server Hello (2)

Length: 94

Version: TLS 1.2 (0x0303)

Random: 44e447964d7e8a7d3b404c4748423f02345241dcc9c7e332...

Session ID Length: 32

Session ID: 7667476d1d698d0a90caa1d9a449be814b89a0b52f470e2d...

Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 (0xc02f)

Compression Method: null (0)

Extensions Length: 22

[other extensions omitted]

Extension: application\_layer\_protocol\_negotiation (len=5)

Type: application\_layer\_protocol\_negotiation (16)

Length: 5

ALPN Extension Length: 3

ALPN Protocol

ALPN string length: 2

ALPN Next Protocol: h2

如上所示，服务器端选择了h2， 最终当客户端和服务器端TLS握手结束之后，会选择使用HTTP2作为后续的应用层数据协议。

NPN和ALPN都是TLS的扩展，相较而言，ALPN更加好用。