IP/Secure: Providing Security on Datagram Delivery for Mobile Host Environment

Takeshi Tanida <tanida@jaist.ac.jp>
Yoichi Shinoda <shinoda@jaist.ac.jp>

Abstract

Security issues has long been an area that the Internet has paid least attention to. There were both historical and technical reasons which justified this action, but we are just about to see these justifications to evaporate, as mechanisms for supporting mobile hosts in the Internet are beginning to emerge.

We propose IP/Secure, a modification of the stock IP (Internet Protocol) to provide a network layer security with application-selectable levels of protection. We introduce a new IP option to convey an authenticator for the IP datagram, that is designed to occupy only a few 32 bit words for minimal impact to performance and network throughput. We also propose ISCP (Internet Security Control Protocol), a support protocol used for key management and delivery that is based on the ID-based key distribution scheme.

I. Introduction

Current Internet pays little attention to security issues. This is mainly due to the fact that the Internet was developed as a network for researchers and engineers. Researchers and engineers are more interested in expanding or adding new features to a network than limiting what it can do. Another reason is that the Internet employs hierarchical routing scheme. Why do we care about internetwork security if a physical network that is full of vital information is deep inside of some autonomous network, where no outsider can physically tap the network media, and where internetwork traffic is checked by some packet filtering mechanism? This situation is now changing as mechanisms to support mobile hosts are introduced to the Internet. Over the last three years several proposals have been made to support mobile hosts in the Internet environment [1, 2, 3, 4]. These proposals each differ from one another, but characteristics common to all proposals are that they allow hosts with local network address to exist in some remote network (after all, this is what is intended) and that they allow hosts with foreign addresses (e.g. hosts with addresses that are not otherwise addressable if none of the proposed mechanisms are used) to virtually exist in a local network.

These characteristics will produce severe security problems as soon as they are put into a practical use. Wire (and wireless) tapping the traffic between a mobile host and its home network comes up immediately as a first attempt. If this is possible, various forms of security attacks such as impersonation of the mobile host, playback attack and datagram alteration are not hard to accomplish. Another example is that a protection based on packet filtering mechanisms based on source addresses will have to be re-coded to allow mobile hosts outside of the packet filter to communicate with their home network, or they no longer provide any protection in some cases. Session and application layer authentication mechanisms such as Kerberos [5] and SecureRPC [6] may seem to be an answer, but they all assume secure delivery of datagrams they use. If impersonation of a host is possible, how can we tell that a key distribution server we have connected is the one we intended to connect? It is obvious that the host authentication at the network layer plays a vital role for any authentication mechanisms in the upper layer to function correctly.

In this paper, we propose a modification of the stock IP (Internet Protocol) to provide a low overhead network layer security, together with ISCP (Internet Security Control Protocol) used for key delivery. The design decisions made are biased so that the modified protocol has least impact to an existing system in the Internet.

Organization of this paper is as follows. We will first discuss design decisions made for key management and delivery in section 2. Section 3 describes the authentication procedure using a key obtained by the key delivery mechanism described in section 2, followed by discussions on the implementation design of our proposed security mechanism in section 4. Section 5 provides some considerations and future research directions as well as concluding remarks.

II. Key Delivery

In this section, we discuss the design decisions made for the key delivery mechanism used in our IP authentication mechanism.

II.A. Design of Key Delivery Mechanism

Authentication mechanism for the IP should be capable of authenticate very large number of hosts each having a unique address. There are numbers of ways to distribute public key certificate used in the authentication, such as centralized key distribution in which a set of key servers service requests for retrieval of key for a specified host, or distributed key distribution in which keys are kept in the memory of communicating entities. However, when peer entity is not kept within a small set of hosts, communication costs, memory costs or costs for maintaining security of the distribution mechanism itself is too high for low level entity such as the IP. More over, we have to realize that we are dealing with network layer functionality, so it is not desirable to invoke key management and distribution functions in the upper layer, which is a severe layer violation. We have to avoid chickens and eggs problem, so the centralized key distribution using upper level protocol is obviously not our choice.

We utilize a variation of the ID-based key distribution system (ID-KDS) [7] with IP addresses as IDs to solve this problem.

II.B. ID Based Key Distribution System

There are two different variations to obtain public key certificate from a peer entity when ID-KDS is used, as shown below.

- *ID CERTIFICATE*: Public key certificate is obtained from the peer entity prior to authentication.
- *ID SIGNATURE*: Public key is tacked on to a data itself. PEM [8] uses variant of this method.

Major characteristic differences between these two methods are shown in table 1. In the ID signature method, public key certificate is send together with the data itself, meaning that every IP datagrams must convey public key certificate that is at least several hundred bits long if we choose this method. This is an enormous overhead for a network layer, and there is no space for the current IP datagram to store this data. Decryption of public key certificate for each datagram that is received also requires many CPU cycles which we cannot afford.

Our choice naturally fell into the use of ID certificate method. The ID certificate method has one disadvantage that it can not be used for multicast style communication directly, but it remains the better choice.

Table 1: Comparison of two ID-KDS

	ID CERT	ID SIGN
Pre-Delivery of PK	no	yes
One to Many Auth.	no	yes
Protection of Info.	yes	no
Low Communication Cost	yes	no
Low Processing Overhead	yes	no

II.C. Public Key Certificate Issurance

To utilize the public key certificate key distribution, a public key certificate that is signed by a reliable certificate authority (CA) is required. Once public key certificate is issued, a proving host can deliver a shared secret key to verifying host by using its own public key certificate and its secret key.

Before we go on, let us get familiarlized with symbols and notations used throughout the rest of this paper. They are summarized in table 2. Note that we use DES for conventional cryptosystem and RSA for public key cryptosystem, but the choice may vary. For example, we could use other conventional cryptosystem such as IDEA or FEAL8 instead of DES to work with the shared secret key *WK*.

Table 2: Symbols used in key distribution

Symbol	Meaning
VP	Validity Period of a Public Certificate
IP_X	Network Address of a host X
WK	Shared DES key
SK_X	RSA Secret Key for a host X
PK_X	RSA Public Key for a host X
$(M)^K$	RSA Encr. of a Message M with Key K

The procedure to issue a public key certificate by CA to a host is illustrated in figure 1(a).

CA keeps its own secret key SK_C and make corresponding public key PK_C available to the public. The procedure for issuing a public key certificate is carried out as follows.

1. A host J willing to receive a public key certificate keeps its secret key SK_J, and sends corresponding public key PK_J and its network address IP_J to CA.
2. The CA adds some data that determines validity period VP to (PK_J, IP_J), and sign (encrypt) it with CA's secret key. Resulting public key certificate $(IP_J, PK_J, VP)^{SK_C}$ is send back to host J.

II.D. Key Delivery Procedures

There are cases that authentication of a peer host need not be symmetric. To deal with these cases, we prepared two different procedures for our key delivery mechanism. When a host A is a prover, and host B is a verifier, this relation is called one-way authentication. When hosts A and B are both prover and verifier at the same time, we call this relation two-way authentication. We describe the procedure for both cases in the following.

One-way Authentication The procedure of key delivery for one-way authentication is illustrated in figure 1(b).

The procedure for one-way authentication is as follows.

Host J		CA
IP_J, PK_J	$\rightarrow$	$(IP_J, PK_J, VP)^{SK_C}$ (signature)
	$\leftarrow$ $(IP_J, PK_J, VP)^{SK_C}$	

(a) Procedure for Public Key Certificat Issuance

Host A		Host B
$(IP_A, PK_A, VP)^{SK_C}$	$\rightarrow$	$(IP_A, PK_A, VP)^{SK_C PK_C}$ (decryption) $\Rightarrow IP_A, PK_A, VP$
$(WK)^{PK_A SK_A}$ (decryption) $\Rightarrow WK$	$\leftarrow$	$(WK)^{PK_A}$

(b) Key Delivery Procedure for One-way Authentication

Host A		Host B
$(IP_A, PK_A, VP)^{SK_C}$	$\rightarrow$	$(IP_A, PK_A, VP)^{SK_C PK_C}$ (decryption) $\Rightarrow IP_A, PK_A, VP$
$\left\{ \begin{array}{l} (IP_B, PK_B, VP)^{SK_C PK_C} \text{ (decryption)} \\ \Rightarrow IP_B, PK_B, VP \\ (WK)^{PK_A SK_B PK_B SK_A} \text{ (decryption)} \\ \Rightarrow WK \end{array} \right\}$	$\leftarrow$	$\left\{ \begin{array}{l} (IP_B, PK_B, VP)^{SK_C} \\ (WK)^{PK_A SK_B} \end{array} \right\}$

(c) Key Delivery Procedure for Two-way Authentication

Figure 1: Key Issuance and Delivery Procedures

1. Host A sends $(IP_A, PK_A, VP)^{SK_C}$ (public key certificate) to host B.

2. Host B calculates $(IP_A, PK_A, VP)^{SK_C\ PK_C} = (IP_A, PK_A, VP)$ to obtain the public key of host A, which is PK_A.

3. Host B generates a secret DES key WK which will be shared with host A, and encrypts its secret key with PK_A and sends resulting $(WK)^{PK_A}$ (sealed secret key) to host A.

Only host A can open and obtain the shared secret key WK by using SK_A.

Two-way Authentication The procedure of key delivery for two-way authentication is illustrated in figure 1(c).

For two-way authentication, one-way authentication is modified to verify a host generating the shared secret key. More specifically, the procedure is as follows.

1. Host A sends $(IP_A, PK_A, VP)^{SK_C}$ (public key certificate) to Host B.

2. Host B calculates $(IP_A, PK_A, VP)^{SK_C\ PK_C} = (IP_A, PK_A, VP)$ and verifies that the public key for the host A is PK_A.

3. Host B then encrypts shared secret key WK with PK_A and its own secret key SK_B, and send resulting $(WK)^{PK_A SK_B}$ (two-way sealed authentication key) together with its $(IP_B, PK_B, VP)^{SK_C}$ (public key certificate) to host A.

4. Host A first obtains the public key for the host B from $(IP_B, PK_B, VP)^{SK_C}$ and decrypts $(WK)^{PK_A SK_B}$ to obtain the two-way authenticated shared secret key.

III. Authentication Procedures

With successful completion of the key delivery by the ID-KDS described in the previous section, two communicating hosts now share a secret DES key WK. In this section, we discuss the datagram authentication procedures in our IP authentication mechanism using the WK.

III.A. Authentication Levels

Protection against impersonation, wire tapping and alteration can be achieved by authenticating datagram origins (IP addresses) and encryption of datagrams that peer hosts are exchanging. However, each peer host or application has its own ideas about the levels of protection they require and the permissible overhead. Therefore, it is desirable to provide a set of authentication procedures with different strength against possible attacks. The IP security mechanism discussed here allows a host to specify a security level of its choice to each end point and provide corresponding protections.

The authentication is done by encrypting of whole or part of datagrams by the shared secret key obtained through ID certificate based key delivery method. Amount of data encrypted depends on the selected security level. Table 3 shows correspondence between the security levels and encrypted fields of a datagram. Boxes with "○" denotes encrypted data, and "(auth)" denotes that the field is encrypted into an appropriate authenticator format.

Table 3: Security Levels and Encrypted Data

Security Level	Encrypted Field		
	Src Addr (auth)	TCP Hdr (auth)	TCP/UDP User Data
1	○		
2	○	○	
3	○	○	○

In the following, we will describe the functionality (degree of protection) and the procedures for each security level.

III.B. Security Level 1

Authentication functionality and procedure between hosts with security level 1 is discussed in the following.

Level 1 Functionality One of the security issues of mobile hosts is present due to the fact that the proposed mechanisms use two different addresses for a mobile host to establish correct delivery of IP datagrams to the mobile host. One of these addresses is a logical address that express the logical extent of a network that the mobile host belongs to, and is permanently assigned to the mobile host. The logical address is the address which the mobile host received its public key certificate from CA. Another is a physical address that is temporary assigned to the mobile host in a remote network. The physical address is used to deliver IP datagrams to the mobile host, so there must be some mechanism to ensure the binding between the logical and the physical address for protection against impersonation using a forged physical address for the mobile host.

In security level 1, the physical address is encrypted using the shared secret key and the result is used as an authenticator to ensure the binding between the physical address and the logical address.

Level 1 Procedure Authentication procedure for security level 1 is illustrated in figure 2. In figure 2, *DG* is a datagram, *g* is a function to extract physical source address from *DG*, *WK* is a shared secret DES encryption key that hosts A and B obtained by ID-KDS public key delivery. *D* is a DES encryption algorithm governed by the DES encryption key *WK* and *S* is a encrypted source address which is an authenticator.

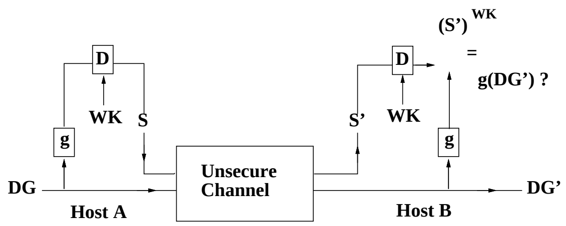

Figure 2: Authentication Procedure (Security Level 1)

Proving host (A) applies the following procedure to outgoing datagram.

1. Physical source address of the datagram is extracted by applying function *g*.
2. The physical source address is encrypted by encryption key *WK*.
3. Encrypted physical source address is attached to the datagram and send to host B.

Verifying host (B) applies the following procedure to incoming datagram.

1. Decrypt authenticator by DES key *WK*.
2. Extract physical source address by applying function *g* to the datagram.
3. Compare the decrypted authenticator and the extracted physical source address.

If the comparison does match, host B can determine that the datagram was originated by host A and pass the datagram to an appropriate upper layer.

Note that this procedure can be used to authenticate the source address of a non-mobile host where logical address and the physical address are identical (in which case we use the only address the non-mobile host was assigned).

Also note that we can also achieve protection against impersonation in the environment in which an un-secure gateway is feeding a false routing information by this mechanism.

III.C. Security Level 2

In security level 2, both level 1 and level 2 procedures are applied. The functionality and the procedure for level 2 is discussed in the following.

Level 2 Functionality Security level 1 supports only a very limited functionality, which is the authentication of a source address of a datagram. It is still possible to perform positive attacks such as impersonation, datagram alteration and play back attack if certain condition is met.

- It is impossible to prevent a host from impersonating another host if the impersonating host can steal an authenticator and is capable of receiving all datagrams destined to impersonated host. This is especially dangerous in the context where mobile host is temporary disconnected from any network segment, because the impersonating host only needs to tap into the last network segment which impersonated host was last recognized and can act as the impersonated host as long as the authenticator is valid.
- Protection against alteration of datagram by intermediate system is not provided.
- Playback attack is always possible.

To solve these problems, level 2 procedure uses an authenticator generated by encrypting and hashing TCP header. It is obvious that incorporating TCP header information in the authenticator results in protection against alteration of TCP header, but it also results in protection against alteration of TCP user data and playback attack for the following reasons.

- Since TCP header contains checksum, if TCP user data is altered, it will be detected in the error correction mechanisms in TCP layer.
- A TCP header contains sequence numbers. Therefore, playback attack using stolen datagram is detected and discarded in the TCP layer as segment having a duplicate sequence number. Playback attack intending to create a new connection with stolen sequences of TCP/IP datagrams will fail because it is unlikely that acknowledgement and window values in TCP header will be the same for the new connection.

Level 2 Procedure Figure 3 shows the authentication procedure for the security level 2. Here, *DG* is a datagram, *g* is a function that extracts TCP header from *DG*, *WK* is a DES key that hosts (A) and (B) obtained by key delivery mechanism, *D* is a DES encryption algorithm which is governed by DES key *WK*, *h* is a regular hash function. and *S* is a encrypted and compressed TCP header, which is an authenticator.

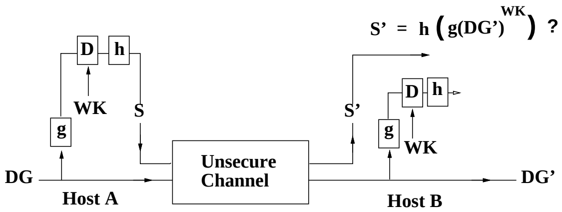

Figure 3: Authentication procedure (Security Level 2)

Proving host (A) applies the following procedure to an outgoing datagram.

1. TCP header of the datagram is extracted using function *g*.
2. The TCP header is encrypted by DES key *WK*.
3. The encrypted TCP header is compressed by regular hash function *h*.
4. The encrypted and compressed TCP header is attached to the datagram and send to the host (B).

Verifying host (B) applies the following procedure to an incoming datagram.

1. A TCP header of the datagram is extracted by using function *g*.
2. The TCP header is encrypted by the DES key *WK*.
3. The encrypted TCP header is compressed by regular hashing function *h*.
4. The encrypted and compressed TCP header is compared with the authenticator.

If the comparison does succeed, host B can determine that the datagram was originated by host A. This comparison also succeeds if the datagram was stolen and a copy is injected into the net, but TCP layer will reject it because TCP header content can not be modified.

Level 2 UDP Peculiarity The use of encrypted TCP header as an authenticator can accomplish highly reliable authentication with relatively low processing overhead. But with level 2 UDP, using encrypted UDP header as an authenticator generally has no meaning because authenticators for a same peer pair will be the same for all datagrams since UDP header does not contain fields that vary from datagram to datagram except when UDP checksum is turned on. Even if checksumming is turned on, we still have a chance to be playback attacked. To make the level 2 UDP as secure as level 2 TCP, we have to perform the following.

- UDP user data is compressed by some hash function, and the resulting value is encrypted.
- Using the resulting data as an authenticator will inhibit alteration of the datagram even if the hash function is known, because the authenticator is encrypted.
- Some form of packet sequencing or time stamping will be necessary to protect against playback attacks. Note that this can not be embedded in the UDP header structure, and it must be encoded as a part of the security protocol header.

This will result in a greater overhead compared to the case with TCP as a upper layer protocol. Further more, protection against of playback attack cannot be accomplished by above procedure, and requires additional mechanisms and overheads to provide it. For this reason, we do not use the procedure for security level 2 if the upper layer protocol is UDP and we fall back to level 1 security in our current implementation.

III.D. Security Level 3

Hosts with security level 3 specified does level 1, level 2 and level 3 authentications. The authentication functionality and procedure for security level 3 is discussed in the following.

Level 3 Functionality With level 2 authentication, we still have a chance for a datagrams to be wire tapped. datagrams. Level 3 authentication solve this problem by encrypting entire user data.

Level 3 Procedure Figure 4 shows the procedure for level 3 datagram authentication. In this figure, *DG* is a datagram, *WK* is a DES key which hosts (A) and (B) obtained by key delivery mechanism, *D* is a DES encryption algorithm governed by DES key *WK*.

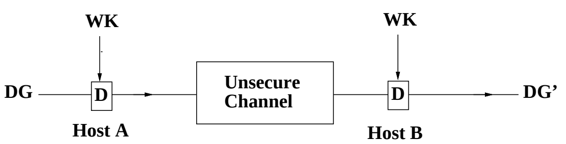

Figure 4: Authentication Procedure (Security Level 3)

Proving host (A) applies the following procedure to an outgoing datagram.

1. User data of the datagram is replaced by encrypted data using DES key *WK*.
2. The datagram is send to host B.

Verifying host (B) applies the following procedure to an incoming datagram.

1. User data of the datagram is replaced by decrypted data using DES key *WK*.

III.E. Summary of Protections

Table 4 summarizes levels of protections that each security level provides. In this table, 'T' stands for TCP and 'U' stands for UDP and denotes that the corresponding functionality is available if the upper layer protocols is the designated type.

Table 4: Summary of Security Levels and Their Functionalities

Functionality (Auth. Func. Level)	Security Level 1	2	3
Impersonation (Level 1)	T/U	T/U	T/U
Impersonation (Level 2)		T	T/U
Datagram Alteration		T	T/U
Playback Attack		T	T
Wiretapping			T/U

IV. IP/Secure

In this section, we will discuss our design for an experimental implementation of the secure IP, the IP/Secure. To implement the IP/Secure, key delivery mechanism and authentication procedures for each security level must be built into the IP module.

In IP/Secure, we introduce the ISCP (Internet Security Control Protocol), a companion protocol that supports session key delivery, in addition to the modification to IP module for authentication mechanism.

IV.A. ISCP

The ISCP is the straight forward implementation of the key delivery mechanism described in section 2.

Protocol Stack in IP/Secure is shown in figure 5. IP module calls ISCP module to obtain encryption keys. Therefore, relationship between IP an ISCP module is analogous to that of IP and ICMP (Internet Control Message Protocol) module. Functionally speaking, relationship between IP/Secure and ISCP can be understood intuitively as analogous to that of IP and ARP (Address Resolution Protocol), because the

FTP,TELNET,SMTP		
TCP,UDP		
IP	⟷	ISCP

Figure 5: Protocol Stack for IP/Secure

encryption key delivered by ISCP is required for datagram transmission under IP/Secure whereas the physical media address delivered by ARP is required for datagram transmission under IP.

ISCP messages are sent as IP client. Format common to all ISCP messages includes 8 bit type field, 8 bit code field that further describes the type field, checksum that contains checksum of the ISCP message, as shown in figure 6. ISCP messages other than ISCP-ERR message which indicates authentication failure have authentication information field. Figure 5 shows general ISCP message structure, and table 6 shows authentication information conveyed by each ISCP message.

0 – 31 (bit)

Type	Code	Checksum
Authenticate Information		

Figure 6: ISCP Message Format

Description of each ISCP message is as follows.

ISCP-KREQ A ISCP-KREQ message is send from an authenticated host to an authenticating host if the encryption key for the authenticating host is not in the ISCP cache.

ISCP-KRPY An authenticating host received either an ISCP-KREQ or an ISCP-KERR message from an authenticated host replies with ISCP-KRPY message to the authenticated host. If one-way authentication is in effect, the ISCP-KRPY message will contain a one-way authentication key (Code = 1), and if two-way authentication is in effect, the ISCP-KRPY message will contain a two-way authentication key (Code = 2).

ISCP-KERR An ISCP-KERR message is generated when an authenticating host received an ISCP-KRPY message (Code=1) from an authenticated host when two-way authentication is in effect.

ISCP-ERR An ISCP-ERR message is generated whenever an authentication error is detected. This includes the cases when datagram without an authenticator is received when one is expected, or an authenticator cannot be correctly decrypted possibly due to key timeout, or when authentication actually failed.

IV.B. ISCP Cache

Encryption keys obtained by exchanging ISCP-KREQ and ISCP-KRPY messages are saved as per destination address entry, and they are referred to as ISCP cache as a whole. Figure 7 shows structure for an ISCP cache table entry. Each field in the entry has the following meanings.

0 – 31 (bit)

Internet address	
key (octet 0-8)	
timer	flags
*holdlist	
public key (octet 0-67)	

Figure 7: ISCP Cache Entry

Internet address: IP address of a host with this key.

key: Key to decrypt authenticator for verifying host, or key to encrypt authenticator for proving host.

timer: Time elapsed since this entry is registered.

flags: Control flags for this entry.

***holdlist:** Pointer to datagram hold area used to hold datagrams during key acquiring process. Used only in an authenticated host.

public key: Public key that is used to generate encryption key. It is used only in an authenticating host to cache the result of decryption of public key certification. It can be omitted if overhead of the above process is allowed or if memory use is limited.

IV.C. IP Option Format

IP option is used to convey authenticator as a part of a datagram. The structure of the option is shown in figure 8.

0 – 31 (bit)

Type	Length	Security Level
Authenticator		

Figure 8: IP option for IP/Secure

Description of the each field in the option is given below.

Table 5: Types and Codes of an ISCP message

Type	ISCP message	Code (C)	Function
1	ISCP-KREQ	1	Request secret key
2	ISCP-KRPY	1	Response for one-way authentication
		2	Response for two-way authentication
3	ISCP-KERR	1	Error detected in response to ISCP-KREQ
4	ISCP-ERR	1	Auth. failure (No authenticator is set)
		2	Auth. failure (Secret key is invalid)

Table 6: Authentication Information

ISCP message	Code (C)	Authentication Information	
ISCP-KREQ	1	Public key certificate	$(IP_A, PK_A, VP)^{SK_C}$
ISCP-KRPY	1	One-way secret key	$(WK)^{PK_A}$
	2	Two-way secret key	$(WK)^{PK_A SK_B}$
ISCP-KERR	1	Public key certificate	$(IP_A, PK_A, VP)^{SK_C}$

Type: Newly allocated value that indicates IP/Secure is in effect. Copy flag will be set to zero to avoid copying in case of fragmentation.

Length: If the security level is 1, the option is 12 octets long, and if the security level is 2 or 3, the option is 20 octets long. If option area is used for other purposes and has space limitations, the length of the option is 12 octets regardless of the security level.

Security Level: Security level (1 to 3).

Authenticator: Authenticator

IV.C.1. Authenticator

An authenticator placed in the "authenticator" field in the IP option has different meanings for each security level. Table 7 shows contents of authenticator for each security level. Box with "○" denotes information used to generate the authenticator.

Table 7: Authenticator Contents

Level	Authenticator	
	Source address	TCP header
1	○	
2	○	○
3	○	○

Source Address Authenticator The source address authenticator is 8 octets long, and this is the source address of the datagram encrypted by secret key *WK*. Here, we refer source address to the physical position of the source host. The format of the source address authenticator before the encryption takes place is shown in figure 9. Lower octets are set to zero, since DES encryption is done in the 8 octets block. The encryption translates this structure into a flat 8 octet authenticator.

0 31

Source IP Address
(Not Used)

(bit)

Figure 9: Source Address Authenticator (Before Encryption)

TCP Header Authenticator The TCP header authenticator is 8 octets long, and is the TCP header encrypted with secret key *WK* and compressed. The format of the TCP header authenticator is shown in figure 10.

0 31

Encrypted and Compressed TCP Header

(bits)

Figure 10: TCP header authenticator

Compressed Source Address/TCP Header Authenticator The compressed TCP header authenticator is 8 octets long, and is a concatenation of source address and the TCP header authenticator re-compressed into 4 octets, re-encrypted by the secret key *WK*. The format of the compressed address/TCP header authenticator is shown in figure 11. This format is used when there are other IP options already in the option area, causing space limitation.

0 31	
Source IP Address	(bits)
Compressed TCP Header Authenticator	

Figure 11: Authenticator for Limited Option Area (Before encryption)

V. Conclusion

Basic mechanisms for key delivery and authentication, and a design for an experimental implementation of the security enhanced IP protocol is presented.

We focused on low overhead in both key delivery and authentication procedure itself. We also paid considerable amount of attention for inter-operability with existing Internet hosts and gateways (aside from the fact that an IP option is either processed very slowly or sometimes even incorrectly at many workstation-based gateways), so that our modification has minimal impact on them. For example, IP/Secure datagrams should have no trouble getting through Internet gateways. The IP/Secure does not modify upper layer headers, so even mechanisms which does peak through such as TCP header compression [9] will continue to work.

At the time of this writing, an initial implementation of IP/Secure is complete for the PC/AT compatible machines running BSD/386, and we are working on the SunOS 4.1.3 with NET/2 network module. Simple performance evaluation using two Gateway2000 (i486DX2 at 66MHz) showed very promising results. For example, shared secret key delivery only required 190 msecs, and is not a source of any performance degration, because this happens only when the key is expired, which will be set at 20 minutes to 4 hours in practice. Memory-to-memory TCP throughput with level 3 security over the Ethernet has gone down to around 33% of stock IP, but it performed 70 to 80% of stock IP over the internet environment where average round trip time is 20msec (which may be considered small for mobile environment.) Overhead of very short packets were also measured, and they were in the order of few msecs, so IP/Secure does not spoil interactive performance in any means.

There are several things as listed below that we are working on at the time of this writing.

- Porting of IP/Secure to architectures other than BSD386 on i386/i486 architecture including SunOS 4.1.3 with NET2 networking code, as stated above.
- Design of a proxy IP/secure agent on the firewall gateway providing protections for machines inside the firewall that cannot be equipped with IP/Secure for some reason such as unavailability of kernel source codes.
- Design of a IP/Secure black box that plugs in between an interface of a machine and an attached network, transparently providing IP/Secure protections for the machine. This device will also help those machines that cannot be equipped with IP/Secure.
- Introduction of hierarchical public key certificate. We are aware of the problem that the current IP/Secure requires a single CA to issue public key certificate for the entire address space. This is not likely to work, especially if the IP/Secure is put into any practical use. We are looking at making our public key certificate into multi-level as incorporated in the PEM. The PEM permits unlimited numbers of hierarchy in the public key certificate, but we are looking at two or three at this point.
- In addition, we are also looking at several proposed mobile IP extensions including [4] to see if IP/Secure works with them.

We believe that proposed IP extension for mobile host support will bring in major security problems to the Internet, and that it is vitally important for these extensions to include a non-adhoc security mechanisms.

Recent attacks which is believed to have captured access informations for tens of thousands of systems across the Internet used the method of wire tapping the clear text password used by traditional remote access applications such as telnet, rlogin and ftp, using the promiscuous mode of a network interface of a once compromised machine. We started our design of IP/Secure to work with the VIP scheme, ended up in what we believe is a more general protection mechanism. IP/Secure, once introduced in the kernel, can provide a protection against this type of attack with least impact to these applications especially under the UNIX process and file descriptor semantics, in which open file descriptors are inherited by the executed program. In such a case, we would only have to provide a wrapper program that opens a remote connection in the secure mode, and execute the remote access applications.

As migration into IPng is just couples of years ahead, an attempt to adapt IP/Secure to the new protocol, or emitting possible feedbacks to the proposed protocol that provides general framework for security service such as I-NLSP[10] remain as important issues.

VI. References

[1] Fumio Teraoka and Mario Tokoro. Host migration transparency in ip networks. In *Computer Communication Review*, pages 45–65, Jan 1993.

[2] Pravin Bhagwat and Charles E. Perkins. A mobile networking system based on internet protocol. In *Proceedings of the Mobile & Location-Independent Computing*, August 1993.

[3] Hiromi Wada, Takashi Yozawa, Tatsuya Ohnishi, and Yasunori Tanaka. Mobile computing environment based on internet packet forwarding. In *Proceedings of Winter USENIX*, pages 503–517, San Diego,CA, Jan 1993.

[4] IETF/mobile-wg. Routing Support for IP Mobile Hosts. Internet Draft, December 1993.

[5] J. G. Steiner, C. Neuman, and J. I. Schiller. Kerberos: An authentication service for open network systems. In *Proceedings of the 1988 Winter USENIX Conference*, pages 191–202, 1988.

[6] Simason Garfinkel and Gene Spafford. *PRACTICAL UNIX SECURITY*. O'Reilly & Associates,Inc., 1991.

[7] D. E. Denning, editor. *Cryptography and Data Security*. Addison Wesley, 1982.

[8] S. Kent, J.Linn, and IAB Privacy Task Force. Privacy Enhancement for Internet Electronic Mail: Part II – Certificate-Based Key Management. RFC 1114, August 1989.

[9] Van Jacobson. Compressing TCP/IP Headers for Low-Speed Serial Links. RFC 1144, February 1990.

[10] K. Robert Glenn. Integrated Network Layer Security Protocol. Internet Draft, September 1993.

Acknowledgement

Authors would like to thank Professor Eiji Okamoto and Professor Tomohiko Uematsu for their valuable advises in the early stage of this study. Technical contributions of this paper was impossible without their extensive knowledge of, and foresight into cryptosystems. We would also like to thank researchers of the WIDE internet project, especially Hiroaki Kikuchi of Fujitsu Laboratories. The piece of code we have today was impossible without their unfailing criticims in protocol design and implementation details.

Author Information

Takeshi Tanida was a master's course student at Japan Advanced Institute of Technology (JAIST) delegated from Fujitsu Hokuriku Systems, when he carried out the work described in this paper under the supervision of Prof. Shinoda. His research interest covers general networking protocols and systems. He received a Master of Information Science from JAIST in 1994 and is continuing his work on network layer security as a visiting researcher at JAIST.

Yoichi Shinoda is an Associate Professor of School of Information Science at Japan Advanced Institute of Science and Technology (JAIST), where he has been leading research activities on networked and distributed systems. He is the chief architect of the FRONTNET/FRONTIER, which is the campus network/computing environment of the JAIST. He received a Doctor of Engineering from Tokyo Institute of Technology in 1989. His research interest spans widely on various aspects of network systems and distributed environment including security systems and mobile support. He is also interested in incorporating massively parallel processors into our daily information activities. Prof. Shinoda is a member of the WIDE Internet Project, ACM, USENIX and Japan Society of Software Science and Technology (JSSST).