Internet Draft
Document: draft-renkel-middlebox-tunnels-00.ps
Expires: December 2002

Jim Renkel
The CommWorks Corp.
a 3Com company
June 2002

NAT and Middlebox Tunnels

Status of this Memo

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

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

This Internet Draft compares Network Address Translation (NAT) and Realm Specific IP (RSIP) tunnels, and the advantages and disadvantages of each. It then shows how the advantages can be combined by implementing NAT as a tunnel in hosts, while minimizing the disadvantages. Based on the advantages of middlebox tunnels in general and implementing NAT as a middlebox tunnel in particular, it then recommends that the Middlebox Communications architecture protocol (MIDCOM) be amended to support tunnels between hosts and middleboxes.

Table of Contents

1. Introduction....3
2. Network Address Translation (NAT)....4
3. Realm Specific IP (RSIP)....6
4. NAT tunneling....7
5. Summary....9
6. Recommendations....10
7. Author's Address....10
8. Full Copyright Statement....11

Table of Figures

Figure 1 - FTP in a network without NAT or tunnels....3
Figure 2 - FTP in a network with NAT....4
Figure 3 - FTP in a network with an RSIP tunnel....6
Figure 4 - FTP in a network with a NAT tunnel....8

1. Introduction

Because of the growth of the Internet, IPv4 addresses have become scarce. To help alleviate the scarcity, network address translation (NAT) can be used between IP realms. While helping to alleviate address scarcity, NAT has introduced other problems for some types of Internet connections.

One type of Internet connection that has been caused problems by NAT is a control connection that is used to, well, control another data connection. The control connection has to refer to one or more IP addresses / ports that define the data connection. But if these data connection addresses / ports are translated between the addressed endpoints, the endpoints may not accurately known the IP addresses / ports by which they are known to the other endpoint and, hence, can't refer to them accurately in the control connection.

An example of such control and data connections is FTP. We first look at how FTP operates without network address translation. This is described in the following figure and text. The control connection is in red, the data connection in blue.

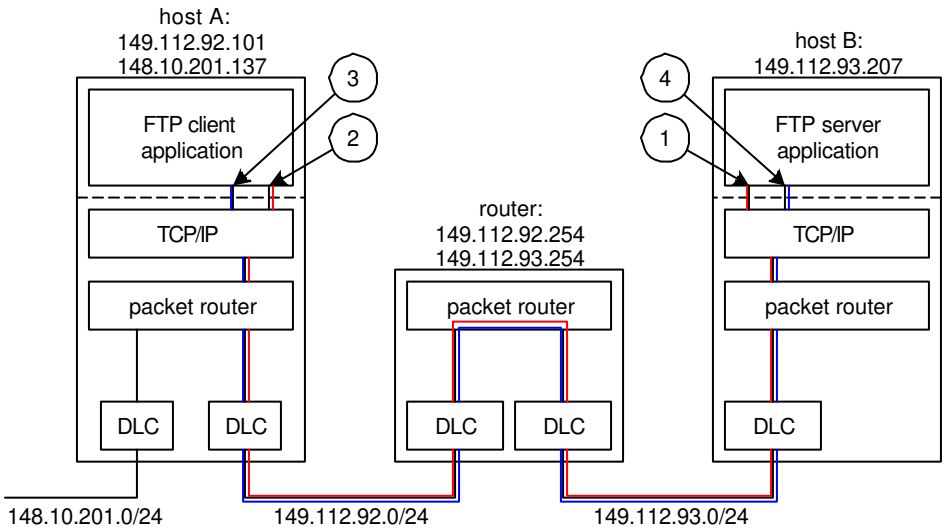

Figure 1 - FTP in a network without NAT or tunnels

FTP is a client / server application. The FTP client is located in host A, which is "dual homed" on to the 148.10.201.0/24 and 149.112.92.0/24 subnets (The dual homing isn't very interesting in this example, but will be in later ones.). The FTP server is located in host B, which is on the 149.112.93.0/24 subnet. The two 149.112.*.* subnets are joined by a router.

The sequence of events in the FTP application, in greatly simplified form, is:

1. The FTP server in host B:
- obtains a service access point (SAP; a.k.a., a "socket") in host B;
- defines it as TCP port 21 (The default FTP server control port.); and
- listens on it for a control connection from a client;

2. The FTP client in host A:
- obtains a SAP with any port number (say, 1037 is allocated), and
- connects to the FTP server using TCP on that SAP / port, specifying 149.112.93.207 and 21 as the remote address and port number;
- the packet router in host A selects the appropriate DLC for the connection;
- the selected DLC in host A:
    - recognizes that the specified remote address is not in the subnet to which it is attached, and

- sends packets transmitted in that connection to the default gateway for the subnet, which is the router connecting the two 149.112.*.* subnets;

- packets in the reverse direction are handled similarly;

3. After logging onto the server using the FTP protocol, the FTP client:
   - obtains another SAP with any port number (say, 1038);
   - listens on it for a TCP data connection from the server; and
   - then sends the server a command to open a data connection to that port;
4. On receiving the command from the FTP client, the FTP server:
   - opens a data connection to the identified address and port;
   - the client sends an FTP data transfer command on the control connection; and
   - the data is transferred over the data connection.

We now look at two solutions to the address scarcity problem, and see how FTP operates with them.

2. Network Address Translation (NAT)

Because of address scarcity, the 149.112.92.0 subnet is made into a private IP realm by adding NAT functionality to the router. This is described in the following figure and text (Changes from the previous figure and text are in italics.).

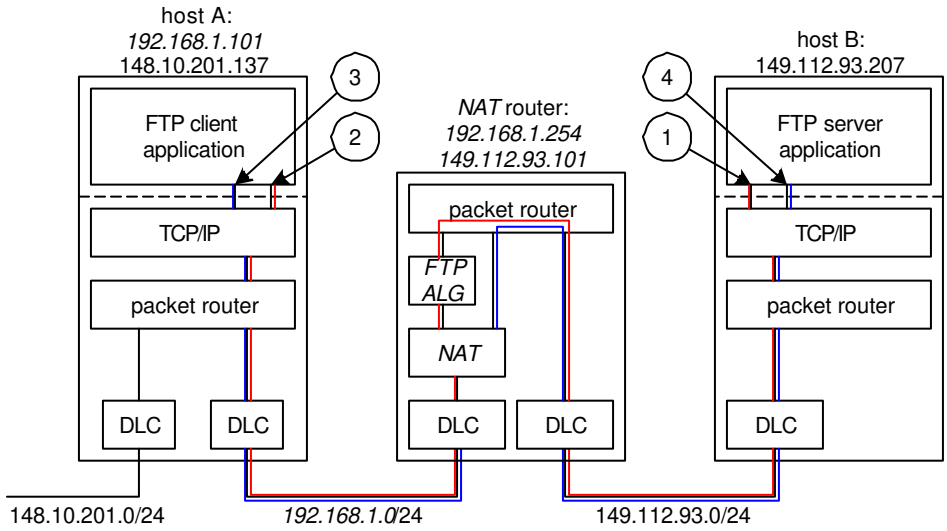

Figure 2 - FTP in a network with NAT

The FTP client is located in host A, which is "dual homed" on to the 148.10.201.0/24 and *192.168.1.0*/24 subnets. The FTP server is located in host B, which is on the 149.112.93.0/24 subnet. The two subnets are joined by a *NAT* router, *which appears as a host (149.112.93.101) to one subnet*. The sequence of events in the FTP application, in greatly simplified form, is:

1. The FTP server in host B obtains a service access point (SAP; a.k.a., a "socket") in host B:
   - defines it as TCP port 21 (The default FTP server control port.); and
   - listens on it for a control connection from a client;
2. The FTP client in host A:
   - obtains a SAP with any port number (say, 1037 is allocated); and
   - connects to the FTP server using TCP on that SAP / port, specifying 149.112.93.207 and 21 as the remote address and port number;
   - the packet router in host A selects the appropriate DLC for the connection;

- the selected DLC in host A:
  - recognizes that the specified remote address is not in the subnet to which it is attached, and
  - sends packets transmitted in that connection to the default gateway for the subnet, which is the *NAT* router connecting the two subnets;
- *the NAT router:*
  - *allocates a port number on the 149.112.93.* subnet (say,2049) to the connection,*
  - *flows the packets through the NAT function (which translates 168.192.1.101:1037 reversibly to 149.112.93.101:2049 in IP headers in the control connection); and*
  - *recognizing that the connection is an FTP control connection, also flows the packets through the FTP application level gateway (ALG);*
- packets in the reverse direction are handled similarly;

3. After logging onto the server using the FTP protocol, the FTP client:
   - obtains another SAP with any port number (say, 1038);
   - listens on it for a TCP data connection from the server; and then
   - sends the server a command to open a data connection to that port *(168.192.1.101:1037)*;

- *the FTP ALG:*
  - *recognizes the command;*
  - *allocates a port number on the 149.112.93.* subnet (say,2050), and*
  - *rewrites the command with the new address and port number;*

4. On receiving the command from the FTP client, the FTP server:
   - opens a data connection to the identified address and port;
   - *the NAT function in the router translates 168.192.1.101:1038 reversibly to 149.112.93.101:2050 in IP headers in the data connection,*
   - the client sends an FTP data transfer command on the control connection and
   - the data is transferred over the data connection.

The benefits of NAT are:

- address scarcity problems have been solved, without any changes to any host application software;
- no overhead is added to the packets between the host and the router.

The costs of NAT are:

- ALGs must be written for every control protocol; and
- NAT router software must be upgraded as new control protocols are introduced.

The effort to implement new ALGs and update NAT router software is enormous: every NAT router type must have an ALG written for it and for every control protocol, and then every NAT router instance must be upgraded. This requirement to develop and upgrade NAT router software seriously impedes and delays the deployment and use of new control protocols.

3. Realm Specific IP (RSIP)

Alternately, the 149.112.92.0 subnet is made into a private IP realm by adding *RSIP* functionality to the router. This is described in the following figure and text (Changes from the previous figure and text describing NAT are in italics.).

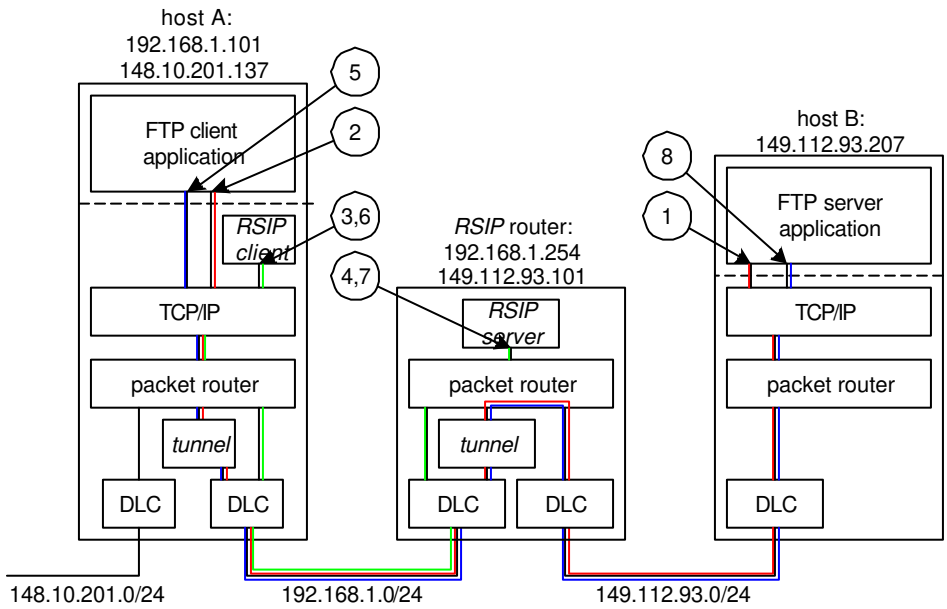

Figure 3 - FTP in a network with an RSIP tunnel

The FTP client is located in host A, which is "dual homed" on to the 148.10.201.0/24 and *192.168.1.0*/24 subnets. The FTP server is located in host B, which is on the 149.112.93.0/24 subnet. The two subnets are joined by a *RSIP* router, which appears as a host (149.112.93.101) to one subnet. *Host A has opened a tunnel to the RSIP router and, thus, appears to be "virtually homed" on 149.112.93.0/24 subnet. Since the FTP client supports multi-homing, the tunnel is treated as just another subnet adapter.* The sequence of events in the FTP application, in greatly simplified form, is:

1. The FTP server in host B:
   - obtains a service access point (SAP; a.k.a., a "socket") in host B;
   - defines it as TCP port 21 (The default FTP server control port.); and
   - listens on it for a control connection from a client;
2. The FTP client in host A obtains an SAP with any port number.
3. *The RSIP client requests an IP address and port number for this connection from the RSIP server in the router (If ports had been previously allocated and are unused, they may be used at this time without interacting with the RSIP server.);*
4. *The RSIP server allocates an IP address and port number* (say, *149.112.93.101:*1037 is allocated).
   - *Back in host A, the FTP client* connects to the FTP server using TCP on that SAP / port, specifying 149.112.93.207 and 21 as the remote address and port number;
   - the packet router in host A selects the appropriate DLC for the connection, *which is the tunnel*;
   - *the tunnel DLC*
     - *adds tunnel, UDP, and IP headers to packets (Details vary with tunnel type.), and*
     - *gives the packets to the subnet DLC (In practice, the tunnel DLC may just add a tunnel header and recirculate the packets through the UDP/IP protocol stack and the packet router.);*

- *the packets are sent to the RSIP router;*
- *the RSIP router:*
  - *removes the IP. UDP, and tunnel headers, and*
  - *forwards the packets to host B*;
  - packets in the reverse direction are handled similarly;

5. After logging onto the server using the FTP protocol, the FTP client obtains another SAP.

6. *The RSIP client requests an IP address and port number for this connection (Again, a previously allocated but unused port may be used without interacting with the RSIP server.);*

7. *The RSIP server allocates an IP address and port number* (say, *149.112.93.101:*1038 is allocated).
   - *The FTP client* listens on the SAP for a TCP data connection from the server; and
   - sends the server a command to open a data connection to that port; ***Note: no rewriting of the FTP control stream by an ALG is required, as it already contains the correct subnet 149.112.93.0/24 subnet address and port number*** *(This last point is emphasized because it is the crux of the value of tunnels in dealing with NAT.);*

8. On receiving the command from the FTP client, the FTP server opens a data connection to the identified address and port;
   - the client sends an FTP data transfer command on the control connection; and
   - the data is transferred over the data connection.

The benefits of RSIP are:
- as with NAT, address scarcity problems have been solved, without any changes to any host application software;
- ALGs need not be written for every control protocol, and
- NAT router software need not be upgraded as new control protocols are introduced.

The costs of RSIP are:
- an RSIP client and a tunnel DLC are required in the host platform,
- an RSIP server and tunnel DLC are required in the router, and
- packets between host and the router have additional tunnel, UDP and IP headers in them (In the case of very small packets, the additional headers can almost double the size of the packets.).

4. NAT tunneling

An ideal solution to the address scarcity problem would combine the benefits of the NAT and RSIP solutions while minimizing development and deployment costs. It would:
- solve address scarcity problems without any changes to any host application software;
- not add overhead to the packets between the host and the router;
- not need ALGs to be written for every control protocol, and
- not need NAT router software to be upgraded as new control protocols are introduced.

The solution is to implement NAT in the host using RSIP's tunneling paradigm rather than NAT's default gateway forwarding paradigm. This is done by implementing an "inverse NAT" tunnel DLC in the host that, rather than adding headers to packets, rewrites the IP header (and UDP or TPC header, if port number translation is done.) in such a way that, when used in combination with the NAT function in the router, packets transmitted by the router are identical to those generated by the host before they were modified by the inverse NAT tunnel DLC, and packets received by the application are identical to those received by the router.

This solution is called "NAT tunneling" and is described in the following figure and text (Changes from the previous figure and text describing RSIP are in italics.).

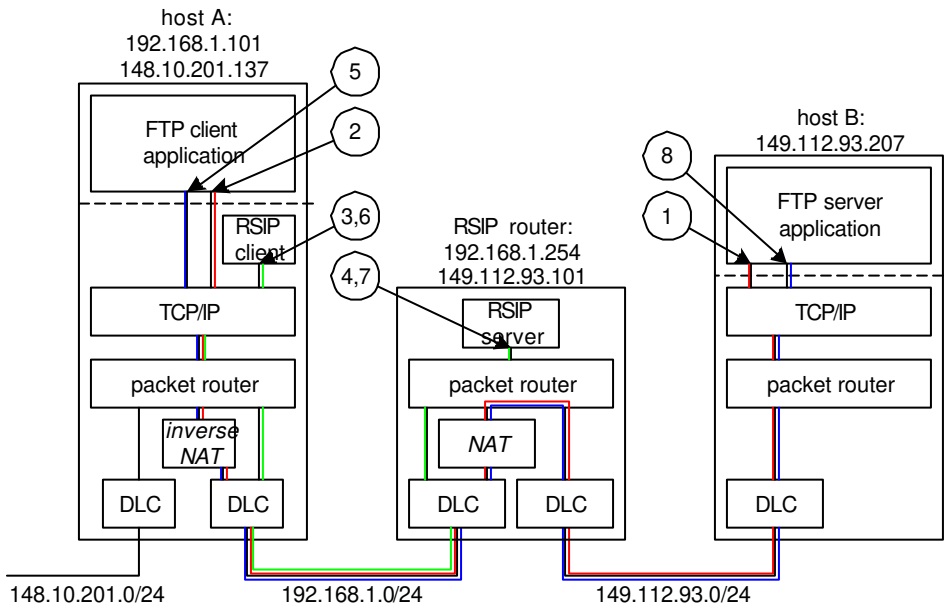

Figure 4 - FTP in a network with a NAT tunnel

The FTP client is located in host A, which is "dual homed" on to the 148.10.201.0/24 and *192.168.1.0*/24 subnets. The FTP server is located in host B, which is on the 149.112.93.0/24 subnet. The two subnets are joined by a *NAT* router *with an RSIP server*, which appears as a host (149.112.93.101) to one subnet. Host A has opened an *"inverse NAT"* tunnel to the RSIP router and, thus, appears to be "virtually homed" on 149.112.93.0/24 subnet. Since the FTP client supports multi-homing, the tunnel is treated as just another subnet adapter. The sequence of events in the FTP application, in greatly simplified form, is:

1. The FTP server in host B:
   - obtains a service access point (SAP; a.k.a., a "socket") in host B;
   - defines it as TCP port 21 (The default FTP server control port.); and
   - listens on it for a control connection from a client;
2. The FTP client in host A obtains an SAP with any port number.
3. The RSIP client requests an IP address and port number for this connection from the RSIP server in the router (If ports had been previously allocated and are unused, they may be used at this time without interacting with the RSIP server.);
4. The RSIP server allocates an IP address and port number (say, 149.112.93.101:1037 is allocated).
   - Back in host A, the FTP client connects to the FTP server using TCP on that SAP / port, specifying 149.112.93.207 and 21 as the remote address and port number;
   - the packet router in host A selects the appropriate DLC for the connection, which is the *inverse NAT* tunnel;
   - the tunnel DLC
     - *rewrites the TCP and IP headers in packets*, and
     - gives the packets to the subnet DLC (In practice, the tunnel DLC may just recirculate the packets through the packet router.);
   - the packets are sent to the RSIP router;
   - the RSIP router:
     - *rewrites the IP and TCP headers (Note: the IP and TCP headers are now identical to those generated by the host prior to rewriting by the inverse NAT tunnel DLC.)*, and
     - forwards the packets to host B;

- packets in the reverse direction are handled similarly;

5. After logging onto the server using the FTP protocol, the FTP client obtains another SAP.
6. The RSIP client requests an IP address and port number for this connection (Again, a previously allocated but unused port may be used without interacting with the RSIP server.);
7. The RSIP server allocates an IP address and port number (say, 149.112.93.101:1038 is allocated).
   - The FTP client listens on the SAP for a TCP data connection from the server; and
   - sends the server a command to open a data connection to that port; Note: no rewriting of the FTP control stream by an ALG is required, as it already contains the correct subnet 149.112.93.0/24 subnet address and port number;
8. On receiving the command from the FTP client, the FTP server opens a data connection to the identified address and port;
   - the client sends an FTP data transfer command on the control connection; and
   - the data is transferred over the data connection.

The benefits of NAT tunneling are:

- as with NAT and RSIP, address scarcity problems have been solved, without any changes to any host application software;
- packets between host and the router have nothing added to them;
- ALGs need not be written for every control protocol, and
- router software need not be upgraded as new control protocols are introduced.

The costs of NAT tunneling are:

- an RSIP client and an inverse NAT tunnel DLC are required in the host platform, and
- an RSIP server is required in the router.

5. Summary

Network Address Translation (NAT) routers can be used to alleviate IP4 address scarcity, but they can cause problems with so-called control protocols, These problems can be solved by including application level gateways (ALGs) in the NAT routers, but this requires one ALG for each control protocol and for each NAT router. If there are n types of NAT routers, then n ALGs must be developed each time another control protocol is to be supported. Additionally, if there are x instances of the n NAT routers deployed, then all x must be upgraded each time a new control protocol is introduced. Thus, the effort to support one additional control protocol is $O(n)+O(x)$. If , over time, m additional control protocols are to be supported, then the total effort is $m*(O(n)+O(x))$.

The IETF MIDCOM working group is attacking this problem in a different way, by defining a middlebox communication protocol (MIDCOM) that allows applications and agents to see into the network translation mechanism and solve the control protocol problem external to the NAT router. Initially, all NAT routers would need to be upgraded to include a MIDCOM server. The effort to do this is the same as adding an ALG for a new control protocol to all NAT routers, $O(n)+O(x)$.

Now, however, when another control protocol is to be supported, the NAT routers need not be modified, only applications using that control protocol. If there are a applications that use the control protocol and y instances of that application deployed, then the effort to support the control protocol is $O(a)+O(y)$. It is a reasonable assumption that $a<n$ and $x<y$. Thus, $O(a)+O(y)<O(n)+O(x)$, assuming the proportionality constants in $O(a)$ and $O(n)$ and in $O(y)$ and $O(x)$ respectively are of the same magnitude. If, over time, m additional control protocols are to be supported, then the total effort with the MIDCOM protocol is $(O(n)+O(x))+m*(O(a)+O(y))$, which is less than $m*(O(n)+O(x))$.

Alternately, in cases such as SIP where the control protocol flows through an intermediate system, e.g., a SIP proxy, the SIP proxy may be made a MIDCOM agent for the host applications, rewriting their control protocol messages as they flow through the proxy. To a first order of magnitude, this requires approximately the same total effort as modifying the applications, and this effort will not be examined further herein.

The RSIP protocol attacks this problem in a completely different way, as described above. While RSIP as defined requires tunnels between hosts and middleboxes, we have shown how NAT can be supported in hosts as a tunnel. Thus RSIP's approach is applicable to NAT support.

Initially, an RSIP server must be implemented for each type of NAT router and each deployed NAT router must be upgraded. As above, this requires *O(n)+O(x)* effort. Additionally, an RSIP client must be developed for each platform type and deployed in each platform. Assuming *p* platform types and *z* deployed platforms, this requires *O(p)+O(z)* effort. While *p* is certainly less than *a*, the relation ship between *x* and *z*, and thus between *O(p)+O(z)* and *O(a)+O(y)* is uncertain. But, once RSIP support is developed and deployed, no additional effort is required for additional control protocols. The total effort to develop and deploy RSIP support is *(O(n)+O(x))+(O(p)+O(z))*. This is certainly less than either *(O(n)+O(x))+m*(O(a)+O(y))* or *m*(O(n)+O(x))*, even for small values of *m*.

If MIDOCM supported tunnels, this same total effort reduction would be true of MIDCOM. It is believed that the effort required to have MIDCOM support tunnels is extremely small.

In the real world, it is unlikely that any of the above strategies will be adopted as the only strategy used to support NAT. But by including the RSIP strategy in the mix, the total effort can potentially be significantly reduced over a mix of strategies that does not include the RSIP strategy.

Beyond a potentially large effort saving, the RSIP tunnel strategy has other advantages. These advantages accrue primarily where there are a large number of connections between two realms. In this case, the collection of connections can be managed as a whole, rather than individually, with respect to aspects such as security, bandwidth reservation, route optimization, QoS, etc. This can result in significant operational savings.

Additionally, where the connections cross more than two realms, RSIP tunnels (excluding NAT tunnels, in this case) can be managed end-to-end between the two end realms, whereas NAT connections, even with MIDCOM support, must be managed in each realm that they traverse. Again, this can result in significant potential operational savings. While implementing true tunnels will no doubt require more effort than just supporting NAT as a tunnel, the extra effort may be out-weighed by the operational savings.

6. Recommendations

Based on the above, we recommend that the MIDCOM architecture and requirements be amended to include support for tunnels, whether true tunnels or NAT tunnels as described herein.

7. Author's Address

Jim Renkel
The CommWorks Corp., a 3Com company
3800 Golf Rd.
Rolling Meadows, IL, USA
Phone: +1.847.262.2539
Email: james_renkel@commworks.com

8. Full Copyright Statement

Copyright (C) The Internet Society (2002). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English. The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns. This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE."