NAT 的会话穿透用法（STUN）

1. 简介

本说明书中说定义的协议—会话穿透 NAT 用法，为处理 NAT 提供了一种工具。它提供了一种为端点决定由 NAT 分配的和端点本身私有地址和端口相关联的 IP 地址和端口的方法。他也提供一种保持 NAT 绑定的方法。本协议的扩展可用于两端点的连接检测[MMUSIC-ICE]，或者两端点之间的包中转[BEHAVE-TURN].

为保持工具的特性，本说明书定义了一种可扩展的包格式，几种传输层协议之上的操作以及两种鉴权形式。

STUN 被试图用于解决一种或者多种 NAT 穿透。这些解决方案即是为我们所知的 STUN 用法。每种用法描述了 STUN 怎样被用于实现 NAT 穿透解决方案。典型地，一种用法指明什么时候 STUN 消息被发送，哪种可选属性被包含，什么服务器被使用，以及什么鉴权方式被使用。交互式连接建立(ICE) [MMUSIC-ICE] 是 STUN 的一种使用方式. SIP Outbound[SIP-OUTBOUND]是另外的一种 STUN 用法。在某些情况下，一种用法需要扩展 STUN，一种 STUN 扩展能以新的方法，属性，或者错误响应码的形式出现。更多的 STUN 用法信息能够在第 14 节找到。

2. 源于 RFC3489 的演变

STUN 最初定义在 RFC3489。RFC3489 有时候被成为“经典的 STUN”，它是以能完全解决 NAT 穿透问题的解决方案来描述的。在该解决方案中一个客户端能够发现它自己是否处在 NAT 的后面，判定 NAT 的类型，发现 NAT 外围的公网 IP 地址和端口，然后利用这个 IP 地址和端口填在协议体中，参见 RFC3261。然而，自从 RFC3489 发布后，经验发现经典的 STUN 简直不能按一种分布式的解决方案来有效地工作。通过经典 STUN 获得的 IP 地址和端口有时候对端到端的通讯有用，但是有时候没有用。经典的 STUN 没有提供方法来是否应该发现，事实上，工作还是不工作，并且在它不能工作的情况下没有提供补救措施。更有甚者的是，经典 STUN 的判断 NAT 类型的分类算法被发现是不完美的，许多 NAT 不完全符合由它定义的类型。

经典 STUN 也存在一种安全弱点—攻击者能够在特定的拓扑和限制下提供错误的映射，这基本上不能通过加密方式来解决。尽管这个问题在本说明书中仍存在，但是通过使用 STUN 更完善的解决方案使得攻击被缓解了。

鉴于这些原因，本说明书淘汰了 RFC3489，并且将 STUN 描述成一种工具用于完全穿透 NAT 解决方案的一部分。ICE [MMUSIC-ICE]是一种为像 SIP 那样基于 offer/answer [RFC3264]策略的协议提供完全的解决方案。SIP Outbound [SIP-OUTBOUND]是一种 sip 信令完全穿透的解决方案，它以完全不同的方法使用 STUN。因此，一个协议可能能够由它自己使用 STUN 作为一种解决方案，这类的用法没在这描述，由于上述原因它被强烈的阻止了。

这里描述的上了线的协议仅是轻微地改自经典 STUN。运行在 TCP 和 UDP 之上的协议的扩展是一个更加结构化的方法。一个极其巧妙的让应用程序使用 STUN 的机制是偷用 RFC3489 中定义的 128 位事物 ID 中的 32 位，允许改变向后兼容。编码的地址映射使用一种新的异或格式。还有其他的，更小的改变，参见 19 节更加完整的描述。

由于使用范围的改变，STUN 已经从"Simple Traversal of UDP through NAT"更名为 "Session Traversal Utilities for NAT"。略缩词仍为大家熟知的 STUN。

3. 操作概览

本节仅是描述性的。

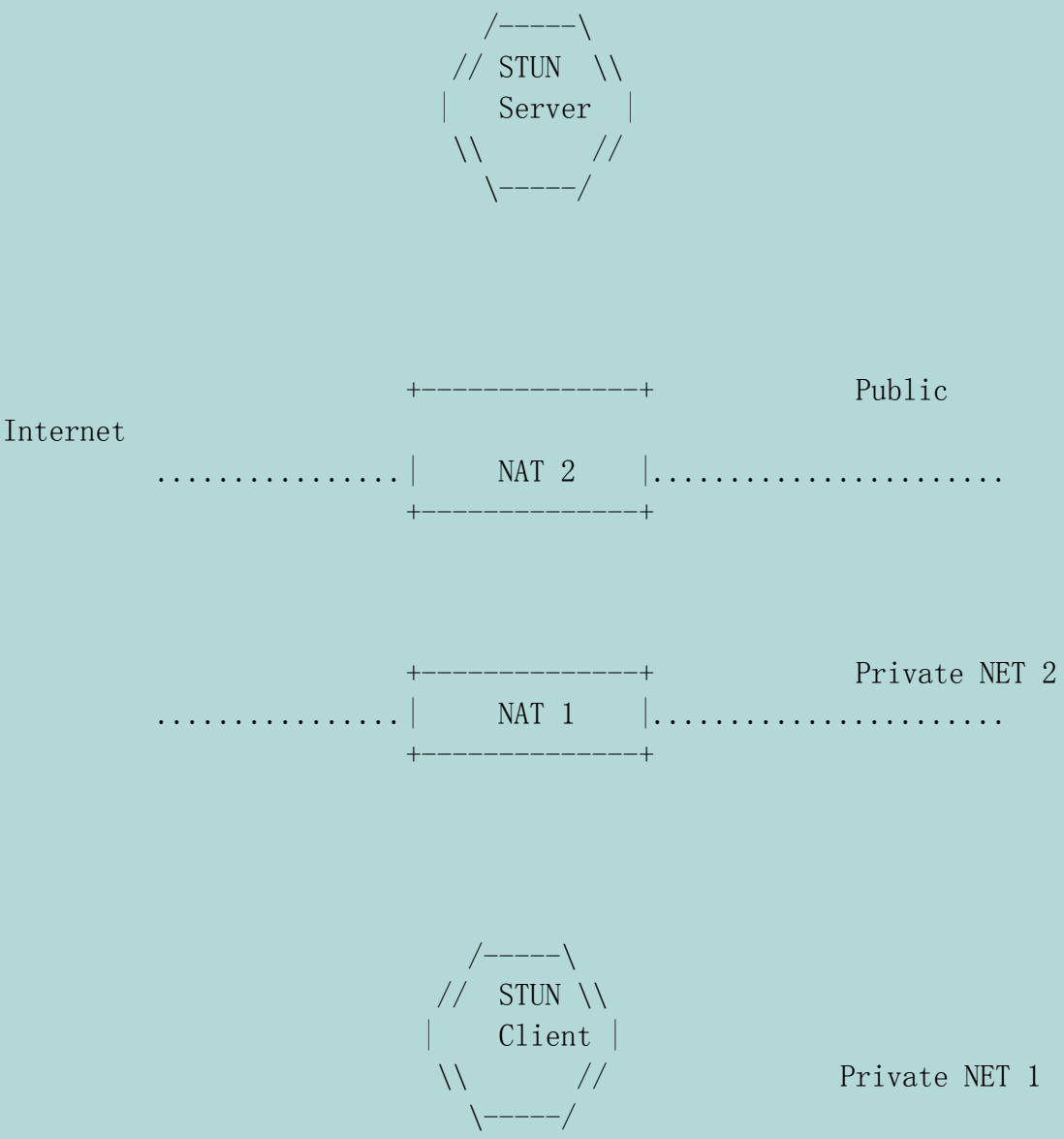


Figure 1: One Possible STUN Configuration

一种可能的 STUN 配置如图 1 所示。在这种配置下，有两种实现 STUN 协议的实体（成为 STUN 代理）。图中底下的代理是客户端，它连接到私网 1。这个私网通过 NAT1 连接到私网 2.私网 2 通过 NAT2 连接到公网。图中最上面的代理是位于公网的服务器。

STUN 是一个 client-server 协议。它支持两种类型的事物。一种是 request/response 事物，在这种事物中，客户发送一个请求到服务器，服务器返回一个响应。第二种是指示事物，在这种事物中，两种代理（client 或者 server）发送一个指示并不产生响应。两种事物都包含一个随机选择的 96 位的事物 ID 号。对请求/相应事物，这个 ID 号允许产生它的客户端将响应和请求相关联；对于指示事物，事物 ID 用于调试。

所有的 STUN 消息都以固定的头部开始，头部包含一个方法，一个类别，和一个事物

ID 号。方法指明各种各样的请求或者指示。本说明书仅仅定义一种方法—Binding，其他的方法有望在其他的文档中定义。类别字段表明该消息是一个请求，一个成功的响应，一个错误响应，还是一个指示。紧跟固定头部的是 0 个或多个的 TLV 属性，为具体消息传递附加信息。

本文档定义一个唯一的称作 Binding 的方法。该方法能被用于请求/响应事物或者指示事物。当被用在请求/响应事物中的时候，方法可以用于决定一个 NAT 分配给 client 的特定的 binding。当被用在请求/响应事物或者指示中的时候，方法可以用于保持这种 binding 有效。在 binding 请求/响应事物中，绑定请求由 STUN 客户端发送到 STUN 服务端。当绑定请求到达 server 的时候，他可能穿过 STUN Client 和 STUN Client 之间的一个或者多个 NAT（在图 1 中有两个这样的 NAT）。当绑定请求通过 Nat 的时候，Nat 将修改数据包的源通讯地址（指源 IP 地址和源端口号）。因而，server 收到的请求的源通讯地址将是由 NAT 创建的接近 server 那边的公网的 IP 和端口。这个地址成为反射通讯地址。STUN server 复制该源通讯地址到一个 STUN 绑定响应的 XOR-MAPPED-ADDRESS 属性中，并返回该绑定响应到客户端。当该响应数据包返回通过 NAT 的时候，NAT 将修改 IP 头的目的地址，但是包含在 STUN 响应数据包体 XOR-MAPPED-ADDRESS 属性中的通讯地址将不会被修改。这样，client 能够学习到最外面 NAT 分配的关于 STUN server 的通讯地址。

在某些情况下，STUN 必须服用其他的协议（如[MMUSIC-ICE], [SIP-OUTBOUND]）。在这些情况下，必须有一种方法来监视一个包，判定它是一个 STUN 包还是不是。STUN 在包头提供三种有固定值的字段来达到该目的。如果这样不够的话，STUN 数据包还可以包含一个 FINGERPRINT 值，能够用于区分数据包。

STUN 定义了一系列的一个用法能够决定使用的可选过程，称为机制。这些机制包括 DNS 发现，对于备选 server 的重定向技术，一个用于复用的指纹属性，以及两个鉴权和消息完整交换。鉴权机制解决一个用户名，密码，以及消息完整值的用途。两种鉴权机制，长期证书机制和短期证书机制被定义在本说明书中。每一种用途详述了该用途允许的机制。

在长期证书机制中，客户和服务器共享一个预先备置的用户名和密码并执行一个源自（但是细节不同）HTTP 定义的领会挑战/响应交换。在短期证书机制中，client 和 server 通过一些优先于 STUN 交换的带外方法交换用户名和密码。这些用于完整性保护和鉴权请求和响应。没有挑战或暂未使用。

4. 术语

在本文档中，关键词”必须”,”不允许”,”要求”,”可以”,”不可以”,”应该”,”不应该”,”建议”,”可能”,”可选” 是根据 BCP14,RFC 2119[2]的规范描述 STUN 实现需要的不同层次。

5. 定义

STUN Agent：一个 STUN Agent 是一个实现了 STUN 协议的实体。该实体既可以是 STUN 客户端，也可以是 STUN 服务端。

STUN Client：一个 STUN client 是一个发送 STUN 请求和接受 STUN 响应的实体。一个 STUN 客户端能够发送指示。在本说明书中，词条 STUN client 和 client 是同义词。

STUN Server：一个 STUN client 是一个发送 STUN 响应和接受 STUN 请求的实体。一个 STUN 客户端能够发送指示。在本说明书中，词条 STUN server 和 server 是同义词。

Transport Address：IP 地址和端口号的组合（如 UDP 和 TCP 端口号）。

Reflexive Transport Address：一个客户端学习到的能够标示该客户端的被 server 看得到的通讯地址。当一个客户端和另一个主机被一个 NAT 阻隔的时候，反射通讯地址描述 NAT 公网一边分给客户端的映射地址。反射通讯地址从 STUN 响应消息中的映射地址属性（MAPPED-ADDRESS or XOR-MAPPED-ADDRESS）学习到。

Mapped Address：和反射地址一个意思。这个词条保留仅仅是由于历史原因以及归功于属性名 MAPPED-ADDRESS 和 XOR-MAPPED-ADDRESS。

Long-Term Credential：一个描述客户端和服务端共享私密的用户名和密码。长期证书通常授给 client 当订阅者注册一个服务的时候并保持到订阅者离开改服务或者显式改变证书关系。

Long-term Password：长期证书的密码。

Short-term Credential：一个描述客户端和服务端共享私密的临时用户名和密码。短期证书通过一些 server 和 client 之间的协议机制俩获得，预处理 STUN 交换。一个短期的证书由一个显式的临时域，该域可能基于一段指定的时间（例如 5 分钟）或者一个事件（例如结束一个 SIP 对话）。短期证书的指定域由应用程序用途定义。

Long-Term Password：短期证书的密码。

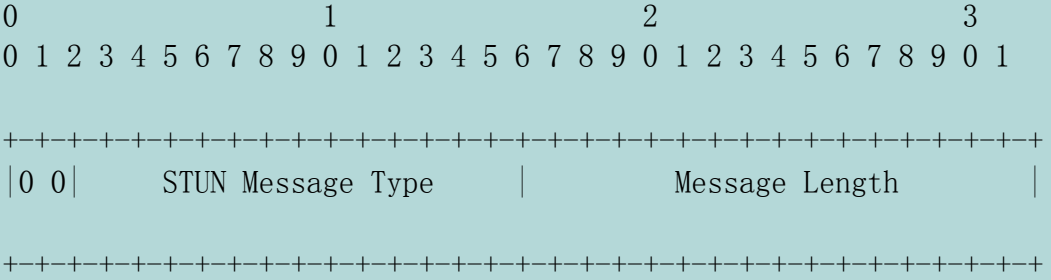
Attribute：一个能附加到 STUN 消息的类型-长度-值对象的 STUN 词条。属性分为两类：comprehension-required 和 comprehension-optional。STUN 代理可以安全地忽略它们不能理解的 comprehension-optional 属性，但是如果它包含一个不能理解的 comprehension-required 属性，它将不能成功处理一个消息。

TRO：重传时间到，定义了一个发送请求和第一次重传请求的初始的时间周期。

6. STUN 消息结构

STUN 消息使用网络顺序的二进制编码（最重要的字节或者八位优先，通常称作 big-endian）。传输顺序在 RFC791【RFC0791】的附录 B 中详细描述。注意，数据常量是十进制的（基数是 10）。

所有的 STUN 消息必须以 20 字节的头部开始，紧跟 0 个或者多个属性。STUN 头部包含一个 STUN 消息类型，巧妙的方法值，事物 ID 和消息长度。



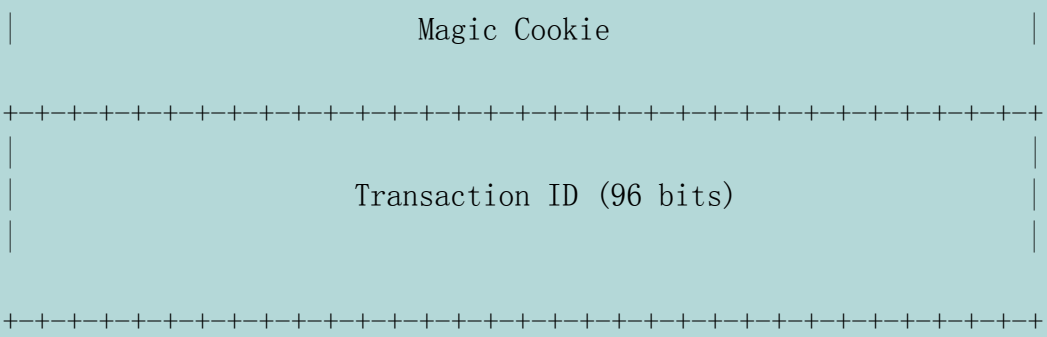


Figure 2: Format of STUN Message Header

每一个 STUN 消息最重要的 2bits 必须为 0。当 STUN 和其他协议在同一端口上复用的时候，这能够区分 STUN 的数据包。

消息类型字段定义了消息类别（请求，成功响应，失败响应，或者重定向）和 STUN 消息的消息方法（基本功能）。尽管有 4 种消息类型，但是在 STUN 中仅有两种事物：请求/响应事物（包含请求消息和响应消息）和指示事物（包含单一的指示消息）。响应类型分为错误和成功的响应来辅助快速处理 STUN 消息。消息类型域被分解在如下的结构中：

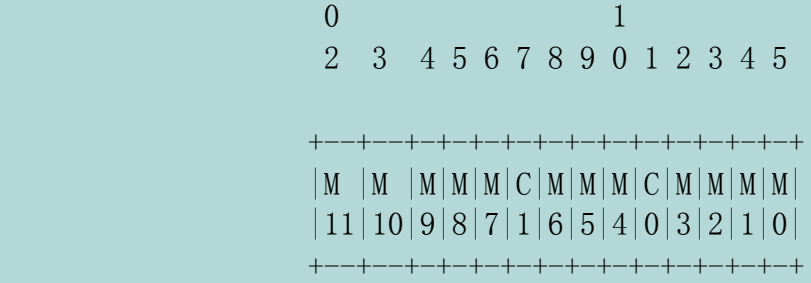


Figure 3: Format of STUN Message Type Field

消息类型域中的比特位从最重要（M11）到最不重要（M0）来显示。M11 到 M0 标示一个方法的 12 位编码。C1 和 C0 标示类型的编码。0b00 的类型是一个请求，0b01 的类型是一个指示，010 类型是一个成功的响应，0b11 是一个错误的响应。本说明书定义了唯一的一个方法：Binding。方法和类别是正交的，因此，对于每个方法，一个请求，成功响应，错误响应和标示都可能是该方法的。扩展定义的新方法必须指明对于改方法哪种类型被允许。

例如，一个绑定请求拥有类别 class=0b00（request）和 method=0b000000000001 (Binding)并且编码到前 16 位的为 0x0101。

注意：这种不幸的编码方式归功于【rfc3489】中值的安排，它没有考虑指示，成功和错误的比特域的使用。

巧妙域必须包含固定的网络序的值 0x2112A442。在 RFC3489 中，这个域是事物 ID 的一部分；把这个域放在这里允许一个主机检测客户端是否理解本修订说明书所附加的特定属性。另外，当 STUN 在相同端口上复用其他协议的时候它辅助区分 STUN 包和其他协议包。

事物 ID 是一个 96bits 的标识符，用于唯一标示一个 STUN 事物。对于请求/响应事物，该标识符被 STUN client 挑选给请求，并且有 server 在响应中传回。对于指示（indication），它由发送它的代理挑选。他主要为关联请求和响应服务，它也为特定的类别免受攻击而有一

定的作用。Server 也使用事物 ID 来作为一个经过所有客户端的每个事物的唯一关键字。由此，事物 ID 必须统一地，随机地从 0—2^96-1 中选取，可以是一个加密的随机串。重传相同的请求使用相同的 ID，但是 client 必须为一个先的事物挑选新的事物 ID，除非新的请求是面向 bit 的与先前请求相同的并且从相同的通讯地址发送到相同的 IP 地址。Success 和 error 响应必须携带和他们相关联的请求的事物 ID。当一个代理在相同端口上即是 server 有是 client 的时候，代理发送的事物 ID 和接受道德事物 ID 没有关系。

消息长度域必须包含不包括 20 字节头部的 STUN 消息的字节大小数。由于所有的 STUN 属性被填充成 4 字节的组合，因而此域的最后两 bit 总是 0.这也提供了一种区分 STUN 数据包和其他协议包的方法。

紧跟 STUN 固定头部的是 0 哥或者多个属性。每个属性都是以 TLV 编码的。编码细节和属性本身将在 15 节给出。

7. 基本的协议处理

本节定义了 STUN 协议的基本处理。他描述了消息是怎么样形成的，怎样发送，当他们到达时怎样被处理。也详细定义了绑定方法的处理。本文档其他的节描述了在特定条件下选择使用可选的处理的一种方法。其他的文档可能会定义其他的 STUN 扩展，通过附加新的方法，新的属性或者新的错误响应码。

7.1 形成一个请求消息或一个指示消息

当形成一个请求或一个指示消息，代理必须遵从第 6 节的规则构造头部。另外，消息类别必须是“Request”或者“Indication”（按实际情况而定），方法必须是绑定或者其他文档定义的方法。

代理添加方法或者用法指定的所有属性。例如，某些用法可能指定代理使用一个鉴权方法（section10）或者 FINGERPRINT 属性（section8）。

如果代理发送一个请求，它可以在请求中添加一个 SOFTWARE 属性。代理可能在指示消息中包含一个 SOFTWARE 属性，这有方法决定。STUN 的扩展应该讨论 SOFTWARE 在新的只是中是否有用。

对于没有鉴权的绑定方法，不需要任何属性除非用法指定了。

在知情的情况下，所有的通过 UDP 发送的 STUN 消息应该短语路径的 MTU。如果不知道路径的 MTU，消息应该小于 Ipv4 首跳的 576 字节【RFC1122】以及 Ipv6 首跳的 1280 字节【RFC2460】。这个值和 IP 包的总大小有关。因此，对于 Ipv4，实际的 STUN 消息长度应该小于 548 字节（576-20 字节的 IP 头部，减 8 字节的 UDP 头部，假设没有 IP 可选项被使用）。STUN 无能力解决请求小于 MTU 但是响应大于 MTU 的情况。不要认为这种限制将是 STUN 的一个问题。MTU 的限制是应该，不是必须的，来证实 STUN 本身被用于探测 MTU 的特征[BEHAVE-NAT]。

除此之外或者类似的应用，MTU 限制必须被遵从。

7.2 发送请求消息或者指示消息

代理将发送请求或者指示消息。本文档说明了怎样通过 UDP，TCP 或者 TCP 之上的 TLS 来发送 STUN 消息；其他的通讯协议将在以后被添加。STUN 用法必须指明那个通讯协议被使用，代理怎样决定接收者的 IP 地址和端口。第 9 节描述了用法可能选择使用的基

于 DNS 的方法来决定 server 的 IP 地址和端口。STUN 可能使用任播地址，但是仅用于 UDP 和鉴权没有使用的用法。

任何时候，一个 client 可能发送多个外发请求到相同的 server（即是，多个不通事物 ID 的事物被处理）。不像其他新事物的几率限制（例如 ICE 连接检查或者当 STUN 运行在 TCP 之上的时候），一个 client 应该经过 RTO 间隔为 server 处理新事物滨且应该限制他自己在去 10 个外发事物对同一个 server。

7.2.1 通过 UDP 发送

当在 UDP 之上运行 STUN，STUN 数据包可能被网络丢掉。可靠的 STUN 请求/响应事物通过客户事物本身来重传请求消息来完成。STUN 只是不会被重传；因此，通过 UDP 的指示事物是不可靠的。

客户端应该间隔一个 RTO 重传一个请求消息，每次重传，重传间隔时间翻倍。RTO 是 round-trip（RTT）的一个估计值，它的计算于 RFC2988[RFC2988]描述，除两个例外。首先，RTO 的初始值应该是可配置的（而不是 RFC2988 中推荐的 3s）并且应该大于 500ms。这种例外的“应该”情况是当请他机制被用来获取拥塞门限（如 ICE 为固定流率定义的那些），或者 STUN 用于不知道网路功能的非 Internet 环境。在固定线路接入链路，500ms 这个值是被推荐的[KARN87]。当应用于 STUN，RTT 估计值不应该根据导致重传请求的 STUN 事物来计算。

客户端结束事物后应该缓存 RTO 值，并且作为下个事物重传到相同 server 的初始值（基于相同 IP 地址的）。该值应该在 10 分钟后被考虑失效并被丢弃。

重传会继续直到一个响应被收到，或者知道一个完全的 Rc 请求被发送。Rc 应该是可配置的并且应该有一个默认值 7。在最后的请求之后，如果持续时间等于 Rm 时间的 RTO 超时而没有收到响应（只要这个终结请求成功，提供了足够的时间来等待一个响应），客户端应该考虑事物已经失败了。Rm 应该是可配置的并且应该由一个默认值 16。如果有一严重的 ICMP 错误，一个 STUN 事物会被考虑失败【RFC1122】。例如，假设 RTO 的值是 500ms，请求会在 0ms，500ms，1500ms，3500ms，，7500ms，和 31500ms 被发送。如果 client 在 39500ms 后没有收到响应，客户端会考虑事物超时。

7.2.2 通过 TCP 或者 TCP 上的 TLS 发送

对于 TCP 或者 TCP 上的 TLS，客户端开启一个连接到 server。

对于 STUN 的某些用法，STUN 只能通过 TCp 协议来发送。在这种情况下，它能够在没有任何辅助的分帧和复用下被发送。在另外一些用法下，有其他的扩展，他可能和其他数据复用 TCP 连接。

在这种情况下，STUN 必须运行在允许代理提取完整的 STUN 消息和完整的应用层消息的用法和扩展指定的某些分帧协议的顶层。STUN 服务运行在众所周之的端口或者由第 9 节中单独为 STUN 描述的 DNS 过程所发现的端口，并不能和其他数据复用。因此，没有分帧协议被用于连接这些 server。当附加的分帧被使用，用法会说明客户端怎样应用它以及哪个

端口来连接。例如，在 ICE 连接检测的情况下，这些信息通过服务器和客户端的带外协商来学习到。

当 STUN 被自己运行在 TCP 之上的 TLS 时，TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite 必须以最小值实现。这中实现也可能支持任何其他的 ciphersuite。当他接受到 TLS 的认证消息后，可弧度啊你应该验证认证并且探测认证所标示的位置。如果认证是无效的或者被取消，或者如果他没有标示合适的组，客户端不允许发送 STUN 消息否则继续进行 STUN 事物。Client 必须验证 server 的身份。为了达到这一点，它遵从 RFC2818【RFC2818】中 3.1 节定义的身份验证过程。该过程假设客户端引用一个 URI。为了此使用，client 处理第 8.1 节中的域名和 IP 地址为被解析引用的 URI 主机的一部分。然而，一个 client 可能被配置一系列可证书的域名和 IP 地址；如果一个标示其中一个域名或者 IP 地址认证被收到，client 认为 server 的身份被验证。

当 STUN 被运行于和其他基于 TCP 之上的 TLS 协议复用连接，代理的密码套件和 TLS 处理过程操作将被其他协议定义。

可靠的通过 TCP 或者 TCP 之上 TLS 的 STUN 将有 TCP 自己处理，在 STUN 协议层没有重传。然而，对于请求/响应事物，如果 client 在它发送 SYN 建立连接后经过 Ti 秒没有接受到响应，它认为事物已经超时。Ti 应该是可配置的并且应该有一个默认值 39.5s。这个值被挑选用来为默认的初始化 RTO 等同 TCP 和 UDP 的超时值。

另外，如果 client 不能建立 TCp 连接，或者在响应被接受前连接被重置或者失败，任何正在进行的请求/响应事物都被认为失败。

Client 可能通过一个 TCP（或者 TLS-over-TCP）连接发送多个事物，并且它可能发送其他请求在接受先前的请求响应之前。Client 应该保持连接是打开的直到：

- 没有进一步的 STUN 请求或者指示通过该连接发送，并且
- 没有计划使用通过该连接发送的 STUN 请求学习到的任何资源（如一个映射地址(MAPPED-ADDRESS or XOR-MAPPED-ADDRESS) 或者一个中传地址[BEHAVE-TURN]），并且
- 如果在该端口复用其他的应用层协议，使用完其他协议之后，并且
- 如果使用从一个已经建立通讯的远程伙伴学习到的被某些TCPNAT 穿越技术需要的端口（e.g., [MMUSIC-ICE-TCP]）。

在 server 端，server 应该保持连接打开，让 client 关闭，除非 server 决定连接已经超时（例如，由于 client 从网络断开）。当连接仍然打开在 NAT 阻拦的时候，Client 学习到的绑定将仍然有效。只有 client 知道它需要绑定多长时间。如果接收到请求的响应没有发送出去，Server 不应该关闭连接。不允许一个 server 为了发送一个响应向 client 另开一个连接。在过载的情况下，Server 应该采用有关连接管理的最佳的做法。

7.3 接受一个 STUN 消息

本节说明一个 STUN 消息的处理过程。这里的处理过程说明是针对本说明书定义的 STUN 消息；附加的向后兼容规则将在第 12 节定义。这些附加的过程是可选的，并且用法

能够选择使用他们。首先，一系列的处理操作被应用取于类别。这遵从具体类别处理，如后面的子节所描述。

当一个 STUN 代理接受到一个 STUN 消息，它首先检查该消息是否遵从第 6 节描述的的规则。它检查首两位是否是 0，然后是 magic cookie 域是否是正确的值，然后消息长度是否是切合实际的，然后检查方法域的值是否是支持的方法。如果消息类别是“success response”或“Error Response”，代理将检查事物 ID 匹配的事物是否仍在处理过程中。如果 FINGERPRINT 扩展被使用，代理检查 FINGERPRINT 属性是否存在并且他的值时都正确。如果任何错误被检测到，消息将毫无声息地被丢弃。在 STUN 和其他协议复用的时候，一个错误可能表示消息不是 STUN 消息；在这种情况下，代理应该试图解析出消息是另外协议的。

STUN 代理为用法指定的鉴权机制做任何需要的检查（参看第 10 节）。

一旦鉴权检查被处理，代理检查消息的未知的属性和熟知的但不期望的属性。未知的但可选理解的属性必须被代理忽略。熟知的但不期望的属性应该被代理忽略。未知而要求理解的属性产生的处理过程依赖消息类别，将在后面描述。

在这种情况下，更进一步的处理依赖于请求的消息类别。

7.3.1 处理一个请求

如果请求包含一个或者多个要求理解的属性，server 回复一个 420 的错误响应码（未知属性）。并且包含一个 UNKNOWN-ATTRIBUTES 属性在未知的要求理解的属性列表响应中。

当运行在 UDP 上的时候，server 接受到的请求可能是一个事物的第一个请求或一个重传。Server 必须响应重传满足如下条件的属性被保留：如果 client 接受到重传的响应并且响应是发送给起始的请求，client 和 server 的全部状态等同于起始重传的响应被收到的情况，或者所有响应都收到（在这种情况下，client 使用先收到的）。Server 最早的满足此要求的方法是在过去的 40s 内记住所有通过 UDP 收到的事物 ID 和他们相关的响应。然而，这要求 server 保存状态，并且对不要求鉴权的请求不合适。另一种方法是重处理请求和冲计算响应。最新的技术必须仅使用于幂等的请求（当一个请求的相同请求能够被安全重复而不影响系统的全局状态的时候，该请求被认为是幂等的）并且对于相同的请求导致相同的响应。绑定方法被认为是等幂的。注意相当少的网络事件能产生反射通讯地址值的改变，STUN 的扩展必须讨论 server 上的不存储事物状态的请求重传的实现。

7.3.1.1 形成一个成功或错误的响应

当形成响应（成功或错误）的时候，server 遵从第 6 节中的规则。响应的方法和请求相同，消息能是“success response”或“Error Response”。

对于错误的响应，server 必须添加上述处理过程指定的包含错误码的 ERROR-CODE 属性。错误原因短语不是固定的，但是应该在某种程度上符合错误码。对于特定的错误，附加属性被添加到消息。这些属性有错误码指定的描述拼出。例如，对于错误码 420（未知属性），

server 必须包含一个 UNKNOWN-ATTRIBUTES 属性。特定的鉴权错误也导致属性被添加（参见第 10 节）。扩展可能定义其他的错误和/或者附加属性在错误的情况下被添加。

如果 server 使用一个鉴权机制鉴权，server 应该添加合适的鉴权属性到响应中（参加第 10 节）。

Server 也添加由具体方法或用法要求的任何属性。另外，server 应该添加 SOFTWARE 属性到消息。

对于绑定方法，不要求额外的检查除非用法指定了。当形成成功的响应，server 添加一个 XOR-MAPPED-ADDRESS 属性到响应，属性内容是请求消息的源地址。对于 UDP，这是请求的源 IP 地址和源 UDP 端口。对于 TCP 和 TLS-over-TCP,这是 TCP 连接的 server 可见的源 IP 地址和源 TCP 端口。

7.3.1.2 发送成功或错误的响应

响应（成功或错误的）在接受请求的相同端口上发送。如果请求是在 UDP 上接受到的，目的地址和端口号就是接收到请求的源 IP 地址和端口号，并且源 IP 地址和端口号就是接受到请求的目的 IP 地址和端口号。如果请求是通过 TCP 或 TLS-over-TCP 接受的，响应从接受请求的相同连接发送。

7.3.2 处理一个指示

如果指示包含未知的要求理解的属性，指示被丢弃或者被停止处理。

代理进行用法和方法要求的额外的任何检查。如果所有的检查成功，代理将处理改指示消息。对指示消息不产生响应。

对于绑定方法，没有要求额外的检查或处理，除非用法指定。代理接受消息仅仅在有 NAT 阻隔的时候刷新“binding”。

当指示消息不是通过 UDP 重传的时候，发送代理没有必要处理指示重传。

7.3.3 处理一个成功响应

如果成功响应包含未知的要求理解的属性，响应被丢弃并且事物被认为失败。

Client 做方法或用法指定的任何额外的检查。如果所有检查通过，client 将处理成功响应。

对于绑定方法，client 检查响应中的 XOR-MAPPED-ADRESS 属性。Client 检查地址族指定的地址。如果他是一个不支持的地址族，属性将被忽略。如果是一个支持的但是不期望的地址族（例如，绑定事物通过 Ipv4 发送，而地址族指定的却 Ipv6），client 可能接受并使用该值。

7.3.4 处理一个错误响应

如果错误响应包含未知的要求理解的属性，或者如果错误响应不包含一个ERROR-CODE 属性，事物被简单地认为失败。

Client 然后做鉴权机制指定的任何处理（参见第 10 节）。这将试图产生一个新的事物。

在这种情况下的处理过程依赖于错误码，方法和用法；一下是默认的规则：

- 如果错误码是 300 到 399，client 应该考虑事物失败除非 ALTERNATE-SERVER 扩展正在被使用。参见第 11 节。
- 如果错误码是 400 到 499，client 声明事物失败，在 420 的情况下（未知属性），响应应该包含一个给出附加信息的 UNKONW-ATTRIBUTES 属性。
- 如果错误码是从 500 到 599 的，client 可能重发请求；client 不必限制他们重发的时间。

任何其他的错误代码使 client 认为事物失败。

8. FINGERPRINT 机制

这一节描述一个可选的 STUN 机制，当两种协议复用相同的通讯地址的时候，这种机制辅助区分 STUN 和其他协议的数据包。这种机制是可选的，并且如果当它被使用的时候 STUN 用法必须描述。FINGERPRINT 机制不向后兼容 RFC3489，并且在兼容被要求的环境下不能被使用。

在一些用法下，STUN 消息和其他协议复用相同的通讯地址，例如实时传输协议（TRP）。为了应用第 7 节中的处理过程，STUN 消息必须首先从应用层数据包分离。

第 6 节描述的 STUN 头部的三个固定域能够为这个目的使用。但是在某些情况下，这三个固定域可能不够用。

当 FINGERPRINT 扩展被使用，一个代理包含 FINGERPRINT 属性的消息被发送到其他代理。第 15.5 节将描述该属性的位置和值。当代理收到一个他认为是一个 STUN 消息的时候，然后，做一些基本检查之后，代理也检查包含 FINGERPRINT 属性的消息以及该属性是否包含正确的值。第 7.3 节描述了一个 STUN 消息全局处理时候的 FINGERPRINT 检查动作。额外的检查帮助代理检测其他协议的消息，这可能类似 STUN 消息。

9. server 的 DNS 发现

这一节描述一个 STUN 的可选过程，允许 client 使用 DNS 决定服务器的 IP 地址和端口号。如果当扩展被使用的时候一个 STUN 用法必须描述。为使用这一过程，client 必须知道一个 server 的域名和一个服务名；用法也必须描述 client 怎样获得他们。万一 server 的域名丢失或者合法的改变或者其他理由的改变，所以 server 的域名硬性编码到软件里是不推荐的。

当一个 client 希望定位一个公网中接受绑定请求/响应事物的 STUN server 的时候，SRV 服务名为“stun”。当他希望定位一个通过 TLS 会话接受绑定请求/响应事物的 STUN server 的时候，SRV 服务名为“stuns”。STUN 用法可能定义额外的 DNS SRV 服务名。

域名由 RFC2782 指定的 SRV 过程被解析成通讯地址。DNS SRV 服务名作为输入该过程的服务名提供的。SRV 查阅的协议是客户端将通过 UDP（“udp”）和 TCP（“tcp”）运行 STUN 的通讯协议。注意仅仅“tcp”在这时候由“stuns”定义。

RFC2782 的过程允许确定待联系的 server。RFC2782 详细描述了 SRV 记录集的排序和使用细节。然而，RFC2782 仅仅陈述了，client 应该“试图连接(protocol, address, service)”却并没有给出如果失败了的细节。当按照这些过程，如果 STUN 事物没有接到响应而超时，client 应该按照 RFC2782 中定义的顺序重新请求下一个 server。当指示事物没有产生响应或超时，此类的重试仅仅对请求/响应传送有可能。

STUN 请求的 TCP，UDP 端口都是 3487。

STUN server 的管理员应该在 SRV 记录中为 TCP 和 UDP 使用这一端口。在所有情况下，DNS 中的端口必须反射 server 所监听的端口。STUN 的 TLS 的默认端口是 5349。如果 server 软件支持决定初始消息是一个 TLS 还是一个 STUN 消息，server 可以在 TCP 上运行 STUN 端口上运行基于 TLS 的 STUN。

如果没有 SRV 记录被发现，client 执行一个域名的 A 或 AAAA 记录查询。结果将是一个 IP 地址列表，其中的每一个都能在默认的端口使用 UDP 或者 TCP 来联系，依赖于 STUN 用法。对于需要 TLS 的用法，client 使用默认的 STUN TLS 端口连接其中一个 IP 地址。

10. 鉴权和消息完整性机制

本节为 STUN 描述 client 和 server 能用来提供鉴权和消息完整性的两个机制；这两个机制即是短期证书机制和长期证书机制。这两种机制是可选的，如果这些机制被使用的时候，用法必须指定。因此，client 和 server 将知道基于哪种应用知识的机制被遵从。例如，一个公网上支持 ICE 的 STUN 服务器可能没有鉴权，因此在代理中支持连接检查的 STUN server 功能可能使用短期证书机制。这两种机制的概览在第 3 节中给出。每一种机制说明了使用该机制的额外处理，扩展处理在第 7 节中说明。额外的处理发生三个不同的地方：当形成一个消息，当一个消息的基本检查执行后即收到一个消息，当详细处理错误消息的时候。

10.1 短期证书机制

短期证书机制假设优先于 STUN 事物，，客户端和服务器使用一些其他协议以用户名和密码的形式交换证书。这种证书是有时间限制的。时间限制由用法定义。例如，在 ICE 用法[MMUSIC-ICE]，两个端点使用带外信令协商用户名和密码，该用户名和密码应用于媒体会话期间。

证书机制用于在每个请求和多个响应中形成一个消息完整性检查。在长期机制中没有任何挑战和响应；因此，重试被证书的虚拟时间限制特征所阻止。

10.1.1 形成一个请求或者指示

对于一个请求或者指示消息，代理在消息中必须包含 USERNAME 和 MESSAGE-INTEGRIT 属性。MESSAGE-INTEGRITY 属性的 HMAC 的计算在 15.4 节中描述。注意密码永远不会包含在请求或指示中。

10.1.2 接受一个请求或指示

代理对消息进行基本的处理后，代理执行下面列出的有序的检查动作：

- 如果消息不包含一个 MESSAGE-INTEGRITY 和 USERNAME 属性：
  1. 如果消息是一个请求，server 必须用一个错误响应拒绝该请求。该响应必须使用一个 400 的错误代码（Bad Request）
  2. 如果消息是一个指示，代理必须毫无声息地丢弃该指示。

- 如果 USERNAME 不包含当前有效的服务器用户名值：
  1. 如果消息是一个请求，server 必须用一个错误响应拒绝该请求。该响应必须使用一个 401 的错误代码（Unauthorized）。
  2. 如果消息是一个指示，代理必须毫无声息地丢弃该指示。

- 使用用户名关联的密码，如第 15.4 节中描述的那样为消息完整性计算值。如果结果值与 MESSAGE-INTEGRITY 属性不匹配：
  1. 如果消息是一个请求，server 必须用一个错误响应拒绝该请求。该响应必须使用一个 401 的错误代码（Unauthorized）。
  2. 如果消息是一个指示，代理必须毫无声息地丢弃该指示。

如果这些检测通过了，代理继续处理请求或指示。Server 产生的任何响应必须包含 MESSAGE-INTEGRITY 属性，使用密码计算其值用来鉴权该请求。响应不允许包含 USERNAME 属性。

如果任何检测失败，server 不允许包含一个 MESSAGE-INTEGRITY 或者 USERNAME 属性在响应消息中。这是因为，在这些失败的情况下，server 不能决定计算 MESSAGE-INTEGRITY 需要的共享私密。

10.1.3 接受一个响应

Client 查找响应中的 MESSAGE-INTEGRITY 属性。如果存在，client 通过 15。4 节中定义的方法计算响应的消息完整性值。如果结果值与 MESSAGE-INTEGRITY 属性的值匹配，响应被认为鉴权了。如果值不匹配，或者如果 MESSAGE-INTEGRITY 是空的，响应必须被丢弃，就好像它从没接受到一样。这意味着重传，如果可用，将继续。

10.2 长期证书机制

长期证书机制依赖于长期证书，以用户名和密码的形式在 client 和 server 间共享。证书

被认为是长期的一旦它提供给用户，并且一直有效到用户不在是系统的订阅者，或者改变了。这是基本的传统的分给用户“登陆”的用户名和密码。

10.2.1 形成一个请求

形成一个请求有两种情况。在第一种情况下，这是第一个从 client 到 server 的请求（由 IP 地址和端口号标示）。在第二种情况下，客户端在先前请求/响应成功完成后提交后续请求。想成请求所导致的 401 或 438 响应将在第 10.2.3 节中描述，它不被认为是“subsequent request”并且不使用 10.2.1.2 中描述的规则。

10.2.1.1 第一个请求

如果 client 没有完成和 server（如主机名标示的，如果第 9 节中描述的 DNS 过程被使用，或如果没使用就是 IP 地址）之间的成功的请求/响应事物，它应该忽略 USERNAME, MESSAGE-INTEGRITY, REALM,和 NONCE 属性。换句换说，第一个请求将被发送就好像没有鉴权或消息完整性应用一样。

10.2.1.2 后续请求

一旦请求/响应事物成功完成了，client 将被 server 暂时置于某个域中，并且选一个鉴权用得用户名和密码。Client 应该暂时缓存和 server 通讯的 username, password, realm。当 client 发送一个后续请求，他应该包含 USERNAME, REALM, 和 NONCE 属性以及他们的临时缓存值。它应该包含一个由第 15.4 节描述的使用缓存密码计算的 MESSAGE-INTEGRITY 属性值。

10.2.2 接受一个请求

Server 处理完基本的请求检查之后，他按下面指定顺序的检查列表执行动作。

- 如果消息不包含一个 MESSAGE-INTEGRITY 属性，server 必须产生一个错误码为 401（为授权）的错误响应。这个响应必须包含一个 REAM 值。该值推荐使用 STUN server 提供商的域名。响应必须包含 server 选择的 NONCE。响应不应该包含 USERNAME 或 MESSAGE-INTEGRITY 属性。
- 如果消息包含一个 MESSAGE-INTEGRITY 属性，但是没有忽略 USERNAME,REAML 或 NONCE 属性，server 必须产生一个错误码为 400（Bad Request）的响应。响应不应该包含 USERNAME,NONCE,REAML 或 MESSAGE-INTEGRITY 属性。
- 如果 NONCE 不在有效，server 必须产生一个错误码为 438（过期的 Nonce）响应。该相应必须包含 NONCE 和 REAML 属性并且不应该包含 USERNAME 或 MESSAGE-INTEGRITY 属性。Server 可以验证临时性为了提供额外的安全。参见【RFC2671】4.3 节的指南。
- 如果 USERNAME 属性中的用户名不再有效，server 必须产生一个错误码为 401（未授权）响应。该相应必须包 REAML 属性。该值推荐使用 STUN server 提供商的域名。该相应必须包含 server 选择的 NONCE。响应不应该包含 USERNAME 或 MESSAGE-INTEGRITY 属性。
- 使用 USERNAME 属性中关联的用户名的密码，按第 154.节描述的计算消息完整性值。

如果结果值与 MESSAGE-INTEGRITY 属性中的值不匹配，server 必须以错误响应拒接请求。该错误响应必须使用错误码 401（未授权）。该相应必须包含 NONCE 和 REAML 属性并且不应该包含 USERNAME 或 MESSAGE-INTEGRITY 属性。

如果这些检测通过，server 继续处理请求。Server 产生的任何响应（如上描述的情形），必须包含 MESSAGE-INTEGRITY 属性，使用用户名和密码计算其值以鉴权请求。REAML,NONCE,和 USERNAME 属性不应该被包含。

10.2.3 接受一个请求

如果响应是一个 401（未授权）的错误响应，client 应该在心事物中重传请求。请求必须包含一个 USERNAME，由 client 决定的作为错误响应中域的合适的用户名。请求必须包含复制自错误响应中的 REAML。请求必须包含复制自错误响应中的 NONCE。请求必须包含 MESSAGE-INTEGRITY 属性，该属性值使用 USERNAME 属性中的用户名关联的密码计算的。如果没有改变先前企图中的 USERNAME 或 REAML 或它关联的密码，Client 不允许执行重试动作。

如果响应是一个响应码为 438 的错误响应（过期的 Nonce），client 必须使用 438 错误响应（过期的 Nonce）中提供的新的 NONCE 来重试请求。这个重试不应该包含 USERNAME,REAML 和 MESSAGE-INTEGRITY。

Client 在响应中查找 MESSAGE-INTEGRITY 属性（无论成功或失败）。如果存在，client 按 15.4 节中描述的方法计算响应的消息完整性，使用他对请求使用的相同的密码。如果计算值和 MESSAGE-INTEGRITY 属性中的内容匹配，响应被认为是鉴权了的。如果不匹配，或者 MESSAGE-INTEGRITY 不存在，响应必须毫无声息地被丢弃。这意味着如果重传可用，将继续重传。

11. ALTERNATE-SERVER 机制

本节描述 STUN 允许一个 server 重定向一个 client 到另一个 server 的机制。本扩展是可选的，当用法使用的时候必须定义。

一个 server 使用本扩展通过回复给请求一个错误码为 300（试图转移）的错误响应消息来重定向开一个 client 到另外的 server。Server 必须包含一个 ALTERNATE-SERVER 属性在错误响应中。错误响应消息可能被鉴权；但是，有 ALTERNATE-SERVER 响应鉴权不可能或不实际的情况。

一个 client 使用本扩展俺如下处理一个 300(试图转移)错误码。Client 在相应中查找 ALTERNATE-SERVER 属性。如果发现了，client 认为本事物已经失败，并且按该属性中指定的 server，使用先前请求相同的通讯协议重新发送请求。如果被鉴权了，该请求必须使用客户端发送请求中使用的而服务端进行重定向的那个相同的证书。如果 server 重定向到的是 client 在过去 5 分钟内已经试过的那个 server，它必须忽略该重定向并且认为事物失败。这阻止了 server 的无限重定向循环。

12. 向后兼容 RFC3489

本节定义允许向后兼容 RFC3489 中定义的原始协议的程度的过程。本机制是可选的，这意味着仅适用于一个新的 client 能够连接旧的 server，反之亦然。本过程使用的时候，用法必须定义它。

第 19 节列出了本说明书和 RFC3489 的所有改变。然而，并不是所有的改变都是重要的，因为“经典 STUN”仅仅按很少的特殊的方式使用。为达到本扩展的目的，重要的改变如下。在 RFC3489 中：

- UDP 是仅仅被支持的通讯协议
- Magic cookie 域作为 128 位长的事物 ID 域的一部分。
- 属性 The XOR-MAPPED-ADDRESS 不存在，绑定方法使用 MAPPED-ADDRESS 素火替代。
- 有三种 comprehension-required 属性，RESPONSE-ADDRESS, CHANGE-REQUEST, 和 CHANGED-ADDRESS 在本说明书中被移除。

  CHANGE-REQUEST 和 CHANGED-ADDRESS 现在是 NAT 行为发现用法[BEHAVE-NAT]的一部分，其他的不支持。

12.1client 处理的改变

客户端想要和一个【RFC3489】的 server 交互操作，它应该发送一个使用 UDP 协议的使用绑定方法的不包含属性的请求消息到 server。如果成功，从 server 收到的成功响应将包含一个 MAPPED-ADDRESS 属性而不是 XOR-MAPPED-ADDRESS 属性。一个 client 寻求和一个老的 server 交互操作，它也必须准备接受消息。此外，client 必须忽略任何可能出现在响应中保留的 comprehension-required 属性。18.2 节中保留的属性，0x0002, 0x0004, 0x0005, 和 0x000B 可能出现在遵从 RFC3489server 的绑定响应。除此改变之外的响应处理合上述描述的过程一样。

12.2server 处理的改变

一个 STUN server 能够根据出现在 magic cookie 域折哦 ingde 正确值检测出给定请求是否是一个 RFC3489client 发送的。当 server 检测到一个 RFC3489 client 的时候，他应该 copy 绑定请求中可见的 magic cookie 域值到绑定响应中该域，并且插入一个 MAPPED-ADDRESS 属性替代一个 XOR-MAPPED-ADDRESS 属性。

在少数情况下，client 可能包含一个 RESPONSE-ADDRESS 或者 CHANGE-REQUEST 属性。在这些情况下，server 将视这些为未知的要求理解的属性，并且回复一个错误响应。尽管使用这些属性的机制不再被支持，这种行为确实可接受的。

RFC3489 版本的 STUN 没有考虑当和其他协议复用时高效地正确标示 STUN 消息的 magic cookie 属性和 FINGERPRINT 属性。因此，STUN 实现了向后兼容 RFC3489 时不应该用于 STUN 和其他协议复用的情况。然而，这对于 RFC3489 中无效的复用也不会成为一个问题。

13. 基本 server 行为

本节定义了基本的独立的 STUN server 的行为。一个基本的 STUN server 通过接受的和回复的 STUN 绑定请求提供给 client 几个反射通讯地址。

STUNserver 必须支持绑定方法。它不应该使用长期或者短期的证书机制。这是因为鉴权请求的工作量大于简单处理它的工作量。同样的理由，它不应该使用 ALTERNATE-SERVER 机制。它必须支持 UDP 和 TCP。它可以支持 STUN over TCP/TLS；然而，TLS 在此基本的模式中提供极小的安全效益。它可以使用 FINGERPRINT 机制但不是必须的。由于单独的 server 仅仅运行 STUN，所以 FINGERPRINT 没有提供利益。强求它可能破坏兼容 RFC3489，但是这些兼容性在单独的 server 中是强烈要求的。单独的 STUN server 应该像 12 节中描述的那样支持向后兼容 RFC3489。

推荐 STUN server 的管理员为 server 提供第 9 节中描述的那样的 DNS 实体。

一个基本的 STUN server 本身不是 NAT 穿透的解决方案。然而，他可用于穿透 NAT 用法的一部分。这将在第 14 节中深入讨论。

14. STUN 用法

STUN 本身不是 NAT 穿透的问题的一个解决方案。但是，STUN 定义了一个工具可用于一个大的解决方案的内部。词条“STUN usage”用于任何使用 STUN 作为一个组件的解决方案。

目前为止，三种 STUN 用法被定义了：交互事连接建立（ICE）【MMUSIC-ICE】，SIP 的客户端初始化连接【SIP-OUTBOUND】和 NAT 行为发现【NEHAVE-NAT】。其他的用法将在以后定义。

一种 STUN 用法定义了怎样实际应用 STUN—什么时候发送请求，怎么处理响应以及哪一种定义的可选的过程（或 STUN 的扩展）被使用。

一种用法也定义：

- 哪种 STUN 方法被使用。
- 什么鉴权和消息完整性机制被使用。
- 如【RFC4107】中讨论的手动与自动的密匙获取的考虑。
- 什么样的机制用来区分 STUN 消息和其他消息。当 STUN 运行在 TCP 上时，分帧机制可能被要求。
- 一个 STUNclient 怎么样确定 STUN server 的 IP 地址和端口。
- 向后兼容 RFC3489 是否被要求。
- 哪些这里定义的或其他扩展中的可选的属性（如 FINGERPRINT 和 ALTERNATE-SERVER）被要求。

另外，任何 STUN 用法必须考虑该用法中的安全实现。大量的针对 STUN 的攻击被获悉（参见本说明书的安全考虑部分），以及考虑这些攻击怎样被阻止和减轻。

最后，一种用法必须考虑它的 STUN 用法是否是一个单方面为主的自我地址固定穿透 NAT 方法的一个例子，如果是的，解决 RFC3424【RFC3424】中提出的问题。

15. STUN 属性

STUN 头部之后的是 0 哥或多个属性。每个属性必须是 16bits 类型，16bits 长度和值域的 TLV 编码。每一个 STUN 属性必须在 32bits 内结束。如上面所提到的，属性中所有域先传送最重要的位。

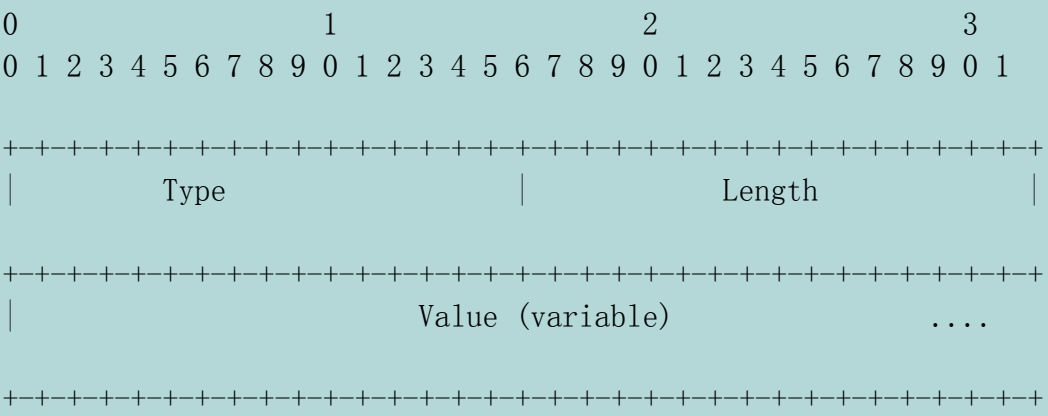


Figure 4: Format of STUN Attributes

长度域中的值必须包含该属性的优先填充的字节度量的长度值。由于 STUN 对齐属性在 32 位以内，不足 4 字节的被填充 1，2 或 3 字节的填充字段从而使其值包含一个 4 字节的组合。填充字段会被忽略，因而可以是任意值。

任何属性类型可能在 STUN 消息中出现不只一次。除非特别指定，否则出现的顺序是很重要的：仅仅首次出现的需要被接收者处理，任何的复制可能会被接收者忽略。

如果需要的话，为了允许本说明书未来的新版本添加新的属性，属性空间被分成两种。属性类型在 0X0000 和 0X7FFF 的是要求理解的属性，就而是说 STUN 代理不能成功处理消息除非它理解改属性。属性类型在 0X8000 和 0XFFFF 之间的是可选理解的属性，就是说如果 STUN 代理不理解这些属性，这些属性将被忽略。

STUN 属性类型集由 IANA 管理。本说明书定义的初始类型集在 18.2 中能找到。

本节余下的部分描述本说明书定义的种各样的属性格式。

15.1MAPPED-ADDRESS

MAPPED-ADDRESS 属性标示一个客户端的反射通讯地址。它包含 8-bit 地址协议族和 16-bit 端口，紧跟着标示 IP 地址的固定长度值。如果 IP 地址家族是 Ipv4，地址字段必须是 32 位。

如果 IP 地址家族是 Ipv6，地址字段必须是 128 位。所有域必须是网络字节序。MAPPED-ADDRESS 属性的格式是：

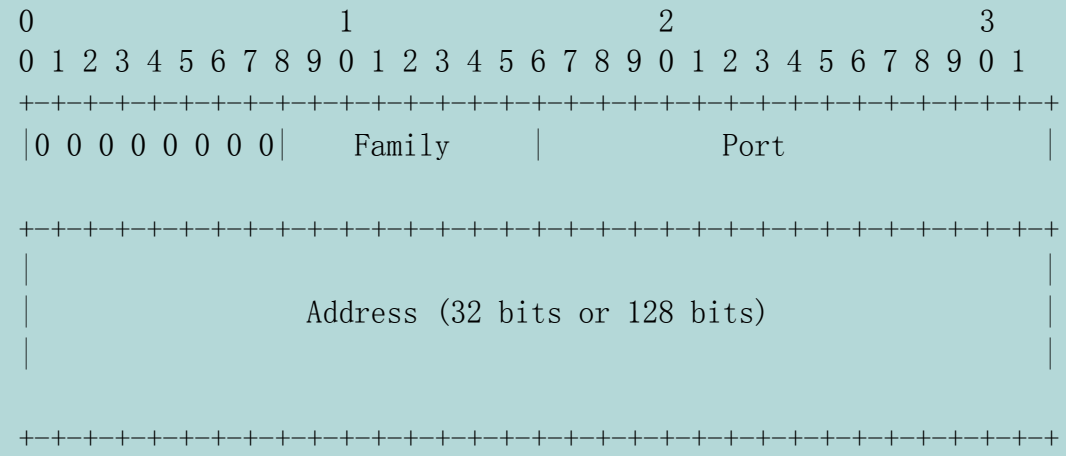


Figure 5: Format of MAPPED-ADDRESS Attribute

地址协议族可以采用下面的值：

0X01：Ipv4

0X02：Ipv6

MAPPED-ADDRESS 的首 8-bit 必须被设成 0 并且必须被接受这忽略。这些比特位代表在 32 位内的对齐参数。

这个属性仅用于向后兼容 RFC3489【RFC3489】的 SERVER。

15.2 XOR-MAPPED-ADDRESS

XOR-MAPPED-ADDRESS 属性和 MAPPED-ADDRESS 属性一样，除了反射通讯地址通过 XOR 函数而模糊不清之外。

XOR-MAPPED-ADDRESS 属性的格式：

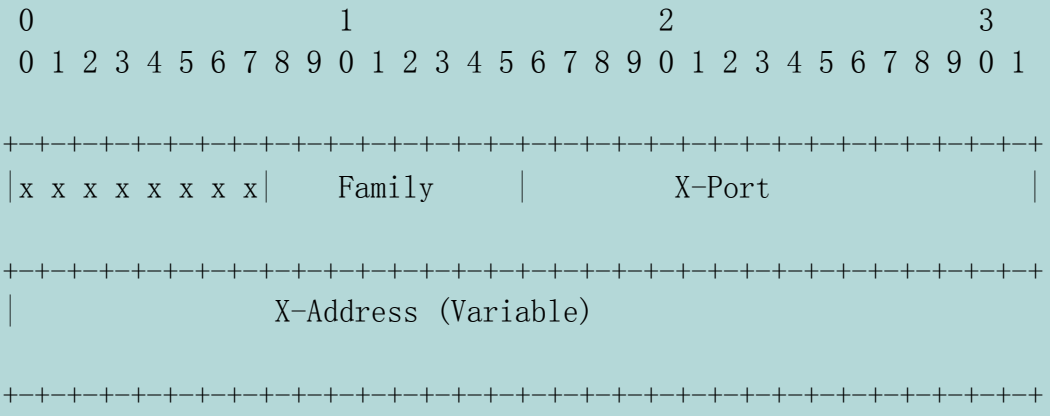


Figure 6: Format of XOR-MAPPED-ADDRESS Attribute

Family 代表 IP 地址族，和 MAPPED-ADDRESS 属性的编码一样。

X-port 采用主机序的映射端口计算，然后将它和 magic cookie 的最重要的 16 相异或，然后将结果转换成网络序。如果 IP 地址族是 Ipv4，X-Address 采用主机序的映射 IP，然后将它和 magic cookie 相异或，然后将结果转换成网络序。如果 IP 地址族是 Ipv6，X-Address 采用主机序的映射 IP，然后将它和 magic cookie 和 96-bit 的事物 ID 值连接串相异或，然后将结果转换成网络序。

属性值的前 8bit 的编码和处理规则，处理属性多次出现的规则以及处理地址族字段的规则与 MAPPED-ADDRESS 相同。

注意：XOR-MAPPED-ADDRESS 和 MAPPED-ADDRESS 仅仅在他们的编码和通讯地址不同。前者的编码是将通讯地址和 magic cooki 想异或。后者的直接使用二进制位。RFC3489 最初指定的仅是 MAPPED-ADDRESS。然而，部署经验发现一些 NAT 重写包含公网 IP 地址的 32 位负荷，如 STUN 的 MAPPED-ADDRESS 属性，善意却误导了试图提供的通用的 ALG 功能。

15.3USERNAME

USERNAME 属性用于消息完整性。他标示消息完整性检测中的用户名和密码组合。

USERNAME 的值是一个可变长度的值，它必须包含一个小于 513 字节的 UTF-8【RFC3629】的编码系列，并且必须使用 SASLprep【RFC4013】处理过。

15.4MESSAGE-INTEWGRITY

MESSAGE-INTEGRITY 属性包含一个 STUN 消息的 HMAC-SHA1【RFC2104】。MESSAGE-INTEGRITY 属性可以在任何类型的 STUN 消息中出现。由于它使用 SHA1 哈西系列，HMAC 将是 20 字节。用于 HMAC 的输入文本是 STUN 消息，宝货消息头，直到包含 MESSAGE-INTEGRITY 属性之前的属性。除了出现在 MESSAGE-INTEGRITY 属性之后的 FINGERPRINT 属性，代理必须忽略其他跟在 MESSAGE-INTEGRITY 后面的属性。

HMAC 的密匙取决于是使用长期证书还是短期证书。对于长期证书，密匙是 16 字节：

Key=MD5(username “:” reaml “:” SASLprep(password))

即是，16 字节的密匙有采用 MD5 的连接下面 5 个域的结果的哈西系列：（1）用户名，从 USERNAME 属性（在 SASLprep 已经被使用的情况下）获取的去掉引号和末尾的空字符；（2）一个冒号；（3）去掉引号和末尾的空字符的域；（4）冒号；（5）去末尾的空字符的并使用 SASLprep 处理过的密码。例如，如果用户名是‘user‘，域名是’reaml‘，以及密码是’pass‘，那么 16 字节的的 HMAC 密匙将是经过 MD5 处理的字符串’user:reaml:pass’哈西系列，处理结果即：0x8493fbc53ba582fb4c044c456bdc40eb。

对于短期证书：

key=SASLprep（Password）

其中 MD5 在 RFC1312【RFC1312】中定义，SASLprep（）在 RFC4013【RFC4013】中定义。

长期证书中使用的密匙方便了系统部署，也适用于 SIP。典型地，SIP 系统使用 SIP 的消化鉴权机制而不真的存储密码到数据库。但是，它们一个等于上述定义密匙的称作 H（A1）的值。

基于上述规则，用于构建 MESSAGE-INTEGRITY 的散列包含来自 STUN 消息头部的长度域。处理散列之前，MESSAGE-INTEGRITY 属性必须插入到消息（包含虚拟内容）中。长度必须设成指向消息长度，并且包含 MESSAGE-INTEGRITY 属性本身，但是之后不包含任何属性。一旦计算执行了，MESSAGE-INTEGRITY 属性的值可以被填充，STUN 消息的长度值可以被设成正确的值—整个消息的长度。类似地，当验证 MESSAGE-INTEGRITY 时，长度域应该调整到指向计算 HMAC 之前的 MESSAGE-INTEGRITY 属性末端。当如 FINGERPRINT 属性出现在 MESSAGE-INTEGRITY 之后的属性存在时这些调整才需要。

15.5FINGERPRINT

FINGERPRINT 属性可能出现在任何 STUN 消息中。属性值的计算像 STUN 消息的 CRC-32 一样直到（但是不包括）FINGERPRINT 属性本身，和 32-bit 值）0X5354554e 异或过（异或帮助应用层也使用 CRC-32 的情况）。32-bit CRC 在 ITU V4.2【ITU.V42.2002】中定义，其包含一个 x32+x26+x23+x22+x16+x12+x11+x10+x8+x7+x5+x4+x2+x+1 产生多项式。当 FINGERPRINT 属性存在，它必须是消息中的最后一个属性，并且出现在 MESSAGE-INTEGRITY 属性之后。

FINGERPRINT 属性可以辅助区分 STUN 包和其他协议包。参见第 8 节。

和 MESSAGE-INTEGRITY 属性一起，FINGERPRINT 属性中使用的 CRC 包括来自 STUN 消息头部的长度域。因此，这个值必须正确并且在计算 CRC 之前包含作为消息长度一部分的 CRC 属性。当在消息中使用 FINGERPRINT 属性的时候，放到消息中的属性属性值是个哑值，然后 CRC 被计算，然后属性值被更新。如果 MESSAGE-INTEGRITY 属性也存在，它必须在 CRC 计算之前设置正确的消息完整性值，因为 CRC 也是在 MESSAGE-INTEGRITY 属性值基础之上计算的。

15.6ERROR-CODE

ERROR 属性用于错误响应消息。它包含一个 UTF-8【RFC3629】编码的在 300 到 699 之间的数字错误代码加上文字性的原因短语，并且包含代码赋值和 SIP【RFC3261】和 HTTP【RFC2616】中的语义。原因短语是为用户看的，可以是适合错误代码的任何东西。推荐的错误代码的原因短语包含在 IANA 注册的错误代码。原因短语必须是短于 128 字符（可以长达 763 字节）的 UTF-8【RFC3629】编码的系列。

0 1 2 3

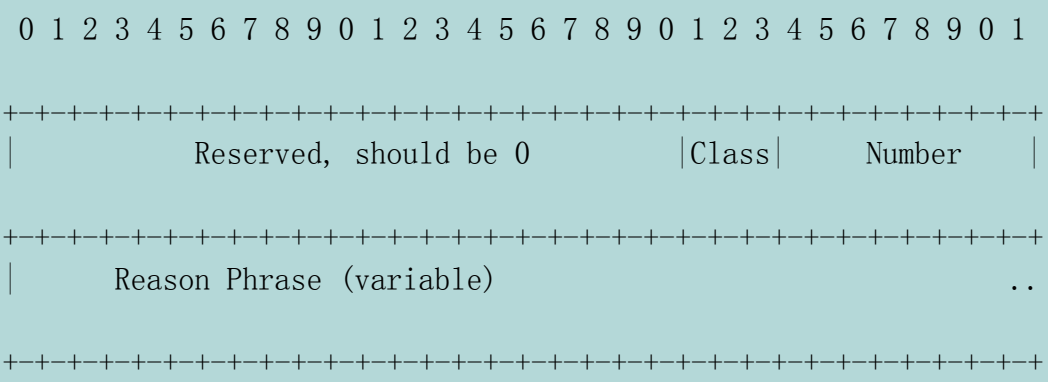


Figure 7: ERROR-CODE Attribute

为了方便处理，错误代码的类型（百位数）按图 7 中所示的分别从如下的代码中编码。

保留位应该是 0，是为了对齐 32-bit。接收者必须忽略这些比特位。类别代表错误代码的百位数。值必须是 3 到 6 之间的。代表错误的数字模除 100，其值必须在 0 到 99 之间。

一下定义错误代码及其推荐的原因短语：

300 试图转移：client 应该为请求联系转移的 SERVER。如果请求包含一个 USERNAME 属性和一个有效的 MESSAGE-INTEGRITY 属性，这个错误响应必须被发送，否则，它不允许被发送并且错误代码 400（Bad Request）被建议发送。这个错误响应必须被 MESSAGE-INTEGRITY 属性保护，接收者必须验证响应的 MESSAGE-INTEGRITY 在它重定向到转移的 SERVER 之前。

NOTE： 300 响应失败的生成和验证消息完整性允许半路的攻击者篡改一个 300 响应因而导致后续的 STUN 消息送到受害者。

400 Bad Request：畸形的请求。Client 不应该不修改先前的企图就重试请求。SERVER 可能不能够产生一个有效的 MESSAGE-INTEGRITY 为此错误，因此 client 不能期望在这个响应上收到一个消息完整性属性。

401 未授权：请求没有包含继续的正确证书。Client 应该采用正确的证书重试请求。

420 未知属性：SERVER 收到一个包含他不认识的单要求理解的属性的 STUN 包。SERVER 必须把这个未知属性放在响应的 UNKONW-ATTRIBUTE 属性域中。

438 过期的暂时值：client 使用的 NONCE 不再有效。Client 应该使用响应中提供的 NONCE 重试请求。

500 服务器错误：SERVER 遭到临时的错误。客户端应该重试一遍。

15.7 域

REAML 属性可能出现在请求和响应中。它包含满足 RFC3261[RFC3261]中描述的“reaml-value”语法的文本但是没有双引号和空格。即是说，它是一个未被引号引起来的 reanl-value（因此是引用文本或引用对的系列）。它必须是一个短于 128 字符（可以长达 763 字节）的 UTF-8【RFC3629】编码的系列。并且必须被 SASLprep【RFC4013】处理过。

请求中出现的 REAML 属性表明长期证书被用于鉴权。出现在特定的错误响应中表明 SERVER 希望使用长期证书来鉴权。

15.8 NONCE

NONCE 属性可能出现在请求和响应中。它包含一个 RFC3261【RFC3261】中定义的引用文本或者引用对系列。注意这意味着 NONCE 属性不包含实际的引用字符。为 SERVER 选择 NONCE 值的指南参见 RFC2617【2617】第 4.3 节。

它必须短于 128 字符（可以长达 763 字节）。

15.9 未知属性

未知属性仅住出现在错误响应中，当 ERROR-CODE 属性的响应码是 420 的时候。

该属性包含一个 16 位的值列表，代表一个不被 SERVER 认识的属性类型。

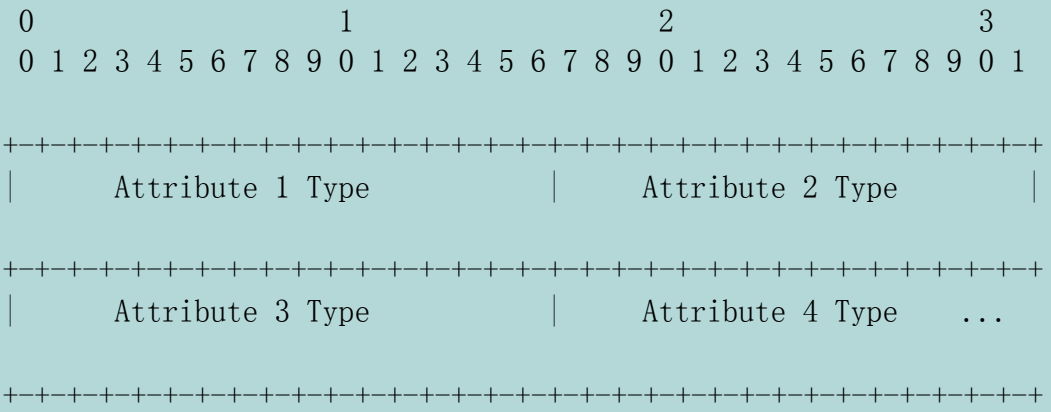


Figure 8: Format of UNKNOWN-ATTRIBUTES Attribute

注意：在【RFC3489】中，这个域通过复制最后一个域来填充的。在本说明书中，使用正规的填充规则。

15.10 软件

软件属性是描述代理发送请求消息软件的。它被客户端和服务器端所使用。它的值应该包含制造商和版本号。该属性不影响协议的操作，仅仅作为诊断和调试的一个工具。该属性值是可变长度的。他必须是短语 128 哥字符（可以长达 763 字节）的 UTF-8【RFC3629】编

码的系列。

15.11 ALTERNATE-SERVER

转移的服务器代表一个 STUN 客户端应该重试的不同的 STUN 服务端的转移通讯地址。

它的编码方法和 MAPPED-ADDRESS 属性一样，因而通过 IP 地址引用一个 SERVER。IP 地址族必须和请求的源地址的 IP 地址族一样。

16. 安全考虑

16.1 针对协议的攻击

16.1.1 外围攻击

攻击者试图在 STUN 消息传输过程中修改 STUN 消息，从而引起 STUN 操作的失败。攻击可以通过使用短期或长期的证书机制来检测请求或响应的攻击。当然，一旦被检测出来，操作的包将被丢弃，导致 STUN 事物完全失败。这种攻击可能仅是半路的攻击。

攻击者能够观察而不修改传送中的 STUN 消息（例如，位于像 Wi-Fi 的共享接入媒介），可以看到一个 STUN 请求，并且立即发送 STUN 响应，典型的是一个错误代码，为了干扰 STUN 处理。这种攻击也阻止利用 MESSAGE-INTEGRITY 的消息。然而，一些错误响应，特别是和鉴权相关的，不能够被 MESSAGE-INTEGRITY 保护。当 STUN 本身运行在安全传输协议之上（例如：TLS），这些攻击完全被减轻。

取决于 STUN 用法的攻击可能影响很小并且不会使消息完整性减轻。例如，STUN 用于 ICE 用法来发现一个基本 STUNSERVER 的服务器反射地址时，当在连接检测阶段攻击被发现的时候鉴权和消息完整性不被要求。然而，连接检测本身需要整个 ICE 合适的操作保护。如 14 节中所描述的，STUN 用法描述了鉴权和消息完整性什么时候被需要。

由于 STUN 鉴权和消息完整性保护使用共享私密的 HMAC 来保护，因而会遭受离线的暴力攻击。当鉴权被使用时，它应该包含一个不被离线的暴力攻击的不可读的健壮的密码。然而，STUN 经常是运行在 UDP 之上的，在这种情况下，健壮的密码是保护 STUN 不受攻击的唯一方法。

16.1.2 内部攻击

一个流氓 client 可能试图通过发起大量的 STUN 请求来对一个 server 进行 Dos 攻击。幸运的是，STUN 请求能够通过 server 无状态地处理，是的这种攻击很难发起。

一个流氓 client 可能使用一个 STUNserver 作为反射器，发送经过篡改源 IP 地址和端口的请求。在这种情况下，响应应该发送到该源地址和端口。这种攻击的数据包的数目不会扩大（Server 是根据 client 的请求来发送响应的），因此数据的增长较小，尽管 STUN 的响应比请求数据包要打。这种攻击通过过滤入口源地址被减轻。

通过 SOFTWARE 属性公开代理的具体软件版本可能使已知有安全漏洞的软件更容易遭

受攻击。实现者应该使 SOFTWARE 属性成为可配置的选项。

16.2 影响用法的攻击

本节列出所有可能发起的 STUN 用法攻击。没中用法必须考虑这些攻击是否可以应用，如果可以，就得讨论反对措施。

绑定请求/响应事物。由于用法的反射地址是用法的一个功能，因而这些攻击的可用性和矫正性是用法指定的。在统称情况下，半路的攻击者很容易修改反射地址。想一下，例如，通常情况下，STUN 直接运行在 UDP 之上。在这种情况下，半路攻击者可以在绑定请求到达 STUN Server 之前修改源 IP 地址。STUN Server 在 XOR-MAPPED-ADDRESS 属性中返回反射 IP 地址，然后发送响应到这个（修改后的）IP 地址和端口。如果攻击者能够拦截这个响应，他可以直接发送到客户端。使用消息完整性检查来避免这种攻击是不可能的，因为一个完整的消息值不可能包含源 IP 地址，也因为中间的 NAT 必须能够修改这个值。取而代之的是，下面列出的解决方案可以验证客户端学习到的反射地址，如 ICE【MMUSIC-ICE】中所做的那样。其他的用法可能使用例外的方法免受攻击。

16.2.1 攻击 I：针对目标的分布式 Dos(Ddos)

在这种攻击中，攻击者提供使用相同的假的指向攻击目标的反射地址的一个或多个客户端。这可能使 STUN 客户端认为他们的反射地址和目标相同。如果客户端分出那个反射地址来接受数据（例如，SIP 消息），数据会取而代之的被送到目标。这种供机能产生大量的扩展，尤其是当使用 STUN 进行多媒体应用的客户端。然而，它仅仅能发起那些数据包从 STUN Server 经过攻击者到目标的攻击，这限制了可能的情况。

16.2.2 攻击 II：隐藏客户端

在这种攻击中，攻击者给 STUN 客户端一个假的反射地址。反射地址是不能路由的通讯地址。因此，客户端不能接受它期望收到的任何数据包当它发包到该反射地址的时候。攻击者对这种剥削不是很感兴趣。它仅影响通常不是期望目标的单个 client。更多的是，任何能够安装攻击的攻击者可以采用其他手段也能拒接 client 的服务，如阻止 client 从 STUN Server，或 DHCP Server 接受任何响应。如 16.2.1 节中的攻击，这种攻击仅在当攻击者在在路劲上等待那些从 STUN Server 发向这个未使用的 IP 地址的数据包的时候才有可能。

16.2.3 攻击 III：假冒 client 的身份

这种攻击类似于攻击 II。然而，假冒的反射地址指向攻击者自身。这允许攻击者接受发向客户端的数据。

16.2.4 攻击 IV：窃听

在这种攻击中，攻击者强制客户端使用反射地址路由到它自身。它转发任何它收到的包到 cilent。这种攻击允许攻击者观察所有发送 client 的包。但是，为了发起攻击，攻击者必须已经能够观察 client 到 STUN server 的数据包。在大多数情况下（如当攻击从接入网络

发起的时候），这意味着，攻击者已经观察到发送到 client 的数据包。因此这种攻击只是针对半路伤的攻击者观察 client 到 STUN server 的数据包，但是通常不是在数据包路由到 client 的半路上的。

16.3 哈希敏捷计划

本说明书使用 HMAC-SHA-1 来计算消息的完整性。如果在早前一段时间，HMAC-SHA-1 被发现是很复杂的，以下的修正将被应用。

我们将定义一个介绍使用新哈希计算消息完整性属性的扩展。客户端应该被要求在它们的请求和响应中都包含新的和老的消息完整性属性。一个新的 server 将使用新的消息完整性属性，而老的 server 将使用老的消息完整性属性。部署混合实现的过渡期之后，其他的说明书可能不赞成老的消息完整性属性，client 也停止在请求中包含它。

注意重要的是使用用户密码的 MD5 串计算的密匙来处理 HMAC。选择 MD5 哈希处理是因为传统数据库中密码是以这种方式存在。如果将来的工作发现一个 MD5 的 HMAC 输入不安全，那么需要不同的哈希处理，这种计划也可能改变。然而，这要求管理员更新他们的数据库。

17. IAB 考虑

IAB 已经研究了单方面为主的自我地址固定（UNSAF）问题，这种问题通常的处理过程是一个客户端试图通过协同工作的反射机制协议确定位于 NAT 另一边其他域中自己的地址（RFC3424【RFC3424】）。如果一个代理在 NAT 的后面，另一个代理在 NAT 的公网的一边，可以使用 STUN 的绑定请求/响应事物来执行这种功能。

IAB 已经委托为此目的开发的协议记录一个特殊的考虑集。因为一些 STUN 用法提供 UNSAF 功能（如 ICE【NNUSIC-ICE】），而其他的不提供（如 SIP Outbound【SIP-OUTBOUND】），这些考虑需要由用法自身处理。

18. IANA 考虑

IANA 已经创建了三个新的注册：一个“STUN 方法注册”，一个“STUN 属性注册”和一个“STUN 错误码注册”。IANA 将分配的 IANA 端口名从“nat-stun-port”改为“stun”。

18.1 STUN 方法注册

一个 STUN 方法是一个范围在）0x000-0xFFF 之间的十六进制数字。编码到 STUN 消息中的 STUN 方法在第 6 节中已经描述。

初始的 STUN 方法有：

0x000：（保留）
0x001：（绑定）

0x002：（保留：是公共密匙）

0x000-0x7FF 之间的 STUN 方法被 IETF 分配审核【RFC5226】。0x800-0xFFF 之间的 STUN 方法由指定专家分配【RFC5226】。专家的责任是验证选择的码点是否被使用，并且请求不是为一个相当大的码点。扩展本身的技术审核不在指定专家的职责范围之内。

18.2 STUN 属性注册

一个 STUN 属性类型是一个在 0x0000-0xFFFF 之间的十六进制数字。0x0000-0x7FFF 之间的属性类型是要求理解的属性；0x8000-0xFFFF 之间的属性是可选理解的属性。一个 STUN 用户代理对未知的要求理解的属性和可选理解的属性分别处理。

初始的 STUN 属性类型是：
要求理解的范围（0x0000-0x7FFF）：

```
0x0000: (Reserved)
0x0001: MAPPED-ADDRESS
0x0002: (Reserved; was RESPONSE-ADDRESS)
0x0003: (Reserved; was CHANGE-ADDRESS)
0x0004: (Reserved; was SOURCE-ADDRESS)
0x0005: (Reserved; was CHANGED-ADDRESS)
0x0006: USERNAME
0x0007: (Reserved; was PASSWORD)
0x0008: MESSAGE-INTEGRITY
0x0009: ERROR-CODE
0x000A: UNKNOWN-ATTRIBUTES
0x000B: (Reserved; was REFLECTED-FROM)
0x0014: REALM
0x0015: NONCE
0x0020: XOR-MAPPED-ADDRESS
```

可选理解范围的（0x8000-0xFFFF）

```
0x8022: SOFTWARE
0x8023: ALTERNATE-SERVER
0x8028: FINGERPRINT
```

在要求理解前半个区间（0x0000-0x3FFF）的属性类型和选择理解前半个区间（0x8000-0xBFFF）的属性类型由 IETF 审核分配【RFC5226】。在要求理解后半个区间（0x0000-0x3FFF）的属性类型和选择理解后半个区间（0x8000-0xBFFF）的属性类型由指定专家分配【RFC5226】。专家的责任是验证选择的码点是否被使用，并且请求不是为一个相当大的码点。扩展本身的技术审核不在指定专家的职责范围之内。

18.3STUN 错误码注册

一个 STUN 的错误码是一个在 0-699 之间的数字。错误码由人类能理解的任何合适的

UTF-8【RFC3629】编码的文本原因短语；本文档仅仅提供一些建议值。

STUN 错误码由分配的码点和 SIP【RFC3261】和 HTTP【RFC2616】语义组成。

本注册的初始值在第 15.6 节中给出。

新的 STUN 错误码分配将由 IETF 审核【RFC5226】。说明书必须仔细考虑为什么客户端不能理解授权请求之前的错误代码代码。参见第 7.3.4 节的规则。

18.4STUN UDP 和 TCP 端口号

IANA 已经为 STUN 分配了端口 3478。这哥端口以名字“nat-stun-port”出现在 IANA 的注册中。为了和使用注册协议服务的 DNS SRV 过程一致，IANA 被要求将端口名字从“nat-stun-port”改成“stun”，文本名从"Simple Traversal of UDP Through NAT (STUN)" to "Session Traversal Utilities for NAT",因此，IANA 端口注册应该读作：

```
stun    3478/tcp    Session Traversal Utilities for NAT (STUN) port
stun    3478/udp    Session Traversal Utilities for NAT (STUN) port
```

另外，IANA 还为“stuns”服务分配了基于 TCP 和 UDP 的端口 5349。UDP 端口当前没有定义；然而保留给将来使用。

19. 从 RFC3489 的改变

本说明书淘汰了 RFC3489[RFC3489]。本说明书在以下的方面不同于 RFC3489：

- 去掉了 STUN 是一个完整的 NAT 穿透解决方案的观念。STUN 现在是一个工具能够用于生成一个 NAT 穿透的解决方案。因此，将协议的名字改成 NAT 的会话穿透用法。
- 介绍了 STUN 用法的概念，并描述了 STUN 必须记录什么样的用法。
- 去掉了 NAT 类型发现和绑定生命周期发现的 STUN 用法。这些用法已经证明对于本文档描述的各种各样的 NAT 设备类型过于脆弱。去掉了 RESPONSE-ADDRESS, CHANGED-ADDRESS,CHANGE-REQUEST, SOURCE-ADDRESS, 以及 REFLECTED-FROM 属性。
- 通过减少 32bit 事物 ID 号来增加一个 32-bit 的 magic cookie。Magic cookie 和先前事物 ID 相同的偏移处开始。
- 增加了 XOR-MAPPED-ADDRESS 属性，如果 magic cookie 出现在请求中，该属性将包含在绑定响应中。否则，RFC3489 行为被保留（即是说，绑定请求包含 MAPPED-ADDRESS）。参见关于这种改变的 XOR-MAPPED-ADDRESS 讨论。
- 介绍了消息类型结构头域的正常结构，是以一对显示的标示请求，响应，错误响应或标志的比特位。因此，消息类型被拆分成类别（前面四种中的一种）和方法。
- 清楚指出 STUN 最重要的两位是 0b00，允许当使用 ICE 时候轻松区分 RTP 包。
- 增加了一个 FINGERPRINT 属性来提供一种当 STUN 和其他协议复用时清楚检测其差别的方法。
- 增加支持 Ipv6，是 Ipv4 客户端很清晰地获得一个 Ipv6 的映射地址，反之亦然。
- 增加基于长期证书的鉴权。

- 增加 SOFTWARE，REAML,NONCE,和 ATERNATE-SERVER 属性。
- 去掉了共享私密方法，增加了 PASSWORD 方法。这种方法从没实现过，当前的用法也不需要。
- 去掉了推荐继续监听 STUN 消息 10 秒来确认是否存在攻击。
- 改变事物定时器，使得其对 TCP 更友好。
- 去掉了围绕控制和媒体平面分离的例子。取而代之的是提供更多用协议使用 STUN 的信息。
- 定义了一种改变长度属性解释的通用的填充机制。在理论上将向后兼容。然而，RFC3489 中的机制从来没为没有在 32 位边界内对齐的新属性。
- REAML,SERVER，原因短语，和 NONCE 现在在 127 字符内，USERNAME 在 513 字符内。
- 改变了 TCP 和 TLS 的 DNS SRV 过程。UDP 和以前仍一样。

20. 贡献者

RFC3489 的最初作者 Christian Huitema 和 Joel Weinberger。

21. 致谢

作者在此感谢 Cedric Aoun, Pete Cordell, Cullen Jennings, Bob Penfield, Xavier Marjou, Magnus Westerlund, Miguel Garcia, Bruce Lowekamp,以及 Chris Sullivan 的评论，还有 Baruch Sterman 和 Alan Hawrylyshen 的初始实现。感谢 Leslie Daigle, Allison Mankin, Eric Rescorla, Henning Schulzrinne 以及 IESG 和 IAB 对此工作所做的努力。