Proceedings of the Special Workshop on Intelligence at the Network Edge

San Francisco, California, USA, March 20, 2000

RSIP: ADDRESS SHARING WITH END-TO-END SECURITY

Michael Borella and Gabriel Montenegro

© 2000 by The USENIX Association. All Rights Reserved. For more information about the USENIX Association: Phone: 1 510 528 8649; FAX: 1 510 548 5738; Email: office@usenix.org; WWW: http://www.usenix.org. Rights to individual papers remain with the author or the author's employer. Permission is granted for noncommercial reproduction of the work for educational or research purposes.This copyright notice must be included in the reproduced paper. USENIX acknowledges all trademarks herein.

RSIP: Address Sharing with End-to-End Security

Michael S. Borella
3Com Corp.
1800 W. Central Rd.
Mount Prospect, IL 60056
mike_borella@3com.com

Gabriel E. Montenegro
Sun Microsystems Laboratories
901 San Antonio Road, MS UMPK15-214
Mountain View, CA 94303
gab@sun.com

Abstract

Realm Specific IP (RSIP) is a new architecture under consideration in the Internet Engineering Task Force (IETF) that can potentially alleviate some of the problems associated with partitioning of the Internet address space due to, for example, the shortage of IPv4 addresses. It is being positioned as a replacement for Network Address Translation (NAT), because, among other things, it can support end-to-end security via IPsec, which NAT cannot. This paper introduces the motivation behind RSIP, the RSIP architecture, and provides a basic overview of the RSIP protocol.

1 Introduction

IPv4, the current version of the Internet Protocol, supports 32 bits of address space, which means that over 4 billion individually addressable hosts can be on the Internet. At the time of its inception, IPv4 designers could not have imagined the explosive growth of the Internet in the mid-to-late 1990's. As a result of this growth, as well as overly generous address allocation schemes of the past, it is becoming increasingly difficult and expensive to obtain IP addresses. Although IPv6, which has 128 bits of address space, has been approved by the Internet Engineering Task Force (IETF), there is no clear upgrade path from IPv4 to IPv6. Concerns that IPv6 may either never deploy, or may only partially deploy, are being expressed [1].

Thus far, network users and administrators have responded to the address shortage by deploying Network Address Translation (NAT) [2] in boundary routers. However, this technology requires that the NAT be aware of any application run across it that transmits IP addresses or TCP/UDP port information in the packet payload. Furthermore, NAT inhibits the use of end-to-end security via IPsec [3]. The latter, in particular, is a major disadvantage, considering the popularity of virtual private networks and the well-known need for security in e-commerce and business communications.

Currently, network administrators must choose between a complete solution that is not supported by many vendors and incompatible with the rest of the Internet (IPv6), and a stop-gap solution that introduces many problems and prevents the deployment of some popular applications (NAT). Realm Specific IP (RSIP) [4] has been proposed as a replacement for NAT that will have little impact on application layer protocols, and will inter-operate with IPsec. RSIP has the additional advantage that it can coexist with NAT, and therefore networks using NAT can be smoothly upgraded, in part or in whole, to use RSIP.

This paper presents an introduction to RSIP, focusing on how it is different from NAT, and how it can support end-to-end IPsec. Since RSIP is still in the draft phase of its journey towards IETF standardization, we include a discussion of architectural and protocol issues that are currently being addressed and evaluated.

2 Background

In this section we discuss the IP address shortage and how it is expected to become a critical issue in the near future. We then describe the operation of NAT, which is required in order to appreciate the alternative

2.1 The Coming IP Address Crunch

The technical media and popular press have been creating much hype about the IP address shortage. Currently, it is still possible to obtain IP addresses in most parts of the world. However, a requester typically has to justify that they actually need the number of addresses that they request, and must demonstrate use of the addresses that they are allocated. Furthermore, addresses have become a commodity of sorts, as ISPs typically charge significantly more for additional addresses beyond the default number that they give to a client.

In the near future, we expect that three trends will cause the rapid exhaustion of the remaining IPv4 address space.

1. *The Internet revolution reaches Asia and developing nations*: Over one third of the world's population currently resides in China and India, two countries that are relatively poorly connected. Given the growing availability of inexpensive access technologies, as well as wireless networking, we can expect a sharp spike in address demand once these countries come online en masse.

2. *Home networking takes off*: All indicators are suggesting that residential networks will be commonplace within the next 2-5 years. The number of multiple PC households is growing, and it is expected that network appliances, from thermostats to physical security systems to microwave ovens, will follow. Potentially, the well-wired household may require dozens of addresses, and a multiple dwelling unit may require hundreds.

3. *Proliferation of cellular IP phones and PDAs*: Several major cellular telephony manufacturers have recently announced that they intend to deploy IP-aware "smart phones," that will replace both current cellular phones and personal digital assistants (PDAs). If this occurs, there will be a sudden worldwide need for millions, if not billions, of addresses.

These trends indicate that within 12-36 months we will begin to feel the IP address crunch in earnest, and that not long thereafter, it may become severe.

2.2 Network Address Translation

The address shortage began to be felt in the early-to-mid 1990's. NAT was developed as an intermediate, temporary solution, that would hold us over until IPv6 deployed. Obviously, IPv6 has not deployed, and even IPv6 advocates admit that unless there is a clearly defined transition phase, it may never deploy. As a result, NATs are becoming widespread, especially in the small enterprise and home networking space.

In principle, a NAT is a boundary router between two different address spaces. Typically, one is a private space of an ISP, residential network, or corporate intranet, and the other is the public space of the Internet. The hosts in the private space use private IP addresses [5] that are unroutable from the public Internet. The NAT performs a one-to-one translation of outgoing (private-to-public) packets from each private address to a unique public IP address, and performs the converse operation for incoming (public-to-private) packets. A popular variation of NAT, Network Address and Port Translation (NAPT) maps all private addresses to one or more public addresses, differentiating amongst these hosts by local port number. Thus, a NAPT device must ensure that all port numbers used on the local side of a session are unique per public address. This requires that some port numbers chosen by a host will be translated along with their address.

The major drawback that NAT introduces is that if an application transmits IP addresses or ports as part of its packets' payloads, the NAT must contain an application layer gateway (ALG) in order for it to support the protocol. The classic example of this is FTP. In the FTP control stream, the client transmits the IP address and port number to which the server should open a socket. In order for FTP to work across a NAT, the NAT must examine the payload of FTP control packets, determine where the address and port information is encoded, and perform the necessary translation. In the worst case, this requires that the NAT also modify the packet length and sequence number fields in the IP and TCP headers, respectively, TCP header checksum, and maintain a running delta of the TCP sequence number for lifetime of the connection.

The need for a protocol-specific ALG in the NAT for each protocol that transmits address or port content is more of a deployment issue than an engineering is-

sue. As long as a protocol payload can be decoded, read, and modified without disrupting end-to-end communications, NAT manufacturers can develop an appropriate ALG. However, deploying this ALG to an installed base of customers can prove to be trying. Since new protocols are being developed at a record pace, a NAT user must perform software or firmware upgrades on a regular and frequent basis. Despite these limitations, the utility of NATs has so far outweighed these drawbacks. Perhaps the true showstoppers for NAT, however, are applications that cryptographically prevent NATs from modifying IP packets. In general, this rules out end-to-end application of IPsec. In particular, NAT breaks all applications of the authentication header (AH), and applications of the Encapsulating Security Payload (ESP) in so-called transport mode. ESP in tunnel mode, however, is impervious to modifications of the outermost IP header. In spite of its deleterious effect on IPsec, NATs do not completely hinder the use of end-to-end data security. Security mechanisms at or above the transport layer, such as TLS or SSH, are unaffected if the applications being run do not transmit addresses or ports in their payloads.

Nevertheless, given the increasing demand for end-to-end network layer security, typically in the form of the virtual private networks that IPsec enables, NAT is seen as a critical roadblock for these services.

3 RSIP

RSIP is an alternative to NAT that operates under the same assumptions of physical architecture and connectivity. While NAT is only defined in terms of operations on a flow of packets, RSIP is defined in terms of operations on the flow, and also a signaling association between the client host (RSIP client) and gateway router (RSIP server). The nature of the operations on the flow of packets is also quite different. NAT gateways must match incoming flows with arbitrary filters. The filters used by RSIP servers use only a very small fixed number of prefixes, which lends itself much more to efficient implementation in hardware.

3.1 Locating RSIP Gateways

Before a client can invoke the services of an RSIP server, it must first locate the RSIP server. That is, it must obtain the RSIP server's IP address. Obviously, this information can be manually configured in each client, but it is highly desirable to automate the discovery process, especially in situations in which a roaming client is visiting a "foreign" network.

Routers between a client and the RSIP server will not intercept and relay RSIP messages like is common for DHCP. Since on most networks, clients will obtain their local addresses with DHCP, the IETF currently is defining a DHCP option that informs clients of the address of the RSIP server [6]. An alternative that is also under consideration is to use the Service Location Protocol (SLP) [7]. This protocol provides a framework for highly dynamic query-based discovery of services. SLP clients (user agents) can discover only those services (service agents) that satisfy certain criteria expressed by the client. Services are defined by the SLP templates. There is now such a template for SLP-based discovery of RSIP servers [8].

3.2 Packet Flow

The key to RSIP is for the RSIP client to prepare packets that are ready for the public network, such that no translation is necessary by the RSIP server. In order to do so, the RSIP client queries the RSIP server for the appropriate public address and port number(s) to utilize (see Section 3.3 for details), and then prepares a packet using these parameters. The RSIP client tunnels this packet over the private network to the RSIP server, which then only has to strip off the tunnel header, and forward the packet on to the public network.

An example RSIP packet flow is shown in Figure 1. An RSIP client (address 10.0.0.4) is connected to a multi-homed RSIP server by a private network. The RSIP server maintains one interface on the private network (address 10.0.0.1) and one interface on the public network (address 149.112.240.55). The figure illustrates a typical HTTP request-reply transaction between the RSIP client and a public WWW server (address 192.156.136.22). It is assumed that the RSIP server has allocated its public IP address

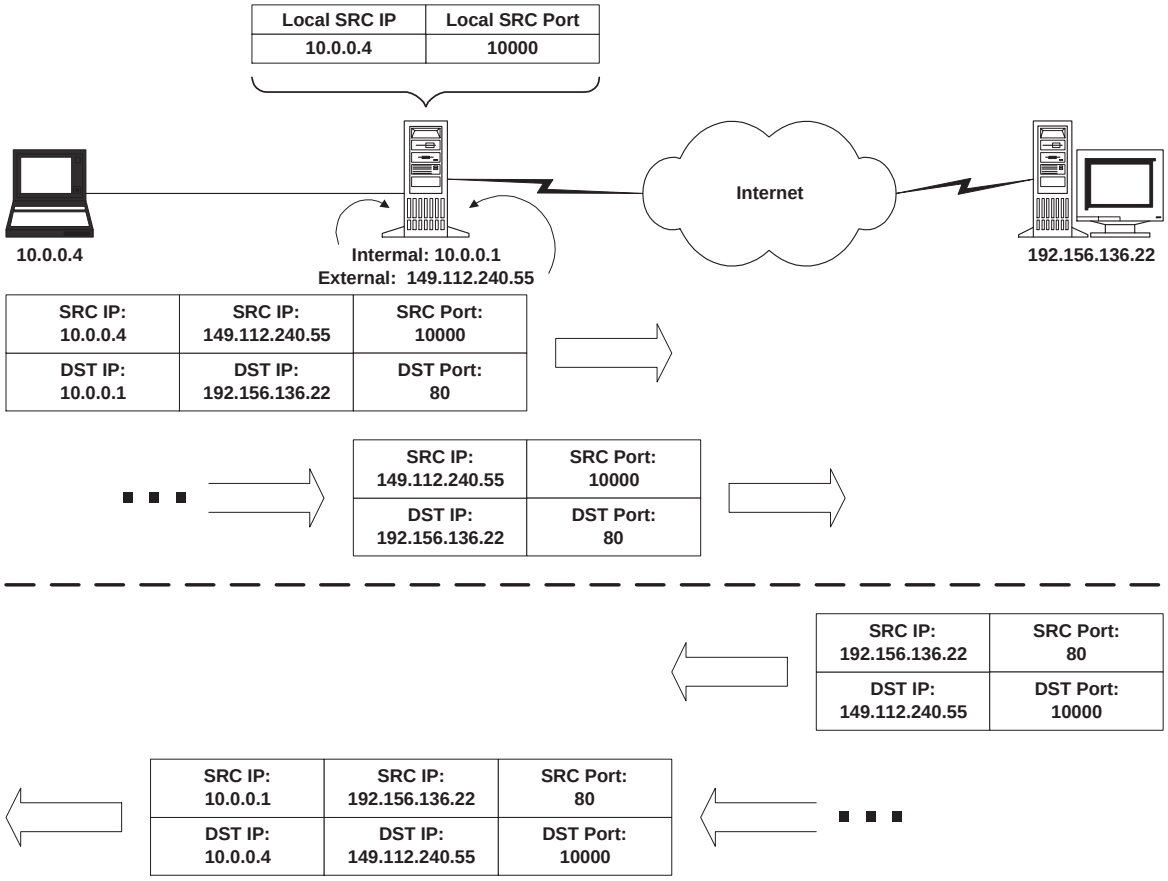

Figure 1: RSIP media flow.

and port 10000 to the RSIP client. When creating the HTTP request packet, the RSIP server uses port 10000 as the source port and 149.112.240.55 as the source IP address. This packet is then encapsulated in a tunnel that delivers it across the local network, to the RSIP server. The RSIP server removes the tunnel header and forwards the packet on the public network. For the incoming response from the WWW server, the RSIP server performs a lookup on the destination port number. Finding port 10000 associated with address 10.0.0.4, the RSIP server forms the tunnel header and transmits the encapsulated packet to the RSIP client.

Since the RSIP client prepares the packet to appear as if it originated from the RSIP server, there is no need for an ALG. Furthermore, even if the RSIP client is using end-to-end network layer encryption with a public server, the transaction will operate properly through the RSIP server, because the RSIP server does not need to examine the payload contents. In general, the RSIP server will allocate more than one port number per RSIP client; thus, the client can utilize protocols, such as FTP, that require multiple simultaneous sessions.

When an RSIP client communicates on the private network, it uses its local (private) address, and is not restricted by RSIP in any way. Thus, an RSIP client must respond to ARPs for its private address, but it must not respond to ARPs for a public address that it is using.

3.3 Signaling

Before an RSIP client can contact a public host, it must establish a signaling association with the RSIP server. The association can be either a TCP connection or a UDP session. It allows the RSIP server to lease address and port bindings to the client, de-allocate these bindings, and otherwise manage resources. The RSIP protocol runs in a simple request-response format. There are three major states that a client may be in: unregistered, reg-

istered, and assigned. All other states indicate that the client has transmitted a request to the server and is waiting for a response. This protocol is described in more detail in [9].

All clients begin in the unregistered state. It is assumed that they have acquired a private IP address, either via DHCP, static assignment, or some other mechanism. Before attempting communication with the public network, they must register with the RSIP server. This registration phase is expected to be performed as part of system initialization. Alternatively, it could be postponed until the client determines that it will, in fact, require RSIP services, and only then will the client request resources from the RSIP server. RSIP clients notify the RSIP server of their presence with a REGISTER_REQUEST message. The server replies with a REGISTER_RESPONSE message that includes a unique CLIENTID token as well as other policy. The client must include this token in all subsequent messages. Upon successful registration, the client enters the registered state. If the registration is not successful, the server will inform the client why with an ERROR_RESPONSE message. All RSIP request messages may be responded to with an ERROR_RESPONSE message, if the request is not granted.

Once registered, a client may request a public IP address and one or more ports with an ASSIGN_REQUEST message. Once an associated ASSIGN_RESPONSE is received from the server, the client enters the assigned state. Each ASSIGN_RESPONSE includes a per-client unique BINDID that identifies the bound resources. In the assigned state, the client may communicate with public hosts, request more resources with another ASSIGN_REQUEST message, free some assigned resources with a FREE_REQUEST message, or de-register with a DE-REGISTER_REQUEST message. From the assigned state, a de-registration frees all resources bound to the client.

Although RSIP is initially targeted at home networks and small to medium enterprises, it may also be deployed in large enterprise networks. These networks may include hundreds of subnets behind an RSIP server, and may require the RSIP client to know whether a given host is on the local or remote side of its RSIP server. To facilitate this situation, a client may transmit a QUERY_REQUEST to the server with the address of a host. The QUERY_RESPONSE from the server will indicate whether the host in question is local or not. In general, it is expected that the server will know of all subnets on the local side because it will be performing firewalling or packet filtering duties as a gateway to the public network.

If an RSIP client is required to act as a server for some application layer protocol, it must inform the RSIP server to pass to it all incoming packets to a particular IP address / port tuple. The client achieves this by transmitting a LISTEN_REQUEST to the server, and the server responding with a LISTEN_RESPONSE.

All resources acquired by an RSIP client are leased for a finite amount of time. Once the lease on a binding has expired, the RSIP server will transmit a FREE_RESPONSE message that informs the client that it may no longer use the resources associated with the binding specified. The server may transmit a FREE_RESPONSE message at any time. Likewise, an RSIP server may transmit a DE-REGISTER_RESPONSE at any time, terminating an RSIP client's registration and all of its bindings.

3.4 IPsec Support

IPsec enables secure end-to-end communication. Packets can either be encrypted, authenticated, or both. In order to use IPsec, two hosts must first establish a security association (SA), perhaps using the Internet Key Exchange (IKE) [10] protocol. All packets protected by an SA have at least one extra header inserted between the IP header and the transport layer header. Figure 2 shows IPsec packets in transport mode. The other of two possible modes is tunnel mode, in which the AH or ESP headers are followed by another IP header. The encapsulating security payload (ESP) header, shown in Figure 2a, encrypts the entire packet payload, and optionally authenticates the entire packet payload except for part of the ESP trailer. The authentication header (AH), shown in Figure 2b, authenticates the entire packet including the immediately preceding IP header (except, of course, for the IP header fields that change per hop). Both ESP and AH may be applied at a packet, as is shown in Figure 2c.

Even though ESP does not include the preceding IP header in its cryptographic calculation, it does include the entire payload, which in transport

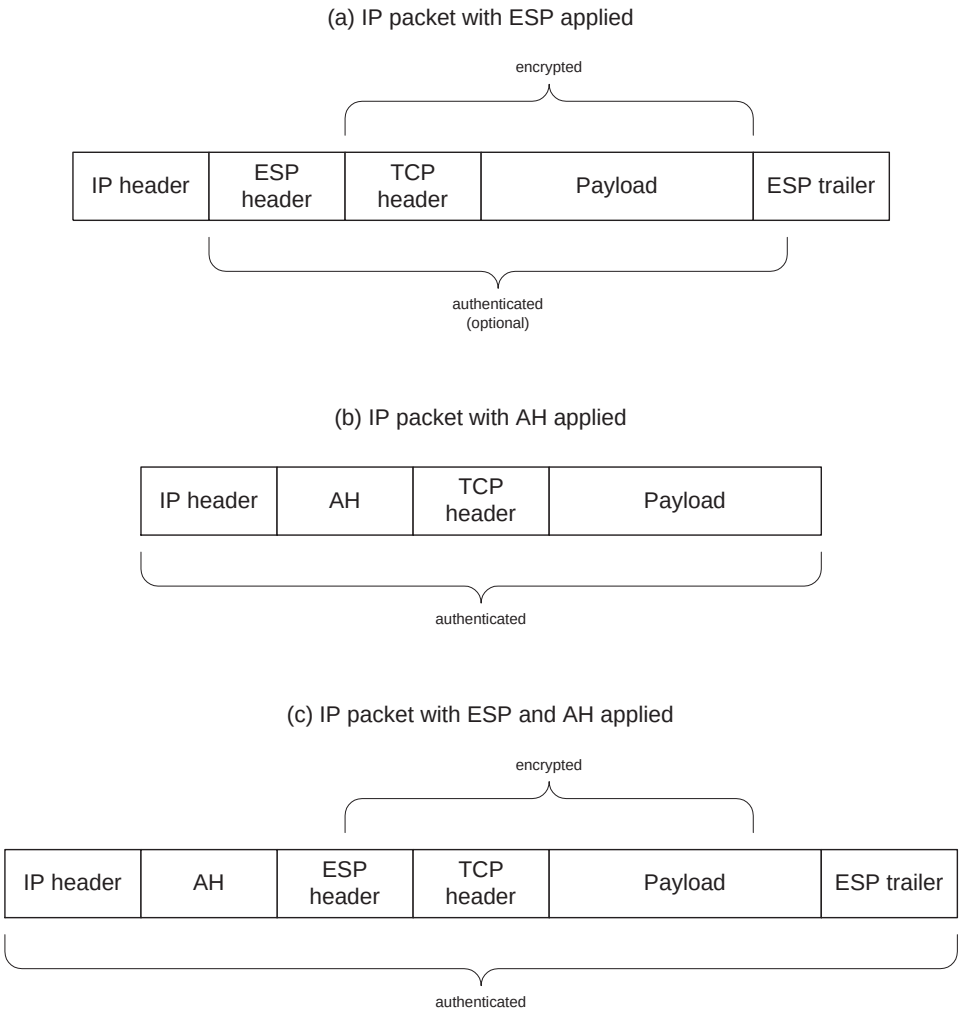

Figure 2: IPsec protection with ESP and AH.

mode includes the transport header. Commonly, the transport protocol is either TCP or UDP, in which case the transport headers include a pseudo header containing, among other things, the source and destination address fields in the preceding IP header. Thus, indirectly, ESP in transport mode renders the IP source and destination addresses immutable, and is broken by NAT gateways.

Given that the keying material used for the encryption and authentication can only be shared between the RSIP client and its public peer, the RSIP-enabled gateway will not be able to read TCP/UDP ports when ESP is used, and will not be able to modify the ports when ESP or AH is used. Thus, IPsec generally will not work through a NAT. However, ESP in tunnel mode is impervious to NATs, in that the outer IP header is not cryptographically protected. Hence, it is possible to modify the outside header without rendering the packet useless. In spite of this, NAT gateways have no surefire way of establishing the appropriate mapping to demultiplex these packets. Recent Linux implementations use temporal association to guess what the right mappings are, based on the assumption that outgoing packets are immediately followed by incoming traffic. And from this, it is possible to guess which client will be expecting the subsequent incoming IPsec packets.

In order to support IPsec through an RSIP-enabled gateway, we need to solve two problems: (1) finding one or more fields in the packet headers to use to demultiplex incoming packets to RSIP clients, and (2) a way of ensuring that these fields are unique per public IP address used for RSIP. The key to address-

ing these issues is that all ESP and AH headers contain a 32-bit value called the security parameter index (SPI), that is unique per SA and is always kept in the clear. Furthermore, the SPI value in packets received by a host is specified by the host during IKE negotiation. Therefore, an RSIP server can allocate mutually exclusive SPI values along with each IP address and port assignment. The RSIP client will tell its peer to use an allocated SPI value. The RSIP server, knowing the SPI values allocated to each client, will be able to route incoming packets to the proper client, by examining the destination address and SPI. Note that even though port numbers do not need to be unique for routing purposes, they do need to be unique in order to avoid socket collisions when two RSIP clients using the same IP address communicate with the same server, on the same destination port.

One more detail needs to be addressed before the RSIP/IPsec integration is complete. IKE implementations currently use port 500 as source and destination for all communication. If concurrent IKE negotiations are taking place between two RSIP clients and the same public host, the RSIP router will not have enough information to route the public host's packets to the proper client just by examining the headers. However, all IKE packets contain an initiator cookie in the first eight bytes of payload. The value of this token is chosen by the initiator of the IKE session; i.e., the RSIP client. Thus, initiator cookies, like ports and SPIs, can be distributed in a mutually exclusive fashion by the RSIP server. A more elegant alternative is to let IKE clients use an ephemeral source port number. An RSIP implementation would then be able to choose a locally-unique port that could be used to demultiplex incoming IKE replies.

The discussions of RSIP/IPsec interactions in this section are necessarily brief due to space constraints. A more complete presentation of these issues, including the RSIP signaling messages for IPsec, is found in [11].

4 Implications and Future Work

RSIP presents a potentially revolutionary concept that can be deployed in an evolutionary fashion. RSIP overcomes the difficulties of NAT in a way that can co-exist with NAT. As NAT gateways are upgraded to support RSIP, legacy NAT clients can continue to use the NAT while RSIP clients take advantage of the benefits offered by RSIP. RSIP gives the network community a more scalable, usable, and secure alternative to NAT as a holdover until IPv6 is deployed.

A number of RSIP issues are currently being resolved in the IETF. While [4] contains a more complete list, we will address the most relevant.

4.1 Incoming IPsec Sessions

Incoming IKE connections are much easier to support if the peer can initiate IKE exchanges to a port other than 500. In this case, the RSIP client would allocate that port at the RSIP server via ASSIGN_REQUEST. Alternatively, if the RSIP client is able to allocate an IP address at the RSIP server, the peer could simply initiate the IKE exchange to port 500 at that address.

If there is only one address that must be shared by the RSIP server and all its clients, and if the peer can only send to port 500, the problem is much more difficult. At any given time, the combination of address and UDP port 500 may be registered and used by only one RSIP system (including clients and server).

Solving this issue requires demultiplexing the incoming IKE connection request based on something other than the port and address combination. It may be possible to do so by first registering an identity with a new RSIP command of LISTEN_RSIP_IKE. Note that the identity could not be that of the IKE responder (the RSIP client), but that of the initiator (the peer). The reason is that IKE Phase 1 only allows the sender to include its own identity, not that of the intended recipient (both, by the way, are allowed in Phase 2). Furthermore, the identity must be in the clear in the first incoming packet for the RSIP server to be able to use it as a demultiplexor. This rules out all variants of Main Mode and Aggressive Mode with Public Key Encryption (and Revised Mode of Public Key Encryption), since these encrypt the ID payload.

The only Phase 1 variants which enable incoming IKE sessions are Aggressive Mode with signatures or with pre-shared keys. Because this scheme involves the RSIP server demultiplexing based on the

identity of the IKE initiator, it is conceivable that only one RSIP client at a time may register interest in fielding requests from any given peer. Furthermore, this precludes more than one RSIP client's being available to any unspecified peer.

Once the IKE session is in place, IPsec is set up as discussed in this document, namely, by the RSIP client and the RSIP server agreeing on an incoming SPI value, which is then communicated to the peer as part of Quick Mode.

The alternate address and port combination must be discovered by the remote peer using methods such as manual configuration, or the use of KX [12] or SRV [13] records. It may even be possible for the DNS query to trigger the above mechanisms to prepare for the incoming and impending IKE session initiation. Such a mechanism would allow more than one RSIP client to be available at any given time, and would also enable each of them to respond to IKE initiations from unspecified peers. Such a DNS query, however, is not guaranteed to occur. For example, the result of the query could be cached and reused after the RSIP server is no longer listening for a given IKE peer's identity.

Due to the limitations implied by having to rely on the identity of the IKE initiator, the only practical way of supporting incoming connections is for the peer to initiate the IKE session to a port other than 500.

4.2 General Disparate Address Space Support for RSIP

An RSIP server is located at the border between two disparate address spaces. In most deployments scenarios, this is the border between the global Internet and a private network. In other scenarios (for example in business to business communications), the address spaces at either side of the RSIP server may have conflicting address ranges. This may happen if both address spaces use net 10.0.0.0 within their network. In this case, the QUERY_REQUEST cannot be resolved by the RSIP server by just examining an IP address. A variant of QUERY_REQUEST that uses DNS names instead of IP addresses may solve this issue.

A general solution probably implies further refinements to the protocol.

4.3 RSIP as a IPv6 Transition Mechanism

In [1], the future of IPv6 deployment is addressed. Three scenarios are explored, one in which IPv6 never deploys, another in which it partially deploys, and a third in which it fully deploys. It is very unlikely that IPv6 will fully deploy without some intermediate form of partial deployment. Thus, a key ingredient for the transition to IPv6 is the existence of transition technologies [14] that allow IPv6 to be deployed on an incremental basis, while coexisting with the legacy IPv4 infrastructure.

In situations in which IPv6 is deployed on some edge networks while backbones and other edge networks remain IPv4-only, RSIP can play a valuable role. The hosts on the IPv6 edge networks may be dual stack (i.e., they simultaneously support both IPv4 and IPv6). By placing RSIP clients in these hosts and an RSIP server on the gateway router between the IPv4 and IPv6 spaces, RSIP can allocate IPv4 addresses to the IPv4 stacks of these hosts when necessary. For example, the following communications can be supported:

- *IPv4/IPv6 dual stack host communicating with another local IPv4/IPv6 dual stack host*: Use IPv6.
- *IPv4/IPv6 dual stack host communicating with another remote IPv4/IPv6 dual stack host*: Use IPv6 locally. When the local edge router receives the packets, it performs IPv6-over-IPv4 tunneling to the remote IPv6 edge network. The remote edge router terminates the tunnel and forwards the IPv6 packet to the destination host.
- *IPv4/IPv6 dual stack host communicating with a remote IPv4 host*: Use RSIP to acquire a public IPv4 address, and use that address to contact the remote IPv4 host. The IPv4 packets are transmitted on the IPv6 edge network using an IPv4-over-IPv6 tunnel, which is terminated at the local edge router.

This system is very similar in spirit to the Dual Stack Transition Mechanism [15] proposal, which uses DHCP for IPv4 address allocation. However, DSTM does not allow multiple clients to share the same IP address from the gateway machine. In

other words, DSTM does not have an equivalent to the RSAP-IP [9] or RSIPSEC [11] methods in RSIP.

5 Conclusion

History has shown us that changing the core of the network to support a new protocol is very difficult. The lack of widespread RSVP, IP multicast, and IPv6 deployment attests to this premise. However, changes to the network edge occur gradually over time. Privacy concerns have led to the deployment of firewalls and host security in almost every edge network. Configuration has been eased by deployment of DHCP. Roaming is supported by Mobile IP. RSIP is another way to upgrade the edge of the network to overcome the limitations of legacy network design.

References

[1] B. Carpenter, "Internet transparency." Internet RFC 2775, Feb. 2000.

[2] P. Srisuresh and M. Holdrege, "IP network address translator (NAT) terminology and considerations." Internet RFC 2663, Aug. 1999.

[3] S. Kent and R. Atkinson, "Security architecture for the Internet Protocol." Internet RFC 2401, Nov. 1998.

[4] M. S. Borella, J. Lo, D. Grabelsky, and G. Montenegro, "Realm Specific IP: Framework." Internet Draft draft-ietf-nat-rsip-framework-03.txt, Dec. 1999. Work in progress.

[5] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear, "Address allocation for private internets." Internet RFC 1918, Feb. 1996.

[6] J. Privat and M. S. Borella, "DHCP next server option." Internet Draft draft-ietf-dhc-nextserver-01.txt, Feb. 2000. Work in progress.

[7] E. Guttman, C. Perkins, J. Veizades, and M. Day, "Service location protocol, version 2." Internet RFC 2608, June 1999.

[8] J. Kempf and G. Montenegro, "Finding an RSIP server with SLP." Internet draft draft-ietf-nat-rsip-slp-00.txt, Feb. 2000. Work in progress.

[9] M. S. Borella, D. Grabelsky, J. Lo, and K. Taniguchi, "Realm Specific IP: Protocol." Internet Draft draft-ietf-nat-rsip-protocol-05.txt, Jan. 2000. Work in progress.

[10] D. Harkins and D. Carrel, "The Internet key exchange (IKE)." Internet RFC 2409, Nov. 1998.

[11] G. Montenegro and M. S. Borella, "RSIP support for end-to-end IPsec." Internet Draft draft-ietf-nat-rsip-ipec-02.txt, Feb. 2000. Work in progress.

[12] R. Atkinson, "Key exchange delegation record for the DNS." Internet RFC 2230, Nov. 1997.

[13] A. Gulbrandsen and P. Vixie, "A DNS RR for specifying the location of services (DNS SRV)." Internet RFC 2052, Oct. 1996.

[14] R. Gilligan and E. Nordmark, "Transition mechanisms for IPv6 hosts and routers." Internet RFC 1933, Apr. 1996.

[15] J. Bound, L. Toutain and H. Afifi, "Dual stack transition mechanism (DSTM)." Internet Draft draft-ietf-ngtrans-dstm-01.txt, March 2000. Work in progress.