DESIGN AND IMPLEMENTATION OF A TRANSPARENT SECURE LAN

Dr. Sufyan T. Faraj and Firas R. Barjas

ABSTRACT

Many attacks may be carried out against communications in Local Area Networks (LANs). However, these attacks can be prevented, or detected, by providing confidentiality, authentication, and data integrity security services to the exchanged data.

This paper introduces a security system that protects a LAN from security attacks. On each host in the protected LAN, the security system transparently intercepts each outbound IP (Internet Protocol) packet, and inserts a crypto header between the packet IP header and payload. This header is used to detect any modification to the content of the packet in transit, and to detect replayed packets. Then, the system encrypts the IP packet payload and some fields of the inserted crypto header. On the other hand, the system transparently intercepts each inbound IP packet, decrypts its encrypted portions, and then uses its crypto header to authenticate the packet. If the packet is properly authenticated, the system indicates it to upper protocols.

To be transparent to applications, the security system part that processes inbound and outbound IP packets was implemented as a NDIS (Network Driver Interface Specification) intermediate driver that resides between the LLC (Logical Link Control) and MAC (Medium Access Control) data link sublayers.

الخلاصة

هنالك العديد من الهجمات التي يمكن تنفيذها ضد اتصالات الشبكات المحلية (LANs). ألا أنه بالأمكان منع أو اكتشاف هذه الهجمات عن طريق توفير السرية للمعلومات المتبادلة، و التأكد من سلامتها و صدق مصدرها.

يقترح هذا البحث نظاماً أمنياً يمكن استعماله لحماية شبكة محلية من الهجمات التي قد تستهدف أمنها. في كل حاسبة مرتبطة بالشبكة المحمية، يقوم النظام الأمني المقترح بمقاطعة مسار كل رزمة خارجة من نوع IP packet بصورة شفافة، ثم يقوم بادخال معلومات أمنية (crypto header) بين الـ IP header و الـ IP payload. تستعمل هذه المعلومات لكشف أي تغيير لمحتويات الرزمة أثناء تنقلها، و لكشف الرزم التي أعيد بثها. بعد ذلك، يقوم النظام بتشفير الـ IP Payload، و بعض الأجزاء من المعلومات الأمنية المضافة. من جهة أخرى، يقوم النظام بمقاطعة مسار كل رزمة داخلة الى الحاسبة بصورة شفافة، ثم يقوم بفتح تشفير الأجزاء المشفرة منها. بعد ذلك، يفحص النظام المعلومات الأمنية الموجودة في الرزمة المُستلمة للتأكد من صدق و سلامة معلوماتها. فأذا كانت الرزمة سليمة، يقوم النظام بتسليمها الى البروتوكولات العليا.

لكي يكون النظام المقترح شفافاً للتطبيقات العليا، فأن ذلك الجزء من النظام الذي يعامل الرزم الخارجة و الداخلة تم تنفيذه كسواقة (NDIS intermediate driver)، و التي تقع بين طبقتي الـــ data link layer : LLC و MAC.

KEYWORDS
LAN, **security**, **encryption**, **authentication**, **transparent**, **NDIS intermediate driver**

ABBREVIATIONS

API:	Application Programming Interface
CIMD:	Cryptographic Intermediate Miniport Driver
DDK:	Driver Development Kit
DIX:	Digital equipment Intel Xerox
DoS:	Denial of Service
ESP:	Encapsulating Security Payload
GUI:	Graphical User Interface
ICMP:	Internet Control Message Protocol
IETF:	Internet Engineering Task Force
IM:	Intermediate Miniport
IP:	Internet Protocol
IPX/SPX:	Internet Packet Exchange/ Sequenced Packet Exchange
IV:	Initialization Vector
LAN:	Local Area Network
LLC:	Logical Link Control
LSP:	Layered Service Provider
MAC:	Medium Access Control
MAC:	Message Authentication Code
MD:	Message Digest
MTU:	Maximum Transmission Unit
NDIS:	Network Driver Interface Specification
NIC:	Network Interface Card
OSI:	Open Systems Interconnection
PMTU:	Path Maximum Transmission Unit
SDL:	Security Descriptors List
SHA:	Secure Hash Algorithm
SSM:	Security System Manager
TCP/IP:	Transport Control Protocol/ Internet Protocol
TDI:	Transport Driver Interface
WLAN:	Wireless LAN

INTRODUCTION

Networks have become indispensable for conducting business in government, commercial, and academic organizations. Networked systems allow people to access needed information rapidly, improve communications while reducing their cost, collaborate with partners, provide better customer services, and conduct electronic commerce. While computer networks revolutionize the way people do business, the risks they introduce can be fatal to a business. Attacks on networks can lead to lost money, time, products, reputation, sensitive information, and even lives [ALL01].

The problem of network security is a very complex issue. By definition, network security means a protection of the network assets from different kinds of threats in the network by implementation of different security services using various security mechanisms [TRC99].

The above definition introduces the following three aspects of network security [STA99]:

- *Security Attack:* Any action that compromises the security of information, such as eavesdropping, masquerading, replay attacks, Denial of Service (DoS) attacks, and active change to the exchanged information.
- *Security Service:* A service that enhances the security of data processing systems and information transfers. It is intended to counter security attacks. A security service makes use of one or more security mechanisms. The most important security services are confidentiality, integrity, and authentication.
- *Security Mechanism:* A mechanism that is designed to detect, prevent, or recover from a security attack. Encryption is the security mechanism that can be used to provide confidentiality, whereas hash functions or Message Authentication Codes (MACs) can be used in different ways to provide authentication and data integrity.

Some important insights in the literature concerning the subject of network security are summarized as follows:

- IPSec (IP Security) is a subset of IPv6 and a set of extension to IPv4. It has been developed by the Internet Engineering Task Force (IETF) to provide secure packet transmission over networks.
- A. Ganz, S. H. Park, and Z. Ganz [GAN00] had developed a security broker for Wireless LANs (WLANs). This system implements a number of security services such as authentication and real-time encryption/decryption for the communications in WLANs. The real time encryption/decryption service is implemented as a Layered Service Provider (LSP), and uses Microsoft's CryptoAPI. The use of a LSP allows a Trojan horse, or a worm to directly call the kernel-mode TCP/IP (Transport Control Protocol/ Internet Protocol) driver via the Transport Driver Interface (TDI), and completely bypass the provided security processing.
- The CIPRESS [RAD01] (Cryptographic Intellectual Property Rights Enforcement SyStem) from Mitsubishi Corporation and Fraunhofer-IGD was an attempt for providing transparent network access control, auditing, and encryption on the Microsoft Windows NT platform. CIPRESS provides these services by enforcing an automatic security policy that was implemented using LSP technique. However, CIPRESS supports only complete files, and does not support streaming data.
- B. S. Shaker [SHA02], Al-Nahrain University developed in 2002 an on-line end-to-end cryptography software system for LANs. An NDIS intermediate driver was used to process sent IP packets, and encrypt each IP packet payload using MARS algorithm if its upper protocol is TCP, whereas decryption of received packets was performed at the IP filter hook driver. While encrypting exchanged TCP data provides confidentiality, and some degree of data integrity (depending on encrypted TCP checksum), this system does not protect networks against masquerading because it does not authenticate the source of received packets. Moreover, replay attacks can be launched freely and successfully against the hosts that use this system. On the other hand, the use of IP filter hook driver for decryption instead of the used NDIS intermediate driver unnecessarily introduces additional processing overhead.

The purpose of this paper is to introduce the design of a security system that can be used to provide confidentiality, authentication, and integrity to the data exchanged within a LAN, so that attacks against the exchanged data are prevented (or detected). The security system provides the security services transparently, i.e. there is no need to change software on a user computer that benefits from the provided security services. Users can continue to use their usual network applications (without any change) while the security system provides security services to the exchanged data.

SYSTEM SPECIFICATIONS AND DESIGN APPROACH

The following points shed the light on the specifications of the proposed security system:

- The security system provides confidentiality to the exchanged data.
- The security system provides authentication and integrity to the exchanged data.
- The security system provides access control.
- If received data is not properly authenticated or if it is replayed data, the security system collects audit information from it, and produces an alert to the network security administrator indicating the reception of such invalid data along with the collected audit information. In turn, the network security administrator could monitor the audit information, and take an appropriate action to prevent a possible attack that might cause the reception of the unauthenticated or replayed data.
- The security system is able to prevent DoS attacks launched by hosts unknown to the system.
- The security system is transparent to applications.
- The security system does not affect the work of routers that may exist in the protected network.
- The security system is able to protect itself from attacks.
- Due to the rapid increase of networks, many people who are not so experienced in network security have become network administrators. For this reason, the security system provides a friendly and easy to use interface for them.

To achieve the features and specifications stated above, the following choices were adopted:

♦ Microsoft Windows 2000 was chosen as the platform under which the proposed security system works, for of its reliability, high performance, and security.

♦ TCP/IP protocol was chosen as the transport protocol for the protected LAN.

♦ The LAN protected by the proposed security system was chosen to be Fast Ethernet. The security system assumes that Ethernet frames encapsulate IP packets according to the Ethernet II (DIX Ethernet) encoding, because by default, Microsoft Windows 2000 TCP/IP stack transmits Ethernet frames using this encoding [MAC00].

♦ The NDIS Intermediate Miniport (IM) driver was chosen to apply the required transparent security processing to inbound and outbound network traffic. NDIS IM driver was chosen for the following reasons:
  1- It is well documented.
  2- It is a kernel mode driver, and all incoming or outgoing packets should pass through this driver. This feature prevents network packets from bypassing the security policy enforced by the security system.
  3- The NDIS IM driver lies below the network layer (between the LLC and MAC data link sublayers [DDK00]). This feature gives NDIS IM drivers a lot of control over network packets, without affecting other network protocol stack components. As an example, the position of NDIS IM drivers make them appropriate to be used to authenticate the source IP address of received IP packets.

The transparent security processing performed by the security system NDIS IM driver mostly follows the way in which the IPSec protocol processes network packets, but it avoids some problems that exist in the IPSec design. These problems are:

1- IPSec is too complex to be secure. The design obviously tries to support many different situations with different options. The number of major modes of operation can be drastically reduced without significant loss of functionality. IPSec is well beyond the level of complexity that can be analyzed or properly implemented with current methodologies. Thus, no IPSec system will achieve the goal of providing a high level of security [SCH99].

2- The ESP (Encapsulating Security Payload) protocol allows the payload to be encrypted without being authenticated. In virtually all cases, encryption without authentication is not useful. ESP should be modified to always provide authentication; only encryption should be optional [SCH99].
3- When both encryption and authentication are provided, IPsec performs the encryption first, and then authenticates the ciphertext. This is the wrong because going by the "Horton principle", the protocol should authenticate what was meant, not what was said. The meaning of the ciphertext still depends on the decryption key used. Authentication should thus be applied to the plaintext, and not to the ciphertext [SCH99].
4- The specifications of IPSec allow predictable -but random- Initialization Vectors (IVs) to be used in IPsec ESP encryption, and explicitly allow the common practice of using the last ciphertext block of encrypted data from an encryption process as the IV for the next encryption process. Predictable initialization vectors compromise IPsec confidentiality. By using an adaptive chosen plaintext attack, an attacker can break low entropy plaintext blocks using brute force, and confirm guesses of the contents of arbitrary plaintext blocks. However, the preconditions of this attack are restrictive, and the vulnerability is thus difficult, but probably not impossible, to exploit in practice [NUO02].
5- Since IPSec is used to secure IP packets only, packets of transport protocols other than TCP/IP (IPX/SPX -Internet Packet Exchange/ Sequenced Packet Exchange- for instance) are not affected by the processing of IPSec. Hence, the packets of these protocols can still be used to launch attacks against a network, if they are installed on the hosts of that network, even if IPSec is deployed.

The developed security system avoids the complexity and many modes of operation inherent in the IPSec protocol, it always provides authentication and encryption for IP packets, it authenticates the plaintext instead of the ciphertext, it uses a random IV for each IP packet, and it prevents communications using transport protocols other than the TCP/IP protocol (the security system can be improved to secure the packets of these transport protocols instead of blocking them). However, the security system implemented now does not provide automatic key exchange, and this can be considered as a future work.

SECURITY SYSTEM ARCHITECTURE AND COMPONENTS

The security system consists mainly of the following three components:

1- The Security Descriptors List (SDL), which stores security descriptors corresponding to hosts in the protected network.
2- The Security System Manager (SSM), which is a Win32 application that allows the network security administrator to control the operation of the CIMD, and alerts him (or her) if a suspicious packet is received.
3- The Cryptographic IM Driver (CIMD) that applies the transparent security processing to inbound and outbound IP packets.

Fig.(1 illustrates the security system architecture and the interaction among its components. The following subsections introduce these components in more details.

The Security Descriptors List (SDL)

The SDL stores security descriptors corresponding to hosts in the protected network. In other words, the SDL is essentially a list of the IP addresses of other hosts in the protected network, and their security parameters. If the network security administrator wants to enable communication with a certain host, he (or she) must specify a security descriptor corresponding to that host. This security descriptor is used by the security system to process packets exchanged with that host. The security policy enforced by the security system blocks all inbound and outbound packets exchanged with a host, if the SDL does not contain a security descriptor corresponding to that host.

The security system stores the SDL in the Windows 2000 Registry encrypted with a master key, so that it is inaccessible to unauthorized users. Moreover, choosing the Registry to store the SDL prevents non-administrator users from modifying the contents of the SDL.

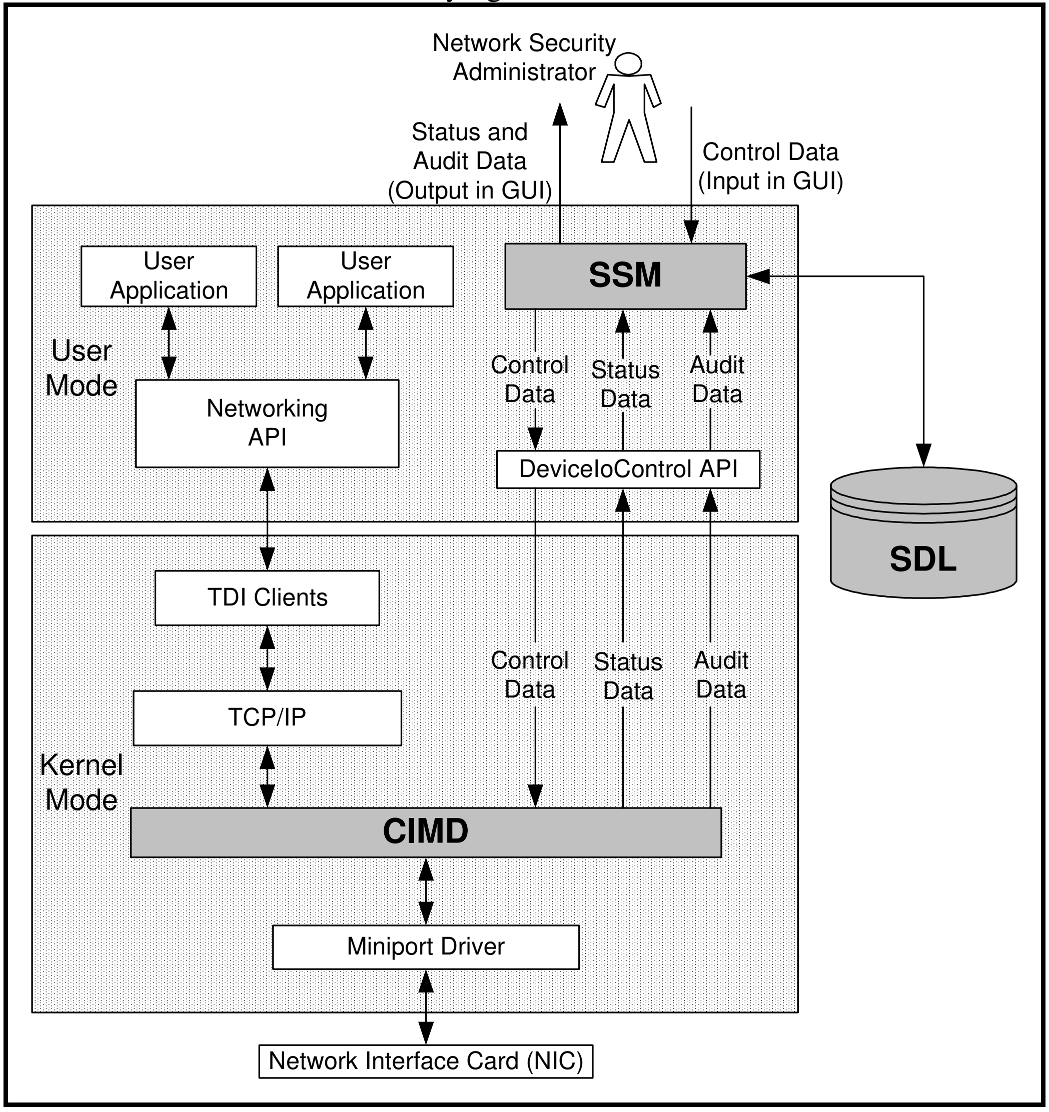

Fig.(1). Security System Architecture

Each security descriptor consists of the following security parameters:

- *IP Address:* This is the IP address of the host represented by the security descriptor.
- *Secure Packets:* This parameter states whether packets exchanged with the host represented by the security descriptor must be secured or not. This parameter is a Boolean flag. If its value is False, all other parameters except the IP Address parameter are ignored.

- *Hash Algorithm:* This parameter specifies the hash algorithm used in the authentication of packets exchanged with the host represented by the security descriptor. The security system implements two hash algorithms: SHA-1 (Secure Hash Algorithm-1) and MD5 (Message Digest 5).
- *Encryption Algorithm:* This parameter specifies the encryption algorithm used to encrypt or decrypt packets exchanged with the host represented by the security descriptor. The security system implements two encryption algorithms: Rijndael and Twofish.
- *Key Length:* This parameter specifies the length of the encryption key used to encrypt or decrypt packets exchanged with the host represented by the security descriptor. The length could be either 128 bits or 256 bits.
- *Encryption Key:* This parameter stores the encryption key used to encrypt or decrypt packets exchanged with the host.
- *Sequence Number Counter:* A monotonically increasing 32-bit value, written in the crypto headers of packets that will be sent to the host represented by the security descriptor. This number is used to detect replayed packets. It is also used as the number of packets, which have been secured with the security descriptor currently used key and algorithms, and which have been sent to the host represented by the security descriptor. This parameter is reset to zero if the security descriptor key or algorithms are changed.
- *Packets Received:* This parameter holds the number of packets that have been received from the host represented by the security descriptor, and processed with the security descriptor currently used key and algorithms. This parameter is reset to zero if the security descriptor key or algorithms are changed.
- *Anti-Replay Window:* A window that is used to detect replayed packets previously received from the host represented by the security descriptor. This parameter is reinitialized if the security descriptor key or algorithms are changed.
- *Bytes Sent:* This parameter holds the number of bytes that have been encrypted with the currently used key and encryption algorithm, and sent to the host represented by the security descriptor.
- *Bytes Received:* This parameter stores the number of bytes received from the host represented by the security descriptor and that have been decrypted with the currently used key and algorithm.

THE SECURITY SYSTEM MANAGER (SSM)

The SSM presents a Graphical User Interface (GUI) to the network security administrator, so that he (or she) can interact with the security system, and control its operation. The SSM provides the following functions:

- ♦ At initialization time, the SSM loads the encrypted SDL from the Registry, decrypts it, and passes its security descriptors to the CIMD. The CIMD stores the passed security descriptors in a linked list in its own memory context for fast access. The CIMD uses the security parameters of each security descriptor to process inbound and outbound packets exchanged with the host represented by that security descriptor, and it updates some of these parameters during processing.
- ♦ The network security administrator can use the SSM to add, remove, or change the parameters of a security descriptor corresponding to a host in the protected network. The SSM updates the SDL accordingly, and uses the DeviceIoControl API (Application Programming Interface) (The DeviceIoControl API sends a control code directly to a specified device driver, causing the corresponding device to perform the corresponding operation [SDK01]) to communicate with the CIMD, and inform it of the changes.
- ♦ The SSM periodically communicates with the CIMD using the DeviceIoControl API to get status data reported by the CIMD, and display it to the security administrator. The status data encompasses the mutable security parameters (that are updated by the CIMD) of all the security

descriptors. These parameters are: Sequence Number Counter, Packets Received, Anti-Replay Window, Bytes Sent, and Bytes Received.

♦ The SSM periodically communicates with the CIMD using the DeviceIoControl API to get the audit data associated with unauthenticated or replayed packets that were received by the CIMD, if any. The audit data associated with each suspicious received packet is stored in an entry in a linked list maintained by the CIMD. The audit data of each suspicious packet includes: the packet source IP address, the packet source MAC address, whether the suspicious packet is unauthenticated or replayed, the upper protocol data encapsulated by the IP packet, the packet total length, and the time when the suspicious packet was received. If there is audit data, the SSM alerts the network security administrator, and indicates the audit data to him (or her).

The Cryptographic NDIS IM Driver (CIMD)

The CIMD consists of two parts (like any other NDIS IM driver): the miniport part and the protocol part. The miniport part exposes miniport entry points (MiniportXxx functions), which NDIS calls to communicate the requests of one or more overlying protocol drivers. The miniport part in turn forwards these requests to the underlying miniport NIC (Network Interface Card) driver after performing the required security processing, if any.

The protocol part exposes protocol entry points (ProtocolXxx functions), which NDIS calls to communicate requests from underlying miniports. The protocol part in turn forwards these requests to overlying protocols after performing the required security processing, if any.

The CIMD inserts into each outbound IP packet that should be secured a 40-byte crypto header between its IP header and payload. This header consists of three fields:

1- The IV field (16 bytes) that holds the random initialization vector used in the encryption.
2- The Hash field (20 bytes) that holds a hash code calculated over the immutable fields of the IP header, the IP payload, and the Sequence Number field of the inserted crypto header. This hash code provides support for data integrity and authentication of the IP packet. The data integrity service detects any modification to the content of the packet in transit. The authentication service enables the receiving host to authenticate the sending host, and thus prevents address spoofing attacks.
3- The Sequence Number field (4 bytes), which is used to detect replayed packets. Each time that a packet is sent to a host, the sender increments the Sequence Number Counter of the security descriptor corresponding to that host, and places the resulting value in the Sequence Number field of the packet crypto header.

Then, the CIMD encrypts the payload of the IP packet along with the Hash and Sequence Number fields of the inserted crypto header using the encryption key of security descriptor representing the packet destination host. The encryption provides the required confidentiality to the exchanged data, and it protects the hash code held in the crypto header. Fig.(2 shows a typical outbound IP packet that becomes as shown in Fig.(3 after being processed by the CIMD.

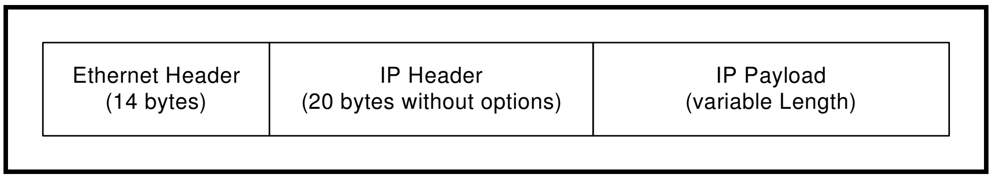

Fig.(2). A Typical Outbound IP Packet

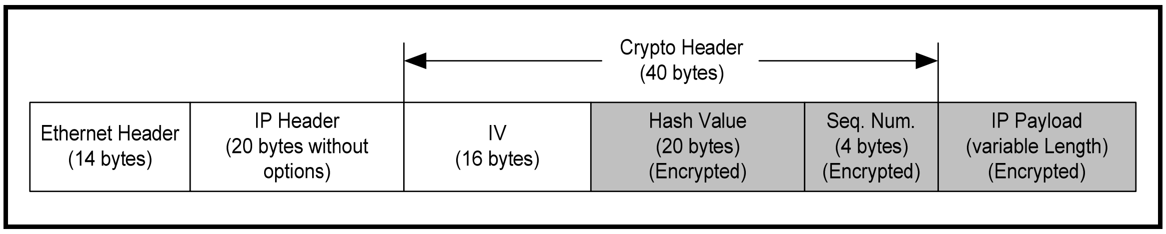

Fig.(3). The Outbound IP Packet after Being Processed by the CIMD

On reception, the CIMD decrypts the encrypted portions of each received IP packet, and then it inspects its crypto header. If the packet is properly authenticated and it is not a replayed packet, the CIMD strips off the crypto header, and then it delivers the packet to upper drivers. Otherwise, if the received packet is not properly authenticated or if it is a replayed packet, the CIMD collects audit information from this packet to be indicated to the network security administrator.

When the CIMD processes an outbound packet, it inserts the crypto header between the packet IP header and IP payload, so that routers in the protected LAN (if any) need not be changed, and they can route the packet properly. Leaving routers unaffected requires also that the CIMD does not encrypt the IP headers of outbound IP packets. However, the CIMD changes the value of the Total Length field of the IP header of each outbound secured packet to reflect the new packet length resulting after the insertion of the crypto header. The Checksum field of the IP header is also changed to a new value calculated over the new IP header.

The security system processes packets in layer 2 (data link layer) of the OSI (Open Systems Interconnection) protocol stack (between the LLC and MAC sublayers). Nevertheless, it does not provide link encryption, and it is considered to be an end-to-end cryptographic system.

LARGE PACKETS RESTRICTIONS

Since the CIMD increases the length of outbound IP packets by the size of the inserted crypto header (40 bytes), the processing of large IP packets may impose the following two problems:

The First Problem

If the CIMD inserts the crypto header into a large IP packet, whose size is larger than 1460 bytes, the size of the packet will become more than 1500 bytes after the insertion of the crypto header. This will cause the sending NIC to drop the packet, because maximum size of IP packets over Ethernet II is 1500 bytes [HOR84].

Hence, to resolve this problem, the security system explicitly sets the MTU (Maximum Transmission Unit) of each network interface to 1460 bytes instead of the default value (i.e. 1500 bytes). This must be done so that the TCP/IP protocol will not send to the CIMD IP packets larger than 1460 bytes. In Windows 2000, the MTU of an *interface* can be changed by modifying the following Registry value [MAC00]:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\interface\MTU.

The Second Problem

This problem results from the solution of the first problem. It occurs when a router that uses the security system routes a secured packet of size larger than 1460 bytes, and which was received from another host. Since the MTUs of the router network interfaces are set to 1460 bytes by the security system installed on it, the router will fragment the received secured packet. Apparently, fragmentation of a secured packet conflicts with the work of the security system. This is because the packet fragments other than the first one will not have crypto headers, and so will be considered unauthentic by the security system on the target host. The first fragment will not be authentic also, because its crypto header originally authenticates the total packet, not only the first fragment. The security system resolves this problem by disabling the Path MTU (PMTU) Discovery [MOG90] that is used by default by Windows 2000 for packets destined to a non-local host [MAC00].

Since the IP layer on the sending host is unaware of the existence of the CIMD that increases the length of outbound packets, PMTU discovery cannot resolve the problem of large packets sent to a non-local host stated above. This is because even if the IP layer adheres to the Next-Hop MTU value in the ICMP (Internet Control Message Protocol) Destination Unreachable messages reported by some router in the packet path, the CIMD still increases the length of outbound packets by the size of the inserted crypto header after it receives them from the IP layer. Thus, resulting packets will have a length 40 bytes more than the Next-Hop MTU reported by the router, and thereby causing the router to drop these packets.

In Windows 2000, when PMTU Discovery is disabled (which is the solution to the problem), an MTU of 576 bytes is used for all non-local destination addresses [MAC00]. Hence, after the CIMD inserts the crypto header into an outbound packet destined to a non-local host, the packet length will not exceed 616 bytes (576 bytes, the MTU + 40 bytes, the size of the crypto header). This packet length is well below the MTU of intermediate routers that use the security system. Thus, packets having this length will not be fragmented by the intermediate routers. The PMTU Discovery can be disabled by setting the following Windows 2000 Registry value to 0 [MAC00]:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\EnablePMTU Discovery

SECURITY SYSTEM IMPLEMENTATION AND PERFORMANCE RESULTS

The CIMD was written in C language and using NDIS library support routines, then it was compiled and built using the Windows 2000 DDK (Driver Development Kit). The SSM was developed using Microsoft Visual C++ language.

To measure the performance results, the security system was installed on 5 machines, which were connected as 100BaseTX Fast Ethernet by using Linksys 10/100 dual speed 16-port stackable Hub. The specifications of each machine were as follows:

• Processor:	Intel Pentium III, 866 MHz
• Physical Memory:	128 Mbytes
• Hard Disk:	Western Digital Caviar, 20 Gbytes
• LAN Card:	Realtek RTL8139(A)
• Operating System:	Windows 2000 Advanced Server

Three tests were performed on two machines (from now on, they will be called A and B) during the transfer of a 710-Mbyte file between them: network performance test, memory usage test, and processor usage test. The file was located in A, while B was used to get the file from A. All the results were measured using the Performance tool of Windows 2000. The tests were repeated on the two machines when they used the security system in each of the following cases:

1- The encryption algorithm is Rijndael, and the hash algorithm is SHA-1.
2- The encryption algorithm is Rijndael, and the hash algorithm is MD5.
3- The encryption algorithm is Twofish, and the hash algorithm is SHA-1.
4- The encryption algorithm is Twofish, and the hash algorithm is MD5.

For comparison purposes, the tests were also repeated in the following cases:

1- When the two machines deployed Windows 2000 IPSec with a pre-shared key as the authentication method, ESP as the used IPSec protocol, 3DES as the encryption algorithm, and SHA-1 as the hash algorithm used by the IPSec HMAC.
2- When nothing was used to secure network traffic. This situation will be referred to as: "Normal Case".

The following subsections present and discuss the results of all the performed tests.

Network Performance Test

The average packets/sec Performance tool counter was used to measure the network performance. Table shows the results of the average packets/sec counter in all the cases.

When the security system is deployed, the degradation in the average packets/sec rate with respect to the Normal Case was about: 31.7% for (Rijndael,MD5), 35.6% for (Rijndael,SHA-1), 44.4% for (Twofish,MD5), and 45.5% for (Twofish,SHA-1). On the other hand, the use of the IPSec service with a pre-shared key as the authentication method, ESP as the used IPSec protocol, 3DES as the encryption algorithm, and SHA-1 as the hash algorithm caused a degradation of about 30.9%. The degradation in the average packets/sec rate is a normal result because of the delay introduced by the security related processing on the two machines.

Table (1). The Results of the Network Performance Test

Case	Packets/sec at A	Packets/sec at B
Normal Case	10574.911	10572.754
IPSec	7307.058	7306.679
Security System (Rijndael,SHA-1)	6807.960	6809.903
Security System (Rijndael,MD5)	7218.649	7217.341
Security System (Twofish,SHA-1)	5761.566	5759.622
Security System (Twofish,MD5)	5879.617	5879.498

Processor Usage Test

Processor usage was measured using the %Processor Time Performance tool counter, which is the percentage of time the processor is executing a non-Idle thread. This counter was designed as a primary indicator of processor activity. It is calculated by measuring the time that the processor spends executing the thread of the Idle process in each sample interval, and subtracting that value from 100% (Each processor has an Idle thread, which consumes cycles when no other threads are ready to run) [MSP01].

Notice that the results of this performance measure may not be too accurate because they may be affected by other services and applications that were running on A and B during the file transfer operation. The results of the test are shown in

Table (1. Before the transfer operation started, the %Processor Time was measured at A and B, and its average was nearly 0.2% at each one of them.

When either the security system or IPSec is used to secure network traffic, %Processor Time increases apparently due to the required security related processing. However, the increase in processor usage does not differ much for all the cases that have security processing.

Table (1 shows that the processor usage at B is larger than its usage at A in all the cases. This may be due to the fact that B was the machine that initiated and drove the file transfer operation.

Table (1). The Results of the Processor Usage Test

Case	%Processor Time at A	%Processor Time at B
Normal Case	41.757 %	47.903 %
IPSec	84.787 %	85.212 %
Security System (Rijndael,SHA-1)	84.522 %	89.804 %
Security System (Rijndael,MD5)	83.674 %	88.590 %
Security System (Twofish,SHA-1)	85.060 %	90.825 %
Security System (Twofish,MD5)	86.638 %	88.600 %

Memory Usage Test

Memory usage was investigated using the Available Mbytes counter of the Performance tool. As for the processor usage test, the results of this performance test may not be too accurate because they may be affected by other services and applications that were running on A and B when the file was transferred between them.

Before the file transfer operation started, the memory available at A was: 25.5 Mbytes, while that available at B was: 42 Mbytes. This difference is due to the fact that different services and applications were running on the two machines. Table (3 shows the results of the Available Mbytes counter at A and B when the file was transferred between them in all the cases. The memory used for file transfer at each machine was calculated by subtracting the available memory during the file transfer from the available memory before the file transfer started.

As shown in the table, the memory used at each machine does not differ much for all the cases. It may also be noticed that the memory used at B is more than that used at A in all the cases. This may be because B is the machine that initiated and drove the file transfer operation.

Table (3). The Results of the Memory Usage Test

Case	Available Mbytes at A	Used Mbytes at A	Available Mbytes at B	Used Mbytes at B
Normal Case	22.500	3	36.383	5.617
IPSec	22.680	2.8	37.030	4.97
Security System (Rijndael,SHA-1)	21.000	4.5	35.230	6.77
Security System (Rijndael,MD5)	21.000	4.5	36.770	5.23
Security System (Twofish,SHA-1)	20.090	5.41	34.000	8
Security System (Twofish,MD5)	21.170	4.33	33.570	8.43

CONCLUSIONS

The most important points concluded throughout the design and implementation of the security system are listed below:

- To provide security to the data exchanged within a network, encryption and message authentication mechanisms must be used. However, these mechanisms impact the average packets/sec rate in the network, which degrades by a percentage depending on the complexity of the used algorithms. Moreover, these security mechanisms increase processor usage on the hosts that perform them approximately to the double.
- The development of an end-to-end cryptographic system that works in layer 2 (data link layer) of the OSI reference model can be made much easier, if the protected protocol that works in layer 3 (network layer) is aware of the existence of the cryptographic system. This may require that the

protected protocol and the cryptographic system be developed by the same party, or by cooperative parties.

- Windows network drivers represent a powerful and efficient method to transparently intercept and process inbound and outbound network data. Among the well-documented network drivers, the NDIS intermediate driver is the more powerful, and more efficient one.
- If a network driver below the IP layer (such as a NDIS intermediate driver) inserts a number of bytes into outbound packets that will be sent on a certain network interface, the MTU of that interface must be decreased by the number of bytes that will be inserted, so that large packets are not dropped by the network interface.
- If a NIC has its NDIS task offload capabilities enabled, these capabilities may improperly affect the work of installed packet-modifying NDIS intermediate drivers. Hence, these capabilities should be turned off for the modifying intermediate drivers to work properly.

REFERENCES

[ALL01] J. H. Allen, *"The CERT Guide to System and Network Security Practices"*, Addison Wesley Professional, June 2001.

[DDK00] Microsoft Corporation, Microsoft Windows 2000 Driver Development Kit, *"Network Drivers"*, 2000.

[GAN00] A. Ganz, S. H. Park, and Z. Ganz, *"Security Broker for Multimedia Wireless LANs"*, Elsevier Science B.V., 2000.

[HOR84] C. Hornig, *“A Standard for the Transmission of IP Datagrams over Ethernet Networks”*, RFC 894, April 1984.

[MAC00] D. MacDonald and W. Barkley, *"Microsoft Windows 2000 TCP/IP Implementation Details"*, Microsoft Corporation, 2000.

[MOG90] J. Mogul and S. Deering, *“Path MTU Discovery”*, RFC 1191, November 1990.

[MSP01] Microsoft Corporation, Microsoft Development Network, Windows 2000 Resource Kit Reference, "Windows 2000 Performance Counters Reference", April 2001.

[NUO02] A. Nuopponen and S. Vaarala, *"Attacking Predictable IPsec ESP Initialization Vectors"*, Helsinki University of Technology, 2002.

[RAD01] E. Rademer and S. D. Wolthusen, *"Transparent Access to Encrypted Data Using Operating System Network Stack Extensions"*, Mitsubishi Corporation and Fraunhofer-IGD, 2001.

[SCH99] B. Schneier and N. Ferguson, *"A Cryptographic Evaluation of IPSec"*, Counterpane Internet Security, Inc., 1999.

[SDK01] Microsoft Corporation, Microsoft Development Network, Platform SDK Documentation, April 2001.

[SHA02] B. S. Shaker, *"Design and Implementation of an Online Cryptography System for LANs"*, M.Sc. thesis, College of Engineering, Al-Nahrain University, 2002.

[STA99] W. Stallings, *"Cryptography and Network Security: Principles and Practice"*, Second Edition, Prentice Hall, 1999.

[TRC99] D. Trcek, T. Klobucar, B. Jerman-Blazic, and F. Bracun, *"CA-Browsing System - a Supporting Application for Global Security Services"*, a white paper, Jozef Stefan Institute, 1999.