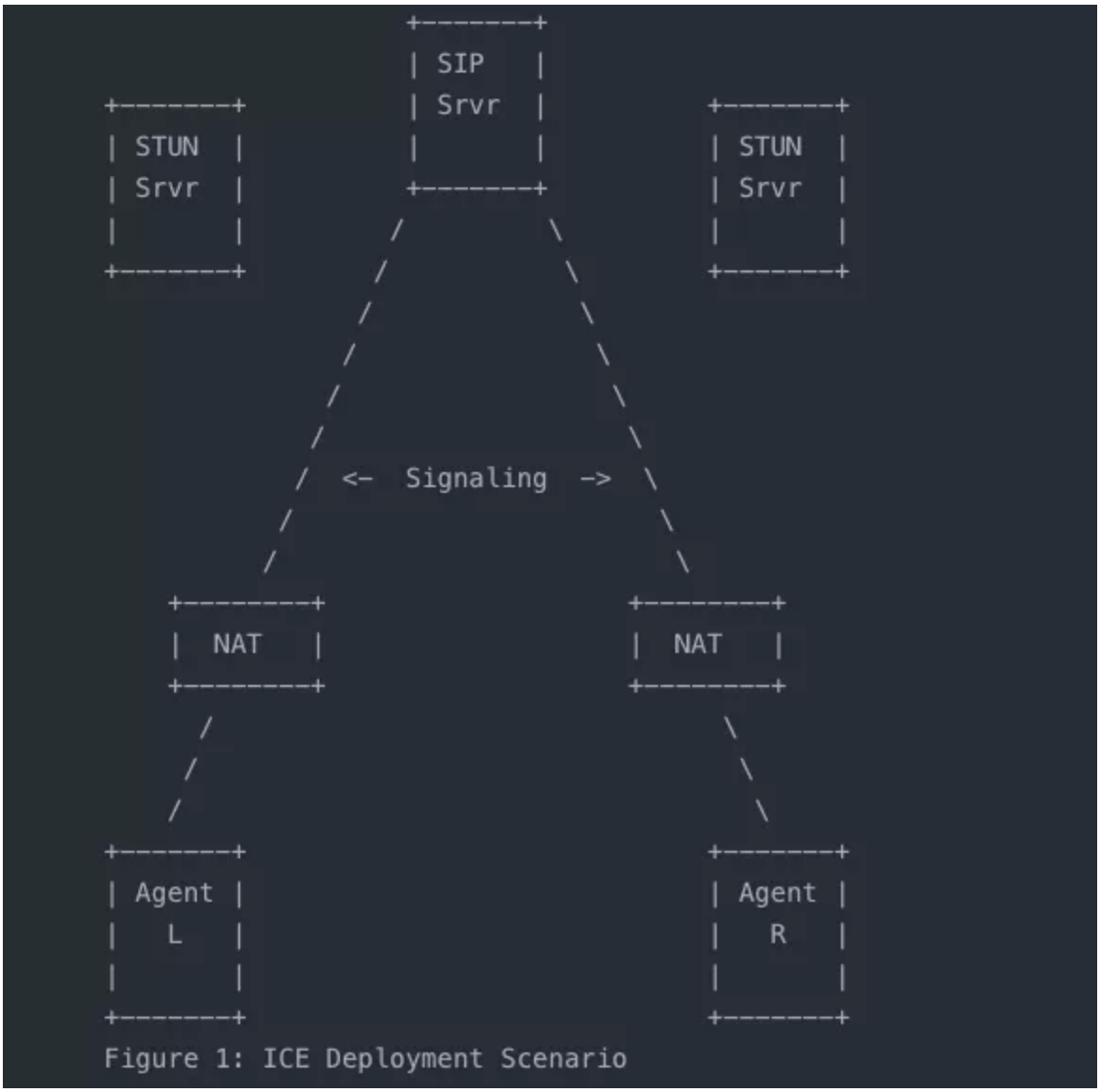
7.0-WebRTC ICE 模块剖析

网页版本：https://www.yuque.com/docs/share/dfffc1ac-4bee-482c-bff8-b19fcc482948?#（密码：gmq1） 《7.0-WebRTC ICE 模块剖析》

1 ICE简介

ICE全称Interactive Connectivity Establishment——交互式连通建设形式。



ICE背后的基本思想如下：每个代理都有各种各样的Candidate Transport 地址（IP地址和端口的组合,特定的传输协议(在此中始终为UDP规范)）。它可以用来与其他代理进行通信。

NAT全称network adress translation，即网络地址转换。其存在的意义是将私有的IP地址转化为公有IP地址。

这些地址包括：

1. 直接连接的网络接口上的传输地址 ——公网IP直连
2. NAT公共端的转换传输地址 ——内网NAT映射
3. 从TURN服务器分配的传输地址 ——中继模式

对于1 公网IP直连这类情况，使用标准socket就可以建立tcp或者udp链接了，这个属于最简单的一种情况，直连上就可以收发数据

对于2 内网NAT映射这类情况，NAT映射的出现是一个ipv4地址不够用，但ipv6还没普及，为了让更多的设备能连接互联网，出现的一种解决办法，到现在也成了ipv6普及的一个重要的绊脚石了，因为互联网中大量的设备都支持nat并且工作正常，导致普及ipv6的积极性被削弱

对于3这种情况，其实就是一个中介，通过中介交换数据的模式，这种模式代价比较大

实际应用中，大部分都是2这种情况，公网IP还是比较昂贵的。

1.1 ICE交互难点

ice交互难得原因，是Nat技术所导致，为了克服这个，有了stun、turn方法，一般来讲，分为**对称型NAT**和**圆锥形NAT**，其中圆锥形NAT又分为完全圆锥型NAT、IP限制圆锥型NAT、Port限制圆锥型NAT，下面分别来讲解这几种情况的NAT互联。

1. **完全圆锥型NAT**

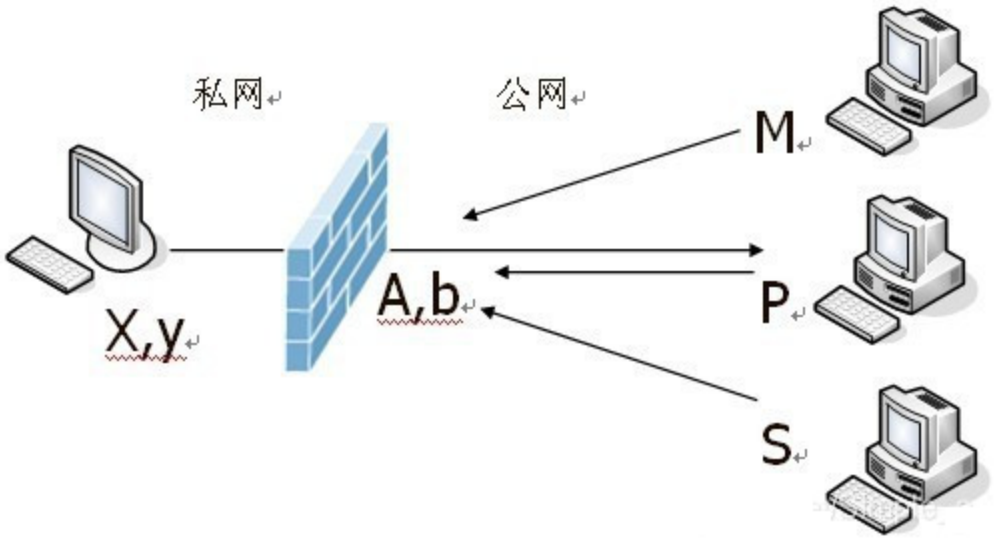
   完全圆锥型NAT是指同一个内网IP1+Port1向任何外网发送数据，在NAT会被映射到同一个外网的IP2+Port2；

   且当外网向IP2+Port2发送数据，在NAT上也会被转换到内网IP1+Port1。一些反向代理服务器的代理节点就是此类型的NAT(比如机房内网)。

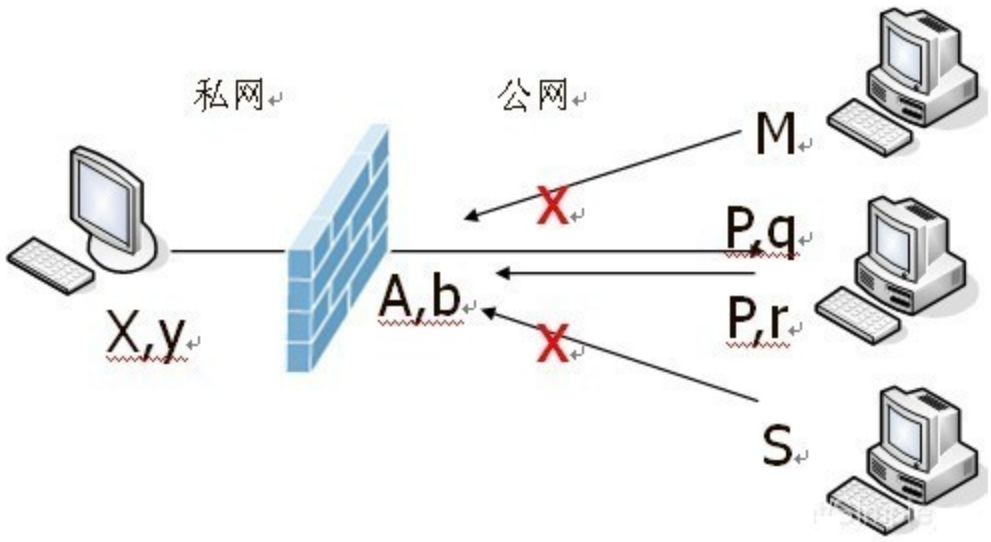
   - 无论私网主机之前是否向公网Ip发送过数据，私网主机都能接收到公网主机发送的数据。

     示例: 私网主机X先访问公网主机M,主机X能成功收到M发送的数据，并且主机X在没有

向主机P发送数据的情况下，也能收到另一台主机P发送的数据。



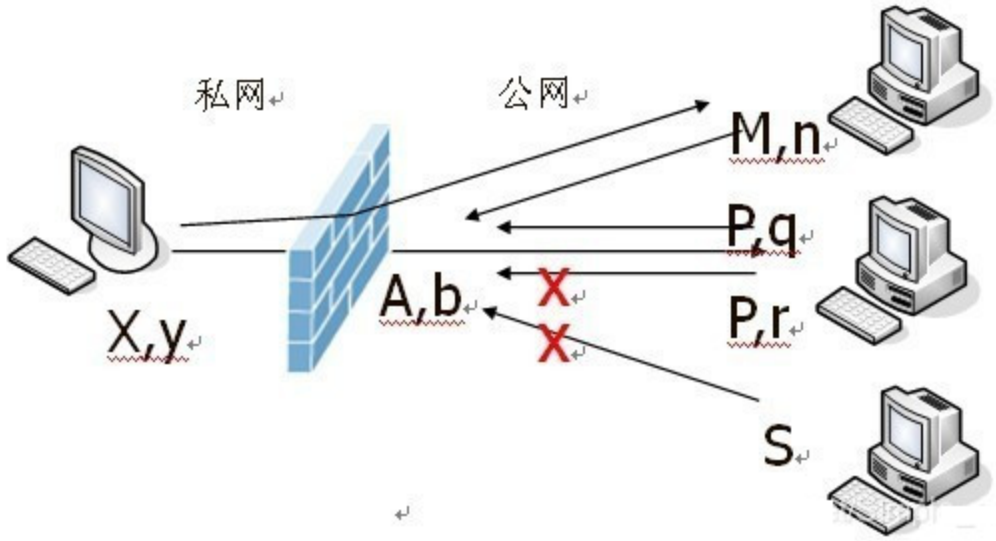
2. **IP限制圆锥型NAT** IP限制圆锥型NAT是指同一个内网IP1+Port1向任何外网发送数据，在NAT会被映射到同一个外网的IP2+Port2；
   但是这种地址映射是与外网目的主机IP关联的，外网主机IP不同，内网主机同一个IP1+Port1在NAT上映射的IP2+Port2也不相同；
   这时候的特性是内网IP1+Port1没有主动向IP3的外网主机发送数据，那么IP3的主机向IP2+Port2发送数据，将会被NAT丢弃，不会转发到内网IP1+Port1上。
   - 私网主机只有先向公网主机发送数据之后，才能接收到公网主机发送的数据。否则接收不到。
     示例: 私网主机X向公网主机P发送数据，然后X能接收到P发送的数据
     在之前没有向M发送过数据时，无法接收来自M的数据。



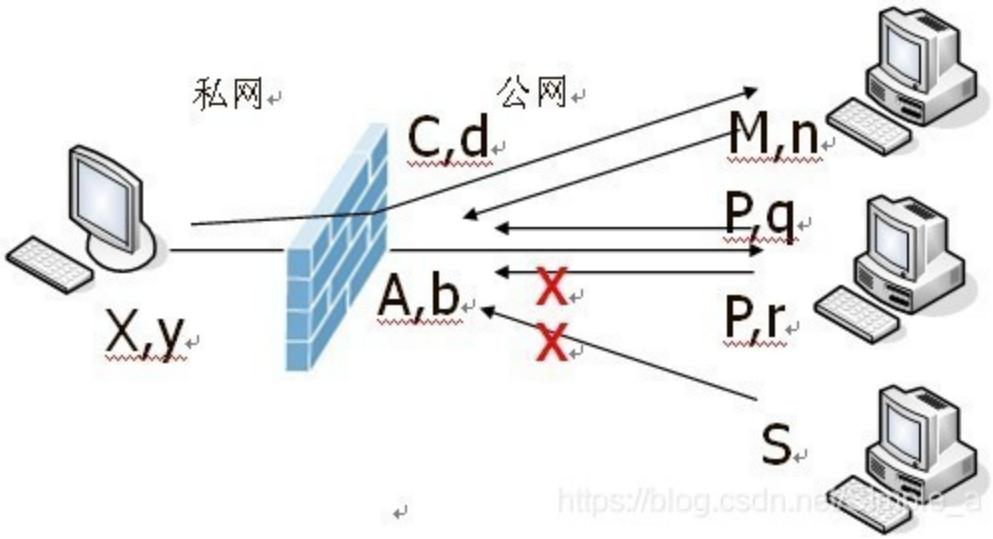
3. **Port限制圆锥型NAT**
   Port限制圆锥型NAT是指同一个内网IP1+Port1向任何外网发送数据，在NAT会被映射到同一个

外网的IP2+Port2；
但是这种地址映射是与外网目的主机IP和端口关联的(比2这种更加严格)，这种的目的外网主机IP不同，内网主机同一个IP1+Port1在NAT上映射的IP2+Port2也不相同，同时目的外网主机IP相同，但端口不同时，内网主机同一个IP1+Port1在NAT上映射的IP2+Port2也不相同；
也就是说当内网IP1+Port1没有主动向IP3的外网主机的Port3发送数据，那么IP3+Port3向IP2+Port2发送数据，将会被NAT丢弃，不会转发到内网IP1+Port1上；
IP限制圆锥型NAT只认姓啥不问名，Port限制圆锥型NAT是既要认姓啥又要看名谁。

- 私网主机只有先向公网主机的某一个端口号发送数据之后，才能接收到公网主机的这一端口号发送的数据。接收数据的公网主机更换端口号，则私网主机接收不到。
  示例: 主机X先向主机P的q端口发送数据，然后只X只能接收到来自主机P的q端口的数据。



4. **对称型NAT**对称型NAT是内网IP1+Port1向外网IP2+Port2发送数据时，在NAT会被映射到一个外网的IP3+Port3；
   当向外网IP4+Port4发送数据时，在NAT会被映射到一个外网的IP5+Port5。这种机制不能保证同一个内网IP和端口向不同外网IP和端口发送数据时，其映射的外网IP和端口的一致性。
   - 私网主机请求指定的公网主机和端口号之后，后续只能接收来自此公网主机的端口号发送来的数据。更换主机和端口号向私网主机发送数据，私网主机都接收不到。
   - 示例: 私网主机X经过NAT先访问公网上主机P(端口号为q)，然后主机能接收到来自P(端口号为q)响应的数据。同样过程访问M(端口号为n)，也能接收到来自M的数据。但是对于没有请求过的主机S，如果S向主机X发送数据，则接收不到。说明此NAT类型为对称型。
   - 和端口限制型不同的是，此种类型的转换会对每一个请求都有一个公网的IP和端口号的映射。而端口限制型还是公用一个映射后的公网IP和端口号。



如果自己实现NAT类型检测的话，NAT类型判断算法整体流程大至是：

1. 先判断防火墙是否阻止所有udp包进来；
2. 再判断是否是公网ip；
3. 再判断是否为全锥型；
4. 然后判断是否是对称型；
5. 最后判断是否是端口或地址限制型；

具体能否打通可以看下表：

ICE协议包括stun和turn协议，turn协议是stun协议的补充，可以简单粗暴理解为如果stun不通，那就走turn，turn可以理解为一个中继代理转发。

Client A0和 Client B建连的大概过程示意图如下：

1.2 Classic STUN（RFC3489）的劣势

Classic STUN 有着诸多局限性，例如:

1. 不能确定获得的公网映射地址能否用于P2P通信；
2. 没有加密方法；

3. 不支持TCP穿越；
4. 不支持对称型NAT的穿越；
5. 不支持IPV6。

1.3 STUN（RFC5389）协议

RFC5389是RFC3489的升级版：

1. 支持UDP/TCP/TLS协议；
2. 支持安全认证。

https://datatracker.ietf.org/doc/html/rfc5389

ICE利用STUN（RFC5389） Binding Request和Response，来获取公网映射地址和进行连通性检查。同时扩展了STUN的相关属性:

1. PRIORITY:在计算candidate pair优先级中使用；
2. USE-CANDIDATE:ICE提名时使用；
3. tie-breaker:在角色冲突时使用。

ICE使用TURN（RFC 5766）协议作为STUN的辅助，在点对点穿越失败的情况下，借助于TURN服务的转发功能，来实现互通。端口与STUN保持一致

TURN消息都遵循 STUN 的消息格式，除了ChannelData消息。TURN扩展了STUN格式：

1. 支持UDP/TCP/TLS协议，适用于UDP被限制的网络；
2. 支持IPV6。

1.4 STUN消息格式

Stun Header：STUN 消息头为 20 字节，后面紧跟 0 或多个属性。STUN 头部包含一 STUN 消息类型、magic cookie、事务 ID 和消息长度。（RFC5389只定义了一个binding方法，其他方法是在其他文档中定义）

每个 STUN 消息的最高位前 2 位必须为 0。当多个协议复用同一个端口的时候，这个可以用于与其他协议区分 STUN 数据包。消息类型确定消息的类别（如请求、成功回应、失败回应、指示 indication）。虽然这里有四种消息类型，但可以分为 2 类事务：请求/响应事务、指示事务。

- magic cookie 为固定值 **0x2112A442**；
- Transaction ID 标识同一个事务的请求和响应。当客户端发送多个 STUN 请求，通过 Transaction ID 识别对应的 STUN 响应。

最高两位：为0，在STUN协议与其他协议端口复用时，用于区分STUN和其他数据包，如RTP数据包。

STUN Message Type（14bits）：消息类型。定义消息类型如下：

Message Length:（16bits）,消息长度，不包含STUN Header的20个字节。所有的STUN属性都是4字节对齐的，该字段值包括STUN属性的padding长度。

Magic Cookie：（32bits），固定值0x2112A442，用于反射地址的异或（XOR）运算。

Transaction ID：（96bits），事务ID标识符，请求对应的响应具有相同的标识符。

Stun Message Type（14bits）还可以分为以下格式：

其中显示的位为从最高有效位M11到最低有效位M0，M11到M0表示方法的12位编码。
C1和C0两位表示类的编码。比如对于binding方法来说:

- **0b00表示request**,
- 0b01表示indication,
- **0b10表示success response**,
- 0b11表示error response

每一个method都有可能对应不同的传输类别。

```C
For example,
a Binding request has class=0b00 (request) and method=0b000000000001
(Binding) and is encoded into the first 16 bits as 0x0001.
A Binding response has class=0b10 (success response) and
method=0b000000000001, and is encoded into the first 16 bits as
   0x0101.
```

STUN是C/S协议，支持两种事务：

1. **Request/Response**事务：由客户端给服务器发送请求，并等待服务端返回响应，用于确定一个NAT给客户端分配的具体绑定。客户端通过事务ID将请求响应连接起来。
2. **Indication transaction**事务：由服务器或者客户端发送指示，另一方不产生响应，用于保持绑定的激活状态。

对于 **Request/Response**事务，Transaction ID 由客户端生成，服务端在 response 中回应与 request 一致的事务 ID。

对于 **Indication** 事务，Transaction ID 由 STUN agent 随机生成，由于 indication 没有响应，因此事务 ID 仅用于辅助 debug。
Transaction ID 的主要作用是将 request 和 response 关联起来。

1.5 STUN属性类型

STUN 消息头后跟着多个属性，每个属性都采用 TLV 编码，type 为 16 位的类型、lenght 为 16 位的长度、value 为属性值。

每个 STUN 属性必须是 4 字节对齐。lenght 字段的值只表示 TLV 中 V(Value) 的长度，既不包括 T(Type) 和 L(length)，又不包括 padding 填充数据的长度。具体如下所示：

STUN 请求和响应都包含消息属性。消息属性分为强制理解属性和可选理解属性，具体如下所示：

在ICE中，包含STUN用到的几个属性，具体如下所示：

1.6 部分属性介绍

MAPPED-ADDRESS:用于表示客户端外部IP地址，如果没有NAT，那么外部IP地址和内部IP地址是相同的。前8位保留，之后8位用于表示IP类型（IPV4/6）。之后16位表示端口号。这里强制使用IPV4版本，所以Address是32位：

- Family：IP类型，0x01-IPV4、0x02-IPV6。
- Port：端口。
- Address：IP地址。

RESPONSE_ADDRESS：用于标示哪一个服务端的IP和Port发送发送的数据。数据格式参考MAPPED-ADDRESS。

XOR-MAPPED-ADDRESS：与MAPPED-ADDRESS属性基本相同，区别在于反射地址经过一次异或（XOR）处理，异或运算是其自身的逆运算，客户端经过一次异或运算获得真实的反射地址。解决ALG篡改地址和端口的问题。

USERNAME：用户名，用于消息完整性，在webrtc中的规则为“对端的ice-ufrag：自己的ice-ufrag”，其中ice-ufrag已通过提议/应答的SDP信息进行交互。

MESSAGE-INTEGRITY：STUN 消息的 HMAC-SHA1 值，长度 20 字节，用于消息完整性认证。详细的计算方式后续进行举例说明。

FINGERPRINT：指纹认证，此属性可以出现在所有的 STUN 消息中，该属性用于区分 STUN 数据包与其他协议的包。属性的值为采用 CRC32 方式计算 STUN 消息直到但不包括FINGERPRINT 属性的的结果，并与 32 位的值 0x5354554e 异或。

ERROR-CODE：属性用于error response报文中。其中包含了300-699表示的错误码，以及一个UTF-8格式的文字出错信息（Reason phrase）。

```
Basic | 复制代码
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  0                     |Class|     Number    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Reason Phrase (variable)                                 ..
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- ERROR-CODE说明

类型	说明
Class	用于存储100的倍数，这个值只能是1~6
Number	100的余数，值的范围是0~99，Error Code的计算是：`Class * 100 + Number`
Reason Phrase	ByteString类型，用于描述错误信息

- Error Code说明

Error Code	说明
400	请求格式不正确
401	请求消息不包含MESSAGE-INTEGRITY属性
420	服务器不理解请求中的强制属性
430	请求消息包含MESSAGE-INTEGRITY属性，但是却使用了一个过期的共享密钥
431	请求消息包含MESSAGE-INTEGRITY属性，但是HMAC校验失败。
500	服务器遇到了一个临时错误。客户端应该重试
600	服务器拒绝满足该请求。 客户端不应重试

REALM：此属性可能出现在请求和响应中。在请求中表示长期资格将在认证中使用。当在错误响应中出现表示服务器希望客户使用长期资格来进行认证。

NONCE：出现在请求和响应消息中的一段字符串。

UNKNOWN-ATTRIBUTES：此属性只在错误代码为420的的错误响应中出现。

SOFTWARE：此属性用于代理发送消息时包含版本的描述。它用于客户端和服务器。它的值包括制造商和版本号。该属性对于协议的运行没有任何影响，仅为诊断和调试目的提供服务。SOFTWARE属性是个可变长度的，采用UTF-8编码的小于128个字符的序列号。

ALTERNATE-SERVER：属性表示 STUN 客户可以尝试的不同的 STUN 服务器地址。属性格式与 MAPPED-ADDRESS 相同。

在Binding请求中通常需要包含一些特殊的属性,以在ICE进行连接性检查的时候提供必要信息，详细的属性如下所示：

- **PRIORITY 和 USE-CANDIDATE**：终端必须在其request中包含PRIORITY属性,指明其优先级，优先级由公式计算而得。如果有需要也可以给出特别指定的候选(即USE-CANDIDATE属性)。第一开始Binding时，可能没有 USE-CANDIDATE这个字段，当这个通道可以使用的时候，也就是ICE提名使用时，STUN消息添加该字段，表示使用该通道开始建联DTLS链接，这时候服务端开始和客户端握手建立安全加密UDP链接。
- **ICE-CONTROLLED 和 ICE-CONTROLLING**：ICE 流程中定义了两种角色:controlling 和 controlled。不同的角色在candidate pair优先级的计算，pair nominate的决策上有所不同。一般流程下，会话的双发各自的角色选择是与会话协商的流程相关的。offerer是controlling，answerer是controlled。ICE-CONTROLLED或者是ICE-CONTROLLING，这两个属性都会携带一个Tie breaker这样的字段，其中包含 一个本机产生的随机值。收到该bind request的一方会检查这两个字段，如果和当前本机的role冲突，则检查本机的tie breaker值和消息中携带的 tie breaker 值进行判定本机合适的role。判定的方法为Tie breaker值大的一方为controlling。如果判定本端变更角色，这直接修改；如果判定对端变更角色，则对此bind request发送487错误响应，收到此错误响应的一方变更角色即可。

下面抓包分析图：

Binding Request

Binding Respond Success

1.7 交互过程

（1）在WebRTC中，STUN客户端内置在浏览器用户代理中，在会话建立之前，先发送stun测试报文，以便浏览器确定其是否位于NAT之后并发现映射地址和端口。

（2）当STUN服务器收到STUN Binding请求时，它会记录Binding请求来自哪个IP地址和端口号，此地址和端口号随后将以STUN Binding响应的形式返回客户端。（通过XOR-MAPPED-ADDRESS属性）。

（3）客户端将响应中发来的IP地址和端口与其发送的IP地址和端口进行比较，以此来判断客户端和服务器之间有没有NAT，若不同，则说明至少有一个NAT，客户端能够识别由最外层的NAT分配的

IP地址和端口。存在多个NAT时，STUN只能识别最外层NAT的相关信息。

（4）STUN服务器将源传输地址复制到STUN Binding响应中XOR-MAPPED-ADDRESS属性中，并将绑定响应发送回STUN客户端。当这个数据包通过NAT返回时，NAT将修改IP报头中的目的传输地址，但是STUN响应主体中XOR-MAPPED-ADDRESS属性中的传输地址将保持不变。通过这种方式，客户端可以了解最外面的NAT相对于STUN服务器分配的反射传输地址。

2 ICE的一些概念

2.1 ICE角色

分为 controlling和controlled。

- offer(主动发起) 一方为controlling角色，
- answer(被动接受)一方为controlled角色。

也就是 full ice agent必须是 controlling role， lite ice agent 是controlled （所以srs（仅支持lite ice） webrtc是客户端发送的offer，然后srs回应answer）。

2.2 ICE的模式

- **FULL ICE:**是双方都要进行连通性检查，完成的走一遍流程。ice客户端实现，这种模式既可以收binding request,也可以发binding reques。

- **Lite ICE:** 在FULL ICE和Lite ICE互通时，**只需要FULL ICE一方进行连通性检查， Lite一方只需回应response消息。**这种模式对于部署在公网的设备比较常用。只接受binding request请求，并且回复。不会主动发送binding request请求给对方，sdp中有 a=ice-lite 字样（比如srs服务器采用lite-ice模式）。

2.3 Candidate地址

媒体传输的候选地址，组成candidate pair做连通性检查，确定传输路径，有如下属性：

Type 类型:

- **Host(**Host Candidate**)**: 这个地址是一个真实的主机，参数中的地址和端口对应一个真实的主机地址，这个地址来源于本地的物理网卡或逻辑网卡上的地址，对于具有公网地址或者同一内网的端可以用。
- **Srvflx(**Server Reflexive Candidate**)**: 这个地址是通过 Cone NAT 反射的类型，参数中的地址和端口是端发送 Binding 请求到 STUN/TURN server 经过 NAT 时，NAT 上分配的地址和端口。
- **Relay(**Relayed Candidate**)**: 这个地址是端发送 Allocate 请求到 TURN server ，由 TURN server 用于中继的地址和端口，该地址和端口是 TURN 服务用于在两个对等点之间转发数据的

地址和端口，是一个中继地址端口。这个地址是端发送 Allocate 请求到 TURN server ，由 TURN server 用于中继的地址和端口（这个可能是本机或 NAT 地址）

- **Prflx**(Peer Reflexive Candidate)：这个地址是通过 发送STUN Binding时，通过Binding获取到的地址。在建连检查期间新发生，参数中的地址和端口是端发送 Binding 请求到 STUN/TURN server 经过 NAT 时，NAT 上分配的地址和端口。这个地址是端发送 Binding 请求到对等端经过 NAT 时，NAT 上分配的地址和端口

  PeerA PeerB （ Full Cone <---> [Restricted Cone, Port Restricted Cone, Symmetric] ）

  如果对方是 Full Cone 类型，对方就可以提供 prflx 类型的 candidate 了，个人理解，仅供参考。

注1：Cone NAT表示锥形NAT， Symmetric NAT表示对称NAT。

注2:：Prflx这种地址类型是两个点建立连接时，通过STUN的检查发现的新地址。

注3：和Srvflx地址区别就是：Srvflx地址是通过信令服务拿到的，但Prflx地址是通过Binding拿到的，但这两个地址可能是一样的，但是获取的途径不一样。

- **Componet ID**
  传输媒体的类型,1代表RTP;2代表 RTCP。
  WebRTC采用Rtcp-mux方式，也就是RTP和RTCP在同一通道内传输，减少ICE的协商和通道的保活。

- **Priority**
  Candidate的优先级。
  如果考虑延时，带宽资源，丢包的因素，Type优先级高低一般建议如下顺序：host > srvflx > prflx > relay。

- **Base**
  是指candidate 的基础地址。
  Srvflx address 的base 是本地host address。
  host address和 relayed address 的base 是自身。

注意：

由本端和远端candidate组成的pair，有自己的优先级。

pair优先级的计算是取决candidate的priority。

priority = 2^32*MIN(G,D) + 2*MAX(G,D) + (G>D?1:0)。

G:controlling candidate 优先级。

D:controlled candidate 优先级。

ICE选择高优先级的candidate pair。

sdp 中 candidate示例：
“a=candidate:1 1 UDP 9654321 212.223.223.223 12345 typ srflx raddr 10.216.33.9 rport 54321”
“a=candidate:foundation dataType protocol 优先级 212.223.223.223 12345 typ srflx raddr 10.216.33.9 rport 54321”

表示 foundation为1，媒体是RTP，采用UDP协议，优先级为9654321，公网映射地址为212.223.223.223:12345，type为srflx，base地址为10.216.33.9:54321

3 ICE过程

3.1 收集 candidates

客户端无法知道自己的外网ip，需要发送stun包给stun服务，stun服务返回对应客户端的出口ip和端口，**返回来的地址和自己本地地址做比对就知道NAT类型**。

根据Componet ID:

- 获取本机host address；
- 从STUN服务器获取 srvflx address；
- 从TURN服务器获取 relay address；
- 同时生成foundation。

3.2 删除重复的candidate

收集地址完成后，需要去掉重复的candidate，如果两个candidate的地址一样，并且Base地址也一样，则删除它。

3.3 交换candidates

ICE交换candidates方式可以使用sdp交换，也可以使用单独信令交换，sdp交换时如下：

- ICE 使用offer/answer方式，双方通过SDP协商交换candidate信息；
- Candidate信息包括type,foundation,base,component id,transport。

SDP a行格式如下:

“a=candidate:1 1 UDP 9654321 212.223.223.223 12345 typ srflx raddr 10.216.33.9 rport 54321”“a=candidate:foundation dataType protocol 优 先 级 212.223.223.223 12345 typ srflx raddr 10.216.33.9 rport 54321”

表示 foundation为1，媒体是RTP，采用UDP协议，优先级为9654321，公网映射地址为212.223.223.223:12345，type为srflx，base地址为10.216.33.9:54321。

示例：

"a=candidate:240568271 1 udp 1686052607 174.139.8.82 64462 typ srflx raddr 10.1.1.19 rport 64462 generation 0 ufrag TWCy network-id 2 network-cost 50"

他的格式具体如下：

- candidate:{foundation} {component} {protocol} {priority} {ip} {port} typ {type}
- 如果存在related ip：raddr {ip} {port}
- generation {generation}

- 如果存在用户名：ufrag {username}
- 如果network_id不是0：network-id {network_id}
- network-cost {network_cost}

每一项的大概意思：

- foundation(240568271)：根据type、ip、protocol、replay_protocol计算出的字符串。一般用于比较两个candidate是否相等。
- component(1)：通道码。RTP通道码是1、RTCP是2，它指示这候选地址关联RTP通道。
- protocol(udp)：传输层类型，upd或tcp。往往认为RTP、RTCP是用UDP，但webrtc其实支持用TCP。

1. priority(1686052607)：优先级，用来和对方的candidate生成地址对后，会使用双方的优先级计算出来一个优先级，然后按照优先级排序地址对，ice选择高优先级的地址对优先建连检查
2. ip(174.139.8.82)：候选IP。它是真正须要的候选地址。当type是反射时，它就是NAT外的公网IP，此时raddr对应内网IP。
3. port(64462)：候选IP关连的端口号。
4. type(srflx)：候选地址类型。它分本地（local）、反射（srflx），中转（relay）。
5. raddr(10.1.1.19)：候选IP基于的IP。对于local类型，它不存在。是反射时，它就是内网IP。
6. rport(64462)：raddr关联的端口。
7. generation(0)：代数。初始值是0，然后会不断+1，大的代数会覆盖掉低代数的候选地址。更新candidate的时候会+1，替换老的candidate。
8. ufrag(TWCy)：用户名。
9. network-id(2)：此网卡IP在网卡集合中的索引，从1始。

如果使用单独信令交换 sdp中应该存在：
a=ice-options:trickle。

使用trickle方式，sdp里面描述媒体信息和ice后选项的信息可以分开传输，先发送sdp过去，在收集地址信息，目的是为了同时进行，而不是等待收集地址信息完成后才开始。

多说一点，其实sdp也是通过信令传输的，理论上sdp是可以不通过信令的，可以在等待两个peer建立完连接后，在交换sdp是可以的。

3.4 生成candidate pairs

在本端收到远端candidates后，将Component ID和transport protocol相同的candidates组成pair。也就是把标记传输同样数据，并且传输协议相同的candidate组成一对地址对，以后就是这两个地址建立连接。

修整candidate pair，如果是srvflx地址，则需要用其base地址替换。对端也是同样的流程。

3.5 连通性检查

将candidate pairs按照优先级排序，供连通性检查使用。其实就是把sdp里面的candidate地址和本地的candidate地址进行排队，组成一个checklist表，生成按优先级排序的链表，按优先顺序发起每个候选地址对的检查。

由连通性检查成功的candidate pair按优先级排序的链表，用于ICE提名和选择最终路径。连通性检查完毕后，开始进行优先级排序。

检查表中的每个候选对都有foundation和state。foundation是 Local的和Remote的的结合。一旦开始检查，就分配已计算每个媒体流的列表，其实就是开始发起打洞流程，比如开始发起stun binding rquest请求，收到bind respond后，认为是成功的可用的，准确的说就是互相发起stun binding request和收到request后在回包给对方Ordinary checks 两端都按照各自checklist分别进行检查。

Triggered checks 收到对端的检查时，也在对应的candidate pair上发起连通性检查，以提高效率。

如果checklist里有relay candidate，则必须首先为relay candidate创建permission。permission其实就是一个许可，如果没有创建许可，发送过去的包将被丢弃(针对TURN时使用)。

然后发送连通性检查请求。

ICE 使用STUN binding request/response，包含Fingerprint检验校验机制。

如果A收到B的response，则代表连通性检查成功，否则需要进行重传直到超时。

在建立连接时，如果没有响应，则会以RTO时间进行重传，每次翻倍，直到最大重传次数。

STUN请求采用STUN short-term credential方式认证，也就过一段时间如果没有stun包发送时，这个连接会过期失效，因此需要不断地发送stun包并收到回复的stun包，用来保持连接有效性。刚开始建联时，大概以50ms间隔频率发送，后期稳定后是以2.5s的间隔频率发送，维持链接有效性。

STUN USERNAME属性 ”RemoteUsername：localUsername”

两端在SDP协商时交换ice-pwd和ice-ufrag，以得对端用户名和密码。计算stun包里面的MESSAGE-INTEGRITY时，需要自己本地的ice-pwd去计算HMAC-SHA1，生成对应的属性值串，用来检查消息的完整性，检验被篡改。

STUN 检查请求中需要检查地址的对称性，请求的源地址是响应的目的地址，请求的目的地址是响应的源地址，否则都设置状态为 Failed。

3.6 生成validlist

将连通性检查成功的candidate pair并按优先级排序加入validlist，这时本地candidate填写的是公网映射地址，remote candidate填写的是对端发送的STUN binding request地址。

3.7 提名candidate pair

由controlling来提名哪对candidate pair为valid pair。提名方式又分为普通提名和进取型提名。

普通提名方式会做两次连通性检查，在第一次做连通性检查时不会带上USE-CANDIDATE属性，而是在生成的validlist里选择pair再进行一次连通性检查，这时会带上USE-CANDIDATE属性，并且置位nominated flag（ICE提名地址对）。

进取型方式则是每次发送连通性检查时都会带上USE-CANDIDATE属性，并且置位nominated flag（ICE提名地址对），不会再去做第二次连通性检查。

3.8 选择最终传输地址

ICE在提名的valid pair里选择优先级最高那对作为本次ICE流程传输地址。然后开始建立DTLS连接，开始握手，交换证书，握手成功。

4 ICE状态

1. **Waiting**：还未开始连通性检查，从checklist中选择合适优先级的pair进行检查；
2. **n-Progress**：连通性检查已经开始，但还未结束；
3. **Succeeded**：该pair 连通性检查已经完成并且成功；
4. **Failed**：失败；
5. **Frozen**：连通性检查还未开始。

5 ICE保活

1. 对于每个ICE通道，都需要为其会话进行保活。
2. 采用STUN binding request或者STUN binding indication。
3. 如果没有收到响应，则会重传，直到最大重传次数。

6 ICE角色冲突解决

1. 当两端角色都为controlling或者controlled角色冲突时，在连通性检查阶段，要求发送binding request消息里必须要带上tie-breaker属性。
2. 当出现冲突时，比较tie-breaker大小，值比较大的则被认为是controlling，同时回应487错误给对端，对端收到487错误后切换角色。

7 参考文档

相关RFC文档：

ICE文档： https://datatracker.ietf.org/doc/html/rfc5245 ICE相关

RFC5389: https://datatracker.ietf.org/doc/html/rfc5389 新版本的STUN

RFC3489: https://datatracker.ietf.org/doc/html/rfc3489 Classic STUN

RFC5766： https://datatracker.ietf.org/doc/html/rfc5766 STUN扩展之TURN协议

RFC8455： https://datatracker.ietf.org/doc/html/rfc8445 ICE收集提名等

STUN协议解析 https://blog.csdn.net/momo0853/article/details/105387675