Implementations of the User Mobility Support over IPSec

Chin-Fu Kuo[†], Yung-Feng Lu[†], Ai-Chun Pang[†*], and Tei-Wei Kuo[†*]
Email: {d89005, d93023, acpang, ktw}@csie.ntu.edu.tw, Fax:+886-223628167
[†] Department of Computer Science and Information Engineering
[*] Graduate Institute of Networking and Multimedia
National Taiwan University, Taipei, Taiwan 106, ROC

Abstract

IPSec, proposed by IETF, is a popular tunneling technology adopted for the Virtual Private Network. It provides the network layer additional security functionality to transform multiple segments of a private network into one. This paper exploits the extension of IPSec to have secured tunnels for users who might move around dynamically without carrying the same machine. We not only explore implementation issues of IPSec over popular operating systems, such as Windows and Linux, but also provide performance evaluation of of the proposed algorithms by a series of experiments.

Keywords: IPSec, Security, Common Criteria, Performance

1 Introduction

Authentication, authorization, and accounting (AAA) for security provides a useful framework for the reasoning and measuring of the security capability of computer systems. Authentication provides a way for the identification of users. Authorization provides users privileges in executing specified tasks. Accounting concerns the auditing of user activities. In June 1993, the United States, the United Kingdom, Germany, France, Canada, and Netherlands started to develop an evaluation standard for a multi-national security market. This standard is known as the "Common Criteria for Information Technology Security Evaluation" (CC-ITSE), and it is usually referred to as the "Common Criteria" (CC). Version 1.0 of the CC was released in January 1996. It became the ISO International Standard 15408 in 1999.

While information sharing over Internet become more and more popular, there is a strong demand for tunneling technologies for the Virtual Private Network (VPN), where the Virtual Private Network (VPN) is an extension of a private network that encompasses network links across public networks (e.g., Internet). Within a VPN, authorized users are allowed to access various network-related services and resources. The Point-to-Point Tunneling Protocol (PPTP), the Layer 2 Tunneling Protocol (L2TP), and the IP Security Protocol (IPSec) are now the main tunneling technologies adopted for VPN. IPSec was proposed by the Internet Engineering Task Force (IETF) to provide the network layer additional security functionality. A certain degree of communication security against eavesdropping, repudiation, and spoofing is provided.

Although IPSec introduces a good flexibility in the security enhancement of higher-layer protocols, the setup of IPSec-based secured tunnels tends to be static in the configuration and restricted in a machine-to-machine fashion. While user mobility has become an important feature for many systems, technologies that provide users a lower cost and flexible way in joining a VPN are in a strong demand. An example approach in this direction is the adopting of the Ensemble system for group communication by having a single shared encryption key for the entire VPN [8]. The major problem for this approach is the vulnerability of the secured tunnels when some malicious user obtains the shared key. Kindred et al. [6] proposed an implementation on the system support for dynamic VPN communities of independently administered and firewall-protected enclaves. It relies on a central authority associated with common VPN mechanisms. Beside work on secured tunnels, researchers have exploited various approached for user and IP mobility with security support in the past decade, e.g., [5, 3, 9]. In particular, Kim et al. [5] proposed to modify an IPSec implementation to provide mobility, where a user must carry the same machine to have mobility support. Berioli, et al. [3, 9] considered the integration of IPSec tunnels and MobileIP.

In this paper, we are interested in the extension of the IPSec implementations to have secured tunnels for users who might move around dynamically without carrying the same machine. With the proposed algorithm, the user mobility could be achieved. We shall not only identify threats against IPSec but also provide a lower-cost and flexible way in joining a virtual private network, compared to a full-

scaled Public Key Infrastructure. An AAA-based negotiation procedure will be proposed for the implementations. The implementation of the proposed algorithm is simple and highly portable. We shall consider the implementation issues of IPSec over popular operating systems, such as Windows and Linux. The overheads of IPSec under different enabled security mechanisms would also be evaluated by a series of experiments.

The rest of this paper is organized as follows: Section 2 summaries CC and identifies threats against IPSec. Section 3 presents a dynamic configuration algorithm to extend IPSec for the support of user mobility. Section 4 explores implementation issues of the proposed algorithm over well-known operating systems. Section 5 is the conclusion.

2 Common Criteria and IPSec

In Common Criteria, a product which is subjected to evaluation is called a *Target of Evaluation* (TOE). The security measures and security requirements of a TOE are defined by two documents: Protection Profile and Security Target. *Protection Profile* (PP) describes the security requirements and allows the consumers and developers to compile standardized sets of security requirements to meet their needs. On the other hand, *Security Target* (ST) specifies the functional requirements and assurance securities for the product developers. The evaluators use ST as the basis for evaluation. In this paper, we follow Common Criteria version 2.1.

A Protection Profile document consists of the following information: *Security Environment* defines the assumptions, security threats, and security policies of the customer organization that uses the TOE. *Security Objective* states the abstract goals of the TOE implementations according to the security environment. *Security Requirement* translates the abstract goals of security objectives into implementation-related goals. *Rationale* describes the relationship between the security environment and the security objectives, and the relationship between the security objectives and the security requirements.

Two threats in CC match the security targets of IPSec: *T.Hack_Comm_Eavesdrop* and *T.Spoofing*. *T.Hack_Comm_Eavesdrop* concerns situations in which hackers obtain user data by eavesdropping on communications lines. *T.Spoofing* considers situations in which attackers trick users into interacting with spurious system services. Based on the CC, the security objectives for each threat and the security requirements of each security objective for IPSec could be collected. The security objectives and requirements are used to choose proper mechanism to overcome the threats for a secure tunnel. The security objectives of other threats are listed in Table 1, and the detailed descriptions of each security objective for the IPSec could be found in Table 2 [2].

With the identification of security objectives, we could better point out the security enhancement features that IPSec provides and explore more complex and advanced services over IPSec. Tradeoffs between performance/overheads and security objectives could be better understood. IPSec encrypts higher-layer protocols by having an authentication header (AH) and an encapsulated security payload (ESP). IPSec can be configured into three operating modes with different security functionalities. The mapping of security objectives and IPSec options is summarized in Table 3. The AH feature satisfies *O.Integrity_Attr_Exch* and *O.Non-repudiation*. The ESP feature serves the requirements of *O.Comm_Trusted_Channel*, *O.Data_Exchange_Conf*, and *O.Crypto_Comm_Channel*. In the next section, we shall propose a user mobility service over IPSec and then present the tradeoffs between performance/overheads and security objectives in a later section.

3 A Dynamic Configuration of Secured Tunnels

The purpose of this section is to extend IPSec to provide a lower-cost and flexible way in joining a virtual private network (VPN). Instead of adopting a full-scaled Public Key Infrastructure, the purpose of this section is to propose an IPSec-based dynamic configuration algorithm for the better support on user mobility[1].

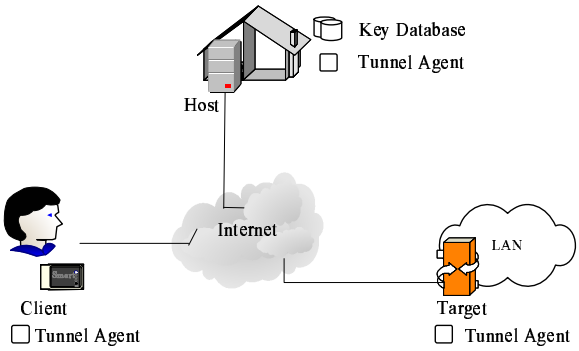

Figure 1. System architecture

As shown in Figure 1, Client, Host, and Target denote the moving subject (e.g., a travel agent), a trusted machine of the subject (e.g., a desktop in the office of the agent), and a trusted machine of the primary working environment of the subject (e.g., a computer in the company of the agent), respectively. The objective is to provide a secured IPSec-based tunnel between Client and Target dynamically. We assume that Host and Target could communicate in a trusted

[1]This algorithm is extended from a preliminary 5-step algorithm presented in the Work-In-Progress Session of the IEEE Real-Time Systems Symposium in 2003

Threat	Security Objective
T.Spoofing	O.Crypto_Comm_Channel O.Comm_Trusted_Channel O.Integrity_Attr_Exch O.Non-repudiation
T.Hack_Comm_Eavesdrop	O.Data_Exchange_Conf

Table 1. The mapping of threats and security objectives for IPSec.

Security Objective	Description
O.Crypto_Comm_Channel	A secure session establishment should be provided between the system and remote systems using encryption functions.
O.Data_Exchange_Conf	User data confidentiality should be protected when data are exchanged between systems.
O.Comm_Trusted_Channel	Concerns the provision of a communications channel between the system and a remote trusted system for security-critical operations.
O.Integrity_Attr_Exch	This is to ensure correct exchanging of attributes with another trusted product. It ensures that the system correctly exchanges security-attribute information with another trusted IT product.
O.Non-repudiation	It is to prevent users from avoiding accountability.

Table 2. Descriptions of security objectives

way. Note that when it is a concern, IPSec could be deployed to resolve the problem. In the algorithm, we use a key-based mechanism to achieve the objective of user mobility. Let Client and Host both have the same set of private and public keys. We assume that when a moving subject leaves Host, he/she would take several key pairs. Each key pair includes a public key and a corresponding private key and could only be used to create a secured tunnel once. We call them one-time key pairs. Whenever a key pair is used, the state of the key pair changes from the valid state to the invalid state. When Client requests a secured tunnel between himself/herself and Target, Client encrypts the following request with the public key of Client and submits it to Host:

Request ID $\|$ **User ID** $\|$ **Objective-ID** $\|$ **Request Info** $\|$ **Authentication Tag**

Request ID, User ID, and Request Info are a uniquely identifiable ID for the request, the ID of Client, and other necessary information for the request, respectively. Objective-ID denotes the security objectives for the tunnelling, e.g., those corresponding to the four rows in Table 3. Authentication Tag is the request's digital signature, that is the request encrypted with the private key of Client.

Figure 2 is a sequence of events for the construction of the secured tunnel, where the numbers correspond to the logical order of the events. We assume that the communications between Host and Target are via a secured connection. The establishment of a secured tunnel between Client and Target could be done as follows:

Step 1: Initial Setup Request: Client uses her Tunnel Agent to submit a request to Host to request for a secured connection.

Step 2: Request Validation: The Tunnel Agent of Host uses Client's private key to decrypt the request and then uses Client's public key to verify the digital signature of the request.

Step 3: Transmission of the User Information: If the state of the key pair is valid, then update the state of the key pair in Key Database from valid to invalid. The Tunnel Agent of Host then passes the user information to Target.

Step 4: User Authentication: The Tunnel Agent of Target authorizes the user to use services provided by Target.

Step 5: Generation of a Pre-share Key: The Tunnel Agent of Target generates a pre-share key for a new tunnel and does the configuration for the tunnel.

Step 6: Passing Back of Tunneling Information and Pre-share Key: The pre-share key and the tunneling information are passed back to the Tunnel Agent of Host.

Step 7: Encryption of Pre-Share Key and Tunneling Information with Public Key: The Tunnel Agent of Host encrypts the following response message (including the tunneling information and session key) with the public key of Client:

IPSec Option	O.Comm_Trusted _Channel	O.Data _Exchange_Conf	O.Crypto_Comm _Channel	O.Integrity _Attr_Exch	O.Non-repudiation
Native (ID-1)	-	-	-	-	-
AH (ID-2)	-	-	-	●	●
ESP (ID-3)	●	●	●	-	-
AH+ESP (ID-4)	●	●	●	●	●

Table 3. The mapping of security objectives and IPSec options (Native:no security service, AH: packets with the AH header, ESP: packets with the ESP format).

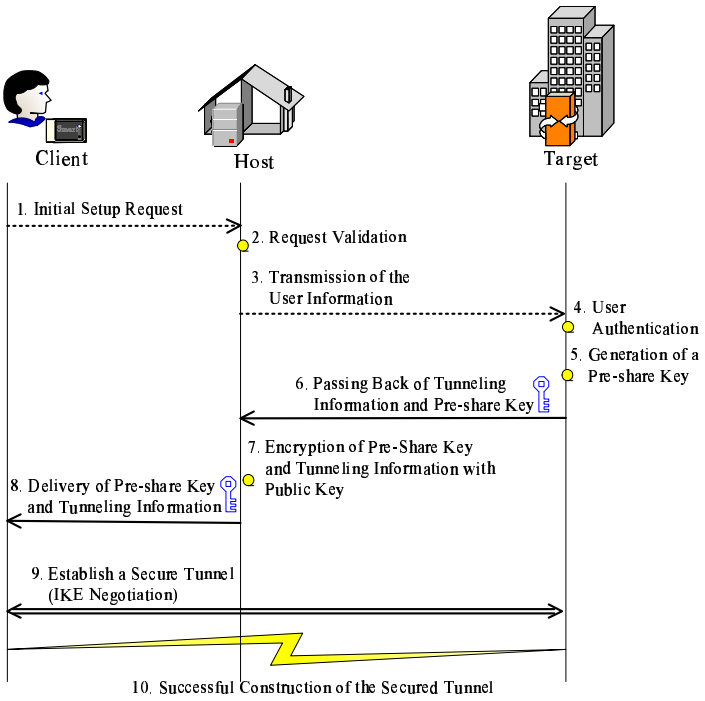

Figure 2. Steps for the set-up of a user-mobility VPN

Request ID ‖ Session Key ‖ Tunneling Info ‖ Authentication Tag

Request ID, Session Key, and Tunneling Info are the request ID, the session key, and tunnelling information, respectively. Authentication Tag is the response message's digital signature, that is the request encrypted with the private key of Client.

Step 8: Delivery of Pre-share Key and Tunneling Information: The Tunnel Agent of Host sends the encrypted information back to Client.

Step 9: Establishment of a Secured Tunnel (IKE Negotiation): The Tunnel Agent of Client decrypts the information with the private key of Client and establishes a tunnel with Target.

Step 10: Successful Construction of the Secured Tunnel

Note that the implementation of the above key mechanism could be done with some piece of hardware containing the private and public keys. Different steps of the algorithm perform the functionality of authentication, authorization, and accounting, as shown in Table 4.

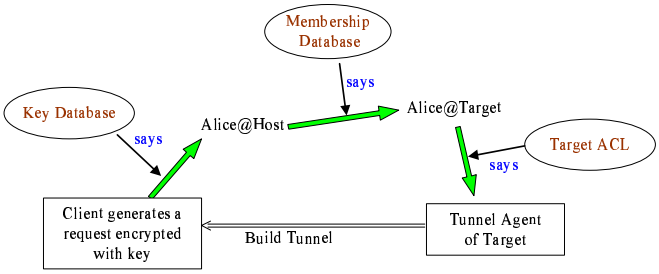

Figure 3. Chain of responsibility

Furthermore, the authentication procedure follows the chains of responsibility in [7]. We could observe a chain of responsibility, as show in Figure 3, running from the request at Client to the Target resource at the other. Client's access request (e.g., a request by Alice) is granted because the request comes with a request information and is encrypted by the public key of Client's key pair. Host certifies the request that is encrypted with Client's key for Alice@Host. Target's membership database shows that Alice@Host is a legal user. The access control list on the Target shows that Client, i.e., Alice, has the authority to own a secured tunnel. The tunnel agent of Target helps in the establishment of a new tunnel.

4 Case Study and Performance Evaluation

The purpose of this section is to explore the implementation issues and provide performance evaluation for the proposed algorithm, especially those over well-known operating systems. Each tunnel establishment of the proposed algorithm might consider a combination of three kinds of options: security objectives, IPSec options, encryption/decryption algorithms. The set of selected security objectives denote the security level expected by users, and

Authentication	Step 2. Host does ticket validation Step 4. Target determines whether the user is valid Step 8. Client authenticates the source of the pre-share key
Authorization	Step 2. After ticket validation, Host forwards the information to Target Step 4. After user validation, Target generates a pre-share key for the client
Accounting	Step 2. Change the state of the used one-time key pair to a used state

Table 4. The relationship between the steps of the construction procedure and AAA

there is a mapping between security objectives and IPSec options (such as AH, ESP, AH+ESP), as shown in Table 3.

The implementation of the proposed IPSec-based algorithm is simple and highly portable. Consider the system architecture, as shown in Figure 1: The responsibilities of Tunnel Agents of Client/Host/Target are for the encryption and decryption of requests/responses, key-pair verification, user authentication, pre-share key generation, and the construction/destruction of secured tunnels. The construction and destruction of secured tunnels could be done by proper modifications to a configuration file for IPSec . Tunnel Agents could trigger the construction and destruction of tunnels with IPSec commands in the shell program. As a result, Tunnel Agents could be implemented pretty independently from many IPSec implementations. In other words, Tunnel Agents can be implemented without any modification to many IPSec implementations. We shall take three well-known operating systems as examples to illustrate the tunnel agent implementation for secured tunnel establishment:

Microsoft Windows: IPSec in Windows consists of three main components: the Policy Agent, the Internet Key Exchange (IKE) module, and the IPSec driver. These three components, in conjunction with other Windows components, such as the TCP/IP driver and cryptoAPI, provide IPSec functionality in Windows 2000 and XP. The command-line tools for the creating of IPSec policies for Microsoft Windows 2000 and XP are ipsecpol.exe and ipseccmd.exe, respectively [11]. ipsecpol.exe has two modes: Dynamic and Static. The dynamic mode defines a policy that could be loaded and enforced for the duration of the Policy Agent. Tunnel agent can use a script with a proper command-line tool in the dynamic mode to setup a secured tunnel whenever it is available.

Linux: FreeS/WAN [4] is an implementation of IPSec and IKE over Linux. With FreeS/WAN, the *ipsec* command is a front-end shellscript that provides control over IPsec activities. With the configuration file for FreeS/WAN, i.e., */etc/ipsec.conf*, Tunnel agent can setup a secured tunnel in Linux. We refer interested readers to the ipsec.conf manual page for details.

FreeBSD: FreeBSD adopts KAME [1], developed in Japan, for IPSec implementations. The racon IKE daemon is used to perform automatic keying [10]. For the establishment of a secured tunnel in FreeBSD, Tunnel Agent needs to edit the configuration file for the racon IKE daemon and execute a script with KAME demands.

The construction of a secured tunnel is divided into two phases: user authentication (i.e., jobs of Tunnel Agents) and tunnel creation (i.e., the IKE negotiation of IPSec). Although the proposed IPSec-based algorithm consists of tunnel setups, key exchanging, and transmissions of data packets, the overheads of the proposed algorithm is dominated by the encryption/decryption and authentication operations for data packets, provided that a significant amount of information flows between Client and Target. In this section, we measured the system performance and system resource usages for data transmissions between two Pentium-IV 1.7GHZ platforms with Linux 2.4.18-14 and FreeS/WAN 2.01. We adopted 100Mbps Ethernet LAN for performance evaluations. A utility iperf was used to do performance evaluation[2].

Figure 4 shows the network throughput under different UDP datagram sizes with IPSec (ESP: MD5+3-DES). IPSec (ESP: MD5+3-DES) was taken for evaluations because it was one of the most popular settings for IPSec. Since the network throughput reaches 20 Mbps, the CPU utilization approached 100%. When the datagram size increased, the throughput kept increasing until the peak (1415 Bytes) because datagram sizes before the peak could well fit in the ESP of an IP packet. The dropping of the network throughput after the peak was because of the splitting of over-sized IP packets. The restoring of the network throughput after the peak had the same reason why the network throughput increased before the peak. The throughput for the receiver was very low (only few megabits per second) probably because of the limitation of buffer sizes and the resulted delaying on decryption work.

[2]More experimental results are reported in a paper appeared in the Work-In-Progress Session of the IEEE Real-Time Systems Symposium, 2003.

Model	Throughput (MB bits/s)	System Time Usage (%)	Overheads (Cycles/Byte)
Native	92.6	9.9	14.5453
AH-MD5-96	91.0	36.9	55.0922
AH-SHA1-96	91.2	60.4	90.0094
ESP-MD5-3-DES	53.9	92.0	231.9884

Table 5. System overheads for the peak throughput over IPSec (TCP) for different encryption algorithms.

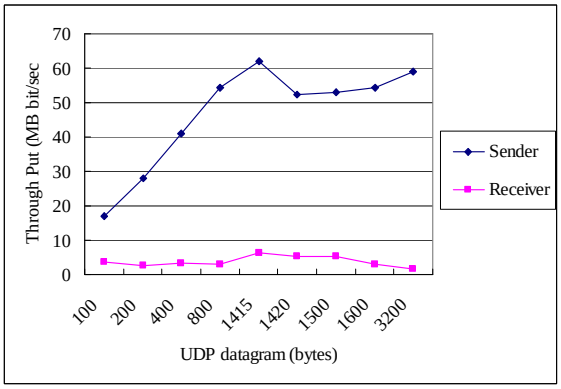

Figure 4. Network throughput under different UDP datagram sizes IPSec (ESP: MND+3-DES

5 Conclusion

This paper exploits the extension of the IPSec implementations to have secured tunnels for users who might move around dynamically. Instead of having a full-scaled Public Key Infrastructure, a lower-cost and flexible way in joining a virtual private network is proposed. An AAA-based negotiation procedure is proposed for the implementations. The implementation of the proposed algorithm is simple and highly portable. We considers the implementation issues of IPSec over popular operating systems, such as Windows and Linux. The overheads of the proposed method were also evaluated by a series of experiments.

For future research, we shall explore dynamic routing methods for security enhancement. Tradeoff between security enhancement and the balancing of network workloads would be a focus of this study.

References

[1] KAME Project. `http://www.kame.net/`.

[2] Common Criteria for Information Technology Security Evaluation (CCITSE), version 2.1. Technical Report CCIMB-99-032, National Institute of Standards and Technology, August 1999. `http://csrc.nist.gov/cc/`.

[3] M. Berioli and F. Trotta. Ip mobility support for ipsec-based virtual private networks: an architectural solution. In *IEEE Global Telecommunications Conference*, 2003.

[4] Internet Engineering Task Force (IETF). FreeS/WAN. `http://www.freeswan.org`.

[5] B.-J. Kim and S. Srinivasan. Simple mobility support for ipsec tunnel mode. In *IEEE 58th Vehicular Technology Conference*, 2003.

[6] D. Kindred and D. Sterne. Dynamic vpn communities: Implementation and experience. In *Proceedings of the DARPA Information Survivability Conference and Exposition(DISCEXII'01)*, 2001.

[7] B. Lampson. Computer security in the real world. *IEEE Computer*, 37(6):37–46, 2004.

[8] O. Rodeh, K. Birman, M. Hayden, and D. Dolev. Dynamic virtual private networks. Technical report, Dept. of Computer Science, Cornell University, March 1999.

[9] E. Sanchez and R. Edwards. On achieving secure seamless mobility. In *2002. 4th International Workshop on Mobile and Wireless Communications Network*, 2002.

[10] J. Tiefenbach and B. Koster. FreeBSD IPsec mini-HOWTO. `http://asherah.dyndns.org/~josh` /ipsec-howto.txt.

[11] C. Weber. Using IPSec in Windows and XP, Part Two. `http://www.securityfocus.com/infocus/1526`.