INTERNET DRAFT
<draft-ietf-midcom-scenarios-02.txt>
Expires November 17, 2001

C. Huitema
Microsoft

May 17, 2001

# MIDCOM Scenarios

Status of this memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet- Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Abstract

As trusted third parties are increasingly being asked to make policy decisions on behalf of the various entities participating in an application's operation, a need has developed for applications to be able to communicate their needs to the devices in the network that provide transport policy enforcement. Examples of these devices include firewalls, network address translators (both within and between address families), signature management for intrusion detection systems, and multimedia buffer management. These devices are a subset of what can be referred to as 'middle boxes.' This document describes traversal scenarios that a 'middle box traversal protocol' should enable.

## 1 Introduction

In order to delineate the requirement of the MIDCOM protocol, we present here a set of scenarios that should be enabled by this protocol. The scenarios include running a server behind a NAT/Firewall, enabling direct connection between peers that exchange addresses in an ad-hoc way, e.g. through an instant messaging service, and enabling peer-to-peer communication with explicit signaling, e.g. using SIP or H.323. These scenarios may include several variants that we will present. We also present the evolution of these scenarios when IPv6 provides global addresses, and

introduce the "6to4 router" scenario required for IPv6 transition, and the IPSEC scenario enabled by IPv6.





The main purpose of this exercise is to explain the nature of the holes that would have to be opened for the applications to work, so as to derive the "functional requirements" of the firewall traversal protocol. It is quite clear that there are other requirements, notably security requirements. The scenarios described what the application needs in order to run; whether the application is actually allowed to run or not is a matter of local policy. In order to meet the security requirements, the protocol will have to enable adequate controls. In order to better understand how a security and control can be applied, the scenarios include examples where authentication is a gating operation.

This memo uses the definitions introduced in [MIDBOXFRAME], in particular the definition of a Firewall/NAT.

## 2 Scenarios

In the following, we document a set of realistic scenarios that should be enabled by a firewall traversal protocol. These scenarios include:

* Placing a server behind a firewall/NAT,

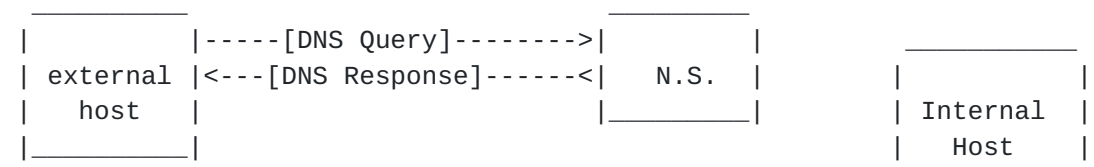
* Enabling ad-hoc peer-to-peer applications,

* Enabling peer-to-peer communication through explicit signaling systems such as SIP or H.323.

We also take into account the deployment of IPv6, which introduces variants of the previous scenarios, as well as new scenarios such as the establishment of tunnels for carrying IPv6 packets through a firewall, or the establishment of IPSEC associations between internal and external hosts.

### 2.1 TCP server behind a firewall/NAT

An internal server wants to receive TCP-IP connections requests from the outside (where outside is some place outside a domain). An example is, running a web server in a domain protected by a firewall.






In this scenario, the internal host publishes the IP address and TCP port number at which it can be joined in a name server, using for example SRV and A records in the DNS. The IP address that is published must be valid in the "external" domain; if this external domain is the current Internet, the published IP address must be valid in the global Internet.

The scenario implies that the following operations happen in sequence:

1) The internal host interacts with the firewall/NAT, using the midcom protocol. As a result of the interaction, the internal host learns the IP address and TCP port that it may advertise to external parties.

2) The internal host publishes the information in a name server.

3) The external host obtains the information from the name server.

4) The external host issues a TCP connection request, and sends a TCP SYN packet.

5) The firewall/NAT receives the packet, performs address translations and port mapping if necessary, and relays the TCP SYN packet to the internal host.

6) After that point, the TCP connection proceeds.

In the diagram, we depict only one external host, but this is an example, not a limitation. Once the address and ports are published in the name server, an unlimited number of external hosts may attempt to connect to the internal server.

It is quite clear that the interaction between the internal host and the firewall/NAT described in the first step of the scenario may require some form of access control. The specific form of access control will depend of the internal domain's policies. In practical examples, these policies varies from allowing all internal hosts to run services, in un-managed domains, to allowing only a specific set of services on a specific set of hosts.

A particular result of the access control is that the first step of this scenario may indeed fail, either because the opening of a hole is not authorized or because the firewall/NAT does not have sufficient resources. The internal host will explicitly learn that it should not advertise any IP address or TCP port to third parties. In this case, the host should not publish information, and external hosts should not attempt to establish connections.

It is important that the behavior of the firewall/NAT be consistent: if the mapping request at step 1 fails, then we expect that an





attempt to establish a connection from an external host will be rejected; conversely, if the mapping request succeed, then we expect the establishment of TCP connections to also succeed.

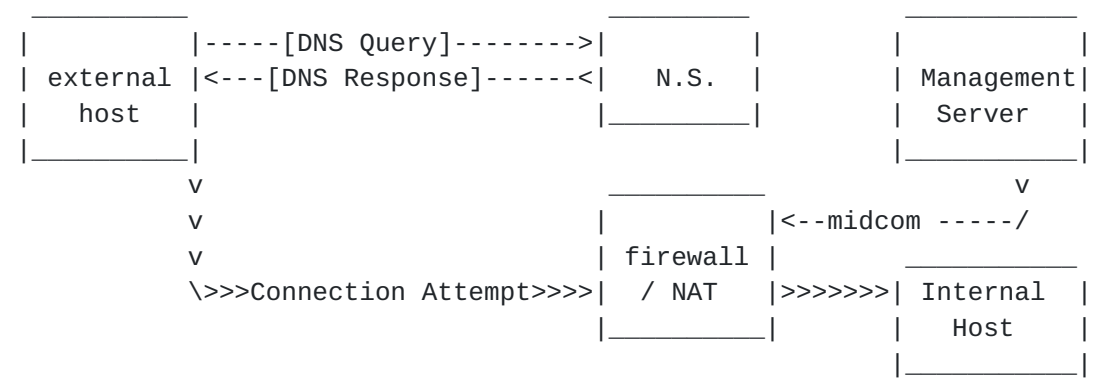
There are two interesting variants of that scenario: the use of UDP instead of TCP, and the control of the firewall/NAT by a third party instead of the internal host.

### 2.1.1 UDP Server behind a firewall/NAT

This scenario is exactly the same as the TCP server scenario, with the difference that the external host issues unsolicited UDP packets, instead of TCP/SYN packets. An example of this scenario is, running a SIP server or a DNS server behind a NAT/Firewall.

### 2.1.2 TCP/UDP server authorized by a third party

This scenario differs from the base scenario in a simple way: the midcom protocol is exercised by a third party instead of the host itself. In the diagram, we call this third party a management server:




The management server will interact with the firewall/NAT, using the midcom protocol, as in the step 1 of the main scenario. The other steps will be unchanged. A key point of this scenario is that the

internal host is unaware of the midcom protocol; in practical deployment, the internal host can be an unmodified server, such as a web server responding to HTTP requests on incoming TCP connections, or a DNS server responding to name requests on incoming UDP packets.

As in the original scenario, the mapping request may be rejected, for example if the "management server" that attempts to establish the mapping is not actually authorized to do so.

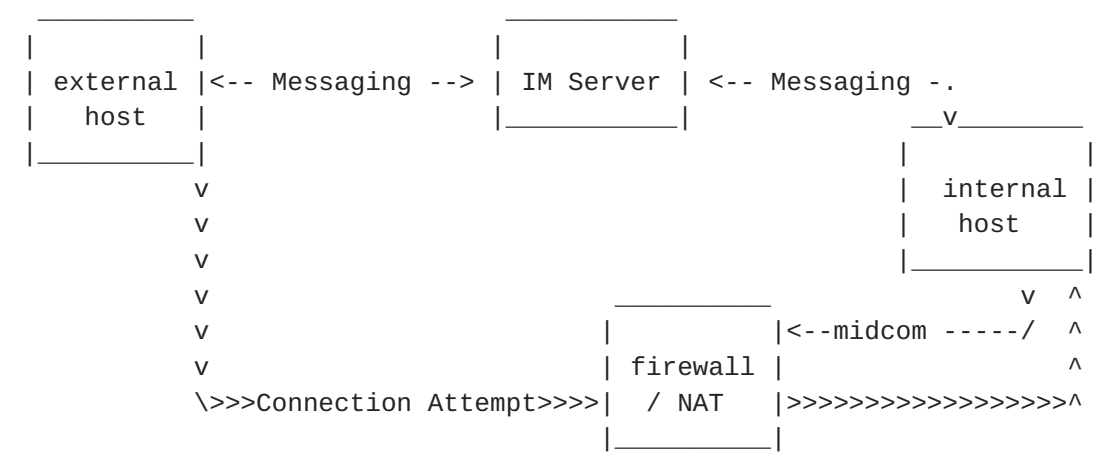
## 2.2 Peer-to-peer communication with ad-hoc rendezvous

The mediated peer-to-peer communication scenario describes hosts that communicate through some external third party, such as an instant messaging service, and then establish a direct communication channel, such as a TCP connection. An example of this scenario is,





starting the exchange of files from an IM service.




This scenario does not involve any particular cooperation between the firewall/NAT and the IM server. The connection between the internal host and the IM system can use any protocol, in particular combinations of TCP, HTTP and TLS.

The scenario implies that the following operations happen in sequence:

1) The internal and external hosts communicate through some form of instant messaging service or chat room. At some point, they decide to establish a direct channel, e.g. to exchange files.

2) The internal host interacts with the firewall/NAT, using the midcom protocol. As a result of the interaction, the internal host learns the IP address and TCP port that it may advertise to the external host.

3) The internal host sends the IP address and the TCP port to the external host.

4) The external host issues a TCP connection request, and sends a TCP SYN packet.

5) The firewall/NAT receives the packet, performs address translations and port mapping if necessary, and relays the TCP SYN packet to the internal host.

6) After that point, the TCP connection proceeds.

7) After the application has finished using the connection, the internal host may interact with the firewall/NAT and close the hole.

In this scenario, the NAT firewall only has to authorize the communication between a single internal host and a well identified external host; the authorization typically only needs to remain





valid for a single TCP connection, or in any case for a limited duration. The request in step 2 may be rejected by the firewall, either for policy reasons, or because there are not sufficient resource available; in this case, the peers should not attempt to establish a connection. As we noted in scenario 2.1.1, it is important that the behavior of the firewall/NAT be consistent: if the mapping request at step 2 fails, then we expect that an attempt to establish a connection at step 5 will be rejected; conversely, if the mapping request succeeds, then we expect the establishment of TCP connection to also succeed.

We expect that the decision to authorize the mapping request or not will depend on a variety of parameters, such as the identity of the internal user, the configuration of the internal system, the identity of the external peer, the purpose of the connection, and the amount of resource requested for the connection. The purpose of the connection may be a generic notation such as "audio" or "video", or a coded description of the application.

There are two variants of that scenario, when the dialog occurs over UDP and when both hosts are hidden behind a firewall/NAT.

### 2.2.1 Peer-to-peer communication using UDP

This scenario is exactly the same as the TCP scenario, with the difference that the external host issues UDP packets, instead of

TCP/SYN packets. An example of this scenario is, streaming audio or video between two peers.

### 2.2.2 Both peers behind firewalls

When both peers are behind firewalls, it is hard to predict the IP address that will be used by the host initiating the TCP connection. In this situation, there are two options:

1) Allow the internal host to accept TCP connections from any external address.

2) Let the "external" host use the midcom protocol to predict the "external" IP address that it will use for the incoming connection.

The first option may look insecure, but the possible insecurity of accepting connections from multiple source is often mitigated by application level protections, such as security tokens exchanged through the IM channel. A variation of this option is to accept connections from multiple sources, but restrict the hole to exactly one source once the connection has been established. As in all other scenarios, the firewall will have the option to accept or refuse the requested hole; it is important that the confirmation or refusal be explicit, and that the behavior of the firewall be consistent, i.e.





actually accept the connection if it accepted to open the hole.

## 2.3 Peer-to-peer communication with explicit signaling

In these scenarios, two peers that want to communicate use a standard signaling protocol such as SIP or H.323. The communication requests for internal host arrive to an internal server, e.g. the "sip proxy" for the internal domain. In the diagram, we call this agent the "internal server". The following description assumes the use of SIP; scenarios that use an H.323 gatekeeper will use a different message flow, but will involve similar interactions between the gatekeeper and the firewall/NAT.

The scenarios imply that the internal server can receive signaling packets from external hosts and servers. This is an application of the previously described scenarios: TCP or UDP server behind a firewall/NAT.

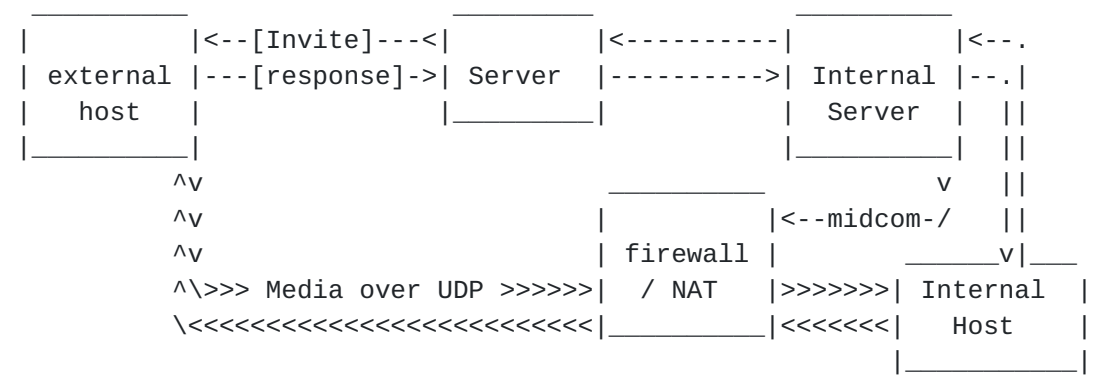
In these scenarios, the "internal server" has to understand the location of the firewall/NAT in order to open the proper holes; this may be difficult in big corporations with multiple firewalls, e.g. in cases when the signaling flow will traverse a different

firewall/NAT than the media path. This will require some form of firewall discovery; however, describing how discovery happens is outside the scope of this document; the scenarios merely assume that discovery somehow has happened, and that the server knows which firewall/NAT will be used.

There are really two scenarios to consider, depending on whether the call initiates from an internal host or from an external host. These two scenarios assume that the firewall/NAT interacts with the internal server. We will then consider a variant, in which the interactions with the firewall/Nat are directly performed by the internal host.

### 2.3.1 Explicit call from an internal host

In this scenario, an internal host calls a third party through the internal server.








The scenario implies that the following operations happen in sequence:

1) The internal host who want to start the call sends an invite message to its preferred internal server. The invite message carries the name of the invited user, and the IP address and UDP ports through which the internal host intends to receive the media, e.g. voice or video.

2) The internal server determines that the target of the invite is located outside the internal domain. If the firewall/NAT performs address and port mapping, the internal server must interact with the firewall/NAT and learn the "external mappings" corresponding to the IP address and UDP ports used by the internal host.

3) The internal server updates the address and port information in the invite message, and relays the call to the "external server."

4) The external server determines that the target of the invite is located in a specific external host. It relays the call to this host.

5) The external host responds to the call. The response provides the IP address and UDP port at which the external host will be expecting to receive the media.

6) The response message is relayed by the external server to the internal server.

7) The internal server receives the response. At this point, it knows the IP addresses and ports used by both the internal and the external host. The internal server interacts with the firewall/NAT using the midcom protocol, to guarantee that the exchange between the internal and external host will be authorized.

8) The response message is relayed by the internal server to the internal host.

9) The external and internal hosts send media packets to the addresses and ports mentioned in the invite and response message; these packets pass through the Firewall/NAT and reach their destination.

We should note that, at step 2, the internal server must learn the external mappings of the internal address and ports; at this stage, it does no know the IP address and ports of the third party.

There is a potential race condition between the signaling message that "responds to the call" and the first media packets sent by the called party. Should the signaling loose the





race, the early media packets will bang against a closed firewall and be clipped. It is in theory possible to design signaling exchange that include a three ways handshake before media transmission can start, followed by a message asking to start ringing only after the availability of all necessary resource has been verified. However, this is not compatible with existing implementations of SIP or H.323, and would require a serious revisiting of the gatewaying between SIP or H.323 and the telephone network.

The description assumes that the hosts use the same UDP ports in both direction of the media communication. This is not necessarily the case. The source IP address may be unpredictable in the case of

multi-homed hosts; the source port may be systematically different from the receive port in some implementation, e.g. parallel processing of the send and receive channels by different software or hardware components.

The scenario does not necessarily require a strict control by the firewall/NAT of the source address and port authorized to send data. Many implementations already support exchange of media level authentication and encryption keys during the call set-up. This provides a level of security that is at least as good as any control of the source address and port: if attackers can manage to read the signaling exchange and get the keys, they can just as well discover the IP addresses and ports, and send forged packets.

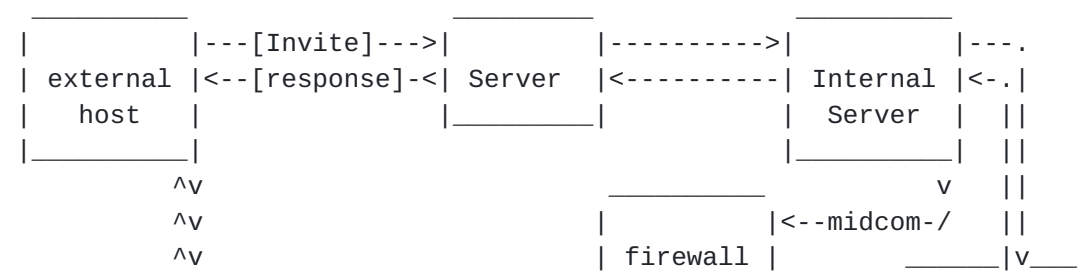
As any other scenario, the firewall/NAT will have the option to accept or refuse the requested hole. In this scenario, we observe two successive interactions between the internal server and the firewall/NAT: to request a mapping at step 2, to provide the address of the external peer at step 7. It is important that the behavior of the firewall/NAT be consistent, and that a hole opening authorized at step 2 not be refused at step 5, when more details are available. If the internal server learns early that the call will be refused, it can terminate it without ever trying to "ring" the external peer. If a call was first authorized and later refused, then the call will proceed, the peer will be ringed and will accept the call, and only at that point discover that there is no way to exchange media; this is obviously very undesirable.

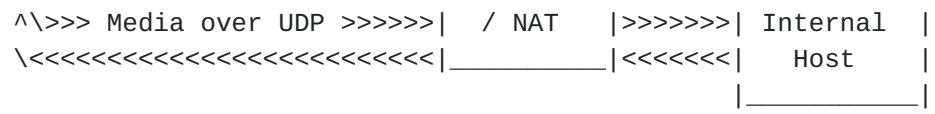
### 2.3.2 Explicit call to an internal host

In this scenario, a third party host calls an internal through the internal server.










The scenario implies that the following operations happen in sequence:

1) The external host who want to start the call sends an invite message to its preferred external server. The invite message carries the name of the invited user, and the IP address and UDP ports through which the external host intends to receive the media, e.g. voice or video.

2) The external server determines that the target of the invite is located in the internal domain. It relays the call to the "internal server."

3) The internal server determines that the target of the invite is located in a specific internal host. It relays the call to this host.

4) The internal host responds to the call. The response provides the "internal" IP address and UDP port at which the internal host will be expecting to receive the media, e.g. voice and video.

5) The internal server receives the host's response. At this point, it knows the IP addresses and ports used by both the internal and the external host.

6) The internal server interacts with the firewall/NAT using the midcom protocol. If the firewall/NAT performs address mapping, the internal server retrieves the mapping of the IP address(es) and port(s) used by the internal host.

7) The internal server prepares an updated response message that reflects the mapping of the internal addresses. It sends the response message to the external server.

8) The response message is relayed to the external host by the external server.

9) The external and internal hosts send media packets to the addresses and ports mentioned in the invite and response message;

these packets pass through the Firewall/NAT and reach their destination.

We note that in this sequence the interaction with the firewall only occurs after the internal host has accepted the call. This can create an annoying effect if the interaction with the firewall fails, equivalent to hearing a void telephone line after picking an incoming call. To avoid this effect, the internal server will have to somehow guarantee that the Firewall/NAT interaction will be successful before relaying the call to the internal host. A possible solution is to include the request of a provisional hole of some sort at step 3, before the call is relayed to the internal host; if the provisional hole is refused by the firewall/NAT, the internal server can refuse the call without disturbing the internal user.

Just like scenario 2.3.1, it may not be possible or desirable to predict or check the source IP address and UDP ports used by the internal and external hosts.

### 2.3.3 Firewall interaction by the internal host

It is possible to update the previous two scenarios so that the internal host interacts directly with the Firewall/NAT, rather than relying on the internal server. This set-up has the advantage of avoiding the "void telephone line" effect mentioned in the previous scenario: the internal host that receives the invite can pick the UDP ports used for audio and video and interact with the firewall/NAT before "ringing" the user; if the interaction fails, the call can be rejected without bothering the user. This set-up however has the disadvantage that all internal hosts must become able to interact with the Firewall/NAT, which in many cases may not be practical.

The direct interaction between the internal host and the NAT/Firewall is already described in the "peer-to-peer" scenarios of the previous section. The only difference between these scenarios is the possibility for the internal server to pass some form of "authorization token" to the internal host.

### 2.3.4 Early media

The "early media" scenario is an important variations of the scenario 2.3.1. Early media designates media transmission sent before the actual completion of the call. Examples are ringing tones and voice messages describing particular network conditions, such as "we are trying to locate your correspondent." In the early media scenario, the following interactions will happen in sequence:

1) The internal host who want to start the call sends an invite

message to its preferred internal server, as in 2.3.1,

2) The internal server determines that the target of the invite is

located outside the internal domain. If the firewall/NAT performs address and port mapping, the internal server must interact with the firewall/NAT and learn the "external mappings" corresponding to the IP address and UDP ports used by the internal host, as in 2.3.1. **In addition, the internal server requests the** authorization to receive packets from a yet unspecified external source.

3) The internal server updates the address and port information in the invite message, and relays the call to the "external server."

4) The external server, or a secondary server acting on its behalf, sends a stream of voice packets towards the "external mappings" of the IP address and UDP ports used by the internal host,

5) The firewall/NAT receives these packets and forwards them to the internal host,

6) The call proceeds as in 2.3.1.

There is a common telephony practice of sending recorded announcements during call set-up; the source IP address of these announcements is not likely to be the same as the source IP address used after call set-up is complete. It is theoretically possible to use the equivalent of call transfer to switch between multiple source in a controlled fashion, but this introduce a lot of signaling complexity, and is incompatible with currently deployed hardware and software. In practice, this scenario requires that the firewall/NAT "opens a hole" without knowing the IP address and port of the external peer.

### 2.3.5 Mobility of the external host

The mobility scenario can be thought as a complication of scenarios 2.3.1 **or 2.3.2, in which the IP address of one of the peers is** allowed to change during a call, due to either mobility or network renumbering. The scenario involves the following exchanges:

1) The external host receives a new IP address, and sends a signaling packet to the "internal server" mentioning the new IP address,

2) The internal server programs the firewall/NAT to start authorizing packets between this new address and the internal host,

3) In parallel with 2, the internal server relays the signaling message to the internal host,

4) The internal and external hosts exchange packets with the new

address; the firewall/NAT authorizes these packets to proceed.

SIP supports that through the re-invite mechanism, but we should note that there is either a gap in the call or a race condition between media packets with the new source address and the signaling message. The external host is likely to source packets with its new address immediately after the address change; if the packets arrive before the firewall/NAT has been programmed to accept them, the packets will bang against the closed firewall/NAT and be dropped.

### 2.3.6 Multiple ports, port ranges

The SIP messages use the encoding defined in SDP [RFC2237] to describe the IP addresses and TCP or UDP ports used my the various media. In many cases, a single media stream will be spread over multiple ports. SDP carries only one port number per media, and states that "other ports used by the media application (such as the RTCP port) should be derived algorithmically from the base media port." When the media is transmitted using RTP [RFC1889], the choice of the port number is very specific: "for UDP and similar protocols, RTP uses an even port number and the corresponding RTCP stream uses the next higher (odd) port number; if an application is supplied with an odd number for use as the RTP port, it should replace this number with the next lower (even) number." This obviously poses a constraint to the allocation of ports and mappings by a NAT.

Most media streams are transmitted using a single pair of RTP and RTCP ports. It is possible however to transmit a single media over several RTP flows, for example using hierarchical encoding. In this case, SDP will encode the port number used by RTP on the first flow, and the number of flows, as in:

```
m=video 49170/2 RTP/AVP 31
```

In this example, the media is sent over 2 consecutive pairs of ports, corresponding respectively to RTP for the first flow (even number, 49170), RTCP for the first flow (odd number, 49171), RTP for the second flow (even number, 49172), and RTCP for the second flow (odd number, 49173). This places a further constraint to any NAT firewall traversal scheme: we must be able to ensure that a consecutive range of N ports starting with an even number is mapped to another consecutive range of N ports, also starting with an even number.

## 2.4 IPv6 Scenarios

All of the scenarios mentioned above can be modified if the domains have been upgraded to run IPv6. One difference between the IPv6 and IPv4 scenarios is that the internal hosts can use global addresses; however, there will also be cases in which address translation is required after the introduction of IPv6, notably if one provides

interoperation between IPv6 and IPv4 using the NAT-PT scheme [RFC2766]. When any form of address translation is required, e.g. between IPv6 and IPv4 addresses, the scenarios are basically

unchanged; what may change is the content of the MIDCOM protocol messages, which will have to include a mix of IPv4 and IPv6 addresses. On the other hand, when only global addresses are used in the exchanges, the scenario are modified; the "middle box" does not necessarily disappears, since in many domains there will still be the need to perform explicit authorizations before letting data go in and out; in these cases the "Firewall/NAT" combination becomes strictly a "Firewall". In this section, we review how the introduction of IPv6 and the use of global addresses can affect the three classes of scenarios mentioned in the previous sections.

The transition to IPv6 will require the introduction of relay routers, as specified in [RFC3056]; we discuss here how the MIDCOM protocol can be used to open holes for the "tunnels" leading to these relay routers.

In addition, the global addressing allows the introduction of another scenario, the use of IPSEC between an internal and an external host.

### 2.4.1 IPv6 TCP or UDP server behind a firewall

In this scenario, the internal host publishes the IP address and TCP port number at which it can be joined in a name server, using for example SRV and A6 records in the DNS. The sequence of operation is the same as in the IPv4 case, but each of the step has a different emphasis:

1) The internal host interacts with the firewall, using the midcom protocol. As a result of the interaction, the firewall learns the IP address and TCP port that the host will use.

2) The internal host publishes the information in a name server.

3) The external host obtains the information from the name server.

4) The external host issues a TCP connection request, and sends a TCP SYN packet.

5) The firewall receives the packet, checks that the destination address and port are authorized, and relays the TCP SYN packet to the internal host.

6) After that point, the TCP connection proceeds.

The only reason for the first step in the scenario is access control. If the domain's policy is to authorize all hosts to receive all traffic, there is no need for this step - indeed, the firewall becomes mostly a transparent IPv6 router. The impact of IPv6 on the two variants of that scenario is obvious: the use of UDP will have

to be authorized if needed, and there may be a need to let a third party perform the authorizations.

### 2.4.2 Peer-to-peer communication with ad-hoc rendezvous and IPv6

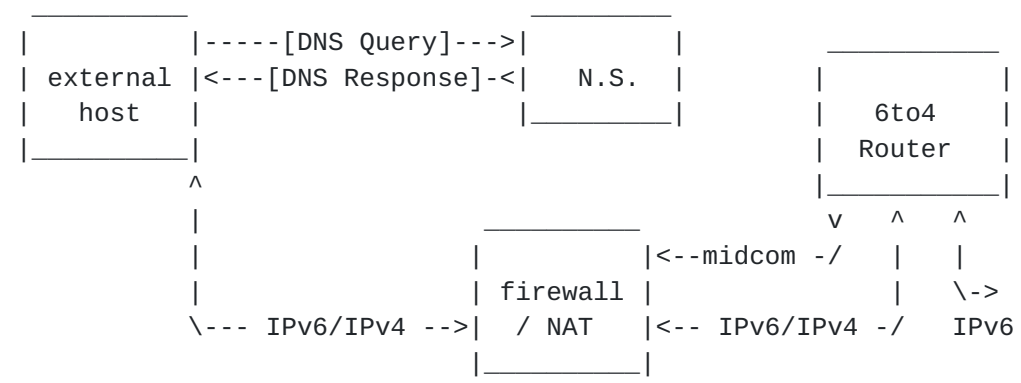
If both peers have a global IPv6 address, they will only have to interact with a firewall if the domain's manager insists on having a firewall control all incoming traffic; there will not be a need for a NAT functionality. The internal host may still need to interact with the firewall in order to "open a hole" for the packets coming from the remote peer, but it will always be able to specify the complete "five tuple" of protocol type, IP addresses and UDP ports; the problem exposed in the case when both hosts were being firewalls disappears.

### 2.4.3 Peer-to-peer communication with explicit signaling and IPv6

This scenario is also made simpler by the availability of global addresses. In the case of a call from an internal host, the internal server will not have to rewrite the addresses in the outgoing "invite"; it will only have to interact with the firewall to open a hole after the reception of the response. In the case of a call to an internal host, the internal server may still have to interact with a firewall if the domain managers insist on requiring this type of protection; it will do so with an explicit knowledge of the IPv6 addresses and UDP ports used by both ends of the connection.

### 2.4.4 IPv6 transition service behind a firewall/NAT

A typical IPv6 transition scenario is described in [RFC3056]. In this scenario, IPv6 is progressively made available by installing in each site a "6to4" router, which receives IPv6 packets through automatic tunnels and forwards them to internal IPv6 hosts.




In this scenario, the 6to4 router provides the internal IPv6 hosts with IPv6 addresses; the IPv6 prefix in these addresses is based on a "global" IPv4 address of the domain. The IPv6 hosts will publish their IPv6 addresses in the DNS. The external hosts will send IPv6 packets encapsulated in IPv4 headers, whose destination will be the

internal 6to4 router; the 6to4 router will receive the packets sent by internal hosts to external hosts, and will encapsulate them with

adequate IPv4 headers.

The scenario implies that the following operations happen in sequence:

1) The 6to4 router interacts with the firewall/NAT, using the midcom protocol. As a result of the interaction, the 6to4 router learns a global IPv4 address that it can use to build a 6to4 prefix.

2) The internal hosts publish IPv6 addresses based on this prefix in a name server.

3) The external host obtains the information from the name server.

4) The external host sends IPv6 packets towards this address.

5) The firewall/NAT receives the packet, notes that these are IPv6 packets carried in IPv4 (protocol type = 41), translates the destination address if necessary and relays the packet to the 6to4 router.

6) The 6to4 router removes the IPv4 header and forwards the IPv6 packet to the internal host.

7) When the 6to4 router receives an IPv6 packet, it determines the adequate IPv4 destination, and uses it to build an encapsulation IPv4 header.

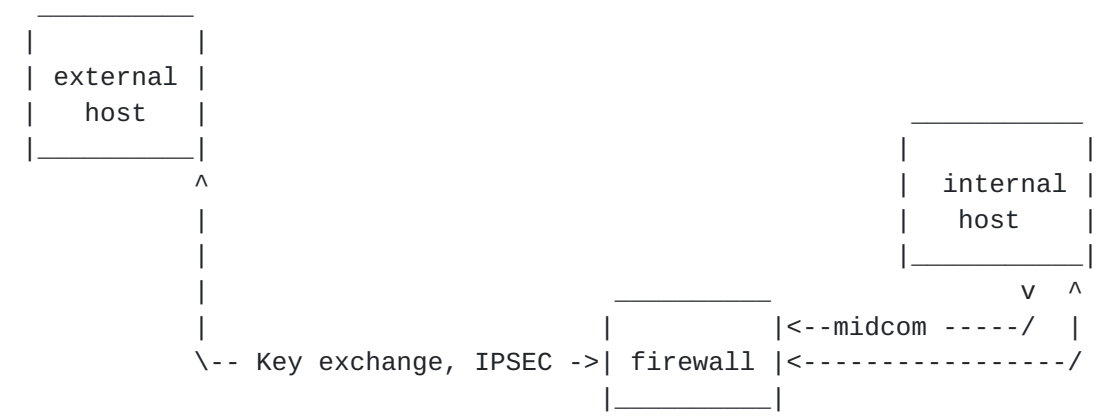
8) The firewall/NAT receives the encapsulated packet. It may perform translation of the source address if needed. It forwards the packet to the IPv4 destination.

In the diagram, we depict only one external host, but this is an example, not a limitation.

It is quite clear that, if fire walling function are desired for the IPv6 traffic, these functions will have to be provided by the 6to4 router.

### 2.4.5 Enabling an IPSEC connection between IPv6 hosts

Once IPv6 provides global addresses to internal hosts, it becomes possible to establish IPSEC associations between an internal host and an external host. The establishment of the association will start by a key exchange, and will continue with the exchange of encrypted traffic.




The scenario implies that the following operations happen in sequence:

1) The internal and external hosts decide to communicate, e.g. after the internal host finds the address of the external host in the DNS.

2) The internal host and the external host exchange key negotiation packets (IKE). The firewall passes these packets.

3) The internal host uses the midcom protocol to signal to the firewall that it is going to exchange encrypted traffic with an external host, and obtains the authorization to proceed.

4) IPSEC packets are exchanged.

5) After the hosts have finished using the IPSEC association, the internal host may interact with the firewall and close the hole.

We should note that this scenario requires that the firewall delegates some of its control functions to the internal host: encrypted traffic cannot be inspected.

As in all other scenarios, the firewall will have to explicitly authorize the opening of a hole for the IPSEC association.

## 3 Security Considerations

Firewalls are used by domain managers to control the traffic that can be exchanged between their domain and the Internet. In the scenarios that we described, this control is relaxed in order to enable certain applications. Relaxing the control has to be a conscious decision of the domain manager.

## 4 IANA Considerations

The purpose of this memo is to document the allocation by IANA of an

Internet; there is no need for any recurring assignment.

## 5 Copyright

The following copyright notice is copied from RFC 2026 [Bradner, 1996], Section 10.4, and describes the applicable copyright for this document.



## 6 Intellectual Property

The following notice is copied from RFC 2026 [Bradner, 1996], Section 10.4, and describes the position of the IETF concerning intellectual property claims made against this document.

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use other technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances

of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such

proprietary rights by implementers or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

## 7 Acknowledgements

The discussion presented here was triggered by the meeting of the MIDCOM working group in Minneapolis. An initial description of the "TCP server" scenario was sent to the group's e-mail list by Eliot Lear.

## 8 References

[RFC3056] B. Carpenter, K. Moore. "Connection of IPv6 Domains via IPv4 Clouds." RFC 3056, February 2001.

[RFC2237] M. Handley, V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

[RFC1889] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson. "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996.

[RFC2766] G. Tsirtsis, P. Srisuresh. "Network Address Translation - Protocol Translation (NAT-PT)", RFC 2766, February 2000.

[MIDBOXFRAME] Middlebox Communication Architecture and Framework. Work in progress.

## 9 Author's Address

Christian Huitema
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399

Email: huitema@microsoft.com