Network Working Group
Request for Comments: 2401
Obsoletes: 1825
Category: Standards Track

S. Kent
BBN Corp
R. Atkinson
@Home Network
November 1998

Security Architecture for the Internet Protocol

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

1.1 Summary of Contents of Document

This memo specifies the base architecture for IPsec compliant systems. The goal of the architecture is to provide various security services for traffic at the IP layer, in both the IPv4 and IPv6 environments. This document describes the goals of such systems, their components and how they fit together with each other and into the IP environment. It also describes the security services offered by the IPsec protocols, and how these services can be employed in the IP environment. This document does not address all aspects of IPsec architecture. Subsequent documents will address additional architectural details of a more advanced nature, e.g., use of IPsec in NAT environments and more complete support for IP multicast. The following fundamental components of the IPsec security architecture are discussed in terms of their underlying, required functionality. Additional RFCs (see Section 1.3 for pointers to other documents) define the protocols in (a), (c), and (d).

a. Security Protocols -- Authentication Header (AH) and Encapsulating Security Payload (ESP)
b. Security Associations -- what they are and how they work, how they are managed, associated processing
c. Key Management -- manual and automatic (The Internet Key Exchange (IKE))
d. Algorithms for authentication and encryption

This document is not an overall Security Architecture for the Internet; it addresses security only at the IP layer, provided through the use of a combination of cryptographic and protocol security mechanisms.

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in RFC 2119 [Bra97].

1.2 Audience

The target audience for this document includes implementers of this IP security technology and others interested in gaining a general background understanding of this system. In particular, prospective users of this technology (end users or system administrators) are part of the target audience. A glossary is provided as an appendix

to help fill in gaps in background/vocabulary. This document assumes that the reader is familiar with the Internet Protocol, related networking technology, and general security terms and concepts.

1.3 Related Documents

As mentioned above, other documents provide detailed definitions of some of the components of IPsec and of their inter-relationship. They include RFCs on the following topics:

a. "IP Security Document Roadmap" [TDG97] -- a document providing guidelines for specifications describing encryption and authentication algorithms used in this system.
b. security protocols -- RFCs describing the Authentication Header (AH) [KA98a] and Encapsulating Security Payload (ESP) [KA98b] protocols.
c. algorithms for authentication and encryption -- a separate RFC for each algorithm.
d. automatic key management -- RFCs on "The Internet Key Exchange (IKE)" [HC98], "Internet Security Association and Key Management Protocol (ISAKMP)" [MSST97],"The OAKLEY Key Determination Protocol" [Orm97], and "The Internet IP Security Domain of Interpretation for ISAKMP" [Pip98].

2. Design Objectives

2.1 Goals/Objectives/Requirements/Problem Description

IPsec is designed to provide interoperable, high quality, cryptographically-based security for IPv4 and IPv6. The set of security services offered includes access control, connectionless integrity, data origin authentication, protection against replays (a form of partial sequence integrity), confidentiality (encryption), and limited traffic flow confidentiality. These services are provided at the IP layer, offering protection for IP and/or upper layer protocols.

These objectives are met through the use of two traffic security protocols, the Authentication Header (AH) and the Encapsulating Security Payload (ESP), and through the use of cryptographic key management procedures and protocols. The set of IPsec protocols employed in any context, and the ways in which they are employed, will be determined by the security and system requirements of users, applications, and/or sites/organizations.

When these mechanisms are correctly implemented and deployed, they ought not to adversely affect users, hosts, and other Internet components that do not employ these security mechanisms for

protection of their traffic. These mechanisms also are designed to be algorithm-independent. This modularity permits selection of different sets of algorithms without affecting the other parts of the implementation. For example, different user communities may select different sets of algorithms (creating cliques) if required.

A standard set of default algorithms is specified to facilitate interoperability in the global Internet. The use of these algorithms, in conjunction with IPsec traffic protection and key management protocols, is intended to permit system and application developers to deploy high quality, Internet layer, cryptographic security technology.

2.2 Caveats and Assumptions

The suite of IPsec protocols and associated default algorithms are designed to provide high quality security for Internet traffic. However, the security offered by use of these protocols ultimately depends on the quality of the their implementation, which is outside the scope of this set of standards. Moreover, the security of a computer system or network is a function of many factors, including personnel, physical, procedural, compromising emanations, and computer security practices. Thus IPsec is only one part of an overall system security architecture.

Finally, the security afforded by the use of IPsec is critically dependent on many aspects of the operating environment in which the IPsec implementation executes. For example, defects in OS security, poor quality of random number sources, sloppy system management protocols and practices, etc. can all degrade the security provided by IPsec. As above, none of these environmental attributes are within the scope of this or other IPsec standards.

3. System Overview

This section provides a high level description of how IPsec works, the components of the system, and how they fit together to provide the security services noted above. The goal of this description is to enable the reader to "picture" the overall process/system, see how it fits into the IP environment, and to provide context for later sections of this document, which describe each of the components in more detail.

An IPsec implementation operates in a host or a security gateway environment, affording protection to IP traffic. The protection offered is based on requirements defined by a Security Policy Database (SPD) established and maintained by a user or system administrator, or by an application operating within constraints

established by either of the above. In general, packets are selected for one of three processing modes based on IP and transport layer header information (Selectors, Section 4.4.2) matched against entries in the database (SPD). Each packet is either afforded IPsec security services, discarded, or allowed to bypass IPsec, based on the applicable database policies identified by the Selectors.

3.1 What IPsec Does

IPsec provides security services at the IP layer by enabling a system to select required security protocols, determine the algorithm(s) to use for the service(s), and put in place any cryptographic keys required to provide the requested services. IPsec can be used to protect one or more "paths" between a pair of hosts, between a pair of security gateways, or between a security gateway and a host. (The term "security gateway" is used throughout the IPsec documents to refer to an intermediate system that implements IPsec protocols. For example, a router or a firewall implementing IPsec is a security gateway.)

The set of security services that IPsec can provide includes access control, connectionless integrity, data origin authentication, rejection of replayed packets (a form of partial sequence integrity), confidentiality (encryption), and limited traffic flow confidentiality. Because these services are provided at the IP layer, they can be used by any higher layer protocol, e.g., TCP, UDP, ICMP, BGP, etc.

The IPsec DOI also supports negotiation of IP compression [SMPT98], motivated in part by the observation that when encryption is employed within IPsec, it prevents effective compression by lower protocol layers.

3.2 How IPsec Works

IPsec uses two protocols to provide traffic security -- Authentication Header (AH) and Encapsulating Security Payload (ESP). Both protocols are described in more detail in their respective RFCs [KA98a, KA98b].

- The IP Authentication Header (AH) [KA98a] provides connectionless integrity, data origin authentication, and an optional anti-replay service.
- The Encapsulating Security Payload (ESP) protocol [KA98b] may provide confidentiality (encryption), and limited traffic flow confidentiality. It also may provide connectionless

integrity, data origin authentication, and an anti-replay service. (One or the other set of these security services must be applied whenever ESP is invoked.)
- Both AH and ESP are vehicles for access control, based on the distribution of cryptographic keys and the management of traffic flows relative to these security protocols.

These protocols may be applied alone or in combination with each other to provide a desired set of security services in IPv4 and IPv6. Each protocol supports two modes of use: transport mode and tunnel mode. In transport mode the protocols provide protection primarily for upper layer protocols; in tunnel mode, the protocols are applied to tunneled IP packets. The differences between the two modes are discussed in Section 4.

IPsec allows the user (or system administrator) to control the granularity at which a security service is offered. For example, one can create a single encrypted tunnel to carry all the traffic between two security gateways or a separate encrypted tunnel can be created for each TCP connection between each pair of hosts communicating across these gateways. IPsec management must incorporate facilities for specifying:

- which security services to use and in what combinations
- the granularity at which a given security protection should be applied
- the algorithms used to effect cryptographic-based security

Because these security services use shared secret values (cryptographic keys), IPsec relies on a separate set of mechanisms for putting these keys in place. (The keys are used for authentication/integrity and encryption services.) This document requires support for both manual and automatic distribution of keys. It specifies a specific public-key based approach (IKE -- [MSST97, Orm97, HC98]) for automatic key management, but other automated key distribution techniques MAY be used. For example, KDC-based systems such as Kerberos and other public-key systems such as SKIP could be employed.

3.3 Where IPsec May Be Implemented

There are several ways in which IPsec may be implemented in a host or in conjunction with a router or firewall (to create a security gateway). Several common examples are provided below:

a. Integration of IPsec into the native IP implementation. This requires access to the IP source code and is applicable to both hosts and security gateways.

b. "Bump-in-the-stack" (BITS) implementations, where IPsec is implemented "underneath" an existing implementation of an IP protocol stack, between the native IP and the local network drivers. Source code access for the IP stack is not required in this context, making this implementation approach appropriate for use with legacy systems. This approach, when it is adopted, is usually employed in hosts.

c. The use of an outboard crypto processor is a common design feature of network security systems used by the military, and of some commercial systems as well. It is sometimes referred to as a "Bump-in-the-wire" (BITW) implementation. Such implementations may be designed to serve either a host or a gateway (or both). Usually the BITW device is IP addressable. When supporting a single host, it may be quite analogous to a BITS implementation, but in supporting a router or firewall, it must operate like a security gateway.

4. Security Associations

This section defines Security Association management requirements for all IPv6 implementations and for those IPv4 implementations that implement AH, ESP, or both. The concept of a "Security Association" (SA) is fundamental to IPsec. Both AH and ESP make use of SAs and a major function of IKE is the establishment and maintenance of Security Associations. All implementations of AH or ESP MUST support the concept of a Security Association as described below. The remainder of this section describes various aspects of Security Association management, defining required characteristics for SA policy management, traffic processing, and SA management techniques.

4.1 Definition and Scope

A Security Association (SA) is a simplex "connection" that affords security services to the traffic carried by it. Security services are afforded to an SA by the use of AH, or ESP, but not both. If both AH and ESP protection is applied to a traffic stream, then two (or more) SAs are created to afford protection to the traffic stream. To secure typical, bi-directional communication between two hosts, or between two security gateways, two Security Associations (one in each direction) are required.

A security association is uniquely identified by a triple consisting of a Security Parameter Index (SPI), an IP Destination Address, and a security protocol (AH or ESP) identifier. In principle, the Destination Address may be a unicast address, an IP broadcast address, or a multicast group address. However, IPsec SA management mechanisms currently are defined only for unicast SAs. Hence, in the

discussions that follow, SAs will be described in the context of point-to-point communication, even though the concept is applicable in the point-to-multipoint case as well.

As noted above, two types of SAs are defined: transport mode and tunnel mode. A transport mode SA is a security association between two hosts. In IPv4, a transport mode security protocol header appears immediately after the IP header and any options, and before any higher layer protocols (e.g., TCP or UDP). In IPv6, the security protocol header appears after the base IP header and extensions, but may appear before or after destination options, and before higher layer protocols. In the case of ESP, a transport mode SA provides security services only for these higher layer protocols, not for the IP header or any extension headers preceding the ESP header. In the case of AH, the protection is also extended to selected portions of the IP header, selected portions of extension headers, and selected options (contained in the IPv4 header, IPv6 Hop-by-Hop extension header, or IPv6 Destination extension headers). For more details on the coverage afforded by AH, see the AH specification [KA98a].

A tunnel mode SA is essentially an SA applied to an IP tunnel. Whenever either end of a security association is a security gateway, the SA MUST be tunnel mode. Thus an SA between two security gateways is always a tunnel mode SA, as is an SA between a host and a security gateway. Note that for the case where traffic is destined for a security gateway, e.g., SNMP commands, the security gateway is acting as a host and transport mode is allowed. But in that case, the security gateway is not acting as a gateway, i.e., not transiting traffic. Two hosts MAY establish a tunnel mode SA between themselves. The requirement for any (transit traffic) SA involving a security gateway to be a tunnel SA arises due to the need to avoid potential problems with regard to fragmentation and reassembly of IPsec packets, and in circumstances where multiple paths (e.g., via different security gateways) exist to the same destination behind the security gateways.

For a tunnel mode SA, there is an "outer" IP header that specifies the IPsec processing destination, plus an "inner" IP header that specifies the (apparently) ultimate destination for the packet. The security protocol header appears after the outer IP header, and before the inner IP header. If AH is employed in tunnel mode, portions of the outer IP header are afforded protection (as above), as well as all of the tunneled IP packet (i.e., all of the inner IP header is protected, as well as higher layer protocols). If ESP is employed, the protection is afforded only to the tunneled packet, not to the outer header.

In summary,

a) A host MUST support both transport and tunnel mode.
b) A security gateway is required to support only tunnel mode. If it supports transport mode, that should be used only when the security gateway is acting as a host, e.g., for network management.

4.2 Security Association Functionality

The set of security services offered by an SA depends on the security protocol selected, the SA mode, the endpoints of the SA, and on the election of optional services within the protocol. For example, AH provides data origin authentication and connectionless integrity for IP datagrams (hereafter referred to as just "authentication"). The "precision" of the authentication service is a function of the granularity of the security association with which AH is employed, as discussed in Section 4.4.2, "Selectors".

AH also offers an anti-replay (partial sequence integrity) service at the discretion of the receiver, to help counter denial of service attacks. AH is an appropriate protocol to employ when confidentiality is not required (or is not permitted, e.g , due to government restrictions on use of encryption). AH also provides authentication for selected portions of the IP header, which may be necessary in some contexts. For example, if the integrity of an IPv4 option or IPv6 extension header must be protected en route between sender and receiver, AH can provide this service (except for the non-predictable but mutable parts of the IP header.)

ESP optionally provides confidentiality for traffic. (The strength of the confidentiality service depends in part, on the encryption algorithm employed.) ESP also may optionally provide authentication (as defined above). If authentication is negotiated for an ESP SA, the receiver also may elect to enforce an anti-replay service with the same features as the AH anti-replay service. The scope of the authentication offered by ESP is narrower than for AH, i.e., the IP header(s) "outside" the ESP header is(are) not protected. If only the upper layer protocols need to be authenticated, then ESP authentication is an appropriate choice and is more space efficient than use of AH encapsulating ESP. Note that although both confidentiality and authentication are optional, they cannot both be omitted. At least one of them MUST be selected.

If confidentiality service is selected, then an ESP (tunnel mode) SA between two security gateways can offer partial traffic flow confidentiality. The use of tunnel mode allows the inner IP headers to be encrypted, concealing the identities of the (ultimate) traffic source and destination. Moreover, ESP payload padding also can be

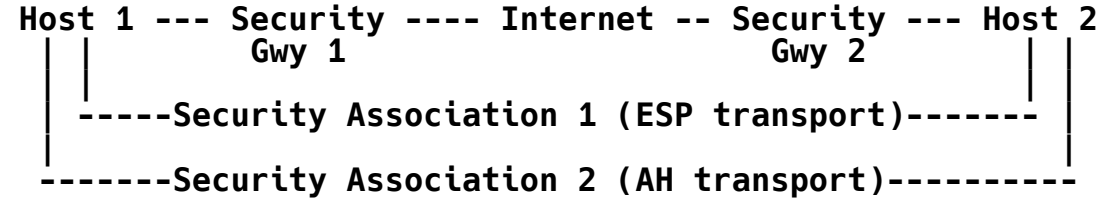
invoked to hide the size of the packets, further concealing the external characteristics of the traffic. Similar traffic flow confidentiality services may be offered when a mobile user is assigned a dynamic IP address in a dialup context, and establishes a (tunnel mode) ESP SA to a corporate firewall (acting as a security gateway). Note that fine granularity SAs generally are more vulnerable to traffic analysis than coarse granularity ones which are carrying traffic from many subscribers.

4.3 Combining Security Associations

The IP datagrams transmitted over an individual SA are afforded protection by exactly one security protocol, either AH or ESP, but not both. Sometimes a security policy may call for a combination of services for a particular traffic flow that is not achievable with a single SA. In such instances it will be necessary to employ multiple SAs to implement the required security policy. The term "security association bundle" or "SA bundle" is applied to a sequence of SAs through which traffic must be processed to satisfy a security policy. The order of the sequence is defined by the policy. (Note that the SAs that comprise a bundle may terminate at different endpoints. For example, one SA may extend between a mobile host and a security gateway and a second, nested SA may extend to a host behind the gateway.)

Security associations may be combined into bundles in two ways: transport adjacency and iterated tunneling.

- Transport adjacency refers to applying more than one security protocol to the same IP datagram, without invoking tunneling. This approach to combining AH and ESP allows for only one level of combination; further nesting yields no added benefit (assuming use of adequately strong algorithms in each protocol) since the processing is performed at one IPsec instance at the (ultimate) destination.

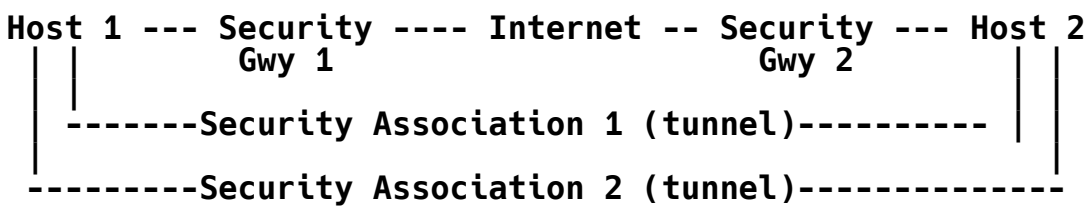


- Iterated tunneling refers to the application of multiple layers of security protocols effected through IP tunneling. This approach allows for multiple levels of nesting, since each tunnel can originate or terminate at a different IPsec

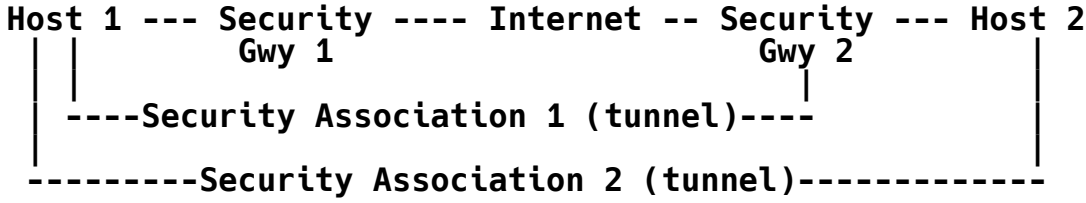
site along the path. No special treatment is expected for ISAKMP traffic at intermediate security gateways other than what can be specified through appropriate SPD entries (See Case 3 in Section 4.5)

There are 3 basic cases of iterated tunneling -- support is required only for cases 2 and 3.:

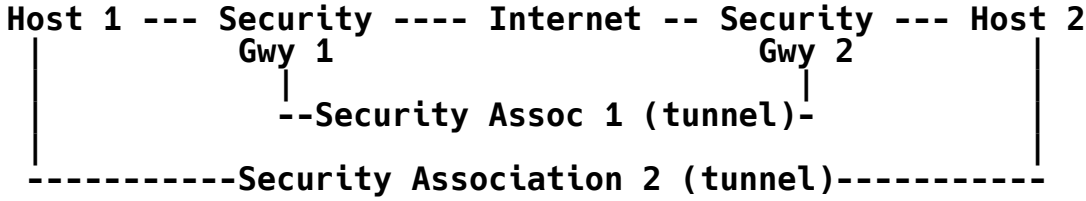
1. both endpoints for the SAs are the same -- The inner and outer tunnels could each be either AH or ESP, though it is unlikely that Host 1 would specify both to be the same, i.e., AH inside of AH or ESP inside of ESP.



2. one endpoint of the SAs is the same -- The inner and uter tunnels could each be either AH or ESP.



3. neither endpoint is the same -- The inner and outer tunnels could each be either AH or ESP.



These two approaches also can be combined, e.g., an SA bundle could be constructed from one tunnel mode SA and one or two transport mode SAs, applied in sequence. (See Section 4.5 "Basic Combinations of Security Associations.") Note that nested tunnels can also occur where neither the source nor the destination endpoints of any of the tunnels are the same. In that case, there would be no host or security gateway with a bundle corresponding to the nested tunnels.

For transport mode SAs, only one ordering of security protocols seems appropriate. AH is applied to both the upper layer protocols and (parts of) the IP header. Thus if AH is used in a transport mode, in conjunction with ESP, AH SHOULD appear as the first header after IP, prior to the appearance of ESP. In that context, AH is applied to the ciphertext output of ESP. In contrast, for tunnel mode SAs, one can imagine uses for various orderings of AH and ESP. The required set of SA bundle types that MUST be supported by a compliant IPsec implementation is described in Section 4.5.

4.4 Security Association Databases

Many of the details associated with processing IP traffic in an IPsec implementation are largely a local matter, not subject to standardization. However, some external aspects of the processing must be standardized, to ensure interoperability and to provide a minimum management capability that is essential for productive use of IPsec. This section describes a general model for processing IP traffic relative to security associations, in support of these interoperability and functionality goals. The model described below is nominal; compliant implementations need not match details of this model as presented, but the external behavior of such implementations must be mappable to the externally observable characteristics of this model.

There are two nominal databases in this model: the Security Policy Database and the Security Association Database. The former specifies the policies that determine the disposition of all IP traffic inbound or outbound from a host, security gateway, or BITS or BITW IPsec implementation. The latter database contains parameters that are associated with each (active) security association. This section also defines the concept of a Selector, a set of IP and upper layer protocol field values that is used by the Security Policy Database to map traffic to a policy, i.e., an SA (or SA bundle).

Each interface for which IPsec is enabled requires nominally separate inbound vs. outbound databases (SAD and SPD), because of the directionality of many of the fields that are used as selectors. Typically there is just one such interface, for a host or security gateway (SG). Note that an SG would always have at least 2 interfaces, but the "internal" one to the corporate net, usually would not have IPsec enabled and so only one pair of SADs and one pair of SPDs would be needed. On the other hand, if a host had multiple interfaces or an SG had multiple external interfaces, it might be necessary to have separate SAD and SPD pairs for each interface.

4.4.1 The Security Policy Database (SPD)

Ultimately, a security association is a management construct used to enforce a security policy in the IPsec environment. Thus an essential element of SA processing is an underlying Security Policy Database (SPD) that specifies what services are to be offered to IP datagrams and in what fashion. The form of the database and its interface are outside the scope of this specification. However, this section does specify certain minimum management functionality that must be provided, to allow a user or system administrator to control how IPsec is applied to traffic transmitted or received by a host or transiting a security gateway.

The SPD must be consulted during the processing of all traffic (INBOUND and OUTBOUND), including non-IPsec traffic. In order to support this, the SPD requires distinct entries for inbound and outbound traffic. One can think of this as separate SPDs (inbound vs. outbound). In addition, a nominally separate SPD must be provided for each IPsec-enabled interface.

An SPD must discriminate among traffic that is afforded IPsec protection and traffic that is allowed to bypass IPsec. This applies to the IPsec protection to be applied by a sender and to the IPsec protection that must be present at the receiver. For any outbound or inbound datagram, three processing choices are possible: discard, bypass IPsec, or apply IPsec. The first choice refers to traffic that is not allowed to exit the host, traverse the security gateway, or be delivered to an application at all. The second choice refers to traffic that is allowed to pass without additional IPsec protection. The third choice refers to traffic that is afforded IPsec protection, and for such traffic the SPD must specify the security services to be provided, protocols to be employed, algorithms to be used, etc.

For every IPsec implementation, there MUST be an administrative interface that allows a user or system administrator to manage the SPD. Specifically, every inbound or outbound packet is subject to processing by IPsec and the SPD must specify what action will be taken in each case. Thus the administrative interface must allow the user (or system administrator) to specify the security processing to be applied to any packet entering or exiting the system, on a packet by packet basis. (In a host IPsec implementation making use of a socket interface, the SPD may not need to be consulted on a per packet basis, but the effect is still the same.) The management interface for the SPD MUST allow creation of entries consistent with the selectors defined in Section 4.4.2, and MUST support (total) ordering of these entries. It is expected that through the use of wildcards in various selector fields, and because all packets on a

single UDP or TCP connection will tend to match a single SPD entry, this requirement will not impose an unreasonably detailed level of SPD specification. The selectors are analogous to what are found in a stateless firewall or filtering router and which are currently manageable this way.

In host systems, applications MAY be allowed to select what security processing is to be applied to the traffic they generate and consume. (Means of signalling such requests to the IPsec implementation are outside the scope of this standard.) However, the system administrator MUST be able to specify whether or not a user or application can override (default) system policies. Note that application specified policies may satisfy system requirements, so that the system may not need to do additional IPsec processing beyond that needed to meet an application's requirements. The form of the management interface is not specified by this document and may differ for hosts vs. security gateways, and within hosts the interface may differ for socket-based vs. BITS implementations. However, this document does specify a standard set of SPD elements that all IPsec implementations MUST support.

The SPD contains an ordered list of policy entries. Each policy entry is keyed by one or more selectors that define the set of IP traffic encompassed by this policy entry. (The required selector types are defined in Section 4.4.2.) These define the granularity of policies or SAs. Each entry includes an indication of whether traffic matching this policy will be bypassed, discarded, or subject to IPsec processing. If IPsec processing is to be applied, the entry includes an SA (or SA bundle) specification, listing the IPsec protocols, modes, and algorithms to be employed, including any nesting requirements. For example, an entry may call for all matching traffic to be protected by ESP in transport mode using 3DES-CBC with an explicit IV, nested inside of AH in tunnel mode using HMAC/SHA-1. For each selector, the policy entry specifies how to derive the corresponding values for a new Security Association Database (SAD, see Section 4.4.3) entry from those in the SPD and the packet (Note that at present, ranges are only supported for IP addresses; but wildcarding can be expressed for all selectors):

a. use the value in the packet itself -- This will limit use of the SA to those packets which have this packet's value for the selector even if the selector for the policy entry has a range of allowed values or a wildcard for this selector.
b. use the value associated with the policy entry -- If this were to be just a single value, then there would be no difference between (b) and (a). However, if the allowed values for the selector are a range (for IP addresses) or

wildcard, then in the case of a range,(b) would enable use of the SA by any packet with a selector value within the range not just by packets with the selector value of the packet that triggered the creation of the SA. In the case of a wildcard, (b) would allow use of the SA by packets with any value for this selector.

For example, suppose there is an SPD entry where the allowed value for source address is any of a range of hosts (192.168.2.1 to 192.168.2.10). And suppose that a packet is to be sent that has a source address of 192.168.2.3. The value to be used for the SA could be any of the sample values below depending on what the policy entry for this selector says is the source of the selector value:

source for the value to be used in the SA	example of new SAD selector value
a. packet	192.168.2.3 (one host)
b. SPD entry	192.168.2.1 to 192.168.2.10 (range of hosts)

Note that if the SPD entry had an allowed value of wildcard for the source address, then the SAD selector value could be wildcard (any host). Case (a) can be used to prohibit sharing, even among packets that match the same SPD entry.

As described below in Section 4.4.3, selectors may include "wildcard" entries and hence the selectors for two entries may overlap. (This is analogous to the overlap that arises with ACLs or filter entries in routers or packet filtering firewalls.) Thus, to ensure consistent, predictable processing, SPD entries MUST be ordered and the SPD MUST always be searched in the same order, so that the first matching entry is consistently selected. (This requirement is necessary as the effect of processing traffic against SPD entries must be deterministic, but there is no way to canonicalize SPD entries given the use of wildcards for some selectors.) More detail on matching of packets against SPD entries is provided in Section 5.

Note that if ESP is specified, either (but not both) authentication or encryption can be omitted. So it MUST be possible to configure the SPD value for the authentication or encryption algorithms to be "NULL". However, at least one of these services MUST be selected, i.e., it MUST NOT be possible to configure both of them as "NULL".

The SPD can be used to map traffic to specific SAs or SA bundles. Thus it can function both as the reference database for security policy and as the map to existing SAs (or SA bundles). (To accommodate the bypass and discard policies cited above, the SPD also

MUST provide a means of mapping traffic to these functions, even though they are not, per se, IPsec processing.) The way in which the SPD operates is different for inbound vs. outbound traffic and it also may differ for host vs. security gateway, BITS, and BITW implementations. Sections 5.1 and 5.2 describe the use of the SPD for outbound and inbound processing, respectively.

Because a security policy may require that more than one SA be applied to a specified set of traffic, in a specific order, the policy entry in the SPD must preserve these ordering requirements, when present. Thus, it must be possible for an IPsec implementation to determine that an outbound or inbound packet must be processed thorough a sequence of SAs. Conceptually, for outbound processing, one might imagine links (to the SAD) from an SPD entry for which there are active SAs, and each entry would consist of either a single SA or an ordered list of SAs that comprise an SA bundle. When a packet is matched against an SPD entry and there is an existing SA or SA bundle that can be used to carry the traffic, the processing of the packet is controlled by the SA or SA bundle entry on the list. For an inbound IPsec packet for which multiple IPsec SAs are to be applied, the lookup based on destination address, IPsec protocol, and SPI should identify a single SA.

The SPD is used to control the flow of ALL traffic through an IPsec system, including security and key management traffic (e.g., ISAKMP) from/to entities behind a security gateway. This means that ISAKMP traffic must be explicitly accounted for in the SPD, else it will be discarded. Note that a security gateway could prohibit traversal of encrypted packets in various ways, e.g., having a DISCARD entry in the SPD for ESP packets or providing proxy key exchange. In the latter case, the traffic would be internally routed to the key management module in the security gateway.

4.4.2 Selectors

An SA (or SA bundle) may be fine-grained or coarse-grained, depending on the selectors used to define the set of traffic for the SA. For example, all traffic between two hosts may be carried via a single SA, and afforded a uniform set of security services. Alternatively, traffic between a pair of hosts might be spread over multiple SAs, depending on the applications being used (as defined by the Next Protocol and Port fields), with different security services offered by different SAs. Similarly, all traffic between a pair of security gateways could be carried on a single SA, or one SA could be assigned for each communicating host pair. The following selector parameters MUST be supported for SA management to facilitate control of SA granularity. Note that in the case of receipt of a packet with an ESP header, e.g., at an encapsulating security gateway or BITW

implementation, the transport layer protocol, source/destination ports, and Name (if present) may be "OPAQUE", i.e., inaccessible because of encryption or fragmentation. Note also that both Source and Destination addresses should either be IPv4 or IPv6.

- Destination IP Address (IPv4 or IPv6): this may be a single IP address (unicast, anycast, broadcast (IPv4 only), or multicast group), a range of addresses (high and low values (inclusive), address + mask, or a wildcard address. The last three are used to support more than one destination system sharing the same SA (e.g., behind a security gateway). Note that this selector is conceptually different from the "Destination IP Address" field in the <Destination IP Address, IPsec Protocol, SPI> tuple used to uniquely identify an SA. When a tunneled packet arrives at the tunnel endpoint, its SPI/Destination address/Protocol are used to look up the SA for this packet in the SAD. This destination address comes from the encapsulating IP header. Once the packet has been processed according to the tunnel SA and has come out of the tunnel, its selectors are "looked up" in the Inbound SPD. The Inbound SPD has a selector called destination address. This IP destination address is the one in the inner (encapsulated) IP header. In the case of a transport'd packet, there will be only one IP header and this ambiguity does not exist. [REQUIRED for all implementations]

- Source IP Address(es) (IPv4 or IPv6): this may be a single IP address (unicast, anycast, broadcast (IPv4 only), or multicast group), range of addresses (high and low values inclusive), address + mask, or a wildcard address. The last three are used to support more than one source system sharing the same SA (e.g., behind a security gateway or in a multihomed host). [REQUIRED for all implementations]

- Name: There are 2 cases (Note that these name forms are supported in the IPsec DOI.)
    1. User ID
        a. a fully qualified user name string (DNS), e.g., mozart@foo.bar.com
        b. X.500 distinguished name, e.g., C = US, SP = MA, O = GTE Internetworking, CN = Stephen T. Kent.
    2. System name (host, security gateway, etc.)
        a. a fully qualified DNS name, e.g., foo.bar.com
        b. X.500 distinguished name
        c. X.500 general name

  NOTE: One of the possible values of this selector is "OPAQUE".

[REQUIRED for the following cases. Note that support for name forms other than addresses is not required for manually keyed SAs.

- o User ID
  - native host implementations
  - BITW and BITS implementations acting as HOSTS with only one user
  - security gateway implementations for INBOUND processing.
- o System names -- all implementations]

- Data sensitivity level: (IPSO/CIPSO labels)
  [REQUIRED for all systems providing information flow security as per Section 8, OPTIONAL for all other systems.]

- Transport Layer Protocol: Obtained from the IPv4 "Protocol" or the IPv6 "Next Header" fields. This may be an individual protocol number. These packet fields may not contain the Transport Protocol due to the presence of IP extension headers, e.g., a Routing Header, AH, ESP, Fragmentation Header, Destination Options, Hop-by-hop options, etc. Note that the Transport Protocol may not be available in the case of receipt of a packet with an ESP header, thus a value of "OPAQUE" SHOULD be supported.
  [REQUIRED for all implementations]

  NOTE: To locate the transport protocol, a system has to chain through the packet headers checking the "Protocol" or "Next Header" field until it encounters either one it recognizes as a transport protocol, or until it reaches one that isn't on its list of extension headers, or until it encounters an ESP header that renders the transport protocol opaque.

- Source and Destination (e.g., TCP/UDP) Ports: These may be individual UDP or TCP port values or a wildcard port. (The use of the Next Protocol field and the Source and/or Destination Port fields (in conjunction with the Source and/or Destination Address fields), as an SA selector is sometimes referred to as "session-oriented keying."). Note that the source and destination ports may not be available in the case of receipt of a packet with an ESP header, thus a value of "OPAQUE" SHOULD be supported.

  The following table summarizes the relationship between the "Next Header" value in the packet and SPD and the derived Port Selector value for the SPD and SAD.

Next Hdr in Packet	Transport Layer Protocol in SPD	Derived Port Selector Field Value in SPD and SAD
ESP	ESP or ANY	ANY (i.e., don't look at it)
-don't care-	ANY	ANY (i.e., don't look at it)
specific value fragment	specific value	NOT ANY (i.e., drop packet)
specific value not fragment	specific value	actual port selector field

If the packet has been fragmented, then the port information may not be available in the current fragment. If so, discard the fragment. An ICMP PMTU should be sent for the first fragment, which will have the port information. [MAY be supported]

The IPsec implementation context determines how selectors are used. For example, a host implementation integrated into the stack may make use of a socket interface. When a new connection is established the SPD can be consulted and an SA (or SA bundle) bound to the socket. Thus traffic sent via that socket need not result in additional lookups to the SPD/SAD. In contrast, a BITS, BITW, or security gateway implementation needs to look at each packet and perform an SPD/SAD lookup based on the selectors. The allowable values for the selector fields differ between the traffic flow, the security association, and the security policy.

The following table summarizes the kinds of entries that one needs to be able to express in the SPD and SAD. It shows how they relate to the fields in data traffic being subjected to IPsec screening. (Note: the "wild" or "wildcard" entry for src and dst addresses includes a mask, range, etc.)

Field	Traffic Value	SAD Entry	SPD Entry
src addr	single IP addr	single,range,wild	single,range,wildcard
dst addr	single IP addr	single,range,wild	single,range,wildcard
xpt protocol*	xpt protocol	single,wildcard	single,wildcard
src port*	single src port	single,wildcard	single,wildcard
dst port*	single dst port	single,wildcard	single,wildcard
user id*	single user id	single,wildcard	single,wildcard
sec. labels	single value	single,wildcard	single,wildcard

* The SAD and SPD entries for these fields could be "OPAQUE" because the traffic value is encrypted.

NOTE: In principle, one could have selectors and/or selector values in the SPD which cannot be negotiated for an SA or SA bundle. Examples might include selector values used to select traffic for

discarding or enumerated lists which cause a separate SA to be created for each item on the list. For now, this is left for future versions of this document and the list of required selectors and selector values is the same for the SPD and the SAD. However, it is acceptable to have an administrative interface that supports use of selector values which cannot be negotiated provided that it does not mislead the user into believing it is creating an SA with these selector values. For example, the interface may allow the user to specify an enumerated list of values but would result in the creation of a separate policy and SA for each item on the list. A vendor might support such an interface to make it easier for its customers to specify clear and concise policy specifications.

4.4.3 Security Association Database (SAD)

In each IPsec implementation there is a nominal Security Association Database, in which each entry defines the parameters associated with one SA. Each SA has an entry in the SAD. For outbound processing, entries are pointed to by entries in the SPD. Note that if an SPD entry does not currently point to an SA that is appropriate for the packet, the implementation creates an appropriate SA (or SA Bundle) and links the SPD entry to the SAD entry (see Section 5.1.1). For inbound processing, each entry in the SAD is indexed by a destination IP address, IPsec protocol type, and SPI. The following parameters are associated with each entry in the SAD. This description does not purport to be a MIB, but only a specification of the minimal data items required to support an SA in an IPsec implementation.

For inbound processing: The following packet fields are used to look up the SA in the SAD:

- Outer Header's Destination IP address: the IPv4 or IPv6 Destination address.
  [REQUIRED for all implementations]
- IPsec Protocol: AH or ESP, used as an index for SA lookup in this database. Specifies the IPsec protocol to be applied to the traffic on this SA.
  [REQUIRED for all implementations]
- SPI: the 32-bit value used to distinguish among different SAs terminating at the same destination and using the same IPsec protocol.
  [REQUIRED for all implementations]

For each of the selectors defined in Section 4.4.2, the SA entry in the SAD MUST contain the value or values which were negotiated at the time the SA was created. For the sender, these values are used to decide whether a given SA is appropriate for use with an outbound packet. This is part of checking to see if there is an existing SA

that can be used. For the receiver, these values are used to check that the selector values in an inbound packet match those for the SA (and thus indirectly those for the matching policy). For the receiver, this is part of verifying that the SA was appropriate for this packet. (See Section 6 for rules for ICMP messages.) These fields can have the form of specific values, ranges, wildcards, or "OPAQUE" as described in section 4.4.2, "Selectors". Note that for an ESP SA, the encryption algorithm or the authentication algorithm could be "NULL". However they MUST not both be "NULL".

The following SAD fields are used in doing IPsec processing:

- Sequence Number Counter: a 32-bit value used to generate the Sequence Number field in AH or ESP headers.
  [REQUIRED for all implementations, but used only for outbound traffic.]
- Sequence Counter Overflow: a flag indicating whether overflow of the Sequence Number Counter should generate an auditable event and prevent transmission of additional packets on the SA.
  [REQUIRED for all implementations, but used only for outbound traffic.]
- Anti-Replay Window: a 32-bit counter and a bit-map (or equivalent) used to determine whether an inbound AH or ESP packet is a replay.
  [REQUIRED for all implementations but used only for inbound traffic. NOTE: If anti-replay has been disabled by the receiver, e.g., in the case of a manually keyed SA, then the Anti-Replay Window is not used.]
- AH Authentication algorithm, keys, etc.
  [REQUIRED for AH implementations]
- ESP Encryption algorithm, keys, IV mode, IV, etc.
  [REQUIRED for ESP implementations]
- ESP authentication algorithm, keys, etc. If the authentication service is not selected, this field will be null.
  [REQUIRED for ESP implementations]
- Lifetime of this Security Association: a time interval after which an SA must be replaced with a new SA (and new SPI) or terminated, plus an indication of which of these actions should occur. This may be expressed as a time or byte count, or a simultaneous use of both, the first lifetime to expire taking precedence. A compliant implementation MUST support both types of lifetimes, and must support a simultaneous use of both. If time is employed, and if IKE employs X.509 certificates for SA establishment, the SA lifetime must be constrained by the validity intervals of the certificates, and the NextIssueDate of the CRLs used in the IKE exchange

for the SA. Both initiator and responder are responsible for constraining SA lifetime in this fashion.
[REQUIRED for all implementations]

NOTE: The details of how to handle the refreshing of keys when SAs expire is a local matter. However, one reasonable approach is:

(a) If byte count is used, then the implementation SHOULD count the number of bytes to which the IPsec algorithm is applied. For ESP, this is the encryption algorithm (including Null encryption) and for AH, this is the authentication algorithm. This includes pad bytes, etc. Note that implementations SHOULD be able to handle having the counters at the ends of an SA get out of synch, e.g., because of packet loss or because the implementations at each end of the SA aren't doing things the same way.

(b) There SHOULD be two kinds of lifetime -- a soft lifetime which warns the implementation to initiate action such as setting up a replacement SA and a hard lifetime when the current SA ends.

(c) If the entire packet does not get delivered during the SAs lifetime, the packet SHOULD be discarded.

o IPsec protocol mode: tunnel, transport or wildcard. Indicates which mode of AH or ESP is applied to traffic on this SA. Note that if this field is "wildcard" at the sending end of the SA, then the application has to specify the mode to the IPsec implementation. This use of wildcard allows the same SA to be used for either tunnel or transport mode traffic on a per packet basis, e.g., by different sockets. The receiver does not need to know the mode in order to properly process the packet's IPsec headers.

[REQUIRED as follows, unless implicitly defined by context:
- host implementations must support all modes
- gateway implementations must support tunnel mode]

NOTE: The use of wildcard for the protocol mode of an inbound SA may add complexity to the situation in the receiver (host only). Since the packets on such an SA could be delivered in either tunnel or transport mode, the security of an incoming packet could depend in part on which mode had been used to deliver it. If, as a result, an application cared about the SA mode of a given packet, then the application would need a mechanism to obtain this mode information.

- Path MTU: any observed path MTU and aging variables. See Section 6.1.2.4
  [REQUIRED for all implementations but used only for outbound traffic]

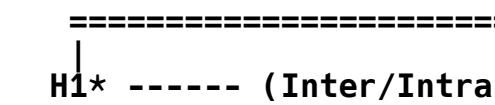
4.5 Basic Combinations of Security Associations

This section describes four examples of combinations of security associations that MUST be supported by compliant IPsec hosts or security gateways. Additional combinations of AH and/or ESP in tunnel and/or transport modes MAY be supported at the discretion of the implementor. Compliant implementations MUST be capable of generating these four combinations and on receipt, of processing them, but SHOULD be able to receive and process any combination. The diagrams and text below describe the basic cases. The legend for the diagrams is:

```
==== = one or more security associations (AH or ESP, transport
       or tunnel)
---- = connectivity (or if so labelled, administrative boundary)
Hx   = host x
SGx  = security gateway x
X*   = X supports IPsec
```

NOTE: The security associations below can be either AH or ESP. The mode (tunnel vs transport) is determined by the nature of the endpoints. For host-to-host SAs, the mode can be either transport or tunnel.

Case 1. The case of providing end-to-end security between 2 hosts across the Internet (or an Intranet).

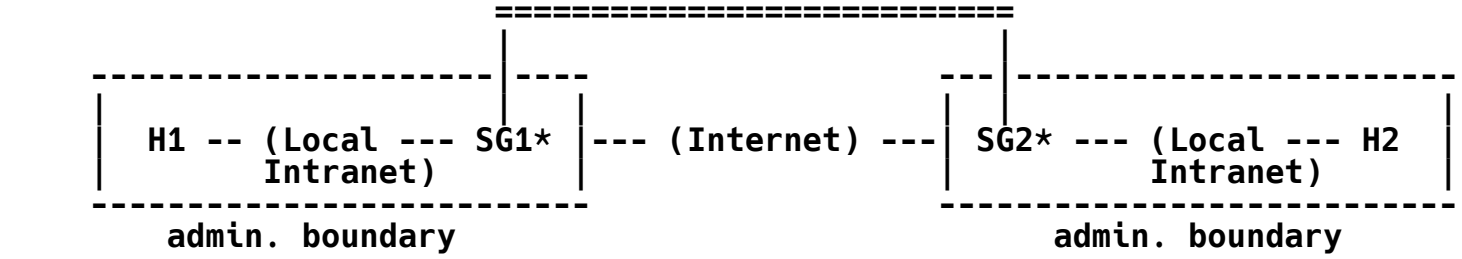


Note that either transport or tunnel mode can be selected by the hosts. So the headers in a packet between H1 and H2 could look like any of the following:

Transport	Tunnel
1. [IP1][AH][upper]	4. [IP2][AH][IP1][upper]
2. [IP1][ESP][upper]	5. [IP2][ESP][IP1][upper]
3. [IP1][AH][ESP][upper]	

Note that there is no requirement to support general nesting, but in transport mode, both AH and ESP can be applied to the packet. In this event, the SA establishment procedure MUST ensure that first ESP, then AH are applied to the packet.

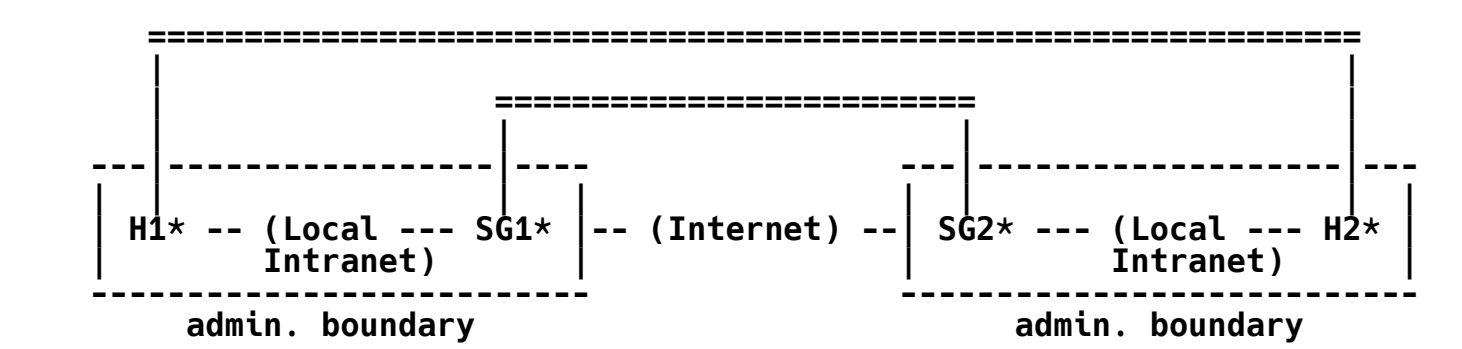
Case 2. This case illustrates simple virtual private networks support.



Only tunnel mode is required here. So the headers in a packet between SG1 and SG2 could look like either of the following:

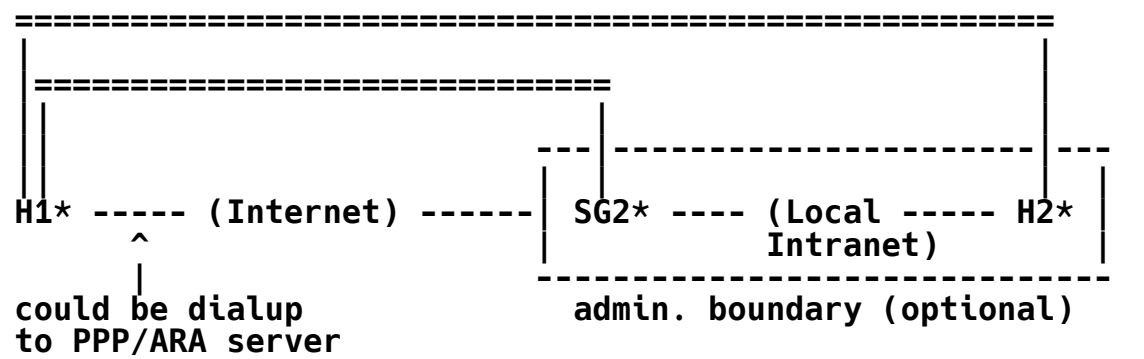
	Tunnel
4.	[IP2][AH][IP1][upper]
5.	[IP2][ESP][IP1][upper]

Case 3. This case combines cases 1 and 2, adding end-to-end security between the sending and receiving hosts. It imposes no new requirements on the hosts or security gateways, other than a requirement for a security gateway to be configurable to pass IPsec traffic (including ISAKMP traffic) for hosts behind it.



Case 4. This covers the situation where a remote host (H1) uses the Internet to reach an organization's firewall (SG2) and to then gain access to some server or other machine (H2). The remote host could be a mobile host (H1) dialing up to a local PPP/ARA server (not shown) on the Internet and then crossing the Internet to the home organization's firewall (SG2), etc. The

details of support for this case, (how H1 locates SG2, authenticates it, and verifies its authorization to represent H2) are discussed in Section 4.6.3, "Locating a Security Gateway".



Only tunnel mode is required between H1 and SG2. So the choices for the SA between H1 and SG2 would be one of the ones in case 2. The choices for the SA between H1 and H2 would be one of the ones in case 1.

Note that in this case, the sender MUST apply the transport header before the tunnel header. Therefore the management interface to the IPsec implementation MUST support configuration of the SPD and SAD to ensure this ordering of IPsec header application.

As noted above, support for additional combinations of AH and ESP is optional. Use of other, optional combinations may adversely affect interoperability.

4.6 SA and Key Management

IPsec mandates support for both manual and automated SA and cryptographic key management. The IPsec protocols, AH and ESP, are largely independent of the associated SA management techniques, although the techniques involved do affect some of the security services offered by the protocols. For example, the optional anti-replay services available for AH and ESP require automated SA management. Moreover, the granularity of key distribution employed with IPsec determines the granularity of authentication provided. (See also a discussion of this issue in Section 4.7.) In general, data origin authentication in AH and ESP is limited by the extent to which secrets used with the authentication algorithm (or with a key management protocol that creates such secrets) are shared among multiple possible sources.

The following text describes the minimum requirements for both types of SA management.

4.6.1 Manual Techniques

The simplest form of management is manual management, in which a person manually configures each system with keying material and security association management data relevant to secure communication with other systems. Manual techniques are practical in small, static environments but they do not scale well. For example, a company could create a Virtual Private Network (VPN) using IPsec in security gateways at several sites. If the number of sites is small, and since all the sites come under the purview of a single administrative domain, this is likely to be a feasible context for manual management techniques. In this case, the security gateway might selectively protect traffic to and from other sites within the organization using a manually configured key, while not protecting traffic for other destinations. It also might be appropriate when only selected communications need to be secured. A similar argument might apply to use of IPsec entirely within an organization for a small number of hosts and/or gateways. Manual management techniques often employ statically configured, symmetric keys, though other options also exist.

4.6.2 Automated SA and Key Management

Widespread deployment and use of IPsec requires an Internet-standard, scalable, automated, SA management protocol. Such support is required to facilitate use of the anti-replay features of AH and ESP, and to accommodate on-demand creation of SAs, e.g., for user- and session-oriented keying. (Note that the notion of "rekeying" an SA actually implies creation of a new SA with a new SPI, a process that generally implies use of an automated SA/key management protocol.)

The default automated key management protocol selected for use with IPsec is IKE [MSST97, Orm97, HC98] under the IPsec domain of interpretation [Pip98]. Other automated SA management protocols MAY be employed.

When an automated SA/key management protocol is employed, the output from this protocol may be used to generate multiple keys, e.g., for a single ESP SA. This may arise because:

- the encryption algorithm uses multiple keys (e.g., triple DES)
- the authentication algorithm uses multiple keys
- both encryption and authentication algorithms are employed

The Key Management System may provide a separate string of bits for each key or it may generate one string of bits from which all of them are extracted. If a single string of bits is provided, care needs to be taken to ensure that the parts of the system that map the string of bits to the required keys do so in the same fashion at both ends of the SA. To ensure that the IPsec implementations at each end of the SA use the same bits for the same keys, and irrespective of which part of the system divides the string of bits into individual keys, the encryption key(s) MUST be taken from the first (left-most, high-order) bits and the authentication key(s) MUST be taken from the remaining bits. The number of bits for each key is defined in the relevant algorithm specification RFC. In the case of multiple encryption keys or multiple authentication keys, the specification for the algorithm must specify the order in which they are to be selected from a single string of bits provided to the algorithm.

4.6.3 Locating a Security Gateway

This section discusses issues relating to how a host learns about the existence of relevant security gateways and once a host has contacted these security gateways, how it knows that these are the correct security gateways. The details of where the required information is stored is a local matter.

Consider a situation in which a remote host (H1) is using the Internet to gain access to a server or other machine (H2) and there is a security gateway (SG2), e.g., a firewall, through which H1's traffic must pass. An example of this situation would be a mobile host (Road Warrior) crossing the Internet to the home organization's firewall (SG2). (See Case 4 in the section 4.5 Basic Combinations of Security Associations.) This situation raises several issues:

1. How does H1 know/learn about the existence of the security gateway SG2?
2. How does it authenticate SG2, and once it has authenticated SG2, how does it confirm that SG2 has been authorized to represent H2?
3. How does SG2 authenticate H1 and verify that H1 is authorized to contact H2?
4. How does H1 know/learn about backup gateways which provide alternate paths to H2?

To address these problems, a host or security gateway MUST have an administrative interface that allows the user/administrator to configure the address of a security gateway for any sets of destination addresses that require its use. This includes the ability to configure:

- the requisite information for locating and authenticating the security gateway and verifying its authorization to represent the destination host.
- the requisite information for locating and authenticating any backup gateways and verifying their authorization to represent the destination host.

It is assumed that the SPD is also configured with policy information that covers any other IPsec requirements for the path to the security gateway and the destination host.

This document does not address the issue of how to automate the discovery/verification of security gateways.

4.7 Security Associations and Multicast

The receiver-orientation of the Security Association implies that, in the case of unicast traffic, the destination system will normally select the SPI value. By having the destination select the SPI value, there is no potential for manually configured Security Associations to conflict with automatically configured (e.g., via a key management protocol) Security Associations or for Security Associations from multiple sources to conflict with each other. For multicast traffic, there are multiple destination systems per multicast group. So some system or person will need to coordinate among all multicast groups to select an SPI or SPIs on behalf of each multicast group and then communicate the group's IPsec information to all of the legitimate members of that multicast group via mechanisms not defined here.

Multiple senders to a multicast group SHOULD use a single Security Association (and hence Security Parameter Index) for all traffic to that group when a symmetric key encryption or authentication algorithm is employed. In such circumstances, the receiver knows only that the message came from a system possessing the key for that multicast group. In such circumstances, a receiver generally will not be able to authenticate which system sent the multicast traffic. Specifications for other, more general multicast cases are deferred to later IPsec documents.

At the time this specification was published, automated protocols for multicast key distribution were not considered adequately mature for standardization. For multicast groups having relatively few members, manual key distribution or multiple use of existing unicast key distribution algorithms such as modified Diffie-Hellman appears feasible. For very large groups, new scalable techniques will be needed. An example of current work in this area is the Group Key Management Protocol (GKMP) [HM97].

5. IP Traffic Processing

As mentioned in Section 4.4.1 "The Security Policy Database (SPD)", the SPD must be consulted during the processing of all traffic (INBOUND and OUTBOUND), including non-IPsec traffic. If no policy is found in the SPD that matches the packet (for either inbound or outbound traffic), the packet MUST be discarded.

NOTE: All of the cryptographic algorithms used in IPsec expect their input in canonical network byte order (see Appendix in RFC 791) and generate their output in canonical network byte order. IP packets are also transmitted in network byte order.

5.1 Outbound IP Traffic Processing

5.1.1 Selecting and Using an SA or SA Bundle

In a security gateway or BITW implementation (and in many BITS implementations), each outbound packet is compared against the SPD to determine what processing is required for the packet. If the packet is to be discarded, this is an auditable event. If the traffic is allowed to bypass IPsec processing, the packet continues through "normal" processing for the environment in which the IPsec processing is taking place. If IPsec processing is required, the packet is either mapped to an existing SA (or SA bundle), or a new SA (or SA bundle) is created for the packet. Since a packet's selectors might match multiple policies or multiple extant SAs and since the SPD is ordered, but the SAD is not, IPsec MUST:

1. Match the packet's selector fields against the outbound policies in the SPD to locate the first appropriate policy, which will point to zero or more SA bundles in the SAD.

2. Match the packet's selector fields against those in the SA bundles found in (1) to locate the first SA bundle that matches. If no SAs were found or none match, create an appropriate SA bundle and link the SPD entry to the SAD entry. If no key management entity is found, drop the packet.

3. Use the SA bundle found/created in (2) to do the required IPsec processing, e.g., authenticate and encrypt.

In a host IPsec implementation based on sockets, the SPD will be consulted whenever a new socket is created, to determine what, if any, IPsec processing will be applied to the traffic that will flow on that socket.

NOTE: A compliant implementation MUST not allow instantiation of an ESP SA that employs both a NULL encryption and a NULL authentication algorithm. An attempt to negotiate such an SA is an auditable event.

5.1.2 Header Construction for Tunnel Mode

This section describes the handling of the inner and outer IP headers, extension headers, and options for AH and ESP tunnels. This includes how to construct the encapsulating (outer) IP header, how to handle fields in the inner IP header, and what other actions should be taken. The general idea is modeled after the one used in RFC 2003, "IP Encapsulation with IP":

- The outer IP header Source Address and Destination Address identify the "endpoints" of the tunnel (the encapsulator and decapsulator). The inner IP header Source Address and Destination Addresses identify the original sender and recipient of the datagram, (from the perspective of this tunnel), respectively. (see footnote 3 after the table in 5.1.2.1 for more details on the encapsulating source IP address.)
- The inner IP header is not changed except to decrement the TTL as noted below, and remains unchanged during its delivery to the tunnel exit point.
- No change to IP options or extension headers in the inner header occurs during delivery of the encapsulated datagram through the tunnel.
- If need be, other protocol headers such as the IP Authentication header may be inserted between the outer IP header and the inner IP header.

The tables in the following sub-sections show the handling for the different header/option fields (constructed = the value in the outer field is constructed independently of the value in the inner).

5.1.2.1 IPv4 -- Header Construction for Tunnel Mode

	<-- How Outer Hdr Relates to Inner Hdr -->	
	Outer Hdr at	Inner Hdr at
IPv4	Encapsulator	Decapsulator
Header fields:	----------	----------
version	4 (1)	no change
header length	constructed	no change
TOS	copied from inner hdr (5)	no change
total length	constructed	no change
ID	constructed	no change
flags (DF,MF)	constructed, DF (4)	no change
fragmt offset	constructed	no change

TTL	constructed (2)	decrement (2)
protocol	AH, ESP, routing hdr	no change
checksum	constructed	constructed (2)
src address	constructed (3)	no change
dest address	constructed (3)	no change
Options	never copied	no change

1. The IP version in the encapsulating header can be different from the value in the inner header.

2. The TTL in the inner header is decremented by the encapsulator prior to forwarding and by the decapsulator if it forwards the packet. (The checksum changes when the TTL changes.)

   Note: The decrementing of the TTL is one of the usual actions that takes place when forwarding a packet. Packets originating from the same node as the encapsulator do not have their TTL's decremented, as the sending node is originating the packet rather than forwarding it.

3. src and dest addresses depend on the SA, which is used to determine the dest address which in turn determines which src address (net interface) is used to forward the packet.

   NOTE: In principle, the encapsulating IP source address can be any of the encapsulator's interface addresses or even an address different from any of the encapsulator's IP addresses, (e.g., if it's acting as a NAT box) so long as the address is reachable through the encapsulator from the environment into which the packet is sent. This does not cause a problem because IPsec does not currently have any INBOUND processing requirement that involves the Source Address of the encapsulating IP header. So while the receiving tunnel endpoint looks at the Destination Address in the encapsulating IP header, it only looks at the Source Address in the inner (encapsulated) IP header.

4. configuration determines whether to copy from the inner header (IPv4 only), clear or set the DF.

5. If Inner Hdr is IPv4 (Protocol = 4), copy the TOS. If Inner Hdr is IPv6 (Protocol = 41), map the Class to TOS.

5.1.2.2 IPv6 -- Header Construction for Tunnel Mode

See previous section 5.1.2 for notes 1-5 indicated by (footnote number).

	<-- How Outer Hdr	Relates Inner Hdr --->
	Outer Hdr at	Inner Hdr at
IPv6	Encapsulator	Decapsulator
Header fields:	---	---
version	6 (1)	no change
class	copied or configured (6)	no change
flow id	copied or configured	no change
len	constructed	no change
next header	AH,ESP,routing hdr	no change
hop limit	constructed (2)	decrement (2)
src address	constructed (3)	no change
dest address	constructed (3)	no change
Extension headers	never copied	no change

6. If Inner Hdr is IPv6 (Next Header = 41), copy the Class. If Inner Hdr is IPv4 (Next Header = 4), map the TOS to Class.

5.2 Processing Inbound IP Traffic

Prior to performing AH or ESP processing, any IP fragments are reassembled. Each inbound IP datagram to which IPsec processing will be applied is identified by the appearance of the AH or ESP values in the IP Next Protocol field (or of AH or ESP as an extension header in the IPv6 context).

Note: Appendix C contains sample code for a bitmask check for a 32 packet window that can be used for implementing anti-replay service.

5.2.1 Selecting and Using an SA or SA Bundle

Mapping the IP datagram to the appropriate SA is simplified because of the presence of the SPI in the AH or ESP header. Note that the selector checks are made on the inner headers not the outer (tunnel) headers. The steps followed are:

1. Use the packet's destination address (outer IP header), IPsec protocol, and SPI to look up the SA in the SAD. If the SA lookup fails, drop the packet and log/report the error.

2. Use the SA found in (1) to do the IPsec processing, e.g., authenticate and decrypt. This step includes matching the packet's (Inner Header if tunneled) selectors to the selectors in the SA. Local policy determines the specificity of the SA selectors (single value, list, range, wildcard). In general, a packet's source address MUST match the SA selector value. However, an ICMP packet received on a tunnel mode SA may have a source address

other than that bound to the SA and thus such packets should be permitted as exceptions to this check. For an ICMP packet, the selectors from the enclosed problem packet (the source and destination addresses and ports should be swapped) should be checked against the selectors for the SA. Note that some or all of these selectors may be inaccessible because of limitations on how many bits of the problem packet the ICMP packet is allowed to carry or due to encryption. See Section 6.

Do (1) and (2) for every IPsec header until a Transport Protocol Header or an IP header that is NOT for this system is encountered. Keep track of what SAs have been used and their order of application.

3. Find an incoming policy in the SPD that matches the packet. This could be done, for example, by use of backpointers from the SAs to the SPD or by matching the packet's selectors (Inner Header if tunneled) against those of the policy entries in the SPD.

4. Check whether the required IPsec processing has been applied, i.e., verify that the SA's found in (1) and (2) match the kind and order of SAs required by the policy found in (3).

   NOTE: The correct "matching" policy will not necessarily be the first inbound policy found. If the check in (4) fails, steps (3) and (4) are repeated until all policy entries have been checked or until the check succeeds.

At the end of these steps, pass the resulting packet to the Transport Layer or forward the packet. Note that any IPsec headers processed in these steps may have been removed, but that this information, i.e., what SAs were used and the order of their application, may be needed for subsequent IPsec or firewall processing.

Note that in the case of a security gateway, if forwarding causes a packet to exit via an IPsec-enabled interface, then additional IPsec processing may be applied.

5.2.2 Handling of AH and ESP tunnels

The handling of the inner and outer IP headers, extension headers, and options for AH and ESP tunnels should be performed as described in the tables in Section 5.1.

6. ICMP Processing (relevant to IPsec)

The focus of this section is on the handling of ICMP error messages. Other ICMP traffic, e.g., Echo/Reply, should be treated like other traffic and can be protected on an end-to-end basis using SAs in the usual fashion.

An ICMP error message protected by AH or ESP and generated by a router SHOULD be processed and forwarded in a tunnel mode SA. Local policy determines whether or not it is subjected to source address checks by the router at the destination end of the tunnel. Note that if the router at the originating end of the tunnel is forwarding an ICMP error message from another router, the source address check would fail. An ICMP message protected by AH or ESP and generated by a router MUST NOT be forwarded on a transport mode SA (unless the SA has been established to the router acting as a host, e.g., a Telnet connection used to manage a router). An ICMP message generated by a host SHOULD be checked against the source IP address selectors bound to the SA in which the message arrives. Note that even if the source of an ICMP error message is authenticated, the returned IP header could be invalid. Accordingly, the selector values in the IP header SHOULD also be checked to be sure that they are consistent with the selectors for the SA over which the ICMP message was received.

The table in Appendix D characterize ICMP messages as being either host generated, router generated, both, unknown/unassigned. ICMP messages falling into the last two categories should be handled as determined by the receiver's policy.

An ICMP message not protected by AH or ESP is unauthenticated and its processing and/or forwarding may result in denial of service. This suggests that, in general, it would be desirable to ignore such messages. However, it is expected that many routers (vs. security gateways) will not implement IPsec for transit traffic and thus strict adherence to this rule would cause many ICMP messages to be discarded. The result is that some critical IP functions would be lost, e.g., redirection and PMTU processing. Thus it MUST be possible to configure an IPsec implementation to accept or reject (router) ICMP traffic as per local security policy.

The remainder of this section addresses how PMTU processing MUST be performed at hosts and security gateways. It addresses processing of both authenticated and unauthenticated ICMP PMTU messages. However, as noted above, unauthenticated ICMP messages MAY be discarded based on local policy.

6.1 PMTU/DF Processing

6.1.1 DF Bit

In cases where a system (host or gateway) adds an encapsulating header (ESP tunnel or AH tunnel), it MUST support the option of copying the DF bit from the original packet to the encapsulating header (and processing ICMP PMTU messages). This means that it MUST be possible to configure the system's treatment of the DF bit (set, clear, copy from encapsulated header) for each interface. (See Appendix B for rationale.)

6.1.2 Path MTU Discovery (PMTU)

This section discusses IPsec handling for Path MTU Discovery messages. ICMP PMTU is used here to refer to an ICMP message for:

IPv4 (RFC 792):
- Type = 3 (Destination Unreachable)
- Code = 4 (Fragmentation needed and DF set)
- Next-Hop MTU in the low-order 16 bits of the second word of the ICMP header (labelled "unused" in RFC 792), with high-order 16 bits set to zero

IPv6 (RFC 1885):
- Type = 2 (Packet Too Big)
- Code = 0 (Fragmentation needed)
- Next-Hop MTU in the 32 bit MTU field of the ICMP6 message

6.1.2.1 Propagation of PMTU

The amount of information returned with the ICMP PMTU message (IPv4 or IPv6) is limited and this affects what selectors are available for use in further propagating the PMTU information. (See Appendix B for more detailed discussion of this topic.)

o PMTU message with 64 bits of IPsec header -- If the ICMP PMTU message contains only 64 bits of the IPsec header (minimum for IPv4), then a security gateway MUST support the following options on a per SPI/SA basis:

   a. if the originating host can be determined (or the possible sources narrowed down to a manageable number), send the PM information to all the possible originating hosts.
   b. if the originating host cannot be determined, store the PMTU with the SA and wait until the next packet(s) arrive from the originating host for the relevant security association. If

the packet(s) are bigger than the PMTU, drop the packet(s), and compose ICMP PMTU message(s) with the new packet(s) and the updated PMTU, and send the ICMP message(s) about the problem to the originating host. Retain the PMTU information for any message that might arrive subsequently (see Section 6.1.2.4, "PMTU Aging").

- o PMTU message with >64 bits of IPsec header -- If the ICMP message contains more information from the original packet then there may be enough non-opaque information to immediately determine to which host to propagate the ICMP/PMTU message and to provide that system with the 5 fields (source address, destination address, source port, destination port, transport protocol) needed to determine where to store/update the PMTU. Under such circumstances, a security gateway MUST generate an ICMP PMTU message immediately upon receipt of an ICMP PMTU from further down the path.

- o Distributing the PMTU to the Transport Layer -- The host mechanism for getting the updated PMTU to the transport layer is unchanged, as specified in RFC 1191 (Path MTU Discovery).

6.1.2.2 Calculation of PMTU

The calculation of PMTU from an ICMP PMTU MUST take into account the addition of any IPsec header -- AH transport, ESP transport, AH/ESP transport, ESP tunnel, AH tunnel. (See Appendix B for discussion of implementation issues.)

Note: In some situations the addition of IPsec headers could result in an effective PMTU (as seen by the host or application) that is unacceptably small. To avoid this problem, the implementation may establish a threshold below which it will not report a reduced PMTU. In such cases, the implementation would apply IPsec and then fragment the resulting packet according to the PMTU. This would result in a more efficient use of the available bandwidth.

6.1.2.3 Granularity of PMTU Processing

In hosts, the granularity with which ICMP PMTU processing can be done differs depending on the implementation situation. Looking at a host, there are 3 situations that are of interest with respect to PMTU issues (See Appendix B for additional details on this topic.):

a. Integration of IPsec into the native IP implementation
b. Bump-in-the-stack implementations, where IPsec is implemented "underneath" an existing implementation of a TCP/IP protocol stack, between the native IP and the local network drivers

c. No IPsec implementation -- This case is included because it is relevant in cases where a security gateway is sending PMTU information back to a host.

Only in case (a) can the PMTU data be maintained at the same granularity as communication associations. In (b) and (c), the IP layer will only be able to maintain PMTU data at the granularity of source and destination IP addresses (and optionally TOS), as described in RFC 1191. This is an important difference, because more than one communication association may map to the same source and destination IP addresses, and each communication association may have a different amount of IPsec header overhead (e.g., due to use of different transforms or different algorithms).

Implementation of the calculation of PMTU and support for PMTUs at the granularity of individual communication associations is a local matter. However, a socket-based implementation of IPsec in a host SHOULD maintain the information on a per socket basis. Bump in the stack systems MUST pass an ICMP PMTU to the host IP implementation, after adjusting it for any IPsec header overhead added by these systems. The calculation of the overhead SHOULD be determined by analysis of the SPI and any other selector information present in a returned ICMP PMTU message.

6.1.2.4 PMTU Aging

In all systems (host or gateway) implementing IPsec and maintaining PMTU information, the PMTU associated with a security association (transport or tunnel) MUST be "aged" and some mechanism put in place for updating the PMTU in a timely manner, especially for discovering if the PMTU is smaller than it needs to be. A given PMTU has to remain in place long enough for a packet to get from the source end of the security association to the system at the other end of the security association and propagate back an ICMP error message if the current PMTU is too big. Note that if there are nested tunnels, multiple packets and round trip times might be required to get an ICMP message back to an encapsulator or originating host.

Systems SHOULD use the approach described in the Path MTU Discovery document (RFC 1191, Section 6.3), which suggests periodically resetting the PMTU to the first-hop data-link MTU and then letting the normal PMTU Discovery processes update the PMTU as necessary. The period SHOULD be configurable.

7. Auditing

Not all systems that implement IPsec will implement auditing. For the most part, the granularity of auditing is a local matter. However, several auditable events are identified in the AH and ESP specifications and for each of these events a minimum set of information that SHOULD be included in an audit log is defined. Additional information also MAY be included in the audit log for each of these events, and additional events, not explicitly called out in this specification, also MAY result in audit log entries. There is no requirement for the receiver to transmit any message to the purported transmitter in response to the detection of an auditable event, because of the potential to induce denial of service via such action.

8. Use in Systems Supporting Information Flow Security

Information of various sensitivity levels may be carried over a single network. Information labels (e.g., Unclassified, Company Proprietary, Secret) [DoD85, DoD87] are often employed to distinguish such information. The use of labels facilitates segregation of information, in support of information flow security models, e.g., the Bell-LaPadula model [BL73]. Such models, and corresponding supporting technology, are designed to prevent the unauthorized flow of sensitive information, even in the face of Trojan Horse attacks. Conventional, discretionary access control (DAC) mechanisms, e.g., based on access control lists, generally are not sufficient to support such policies, and thus facilities such as the SPD do not suffice in such environments.

In the military context, technology that supports such models is often referred to as multi-level security (MLS). Computers and networks often are designated "multi-level secure" if they support the separation of labelled data in conjunction with information flow security policies. Although such technology is more broadly applicable than just military applications, this document uses the acronym "MLS" to designate the technology, consistent with much extant literature.

IPsec mechanisms can easily support MLS networking. MLS networking requires the use of strong Mandatory Access Controls (MAC), which unprivileged users or unprivileged processes are incapable of controlling or violating. This section pertains only to the use of these IP security mechanisms in MLS (information flow security policy) environments. Nothing in this section applies to systems not claiming to provide MLS.

As used in this section, "sensitivity information" might include implementation-defined hierarchic levels, categories, and/or releasability information.

AH can be used to provide strong authentication in support of mandatory access control decisions in MLS environments. If explicit IP sensitivity information (e.g., IPSO [Ken91]) is used and confidentiality is not considered necessary within the particular operational environment, AH can be used to authenticate the binding between sensitivity labels in the IP header and the IP payload (including user data). This is a significant improvement over labeled IPv4 networks where the sensitivity information is trusted even though there is no authentication or cryptographic binding of the information to the IP header and user data. IPv4 networks might or might not use explicit labelling. IPv6 will normally use implicit sensitivity information that is part of the IPsec Security Association but not transmitted with each packet instead of using explicit sensitivity information. All explicit IP sensitivity information MUST be authenticated using either ESP, AH, or both.

Encryption is useful and can be desirable even when all of the hosts are within a protected environment, for example, behind a firewall or disjoint from any external connectivity. ESP can be used, in conjunction with appropriate key management and encryption algorithms, in support of both DAC and MAC. (The choice of encryption and authentication algorithms, and the assurance level of an IPsec implementation will determine the environments in which an implementation may be deemed sufficient to satisfy MLS requirements.) Key management can make use of sensitivity information to provide MAC. IPsec implementations on systems claiming to provide MLS SHOULD be capable of using IPsec to provide MAC for IP-based communications.

8.1 Relationship Between Security Associations and Data Sensitivity

Both the Encapsulating Security Payload and the Authentication Header can be combined with appropriate Security Association policies to provide multi-level secure networking. In this case each SA (or SA bundle) is normally used for only a single instance of sensitivity information. For example, "PROPRIETARY - Internet Engineering" must be associated with a different SA (or SA bundle) from "PROPRIETARY - Finance".

8.2 Sensitivity Consistency Checking

An MLS implementation (both host and router) MAY associate sensitivity information, or a range of sensitivity information with an interface, or a configured IP address with its associated prefix (the latter is sometimes referred to as a logical interface, or an

interface alias). If such properties exist, an implementation SHOULD compare the sensitivity information associated with the packet against the sensitivity information associated with the interface or address/prefix from which the packet arrived, or through which the packet will depart. This check will either verify that the sensitivities match, or that the packet's sensitivity falls within the range of the interface or address/prefix.

The checking SHOULD be done on both inbound and outbound processing.

8.3 Additional MLS Attributes for Security Association Databases

Section 4.4 discussed two Security Association databases (the Security Policy Database (SPD) and the Security Association Database (SAD)) and the associated policy selectors and SA attributes. MLS networking introduces an additional selector/attribute:

- Sensitivity information.

The Sensitivity information aids in selecting the appropriate algorithms and key strength, so that the traffic gets a level of protection appropriate to its importance or sensitivity as described in section 8.1. The exact syntax of the sensitivity information is implementation defined.

8.4 Additional Inbound Processing Steps for MLS Networking

After an inbound packet has passed through IPsec processing, an MLS implementation SHOULD first check the packet's sensitivity (as defined by the SA (or SA bundle) used for the packet) with the interface or address/prefix as described in section 8.2 before delivering the datagram to an upper-layer protocol or forwarding it.

The MLS system MUST retain the binding between the data received in an IPsec protected packet and the sensitivity information in the SA or SAs used for processing, so appropriate policy decisions can be made when delivering the datagram to an application or forwarding engine. The means for maintaining this binding are implementation specific.

8.5 Additional Outbound Processing Steps for MLS Networking

An MLS implementation of IPsec MUST perform two additional checks besides the normal steps detailed in section 5.1.1. When consulting the SPD or the SAD to find an outbound security association, the MLS implementation MUST use the sensitivity of the data to select an

appropriate outbound SA or SA bundle. The second check comes before forwarding the packet out to its destination, and is the sensitivity consistency checking described in section 8.2.

8.6 Additional MLS Processing for Security Gateways

An MLS security gateway MUST follow the previously mentioned inbound and outbound processing rules as well as perform some additional processing specific to the intermediate protection of packets in an MLS environment.

A security gateway MAY act as an outbound proxy, creating SAs for MLS systems that originate packets forwarded by the gateway. These MLS systems may explicitly label the packets to be forwarded, or the whole originating network may have sensitivity characteristics associated with it. The security gateway MUST create and use appropriate SAs for AH, ESP, or both, to protect such traffic it forwards.

Similarly such a gateway SHOULD accept and process inbound AH and/or ESP packets and forward appropriately, using explicit packet labeling, or relying on the sensitivity characteristics of the destination network.

9. Performance Issues

The use of IPsec imposes computational performance costs on the hosts or security gateways that implement these protocols. These costs are associated with the memory needed for IPsec code and data structures, and the computation of integrity check values, encryption and decryption, and added per-packet handling. The per-packet computational costs will be manifested by increased latency and, possibly, reduced throughout. Use of SA/key management protocols, especially ones that employ public key cryptography, also adds computational performance costs to use of IPsec. These per-association computational costs will be manifested in terms of increased latency in association establishment. For many hosts, it is anticipated that software-based cryptography will not appreciably reduce throughput, but hardware may be required for security gateways (since they represent aggregation points), and for some hosts.

The use of IPsec also imposes bandwidth utilization costs on transmission, switching, and routing components of the Internet infrastructure, components not implementing IPsec. This is due to the increase in the packet size resulting from the addition of AH and/or ESP headers, AH and ESP tunneling (which adds a second IP header), and the increased packet traffic associated with key management protocols. It is anticipated that, in most instances,

this increased bandwidth demand will not noticeably affect the Internet infrastructure. However, in some instances, the effects may be significant, e.g., transmission of ESP encrypted traffic over a dialup link that otherwise would have compressed the traffic.

Note: The initial SA establishment overhead will be felt in the first packet. This delay could impact the transport layer and application. For example, it could cause TCP to retransmit the SYN before the ISAKMP exchange is done. The effect of the delay would be different on UDP than TCP because TCP shouldn't transmit anything other than the SYN until the connection is set up whereas UDP will go ahead and transmit data beyond the first packet.

Note: As discussed earlier, compression can still be employed at layers above IP. There is an IETF working group (IP Payload Compression Protocol (ippcp)) working on "protocol specifications that make it possible to perform lossless compression on individual payloads before the payload is processed by a protocol that encrypts it. These specifications will allow for compression operations to be performed prior to the encryption of a payload by IPsec protocols."

10. Conformance Requirements

All IPv4 systems that claim to implement IPsec MUST comply with all requirements of the Security Architecture document. All IPv6 systems MUST comply with all requirements of the Security Architecture document.

11. Security Considerations

The focus of this document is security; hence security considerations permeate this specification.

12. Differences from RFC 1825

This architecture document differs substantially from RFC 1825 in detail and in organization, but the fundamental notions are unchanged. This document provides considerable additional detail in terms of compliance specifications. It introduces the SPD and SAD, and the notion of SA selectors. It is aligned with the new versions of AH and ESP, which also differ from their predecessors. Specific requirements for supported combinations of AH and ESP are newly added, as are details of PMTU management.

Acknowledgements

Many of the concepts embodied in this specification were derived from or influenced by the US Government's SP3 security protocol, ISO/IEC's NLSP, the proposed swIPe security protocol [SDNS, ISO, IB93, IBK93], and the work done for SNMP Security and SNMPv2 Security.

For over 3 years (although it sometimes seems *much* longer), this document has evolved through multiple versions and iterations. During this time, many people have contributed significant ideas and energy to the process and the documents themselves. The authors would like to thank Karen Seo for providing extensive help in the review, editing, background research, and coordination for this version of the specification. The authors would also like to thank the members of the IPsec and IPng working groups, with special mention of the efforts of (in alphabetic order): Steve Bellovin, Steve Deering, James Hughes, Phil Karn, Frank Kastenholz, Perry Metzger, David Mihelcic, Hilarie Orman, Norman Shulman, William Simpson, Harry Varnis, and Nina Yuan.

Appendix A -- Glossary

This section provides definitions for several key terms that are employed in this document. Other documents provide additional definitions and background information relevant to this technology, e.g., [VK83, HA94]. Included in this glossary are generic security service and security mechanism terms, plus IPsec-specific terms.

Access Control
Access control is a security service that prevents unauthorized use of a resource, including the prevention of use of a resource in an unauthorized manner. In the IPsec context, the resource to which access is being controlled is often:

- for a host, computing cycles or data
- for a security gateway, a network behind the gateway or bandwidth on that network.

Anti-replay
[See "Integrity" below]

Authentication
This term is used informally to refer to the combination of two nominally distinct security services, data origin authentication and connectionless integrity. See the definitions below for each of these services.

Availability
Availability, when viewed as a security service, addresses the security concerns engendered by attacks against networks that deny or degrade service. For example, in the IPsec context, the use of anti-replay mechanisms in AH and ESP support availability.

Confidentiality
Confidentiality is the security service that protects data from unauthorized disclosure. The primary confidentiality concern in most instances is unauthorized disclosure of application level data, but disclosure of the external characteristics of communication also can be a concern in some circumstances. Traffic flow confidentiality is the service that addresses this latter concern by concealing source and destination addresses, message length, or frequency of communication. In the IPsec context, using ESP in tunnel mode, especially at a security gateway, can provide some level of traffic flow confidentiality. (See also traffic analysis, below.)

Encryption
Encryption is a security mechanism used to transform data from an intelligible form (plaintext) into an unintelligible form (ciphertext), to provide confidentiality. The inverse transformation process is designated "decryption". Oftimes the term "encryption" is used to generically refer to both processes.

Data Origin Authentication
Data origin authentication is a security service that verifies the identity of the claimed source of data. This service is usually bundled with connectionless integrity service.

Integrity
Integrity is a security service that ensures that modifications to data are detectable. Integrity comes in various flavors to match application requirements. IPsec supports two forms of integrity: connectionless and a form of partial sequence integrity. Connectionless integrity is a service that detects modification of an individual IP datagram, without regard to the ordering of the datagram in a stream of traffic. The form of partial sequence integrity offered in IPsec is referred to as anti-replay integrity, and it detects arrival of duplicate IP datagrams (within a constrained window). This is in contrast to connection-oriented integrity, which imposes more stringent sequencing requirements on traffic, e.g., to be able to detect lost or re-ordered messages. Although authentication and integrity services often are cited separately, in practice they are intimately connected and almost always offered in tandem.

Security Association (SA)
A simplex (uni-directional) logical connection, created for security purposes. All traffic traversing an SA is provided the same security processing. In IPsec, an SA is an internet layer abstraction implemented through the use of AH or ESP.

Security Gateway
A security gateway is an intermediate system that acts as the communications interface between two networks. The set of hosts (and networks) on the external side of the security gateway is viewed as untrusted (or less trusted), while the networks and hosts and on the internal side are viewed as trusted (or more trusted). The internal subnets and hosts served by a security gateway are presumed to be trusted by virtue of sharing a common, local, security administration. (See "Trusted Subnetwork" below.) In the IPsec context, a security gateway is a point at which AH and/or ESP is implemented in order to serve

a set of internal hosts, providing security services for these hosts when they communicate with external hosts also employing IPsec (either directly or via another security gateway).

SPI

Acronym for "Security Parameters Index". The combination of a destination address, a security protocol, and an SPI uniquely identifies a security association (SA, see above). The SPI is carried in AH and ESP protocols to enable the receiving system to select the SA under which a received packet will be processed. An SPI has only local significance, as defined by the creator of the SA (usually the receiver of the packet carrying the SPI); thus an SPI is generally viewed as an opaque bit string. However, the creator of an SA may choose to interpret the bits in an SPI to facilitate local processing.

Traffic Analysis

The analysis of network traffic flow for the purpose of deducing information that is useful to an adversary. Examples of such information are frequency of transmission, the identities of the conversing parties, sizes of packets, flow identifiers, etc. [Sch94]

Trusted Subnetwork

A subnetwork containing hosts and routers that trust each other not to engage in active or passive attacks. There also is an assumption that the underlying communications channel (e.g., a LAN or CAN) isn't being attacked by other means.

Appendix B -- Analysis/Discussion of PMTU/DF/Fragmentation Issues

B.1 DF bit

In cases where a system (host or gateway) adds an encapsulating header (e.g., ESP tunnel), should/must the DF bit in the original packet be copied to the encapsulating header?

Fragmenting seems correct for some situations, e.g., it might be appropriate to fragment packets over a network with a very small MTU, e.g., a packet radio network, or a cellular phone hop to mobile node, rather than propagate back a very small PMTU for use over the rest of the path. In other situations, it might be appropriate to set the DF bit in order to get feedback from later routers about PMTU constraints which require fragmentation. The existence of both of these situations argues for enabling a system to decide whether or not to fragment over a particular network "link", i.e., for requiring an implementation to be able to copy the DF bit (and to process ICMP PMTU messages), but making it an option to be selected on a per interface basis. In other words, an administrator should be able to configure the router's treatment of the DF bit (set, clear, copy from encapsulated header) for each interface.

Note: If a bump-in-the-stack implementation of IPsec attempts to apply different IPsec algorithms based on source/destination ports, it will be difficult to apply Path MTU adjustments.

B.2 Fragmentation

If required, IP fragmentation occurs after IPsec processing within an IPsec implementation. Thus, transport mode AH or ESP is applied only to whole IP datagrams (not to IP fragments). An IP packet to which AH or ESP has been applied may itself be fragmented by routers en route, and such fragments MUST be reassembled prior to IPsec processing at a receiver. In tunnel mode, AH or ESP is applied to an IP packet, the payload of which may be a fragmented IP packet. For example, a security gateway, "bump-in-the-stack" (BITS), or "bump-in-the-wire" (BITW) IPsec implementation may apply tunnel mode AH to such fragments. Note that BITS or BITW implementations are examples of where a host IPsec implementation might receive fragments to which tunnel mode is to be applied. However, if transport mode is to be applied, then these implementations MUST reassemble the fragments prior to applying IPsec.

NOTE: IPsec always has to figure out what the encapsulating IP header fields are. This is independent of where you insert IPsec and is intrinsic to the definition of IPsec. Therefore any IPsec implementation that is not integrated into an IP implementation must include code to construct the necessary IP headers (e.g., IP2):

- o AH-tunnel --> IP2-AH-IP1-Transport-Data
- o ESP-tunnel --> IP2-ESP_hdr-IP1-Transport-Data-ESP_trailer

****

Overall, the fragmentation/reassembly approach described above works for all cases examined.

	AH Xport		AH Tunnel		ESP Xport		ESP Tunnel	
Implementation approach	IPv4	IPv6	IPv4	IPv6	IPv4	IPv6	IPv4	IPv6
Hosts (integr w/ IP stack)	Y	Y	Y	Y	Y	Y	Y	Y
Hosts (betw/ IP and drivers)	Y	Y	Y	Y	Y	Y	Y	Y
S. Gwy (integr w/ IP stack)			Y	Y			Y	Y
Outboard crypto processor *								

* If the crypto processor system has its own IP address, then it is covered by the security gateway case. This box receives the packet from the host and performs IPsec processing. It has to be able to handle the same AH, ESP, and related IPv4/IPv6 tunnel processing that a security gateway would have to handle. If it doesn't have it's own address, then it is similar to the bump-in-the stack implementation between IP and the network drivers.

The following analysis assumes that:

1. There is only one IPsec module in a given system's stack. There isn't an IPsec module A (adding ESP/encryption and thus) hiding the transport protocol, SRC port, and DEST port from IPsec module B.
2. There are several places where IPsec could be implemented (as shown in the table above).
   a. Hosts with integration of IPsec into the native IP implementation. Implementer has access to the source for the stack.
   b. Hosts with bump-in-the-stack implementations, where IPsec is implemented between IP and the local network drivers. Source access for stack is not available; but there are well-defined interfaces that allows the IPsec code to be incorporated into the system.

c. Security gateways and outboard crypto processors with integration of IPsec into the stack.

3. Not all of the above approaches are feasible in all hosts. But it was assumed that for each approach, there are some hosts for whom the approach is feasible.

For each of the above 3 categories, there are IPv4 and IPv6, AH transport and tunnel modes, and ESP transport and tunnel modes -- for a total of 24 cases (3 x 2 x 4).

Some header fields and interface fields are listed here for ease of reference -- they're not in the header order, but instead listed to allow comparison between the columns. (* = not covered by AH authentication. ESP authentication doesn't cover any headers that precede it.)

IPv4	IPv6	IP/Transport Interface (RFC 1122 -- Sec 3.4)
Version = 4	Version = 6	
Header Len		
*TOS	Class,Flow Lbl	TOS
Packet Len	Payload Len	Len
ID		ID (optional)
*Flags		DF
*Offset		
*TTL	*Hop Limit	TTL
Protocol	Next Header	
*Checksum		
Src Address	Src Address	Src Address
Dst Address	Dst Address	Dst Address
Options?	Options?	Opt

? = AH covers Option-Type and Option-Length, but might not cover Option-Data.

The results for each of the 20 cases is shown below ("works" = will work if system fragments after outbound IPsec processing, reassembles before inbound IPsec processing). Notes indicate implementation issues.

a. Hosts (integrated into IP stack)
   - o AH-transport --> (IP1-AH-Transport-Data)
     - IPv4 -- works
     - IPv6 -- works
   - o AH-tunnel --> (IP2-AH-IP1-Transport-Data)
     - IPv4 -- works
     - IPv6 -- works

o ESP-transport --> (IP1-ESP_hdr-Transport-Data-ESP_trailer)
- IPv4 -- works
- IPv6 -- works
o ESP-tunnel --> (IP2-ESP_hdr-IP1-Transport-Data-ESP_trailer)
- IPv4 -- works
- IPv6 -- works

b. Hosts (Bump-in-the-stack) -- put IPsec between IP layer and network drivers. In this case, the IPsec module would have to do something like one of the following for fragmentation and reassembly.
- do the fragmentation/reassembly work itself and send/receive the packet directly to/from the network layer. In AH or ESP transport mode, this is fine. In AH or ESP tunnel mode where the tunnel end is at the ultimate destination, this is fine. But in AH or ESP tunnel modes where the tunnel end is different from the ultimate destination and where the source host is multi-homed, this approach could result in sub-optimal routing because the IPsec module may be unable to obtain the information needed (LAN interface and next-hop gateway) to direct the packet to the appropriate network interface. This is not a problem if the interface and next-hop gateway are the same for the ultimate destination and for the tunnel end. But if they are different, then IPsec would need to know the LAN interface and the next-hop gateway for the tunnel end. (Note: The tunnel end (security gateway) is highly likely to be on the regular path to the ultimate destination. But there could also be more than one path to the destination, e.g., the host could be at an organization with 2 firewalls. And the path being used could involve the less commonly chosen firewall.) OR
- pass the IPsec'd packet back to the IP layer where an extra IP header would end up being pre-pended and the IPsec module would have to check and let IPsec'd fragments go by.

OR

- pass the packet contents to the IP layer in a form such that the IP layer recreates an appropriate IP header

At the network layer, the IPsec module will have access to the following selectors from the packet -- SRC address, DST address, Next Protocol, and if there's a transport layer header --> SRC port and DST port. One cannot assume IPsec has access to the Name. It is assumed that the available selector information is sufficient to figure out the relevant Security Policy entry and Security Association(s).

o AH-transport --> (IP1-AH-Transport-Data)
- IPv4 -- works
- IPv6 -- works
o AH-tunnel --> (IP2-AH-IP1-Transport-Data)
- IPv4 -- works
- IPv6 -- works
o ESP-transport --> (IP1-ESP_hdr-Transport-Data-ESP_trailer)
- IPv4 -- works
- IPv6 -- works
o ESP-tunnel --> (IP2-ESP_hdr-IP1-Transport-Data-ESP_trailer)
- IPv4 -- works
- IPv6 -- works

c. Security gateways -- integrate IPsec into the IP stack

NOTE: The IPsec module will have access to the following selectors from the packet -- SRC address, DST address, Next Protocol, and if there's a transport layer header --> SRC port and DST port. It won't have access to the User ID (only Hosts have access to User ID information.) Unlike some Bump-in-the-stack implementations, security gateways may be able to look up the Source Address in the DNS to provide a System Name, e.g., in situations involving use of dynamically assigned IP addresses in conjunction with dynamically updated DNS entries. It also won't have access to the transport layer information if there is an ESP header, or if it's not the first fragment of a fragmented message. It is assumed that the available selector information is sufficient to figure out the relevant Security Policy entry and Security Association(s).

o AH-tunnel --> (IP2-AH-IP1-Transport-Data)
- IPv4 -- works
- IPv6 -- works
o ESP-tunnel --> (IP2-ESP_hdr-IP1-Transport-Data-ESP_trailer)
- IPv4 -- works
- IPv6 -- works

****

B.3 Path MTU Discovery

As mentioned earlier, "ICMP PMTU" refers to an ICMP message used for Path MTU Discovery.

The legend for the diagrams below in B.3.1 and B.3.3 (but not B.3.2) is:

==== = security association (AH or ESP, transport or tunnel)

```
        ---- = connectivity (or if so labelled, administrative boundary)
        .... = ICMP message (hereafter referred to as ICMP PMTU) for

                IPv4:
                - Type = 3 (Destination Unreachable)
                - Code = 4 (Fragmentation needed and DF set)
                - Next-Hop MTU in the low-order 16 bits of the second
                  word of the ICMP header (labelled unused in RFC 792),
                  with high-order 16 bits set to zero

                IPv6 (RFC 1885):
                - Type = 2 (Packet Too Big)
                - Code = 0 (Fragmentation needed and DF set)
                - Next-Hop MTU in the 32 bit MTU field of the ICMP6

        Hx   = host x
        Rx   = router x
        SGx  = security gateway x
        X*   = X supports IPsec
```

B.3.1 Identifying the Originating Host(s)

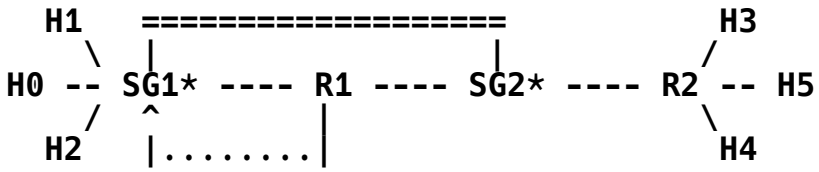
The amount of information returned with the ICMP message is limited and this affects what selectors are available to identify security associations, originating hosts, etc. for use in further propagating the PMTU information.

In brief... An ICMP message must contain the following information from the "offending" packet:

- IPv4 (RFC 792) -- IP header plus a minimum of 64 bits

Accordingly, in the IPv4 context, an ICMP PMTU may identify only the first (outermost) security association. This is because the ICMP PMTU may contain only 64 bits of the "offending" packet beyond the IP header, which would capture only the first SPI from AH or ESP. In the IPv6 context, an ICMP PMTU will probably provide all the SPIs and the selectors in the IP header, but maybe not the SRC/DST ports (in the transport header) or the encapsulated (TCP, UDP, etc.) protocol. Moreover, if ESP is used, the transport ports and protocol selectors may be encrypted.

Looking at the diagram below of a security gateway tunnel (as mentioned elsewhere, security gateways do not use transport mode)...



Suppose that the security policy for SG1 is to use a single SA to SG2 for all the traffic between hosts H0, H1, and H2 and hosts H3, H4, and H5. And suppose H0 sends a data packet to H5 which causes R1 to send an ICMP PMTU message to SG1. If the PMTU message has only the SPI, SG1 will be able to look up the SA and find the list of possible hosts (H0, H1, H2, wildcard); but SG1 will have no way to figure out that H0 sent the traffic that triggered the ICMP PMTU message.

original packet	after IPsec processing	ICMP packet
		IP-3 header (S = R1, D = SG1)
		ICMP header (includes PMTU)
	IP-2 header	IP-2 header (S = SG1, D = SG2)
	ESP header	minimum of 64 bits of ESP hdr (*)
IP-1 header	IP-1 header	
TCP header	TCP header	
TCP data	TCP data	
	ESP trailer	

(*) The 64 bits will include enough of the ESP (or AH) header to include the SPI.
- ESP -- SPI (32 bits), Seq number (32 bits)
- AH -- Next header (8 bits), Payload Len (8 bits), Reserved (16 bits), SPI (32 bits)

This limitation on the amount of information returned with an ICMP message creates a problem in identifying the originating hosts for the packet (so as to know where to further propagate the ICMP PMTU information). If the ICMP message contains only 64 bits of the IPsec header (minimum for IPv4), then the IPsec selectors (e.g., Source and Destination addresses, Next Protocol, Source and Destination ports, etc.) will have been lost. But the ICMP error message will still provide SG1 with the SPI, the PMTU information and the source and destination gateways for the relevant security association.

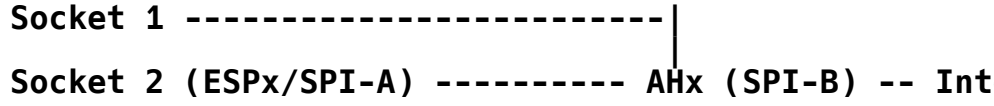
The destination security gateway and SPI uniquely define a security association which in turn defines a set of possible originating hosts. At this point, SG1 could:

a. send the PMTU information to all the possible originating hosts. This would not work well if the host list is a wild card or if many/most of the hosts weren't sending to SG1; but it might work if the SPI/destination/etc mapped to just one or a small number of hosts.
b. store the PMTU with the SPI/etc and wait until the next packet(s) arrive from the originating host(s) for the relevant security association. If it/they are bigger than the PMTU, drop the packet(s), and compose ICMP PMTU message(s) with the new packet(s) and the updated PMTU, and send the originating host(s) the ICMP message(s) about the problem. This involves a delay in notifying the originating host(s), but avoids the problems of (a).

Since only the latter approach is feasible in all instances, a security gateway MUST provide such support, as an option. However, if the ICMP message contains more information from the original packet, then there may be enough information to immediately determine to which host to propagate the ICMP/PMTU message and to provide that system with the 5 fields (source address, destination address, source port, destination port, and transport protocol) needed to determine where to store/update the PMTU. Under such circumstances, a security gateway MUST generate an ICMP PMTU message immediately upon receipt of an ICMP PMTU from further down the path. NOTE: The Next Protocol field may not be contained in the ICMP message and the use of ESP encryption may hide the selector fields that have been encrypted.

B.3.2 Calculation of PMTU

The calculation of PMTU from an ICMP PMTU has to take into account the addition of any IPsec header by H1 -- AH and/or ESP transport, or ESP or AH tunnel. Within a single host, multiple applications may share an SPI and nesting of security associations may occur. (See Section 4.5 Basic Combinations of Security Associations for description of the combinations that MUST be supported). The diagram below illustrates an example of security associations between a pair of hosts (as viewed from the perspective of one of the hosts.) (ESPx or AHx = transport mode)



In order to figure out the PMTU for each socket that maps to SPI-B, it will be necessary to have backpointers from SPI-B to each of the 2 paths that lead to it -- Socket 1 and Socket 2/SPI-A.

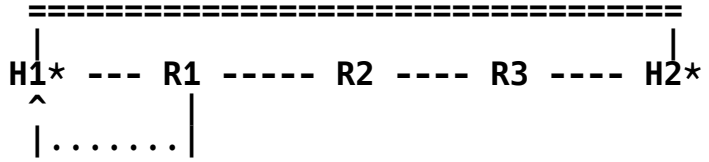
B.3.3 Granularity of Maintaining PMTU Data

In hosts, the granularity with which PMTU ICMP processing can be done differs depending on the implementation situation. Looking at a host, there are three situations that are of interest with respect to PMTU issues:

a. Integration of IPsec into the native IP implementation
b. Bump-in-the-stack implementations, where IPsec is implemented "underneath" an existing implementation of a TCP/IP protocol stack, between the native IP and the local network drivers
c. No IPsec implementation -- This case is included because it is relevant in cases where a security gateway is sending PMTU information back to a host.

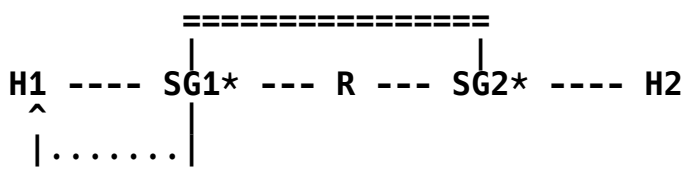
Only in case (a) can the PMTU data be maintained at the same granularity as communication associations. In the other cases, the IP layer will maintain PMTU data at the granularity of Source and Destination IP addresses (and optionally TOS/Class), as described in RFC 1191. This is an important difference, because more than one communication association may map to the same source and destination IP addresses, and each communication association may have a different amount of IPsec header overhead (e.g., due to use of different transforms or different algorithms). The examples below illustrate this.

In cases (a) and (b)... Suppose you have the following situation. H1 is sending to H2 and the packet to be sent from R1 to R2 exceeds the PMTU of the network hop between them.



If R1 is configured to not fragment subscriber traffic, then R1 sends an ICMP PMTU message with the appropriate PMTU to H1. H1's processing would vary with the nature of the implementation. In case (a) (native IP), the security services are bound to sockets or the equivalent. Here the IP/IPsec implementation in H1 can store/update the PMTU for the associated socket. In case (b), the IP layer in H1 can store/update the PMTU but only at the granularity of Source and Destination addresses and possibly TOS/Class, as noted above. So the result may be sub-optimal, since the PMTU for a given SRC/DST/TOS/Class will be the subtraction of the largest amount of IPsec header used for any communication association between a given source and destination.

In case (c), there has to be a security gateway to have any IPsec processing. So suppose you have the following situation. H1 is sending to H2 and the packet to be sent from SG1 to R exceeds the PMTU of the network hop between them.



As described above for case (b), the IP layer in H1 can store/update the PMTU but only at the granularity of Source and Destination addresses, and possibly TOS/Class. So the result may be sub-optimal, since the PMTU for a given SRC/DST/TOS/Class will be the subtraction of the largest amount of IPsec header used for any communication association between a given source and destination.

B.3.4 Per Socket Maintenance of PMTU Data

Implementation of the calculation of PMTU (Section B.3.2) and support for PMTUs at the granularity of individual "communication associations" (Section B.3.3) is a local matter. However, a socket-based implementation of IPsec in a host SHOULD maintain the information on a per socket basis. Bump in the stack systems MUST pass an ICMP PMTU to the host IP implementation, after adjusting it for any IPsec header overhead added by these systems. The determination of the overhead SHOULD be determined by analysis of the SPI and any other selector information present in a returned ICMP PMTU message.

B.3.5 Delivery of PMTU Data to the Transport Layer

The host mechanism for getting the updated PMTU to the transport layer is unchanged, as specified in RFC 1191 (Path MTU Discovery).

B.3.6 Aging of PMTU Data

This topic is covered in Section 6.1.2.4.

Appendix C -- Sequence Space Window Code Example

This appendix contains a routine that implements a bitmask check for a 32 packet window. It was provided by James Hughes (jim_hughes@stortek.com) and Harry Varnis (hgv@anubis.network.com) and is intended as an implementation example. Note that this code both checks for a replay and updates the window. Thus the algorithm, as shown, should only be called AFTER the packet has been authenticated. Implementers might wish to consider splitting the code to do the check for replays before computing the ICV. If the packet is not a replay, the code would then compute the ICV, (discard any bad packets), and if the packet is OK, update the window.

```
#include <stdio.h>
#include <stdlib.h>
typedef unsigned long u_long;

enum {
    ReplayWindowSize = 32
};

u_long bitmap = 0;                  /* session state - must be 32 bits */
u_long lastSeq = 0;                      /* session state */

/* Returns 0 if packet disallowed, 1 if packet permitted */
int ChkReplayWindow(u_long seq);

int ChkReplayWindow(u_long seq) {
    u_long diff;

    if (seq == 0) return 0;              /* first == 0 or wrapped */
    if (seq > lastSeq) {                 /* new larger sequence number */
        diff = seq - lastSeq;
        if (diff < ReplayWindowSize) {  /* In window */
            bitmap <<= diff;
            bitmap |= 1;                 /* set bit for this packet */
        } else bitmap = 1;          /* This packet has a "way larger" */
        lastSeq = seq;
        return 1;                        /* larger is good */
    }
    diff = lastSeq - seq;
    if (diff >= ReplayWindowSize) return 0; /* too old or wrapped */
    if (bitmap & ((u_long)1 << diff)) return 0; /* already seen */
    bitmap |= ((u_long)1 << diff);               /* mark as seen */
    return 1;                            /* out of order but good */
}

char string_buffer[512];
```

```
#define STRING_BUFFER_SIZE sizeof(string_buffer)

int main() {
    int result;
    u_long last, current, bits;

    printf("Input initial state (bits in hex, last msgnum):\n");
    if (!fgets(string_buffer, STRING_BUFFER_SIZE, stdin)) exit(0);
    sscanf(string_buffer, "%lx %lu", &bits, &last);
    if (last != 0)
    bits |= 1;
    bitmap = bits;
    lastSeq = last;
    printf("bits:%08lx last:%lu\n", bitmap, lastSeq);
    printf("Input value to test (current):\n");

    while (1) {
        if (!fgets(string_buffer, STRING_BUFFER_SIZE, stdin)) break;
        sscanf(string_buffer, "%lu", &current);
        result = ChkReplayWindow(current);
        printf("%-3s", result ? "OK" : "BAD");
        printf(" bits:%08lx last:%lu\n", bitmap, lastSeq);
    }
    return 0;
}
```

Appendix D -- Categorization of ICMP messages

The tables below characterize ICMP messages as being either host generated, router generated, both, unassigned/unknown. The first set are IPv4. The second set are IPv6.

IPv4

Type	Name/Codes	Reference
HOST GENERATED:		
3	Destination Unreachable	
	2 Protocol Unreachable	[RFC792]
	3 Port Unreachable	[RFC792]
	8 Source Host Isolated	[RFC792]
	14 Host Precedence Violation	[RFC1812]
10	Router Selection	[RFC1256]

Type	Name/Codes	Reference
ROUTER GENERATED:		
3	Destination Unreachable	
	0 Net Unreachable	[RFC792]
	4 Fragmentation Needed, Don't Fragment was Set	[RFC792]
	5 Source Route Failed	[RFC792]
	6 Destination Network Unknown	[RFC792]
	7 Destination Host Unknown	[RFC792]
	9 Comm. w/Dest. Net. is Administratively Prohibited	[RFC792]
	11 Destination Network Unreachable for Type of Service	[RFC792]
5	Redirect	
	0 Redirect Datagram for the Network (or subnet)	[RFC792]
	2 Redirect Datagram for the Type of Service & Network	[RFC792]
9	Router Advertisement	[RFC1256]
18	Address Mask Reply	[RFC950]

IPv4

Type	Name/Codes	Reference
BOTH ROUTER AND HOST GENERATED:		
0	Echo Reply	[RFC792]
3	Destination Unreachable	
	1 Host Unreachable	[RFC792]
	10 Comm. w/Dest. Host is Administratively Prohibited	[RFC792]
	12 Destination Host Unreachable for Type of Service	[RFC792]
	13 Communication Administratively Prohibited	[RFC1812]
	15 Precedence cutoff in effect	[RFC1812]
4	Source Quench	[RFC792]
5	Redirect	
	1 Redirect Datagram for the Host	[RFC792]
	3 Redirect Datagram for the Type of Service and Host	[RFC792]
6	Alternate Host Address	[JBP]
8	Echo	[RFC792]
11	Time Exceeded	[RFC792]
12	Parameter Problem	[RFC792,RFC1108]
13	Timestamp	[RFC792]
14	Timestamp Reply	[RFC792]
15	Information Request	[RFC792]
16	Information Reply	[RFC792]
17	Address Mask Request	[RFC950]
30	Traceroute	[RFC1393]
31	Datagram Conversion Error	[RFC1475]
32	Mobile Host Redirect	[Johnson]
39	SKIP	[Markson]
40	Photuris	[Simpson]

Type	Name/Codes	Reference
UNASSIGNED TYPE OR UNKNOWN GENERATOR:		
1	Unassigned	[JBP]
2	Unassigned	[JBP]
7	Unassigned	[JBP]
19	Reserved (for Security)	[Solo]
20-29	Reserved (for Robustness Experiment)	[ZSu]
33	IPv6 Where-Are-You	[Simpson]
34	IPv6 I-Am-Here	[Simpson]
35	Mobile Registration Request	[Simpson]
36	Mobile Registration Reply	[Simpson]
37	Domain Name Request	[Simpson]
38	Domain Name Reply	[Simpson]
41-255	Reserved	[JBP]

IPv6

Type	Name/Codes	Reference
HOST GENERATED:		
1	Destination Unreachable	[RFC 1885]
	4 Port Unreachable	

Type	Name/Codes	Reference
ROUTER GENERATED:		
1	Destination Unreachable	[RFC1885]
	0 No Route to Destination	
	1 Comm. w/Destination is Administratively Prohibited	
	2 Not a Neighbor	
	3 Address Unreachable	
2	Packet Too Big	[RFC1885]
	0	
3	Time Exceeded	[RFC1885]
	0 Hop Limit Exceeded in Transit	
	1 Fragment reassembly time exceeded	

Type	Name/Codes	Reference
BOTH ROUTER AND HOST GENERATED:		
4	Parameter Problem	[RFC1885]
	0 Erroneous Header Field Encountered	
	1 Unrecognized Next Header Type Encountered	
	2 Unrecognized IPv6 Option Encountered	

References

[BL73] Bell, D.E. & LaPadula, L.J., "Secure Computer Systems: Mathematical Foundations and Model", Technical Report M74-244, The MITRE Corporation, Bedford, MA, May 1973.

[Bra97] Bradner, S., "Key words for use in RFCs to Indicate Requirement Level", BCP 14, RFC 2119, March 1997.

[DoD85] US National Computer Security Center, "Department of Defense Trusted Computer System Evaluation Criteria", DoD 5200.28-STD, US Department of Defense, Ft. Meade, MD., December 1985.

[DoD87] US National Computer Security Center, "Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria", NCSC-TG-005, Version 1, US Department of Defense, Ft. Meade, MD., 31 July 1987.

[HA94] Haller, N., and R. Atkinson, "On Internet Authentication", RFC 1704, October 1994.

[HC98] Harkins, D., and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[HM97] Harney, H., and C. Muckenhirn, "Group Key Management Protocol (GKMP) Architecture", RFC 2094, July 1997.

[ISO] ISO/IEC JTC1/SC6, Network Layer Security Protocol, ISO-IEC DIS 11577, International Standards Organisation, Geneva, Switzerland, 29 November 1992.

[IB93] John Ioannidis and Matt Blaze, "Architecture and Implementation of Network-layer Security Under Unix", Proceedings of USENIX Security Symposium, Santa Clara, CA, October 1993.

[IBK93] John Ioannidis, Matt Blaze, & Phil Karn, "swIPe: Network-Layer Security for IP", presentation at the Spring 1993 IETF Meeting, Columbus, Ohio

[KA98a] Kent, S., and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.

[KA98b] Kent, S., and R. Atkinson, "IP Encapsulating Security Payload (ESP)", RFC 2406, November 1998.

[Ken91] Kent, S., "US DoD Security Options for the Internet Protocol", RFC 1108, November 1991.

[MSST97] Maughan, D., Schertler, M., Schneider, M., and J. Turner, "Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2408, November 1998.

[Orm97] Orman, H., "The OAKLEY Key Determination Protocol", RFC 2412, November 1998.

[Pip98] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP", RFC 2407, November 1998.

[Sch94] Bruce Schneier, Applied Cryptography, Section 8.6, John Wiley & Sons, New York, NY, 1994.

[SDNS] SDNS Secure Data Network System, Security Protocol 3, SP3, Document SDN.301, Revision 1.5, 15 May 1989, published in NIST Publication NIST-IR-90-4250, February 1990.

[SMPT98] Shacham, A., Monsour, R., Pereira, R., and M. Thomas, "IP Payload Compression Protocol (IPComp)", RFC 2393, August 1998.

[TDG97] Thayer, R., Doraswamy, N., and R. Glenn, "IP Security Document Roadmap", RFC 2411, November 1998.

[VK83] V.L. Voydock & S.T. Kent, "Security Mechanisms in High-level Networks", ACM Computing Surveys, Vol. 15, No. 2, June 1983.

Disclaimer

The views and specification expressed in this document are those of the authors and are not necessarily those of their employers. The authors and their employers specifically disclaim responsibility for any problems arising from correct or incorrect implementation or use of this design.

Author Information

Stephen Kent
BBN Corporation
70 Fawcett Street
Cambridge, MA 02140
USA

Phone: +1 (617) 873-3988
EMail: kent@bbn.com

Randall Atkinson
@Home Network
425 Broadway
Redwood City, CA 94063
USA

Phone: +1 (415) 569-5000
EMail: rja@corp.home.net