Comparison of Network-Level Security Protocols

Mark H. Linehan
IBM T. J. Watson Research Center, Hawthorne, NY
linehan@watson.ibm.com

This paper discusses the concept of applying cryptographic techniques at the network layer of a communications system, and reviews and compares several proposals that have been made to the IPSEC working group of the IETF.

Introduction

In late 1992, the Internet Engineering Task Force chartered an *Internet Protocol Security Protocol* Working Group (IPSEC) to "... develop mechanisms to protect client protocols of IP" [10]. The objective was to provide "... authentication, integrity, access control, and confidentiality" at the network layer. In the ~18 months since then, several schemes have been proposed to address this objective. This paper reviews those proposals for which documentation is readily available.

The proposals discussed here focus on mechanisms for adding security functions to the network layer. In most cases, key management and access policy are provided by higher layer application. This paper addresses only the security functions in the network layer.

Outline of Paper

The paper first defines security-related terms as background for the remaining discussion. The paper then describes common aspects of the network layer security proposals, before discussing the details of each proposal. The paper then discusses the generic question of why encryption might be used at the network layer as opposed to other layers.

Definitions

A variety of terminology is utilized for security technology. For clarity, the key terms used in this paper are defined here.

Access Control

Access control is the application of policies such as requiring that "Top Secret" data be encrypted, or that unencrypted data may be transmitted between a given pair of end systems, or that electronic purchase orders may only be placed by authenticated users who are included in a particular list.

Bypass Traffic

Bypass traffic is communication that avoids security mechanisms. Whether bypass traffic is permitted is an access control policy issue. Some implementations may require bypass traffic for purposes such as key management.

Message Privacy

Privacy, or message confidentiality, is ensuring that only the intended recipients can read messages. Encryption addresses this by scrambling messages such that only those that have the right key can make sense of the message contents.

Message Integrity

Message integrity is guaranteeing that the bits received are exactly the bits sent. One way to achieve this is to take a checksum of the message data, and encrypt the result. Only those that have the right key can modify the message and then adjust the checksum to match.

Padding

Padding is the addition of extra bytes to a message for any of several purposes:

1. To extend data to the next cryptographic block boundary.
2. To align data fields for convenient or more efficient processing.
3. To frustrate some forms of traffic analysis.

Reflection Protection

Reflection protection is a form of message integrity that specifically addresses the possibility that a message may be transmitted ("reflected") back to its origin. (In the author's opinion, this should not require special consideration in most cases. Equivalent protection is automatically achieved if the destination address of a message is included in the protected portion of the message, or if different keys are used for each direction of communication.)

Replay Protection

Replay protection is a form of message integrity checking that protects against the possibility that an attacker might duplicate ("replay") packets sent to a recipient. Typically, this is achieved by placing a sequence number or timestamp in the protected portion of each packet.

Signatures

Electronic signatures are the analogue of paper signatures: they prove that something was acknowledged by someone, perhaps as of a particular date.

Source Authentication

Source authentication is proving the origin of a message to the recipient. This is implicit when symmetric-key encryption is used and keys are distributed pair-wise.

It is important to differentiate several types of message sources: users (as in e-mail from someone), processes or services (as in a file server), machines (as in a particular workstation), and locations (as in a specific employer). Access control policies may be keyed to the type of message source as well as to particular sources.

Security Labels

Governments and many corporations label material with indications such as "Top Secret". They may wish messages to carry such labels to indicate to recipients the desired handling of message contents.

Traffic Analysis

Traffic analysis is the process of generating hypotheses about message contents from the observed patterns of messages transmitted or received. Traffic analysis may be frustrated by various techniques such as padding messages, generating fake messages, and hiding the sending and receiving addresses in messages.

Truncation Protection

Truncation protection is a message integrity service that specifically ensures that the last message(s) of a communication are not lost. (Usually, this function is provided at the transport or session layer, rather than as a security service.)

Common Aspects of Network Layer Security Proposals

The various schemes proposed to add security functions to the network layer have many points in common. For clarity and brevity, these common aspects are described here.

Motivation

The network layer, unlike the other layers, has access to all packets for all sources and destinations -- including packets that are routed through intermediate systems. This creates the opportunity for several security configurations.

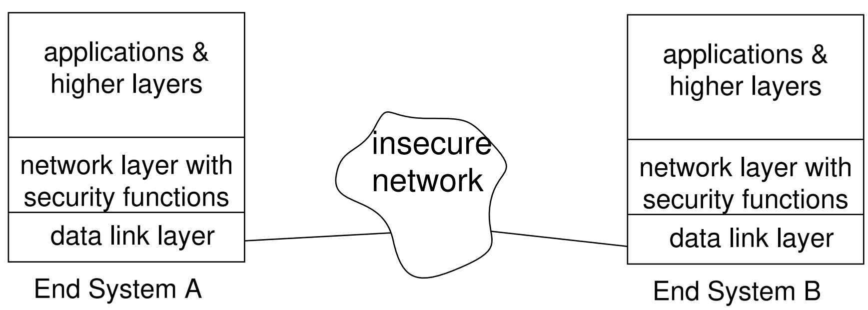

Figure 1: End-to-end security through an insecure

Figure 1 shows how two end systems can employ security functions at the network layer to gain end-to-end security through an insecure network. Message privacy, message integrity, source machine authentication, and access control can all be achieved in this manner. The key point is that no changes are needed to the intermediate network, since the packets generated by the end systems are in network format. The fact that they contain various additional fields to support the security functions is completely transparent to the network.

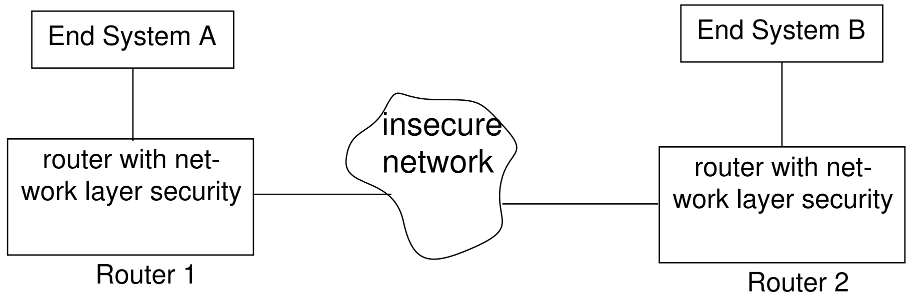

Figure 2: Secure communications provided by “tunneling” through routers.

When implemented in intermediate systems, network-layer security can be provided on behalf of end systems which themselves do not contain security functions. For example, in figure 2, end systems A and B communicate via (trusted) routers 1 and 2. The routers provide security for those packets that traverse the insecure network. The data travels in the clear between the end systems and the routers, which encapsulate the packets for transmission over the network, and decapsulate them before forwarding them to the peer end system.

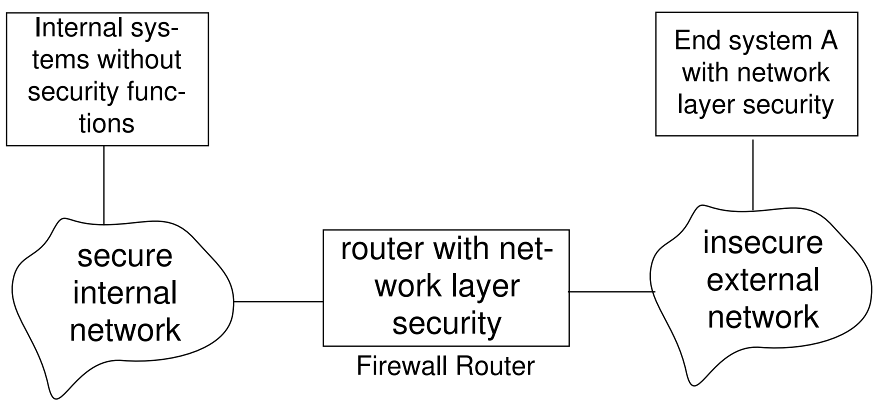

Figure 3: Firewall Router using Network Layer Security.

In another possible configuration, as shown in figure 3, a router can provide access control for a secure internal network. In this role, a router utilizes access control functions embedded in the network layer to provide a “firewall” function for a protected network. End system A gains access to the internal systems on the secure network by authenticating itself to the firewall router. Other external systems either cannot authenticate themselves to the firewall router or do not have access permission in the router. No security function need be incorporated in the internal systems themselves.

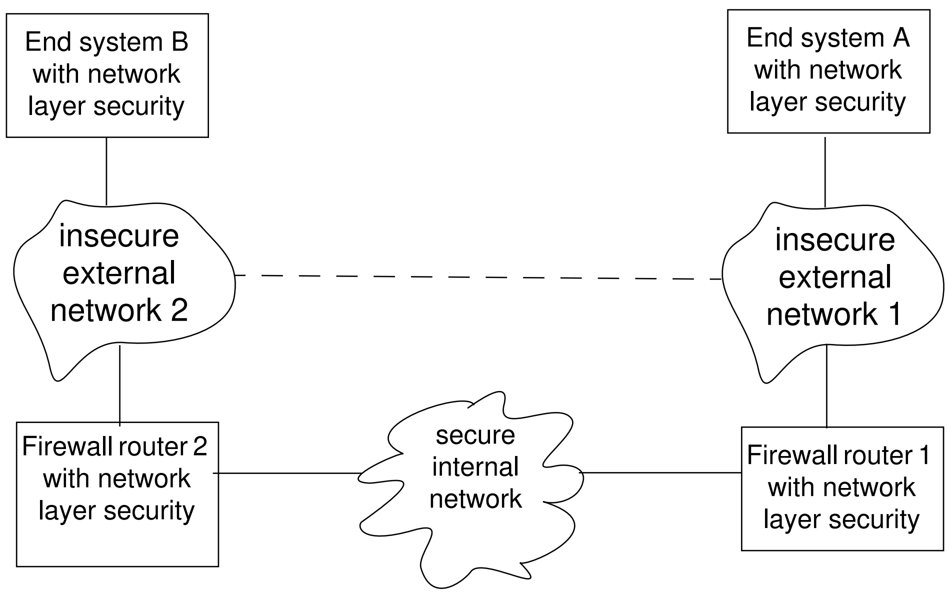

Figure 4: Communication through multiple firewall routers.

One can also imagine more complex configurations. Figure 4 shows an example in which end systems A and B communicate through insecure networks and firewall routers to a secure internal network. When the two end systems communicate with each other, the messages pass through both firewall routers. For example, a message originating at A would be encrypted at A, decrypted at firewall router 1, re-encrypted at router 2, and decrypted a second time at end system B.

One might ask why - in this example - one wouldn’t establish a direct link between the two insecure networks, as shown in the dashed line. The practical answer is that economics, access policy issues, or application configurations may preclude the use of such a link even if it already existed.

One might also ask why end system A might not interact with firewall router 2 when communicating to system B. This would involve routing packets through router 1 without utilizing the firewall functions of router 1. In other words, router 2 would operate as a firewall router on behalf of end system A. The answer is that this mode of operation is certainly possible. The choice among these is determined by access policy issues, not by the technology.

Functions Provided

Security functions implemented at the network layer can include message privacy, message integrity, source machine or network authentication, access control, reflection protection, security labels, padding, and methods of avoiding traffic analysis. Since the network layer doesn't have visibility to the contents of packets, the same security processing must be applied to all messages sent to a given destination. Hence there is no way for the network layer to provide application-specific security functions such as encrypting only the most sensitive parts of e-mail.

Some network layer schemes provide a way for the higher layers to explicitly associate a security label with each packet. When this is provided, the label contents can indicate which security functions should be provided. This provides a way for applications to select which security functions should be implemented at the network level.

Typically, there is a need to provide for bypass traffic for purposes such as key management and host name resolution. Some implementations accomplish this simply by assigning different destination addresses to the servers that provide these functions. Other mechanisms provide an explicit method of bypassing the network layer security functions in the outbound path of end systems.

Packet Encapsulation

The principal technique applied by network layer security is packet encapsulation, as shown in figure 5. Clear-text packets (coming from a network interface or higher layers) are encrypted and then enclosed in an outer network header that is used to route the packet through the network. At the peer system, the outer header is stripped off, and the inner packet is decrypted and forwarded to its destination. The latter may be the local system or may be reached by further communication through the network.

outer packet header	clear-text security header	protected security header	inner (protected) packet

Figure 5: Typical network layer security packet format.

Note that destination address of the inner and outer packets may differ. For example in figure 4, packets generated by system B have system A's address in the destination field of the inner packet, but have the address of firewall router 2 in the outer packet. The firewall router decapsulates the inner packet and forwards it according to the destination address of the inner packet - in this case, to firewall router 1, which re-encapsulates the packet before forwarding it to system A.

The clear-text security header identifies the fact that network layer security has been applied to the packet, indicates what security functions (encryption, authentication, etc.) have been applied, and contains a key identifier. The protected security header contains encrypted and/or integrity-checked fields such as a packet sequence number or an authenticator. The details differ among the various network layer security mechanisms.

Function Placement

In order to support the configurations shown above, security functions are added to both the inbound and outbound packet processing paths of the network layer, as shown in figure 6. Network layer security components are shaded.

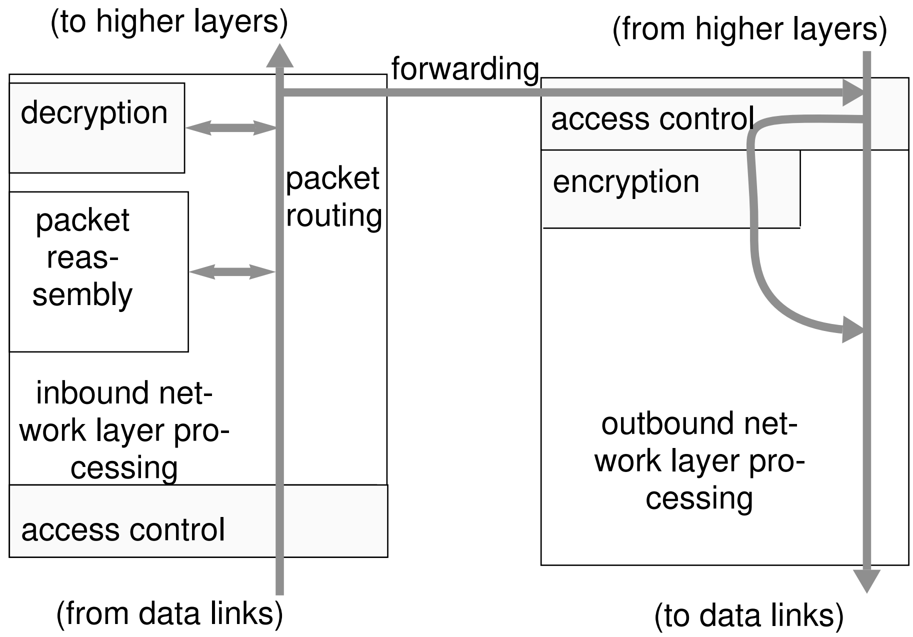

Figure 6: Security functions in the network layer.

The left-hand side of figure 6 shows the network layer's inbound processing path. All incoming packets pass through the access control function, which determines whether each packet should be accepted or ignored according to the access policy and the source address or other information in the packet header.

Accepted packets fall into any of three cases:

Packet destination	**Packet has security header?** no	yes
remote	1	2
local	1	3

If the incoming packet does not have a security header (case 1), it is routed to its local or remote destination if the access control policy permits non-secured packets from the source address given in the packet. If the packet has a security header and the destination is remote (case 2), the packet is also routed according to the security policy.

In the case where the packet destination is local, the packet is processed through several steps:

1. If the packet is fragmented, it must be reassembled before it can be decrypted or integrity checked.
2. The inner packet is decapsulated.
3. Security processing (decryption, integrity checking, source authentication checking) is applied.
4. If the inner packet has a remote destination address, the packet is resubmitted to the network layer packet routing function. If the inner packet has a local address, the packet is passed up to higher layers.

Note that a packet may be routed twice within a single system: once based on the destination address of the outer packet, and a second time according to the destination specified in the inner packet.

The outbound direction is much simpler. Packets are processed through access control to determine whether they require security processing. Then encryption and related functions are applied and packets are encapsulated in outer packets. The destination address for the outer packets is determined according to the access policy table. Packets are submitted for packet forwarding according to the destination addresses in the outer packets.

Key Management and Access Control Policies

Network layer security functions implement, but do not determine, key management and access control policies. These policies are provided by higher-level functions to the network layer in some tabular form. For example, when the outbound encryption function needs a key, it access a key table that is indexed by destination address or security label. When the inbound access control function needs to determine whether to accept an incoming packet, it might reference a policy table indexed by the source address and/or key identifier in the packet.

Separating key management and access control policy mechanisms from the network layer minimizes the function that must be added to that layer. Relegating them to higher layers provides the flexibility to support many different key management techniques and access policies.

Cryptographic Algorithms

Network layer security protocols generally provide for the possibility of having multiple encryption and integrity checking algorithms. Typically, a field in the clear-text security header indicates which algorithms have been applied to a specific packet.

Transparency

One great advantage of network layer security is that it can be provided without requiring changes to applications, to any other communications layers, or to network components that do not need the security functions. The disadvantage is that (in the absence of per-packet security labels) the network layer cannot discriminate between packets that belong to different applications. Hence it is constrained to use the same encryption keys and access policies for all packets destined to the same address. This may not provide the function desired, and can be costly in performance.

Performance

Network layer security functions are almost necessarily implemented in software since the network layer itself generally is all software. This makes it hard to make effective use of cryptographic hardware: the bits to be encrypted must be read from and rewritten to a buffer, so performance is necessarily limited to memory cycle times. Furthermore, the encryption is an additional step added to the network layer processing path. This is unlike encryption applied at the link layer, where the bits can be encrypted as they are transmitted and received, with no additional memory references. Hence, the security functions are likely to set a limit on the performance of the network layer.

SP3 - Security Protocol 3

SP3 is a network-layer security mechanism defined by the National Security Agency (NSA) and the National Institute of Science and Technology (NIST) as part of the “Secure Data Network System” (SDNS) suite of security protocols [17]. It uses encryption to provide message privacy and integrity at the network layer of the connectionless version of the OSI network protocol.

As shown in figure 7, SP3 is a software component that exists at the top sublayer of the network layer. In OSI terms, SP3 is a subnetwork independent convergence protocol (SNICP). To the transport layer, SP3 provides a standard OSI network service with a few additional interface options. To the other network sublayers, SP3 looks like a SNICP -- the component that actually provides networking functions such as routing. Messages generated by the transport layer are processed by SP3 before they are passed to the lower network sublayers. At another end-system, incoming messages pass through SP3 before they reach the transport layer.

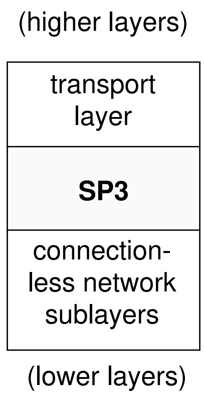

Figure 7: SP3.

Functions

SP3 provides message privacy, message integrity, access control, padding, reflection protection, and security label processing. In the outbound direction, encryption keys and access policies are selected by the destination NSAP address and security label (if any) associated with an NSDU. In the inbound direction, keys and access policy are selected by the source NSAP address, the key-id, and the security label contained in the NPDU.

The access control policy is fixed at the network layer: communication between pairs of systems is prevented if an appropriate key is not available. Message integrity processing is automatically imposed in most circumstances.

Higher-layer SDNS key management and access control services are specified in detail in [15] and [16].

Higher layers can specify, as Quality of Service options, the type of security processing desired for each NSDU. This makes it possible for the security processing to be optimized to the requirements of individual applications.

Packet Formats

SP3 specifies four different packet formats:

SP3-N is used between two end-systems. Neither source nor destination address is included in the encapsulated packet. (I think this is why reflection protection is provided as an explicit service in SP3.)

SP3-A is used when two end-systems communicate via a network. The encapsulated packet carries source and destination addresses that can be used for routing at intermediate systems and source authentication at destination end systems.

SP3-I is like SP3-A, except that an encapsulated CLNP header is included.

SP3-D is like SP3-A, except that an encapsulated IP header is included.

In keeping with the usual OSI packet format, all the SP3 fields are variable-length, and may be presented in an NPDU in any order. Naturally, this flexibility can considerably complicate implementations and has a significant performance cost.

Network Configurations

SP3-N can only be used in the type of end-to-end communications shown in figure 1. The other three NPDU formats can support operations with intermediate systems.

As shown in figure 8, SP3 can be interfaced to other types of SNICP’s. This ability is limited by the lack of standardization of the routing and relay function in OSI networking. It is not clear whether

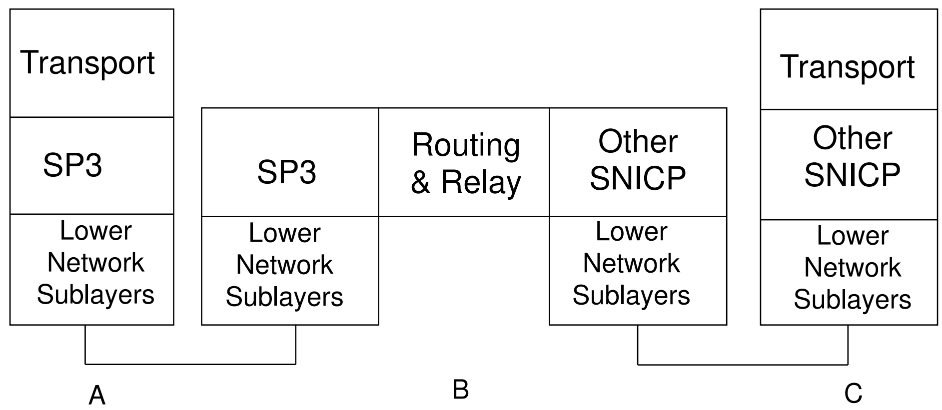

Figure 8: SP3 and the network layer (from page 9 of [17]).

every configuration shown in figures 1 through 4 are actually possible.

SP3-A operates only on complete NSDU's. When SP3-A is used, if an NPDU is fragmented in the path between C and B, it must be reassembled at B before SP3 can process it.

When SP3-I is used, SP3 can carry individual CLNP fragments in separate NPDU's from B to A. In this case, the implementation of SP3 at B must be intimately tied into the rest of the CLNP processing.

When SP3-D is used, the network components between B and C can use IP. In this mode, individual IP fragments can also be forwarded by SP3 from B to A. This means that the SP3 component in B must be integrated into the IP fragmentation processing, rather than existing as a separate function above the IP code as implied by figure 7.

Cooperating Families

SP3 also defines the concept of "cooperating families" of intermediate systems. It addresses the possibility that there may be multiple "firewall" routers providing parallel paths between networks. It ensures that all the members of the family share keys so that a packet can be decrypted by any member of the cooperating family.

It requires that packets not be fragmented since the fragments could be routed through different routers.

In this author's opinion, the fact that packet fragmentation must be suppressed makes this scheme impractical.

I-NLSP: Integrated Network Layer Security Protocol

I-NLSP is a protocol based on the ISO 11577 draft international standard called *Network Layer Security Protocol* (NLSP) [12]. According to [14], NLSP (and hence I-NLSP) is an incompatible descendent of SP3. The primary objective of I-NLSP is to "... provide a set of security services for both IPv4 and CLNP (page 1 of [1]).

The fundamental concept of I-NLSP is that it can accept as input either an IP or CLNP packet, encapsulate it, add a security header, and then transmit the result on either an IP or CLNP network. Of course a real implementation would probably generate either IP or CLNP outer packets, not both. But the concept is that a single integrated protocol definition can cover both the IP and CLNP worlds.

Functions

I-NLSP provides message privacy, message integrity, source authentication, access control, padding, security labels, reflection protection, and methods of frustrating traffic analysis. The function is roughly similar to that of SP3, although many of the details are different.

One difference from other security protocols is that three distinct padding fields are defined for three specific purposes: padding to encryption block lengths, padding to integrity check block lengths, and padding to frustrate traffic analysis. The RFC editor sensibly comments that only one of these is really needed.

I-NLSP includes a capability to perform "in-band" key and access policy negotiation.

Packet Format

The I-NLSP packet is formatted in the usual OSI network fashion, with multiple type-length-value fields. Parsing the packet is clearly quite complex; in fact the RFC editor proposes that many of the length fields be given fixed values to simplify preparation of I-NLSP packets. The RFC doesn't say whether implementations must fully parse incoming packets, or may reject packets without the expected constant length field values.

swIPe

swIPe is an IP network layer security mechanism that has been proposed to the IETF in [8] and [9]. At least two implementations of swIPe exist: one is described in [7], and another is mentioned but not described in any detail in [13]. This makes it possible to review swIPe from both an architectural and implementation viewpoint.

The swIPe implementation of [7] was made available on 15 June 1994 via anonymous ftp from ftp.csua.berkeley.edu in the file /pub/cypherpunks/swIPe/swipe.tar.Z. This implementation was not reviewed during the preparation of this paper.

Functions

swIPe provides message privacy, message integrity, source network address authentication, replay protection, and padding. It provides a mechanism for identifying bypass traffic.

In the outbound direction, the key and access policy is selected by the destination address, IP protocol, and other transport-layer information available at the IP level. In the inbound direction, the key and access policy are identified by a "policy identifier" field in the clear header, and by the source address and other data in the IP packet.

The protected security header contains a sequence number used to defeat replay attacks.

Function Placement

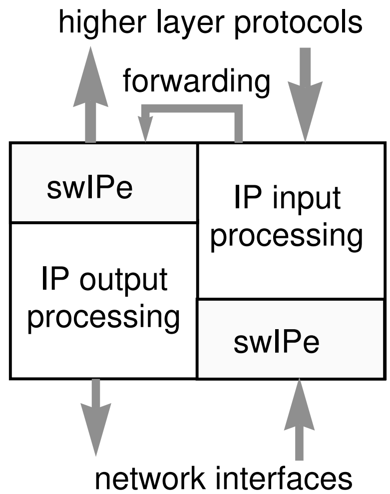

Figure 9: swIPe architecture (from page 3 of [7]).

As shown in figure 9, inbound swIPe processing occurs as incoming packets are received from the data link layer. This permits swIPe to implement access control on inbound packets. swIPe packets are recognized by a unique IP protocol number.

What is not clear in any of the documentation is how swIPe handles fragmented packets. Of course, this is an implementation issue, not an architectural problem.

Modularity

The swIPe architecture defines three conceptual “engines”: the protocol engine, the key engine, and the policy engine. The protocol engine is the function added to the network layer to handle incoming and outbound packets. The key management and policy engines are invoked by the protocol engine when the latter requires keys or access policy definitions.

Implementation Characteristics

Reference [7] describes an implementation of swIPe in some detail. The protocol engine is incorporated in the IP network layer, which operates in the Unix kernel. The key and policy engines are implemented as regular user processes. This makes it practical to implement as sophisticated a key management or access policy as might be desired. In particular, either engine can negotiate with its remote peer.

The interface between the protocol engine and the key and policy engines is provided by an “upcall” mechanism and special ioctl calls. The protocol engine caches the information provided by the other two engines to improve performance. If the key or policy entry for a packet is not found in the cache, the packet is forwarded to the key or policy engine (as required) and is otherwise dropped. swIPe depends upon higher-layer communications functions to retransmit such discarded packets.

Performance

Reference [7] reports performance measurements taken under SunOS on a SparcStation-2. The fixed overhead per packet is 100 microseconds. The MD5 integrity checking takes about 1 microsecond per byte, and DES encryption requires about 10 microseconds per byte. TCP throughput without swIPe is reported at 6000kbps. With MD5, the throughput is 3400kbps, and with DES, the throughput is 440kpbs.

These numbers are quite encouraging. The fixed overhead is reported as half the base overhead for a small packet. The integrity checking cost is also quite reasonable. The DES overhead is significant, but throughput with DES is still comparable to typical distributed file system performance.

SIPP: Simple Internet Protocol plus

SIPP is one of several candidates proposed to the IETF for the next generation of IP (IPng) [6]. The definition incorporates mechanisms for both message integrity and privacy functions provided at the network layer [3]. Packets contain IP option fields that define which of these functions should be applied for incoming packets.

SIPP Authentication

SIPP uses the term “authentication” to refer to both message integrity and source authentication. The SIPP authentication function is described in detail in [1]. An “authentication header” or “payload” is added as an IP security option field. The header contains an integrity check value and a key identifier (called a Security Association ID, or SAID).

Calculation of the integrity check is complicated and likely to be a source of incompatibilities. The integrity check is calculated “...

over the invariant portions of the entire SIPP datagram" (page 2 of [1]), but these are not explicitly specified. An optional SIPP field called the Hop-by-Hop header contains fields that may be selectively included in the integrity check as identified by a bit flag. If source routing is specified in the packet, the route must be reversed by the sender for the integrity check calculation so that the route is included as it would be by the receiver. This latter feature deals with the possibility of a host masquerading attack at the cost of some complexity.

The specification specifies a way to use MD5 for authentication, but also permits implementations to include other authentication algorithms.

SIPP Privacy

SIPP provides for message privacy via an "Encapsulating Security Payload" [2]. The technique involves encapsulating either an entire SIPP packet or only the higher-layer data from a packet, much like SP3. DES is required of all implementations, but other encryption functions may also be used.

A sequence number is included in the encrypted header to protect against replay attacks.

Summary

In style, the SIPP security features are much like swIPe.

Unlike other schemes, the authentication and privacy functions of SIPP each use optional packet fields. When both authentication and privacy are used, the authentication header may be placed inside the encrypted payload, or may be in the cleartext section, or both. The order of processing of authentication versus encryption is implied by the placement of the authentication header.

Because authentication and privacy are carried in separate "payloads", they have separate SAID's (key id's). This gives the option of using different SAID values for each function. This may have a benefit in simplifying the administration of SAID's.

PIPS: Practical IP Security

PIPS [4] is yet another approach to adding security functions to IP. Like swIPe, it defines a new IP protocol number; hence incoming packets are recognized for PIPS processing by the IP protocol number field.

PIPS provides message privacy and integrity, padding, security labels, and data compression. The provision of compression is unique to PIPS; it caters to the needs of slow-speed links as might be found with mobile and dial-up data links. (To be effective, compression must be performed before encryption. Hence, if encryption is implemented at the IP level, then any compression must happen at no lower layer.)

PIPS defines three varieties of Security Association Identifiers, or SAID's: global, dictated, and negotiated. Global SAID's are standardized and provide for two types of typical access policies. Dictated SAID's are key id's that are unique with respect to the packet's source address; these are good for multicast communication. Negotiated SAID's are identification numbers that are mutually agreed by two communicating parties, and are unique with respect to the destination address.

Unlike other schemes, the source and destination addresses contained in the encapsulated packet may be specified in a number of formats: as DNS resource records containing signed keys; as DNS host names, as DNS account names; as RSA public keys; and as X.509 certificates. This provides for considerable flexibility in the protocol implementation.

The protocol provides for negotiation and utilization of a choice of algorithms for encryption, authentication, and compression. The default encryption algorithm is RSA, which is extremely compute intensive.

Security label support is referenced in the (incomplete) draft, but is not described.

Unlike the other mechanisms, PIPS defines in detail how peers may negotiate SAIDs and algorithm choices. This considerably complicates the protocol specification and reduces the modularity of the definition. To help with this complexity, PIPS defines four implementation levels that contain various subsets of PIPS function.

PIPS calls for a new ICMP error code to indicate when a receiving system is unwilling to accept a packet that was not secured using PIPS.

Security at other Layers

Traditionally, security functions have been placed at either the presentation or data link layer. So it is appropriate to review the options at these other layers, and compare them to what can be achieved at the network layer.

Data Link Layer

Encryption at the data link layer is simply encrypting each byte of data at the point where it leaves a communicating station for transmission over the physical media to another station; decryption is performed at the equivalent point on the receiving end. This ensures that clear text cannot be intercepted on the link.

The advantage of encryption at this layer is that it can be applied to all data without changes to applications or any of the higher communications layers. For example, encrypting modems can provide link security with zero change to communicating stations.

Another advantage of encryption at the data link layer is that it is relatively easy to incorporate cryptographic hardware directly in the data transmission and reception paths. This minimizes performance issues.

A third advantage of encryption at the data link layer is that it can work well with data compression implemented in the same layer or any higher layer.

The primary disadvantage of link layer encryption is that it only applies between two directly-connected points. Systems communicating over multiple-hop networks care about end-to-end security; encryption on individual links does not guarantee that the entire path -- including the routers -- is secure.

Another disadvantage is that link layer security has not been provided (to the knowledge of the author) on local area networks. In principle, there is no reason why a LAN interface card could not perform pair-wise encryption with multiple other such cards. It would have to maintain a table of encryption keys indexed by LAN address, and be able to automatically load the appropriate key as needed to process inbound or outbound traffic. However, such cards apparently do not exist on the market -- presumably since much LAN traffic ultimately leaves a local LAN. In other words, the real need is for encryption of traffic across an internet of LAN's and WAN's.

Encryption at the data link layer provides message privacy, and can insure message integrity if the link is frame-oriented, rather than byte-oriented. Source authentication can be provided only at the machine level. Traffic analysis can be frustrated by producing fake traffic during idle periods. Signatures don't make sense at this layer.

Transport Layer

Security can be provided in the transport layer, such as in *SP4* [15]. This permits transport layer peers to communicate securely over insecure networks, in a fashion similar to figure 1. The primary advantage of having the function in the transport layer is that different security policies and keys can be used for different sets of communicating applications. The primary disadvantage is that the "tunneling" scheme shown in figure 2 and the "firewall" configuration of figure 3 are not possible at the transport layer.

Session and Presentation Layer

The functions provided above for the transport layer could also be provided in the OSI session layer, but the author knows of no proposals to do so. Rather, the OSI model [11] calls for encryption and similar functions to be provided at the presentation layer.

Remote procedure call mechanisms can be considered to fit into the session layer. For example, NIS+ [19] provides source authentication, and OSF DCE [18] offers message privacy, integrity, and source authentication functions.

Advantages of providing security functions at these layers are that they are made available to all applications, and they can be selectively invoked by application functions as needed (thus limiting the performance cost). The capability for applications to choose whether to utilize the security functions makes it harder to employ restrictive access policies.

Application Layer

Implementing security functions directly in applications provides the most flexible way to handle individual application requirements. For example, a mail system may need to sign specific portions of outgoing mail. Security functions provided at the lower layers would not, in general, know anything about the structure of mail or which portions should be signed. So there is a good reason for providing security functions in applications, regardless of what capabilities may be provided in the underlying networks.

Conclusions

This author suggests that there is a real value in having security functions located at some lower layer in addition to the application or presentation layers. The increasing use of firewalls shows that there is a strong demand for communications security in addition to what is provided within systems and applications. Short of rewriting many systems and applications, the only feasible design is to provide security functions at one of the lower communications layers. Of the various choices available, only the network layer offers the variety of configurations shown in figures 1-4.

The various protocols reviewed in this paper are more alike than they are different. The basic choice is between an IP-only scheme such as swIPe and a mechanism that tries to integrate IP and CLNP, as with I-NLSP. The fundamental difference is complexity of description and implementation. The greater complexity of I-NLSP is very likely to adversely affect performance.

I-NLSP also provides some additional functionality, such as security label processing. This is attractive since it provides a way for applications to influence the security versus performance tradeoff. However, this function is only meaningful in the CLNP world, where QOS parameters are visible to the network layer. In a typical IP implementation, such parameters don’t exist at the network interface and hence cannot be used to make access policy and key choices. Nevertheless, it would be reasonable to provide some capability in an IP security protocol to accommodate security labels in case an IP implementation chose to support them.

An interesting idea introduced by PIPS is the option of providing data compression. This may be a real advantage for slow-speed data links. However, this implies that the network layer function has knowledge of the characteristics of the data links over which a packet will pass. That violates the standard OSI reference model.

This last point illustrates the fundamental issue in providing security functions at the network level: the tradeoff between the performance costs of software implementations of security and the function provided by placing these features at the network layer. Whether any of these schemes succeed depends upon finding the right balance point in that tradeoff. All these schemes provide the flexibility to choose different balance points. Hence, real success depends as much on implementation characteristics (such as use of hardware assists in routers) as upon protocol design.

References

[1] Atkinson, Randall. *SIPP Authentication Header*. Internet draft file <draft-ietf-sipp-ap-03.txt>, 19 April 1994. (working draft)

[2] Atkinson, Randall. *SIPP Encapsulating Security Payload (ESP)*. Internet draft file <draft-ietf-sipp-esp-03.txt>, 19 April 1994. (working draft)

[3] Atkinson, Randall. *SIPP Security Architecture*. Internet draft file <draft-ietf-sipp-sa-02.txt>, 19 April 1994. (working draft)

[4] Eastlake, Donald E. PIPS: *Practical IP Security*. contained in an incomplete draft sent as an e-mail message to the ipsec mailing list (ipsec@ans.net), dated 31 March 1994.

[5] Glenn, K. Robert. Integrated Network Layer Security Protocol. Internet draft file <draft-glenn-layer-security-00.txt>, 23 September 1993. (working draft)

[6] Hinden, Robert M. *Simple Internet Protocol Plus Working Paper*. Internet draft file <draft-ietf-sipp-whitepaper-00.txt>, 1 February 1994. (working draft)

[7] Ioannidis, John; and Blaze, Matt. *The Architecture and Implementation of Network-Layer Security under Unix*, Proceedings of the 4th USENIX Security Symposium, Santa Clara, Ca., October 1993.

[8] Ioannidis, John; and Blaze, Matt. *The swIPe IP Security Proposal*, Internet draft file <draft-ipsec-swipe-01.txt>, 3 December, 1993. (working draft)

[9] Ioannidis, John; Blaze, Matt; and Karn, Phil. *swIPe: Network-Layer Security for IP*, presentation at 26th IETF, March 1993.

[10] Internet Engineering Task Force, *Internet Protocol Security Protocol Working Group Charter*, late 1992.

[11] International Standards Organization. *Information Processing Systems - Open Systems Interconnection - Basic Reference Model*, international standard 7498, 15 October 1987.

[12] International Standards Organization. *Information Technology - Telecommunications and Information Exchange between Systems - Network Layer Security Protocol,* draft international standard 11577, November 1992.

[13] Lambert, Paul, *Minutes of the IPSEC Working Group from the 28th IETF*, November 1993.

[14] Maughan, W. Douglas. Standards for Computer Systems Security, *An Interoperable Analysis of SDNS SP3 and ISO NLSP*, Proceedings of the Eighth Annual Computer Security Applications Conference, San Antonio, Texas, 30 November to 4 December, 1992, pp 193-201.

[15] National Institute of Standards and Technology. *Secure Data Network System (SDNS) Access Control*, NISTIR 90-4259, February 1990.

[16] National Institute of Standards and Technology. *Secure Data Network System (SDNS) Key Management*, NISTIR 90-4262, February 1990.

[17] National Institute of Standards and Technology. *Secure Data Network System (SDNS) Network, Transport, and Message Security Protocols*, NISTIR 90-4250, February 1990.

[18] Open Systems Foundation. *OSF DCE Application Development Guide Revision 1.0,* Prentice-Hall, 1993, ISBN 0-13-643826-1.

[19] SunSoft, Inc. *Solaris ONC Network Information Service Plus (NIS+): A White Paper,* 91027-0, September 1991.