Establishment of a secure networking between Secure OSs

Jae-Deok Lim*, Joon-Suk Yu*, and Jeong-Nyeo Kim*

* Network Security Department, Information Security Research Division,
Electronics and Telecommunications Research Institute(ETRI)
(Tel : +82-42-860-1522; E-mail: {jdscol92, jsyu92, jnkim}@etri.re.kr)

Abstract: Many studies have been done on secure operating system using secure kernel that has various access control policies for system security. Secure kernel can protect user or system data from unauthorized and/or illegal accesses by applying various access control policies like DAC(Discretionary Access Control), MAC(Mandatory Access Control), RBAC(Role Based Access Control), and so on. But, even if secure operating system is running under various access control policies, network traffic among these secure operating systems can be captured and exposed easily by network monitoring tools like packet sniffer if there is no protection policy for network traffic among secure operating systems. For this reason, protection for data within network traffic is as important as protection for data within local system. In this paper, we propose a secure operating system trusted channel, SOSTC, as a prototype of a simple secure network protocol that can protect network traffic among secure operating systems and can transfer security information of the subject. It is significant that SOSTC can be used to extend a security range of secure operating system to the network environment.

Keywords: access control, packet protection, secure operating system, trusted channel

1. INTRODUCTION

In recent years, business environment of government organizations and companies is being changed steadily into network-based environment like LAN(Local Area Network) or KMS(Knowledge Management System). Accordingly, the information can be simply delivered to the inside and/or outside and can be easily accessed in network-based business environment. But this increases the danger of a leakage of information. Especially a drain of information by internal intruder is being more increased than an external intrusion by hackers.

In this regard, many security systems are under development for protecting data and system from intruder, such as IDS(Intrusion Detection System) and Firewall. But they are exposed to the limitation that the IDS do not protect from new attack methods, and the firewall is defenseless if the intruder is already in system. And system security patches and version upgrades were being applied but they're also exposed to some limitations. Being increased the needs of the fundamental data protection and security, has been issued a secure OS(operating system) that applied various access control policies like DAC(Discretionary Access Control), MAC(Mandatory Access Control), RBAC(Role Based Access Control) and so on[1-3]. The degree of their functionalities becomes a evaluation basis of system security level[4]. Access control policy defines an access permission of a subject to an object based on their relationship. When access control policy is applied, the local system will be protected from the access of unauthorized users.

But, even if secure OS is running under various access control policies, network traffic among these secure OSs can be captured and exposed easily by network monitoring tools like packet sniffer if there is no protection policy for network traffic among secure OSs. For this reason, protection for data within network traffic is as important as protection for data within local system.

Now, many techniques for protecting network traffic have been studied and introduced and IPSec protocol is representative of network security in IP level[5]. It is standardized technology for guaranteeing integrity and confidentiality of network traffic. However care of management and configuration of security policy is required to the manager and it is indicated as a weak point that complexity and overhead of key exchange mechanism. Also, in order to provide a logically single secure computing environment when secure OSs are connected together, it must be provided that the ability of delivery of security information that is attached to the process(user) transferring data. But IPSec protocol is not suitable for delivery of security information.

In this paper, we propose a secure operating system trusted channel, SOSTC, as a prototype of secure network protocol that can provide integrity and confidentiality of network traffic among secure OSs and can maintain consistency of security information by transferring security information of the process(user) transferring data. It is significant that SOSTC can be used to extend a security range of SOS to the network environment by providing secure communication channel that guarantees integrity and confidentiality of network traffic among secure OSs. For this, we provide the encapsulation header, called SOSTC header, and minimize the overhead by simplifying network traffic processing.

The remainder of this paper is organized as follows. Section 2 outlines secure OS in which SOSTC is implemented. Section 3 describes the design and implementation of SOSTC. Section 4 shows the performance of SOSTC and finally we conclude with summary and future works in section 5.

2. SECURE OPERATING SYSTEM

SOSTC is implemented and integrated into secure OS that is introduced in [1] and provides various access control policies. In [1], when a common user is created, he is assigned a pair of security class and category for MAC policy and roles for RBAC policy from a security manager who can set security configurations[1-3]. The security class and the category of MAC policy represents a hierarchical concept and a horizontal group concept respectively and defines a security level of user in secure OS. A security manager is a user who has a security manager role which defines in [1] to be able to set a security configuration and is distinguished from system manager who is so-called 'root' user. A user who want to be a security manager should try to login with security manager role. Of course, the user should be conFig.d to be able to have a security manager role in advance. The role of security manager is defined as '#security_manager' in [1].

In most conventional systems, root user can set any configuration of local system. But the right of root user is reduced in SOS because configuration related to security is distinguished from configuration related to system operation. It can be examples of configuration related to security that MAC, RBAC, audit trail and SOSTC which is proposed in this paper. This concept can be effective provision against most attacks that can occur by acquiring the right of root user using many security holes of system. A user should try to login within the range of his own security information that is MAC and RBAC labels which are assigned from a security manager when user is created. This information is used when access control polices are applied to the user. When a user try to login with default information(security class:0, category:0, role:none), he has the same right as one in conventional system and, although he has the right of root user, he cannot access to objects, e.g. files, directories, and so on, that are protected with MAC and RBAC policies.

In this point, when the object, namely the file, is transferred to other secure OS, the security information, i.e. MAC information, attached to the object to be transferred should be transferred together with the object. This is why the consistency of security information should be maintained among secure OSs. Also, while the object is transferred, its contents should not be exposed and forged by any means. SOSTC proposed in this paper can transfer the security information of object and can provide integrity and confidentiality of network traffic.

3. SECURE OS TRUSTED CHANNEL(SOSTC)

3.1 Design and Implementation

SOSTC was implemented to provide a part of items recommended in CC(Common Criteria) within FreeBSD-based secure OS introduced in [1]. The items provided in SOSTC are some functions of 'Class FTP: Trusted path/channels' in Part II of CC[4]. Because SOSTC conducts encryption and authentication of packet within kernel level, it guarantees confidentiality and integrity of network packets and provides transparency to users. The architecture of SOSTC is shown in Fig. 1. SOSTC is working within IP protocol stack.

SOSTC consists of three parts which are as follows.

STIP(SOSTC Initialization Part) loads the initial data needed for SOSTC into kernel memory at the system boot-time. The initial data are an encryption key for packet encryption, an authentication key for packet authentication and IP addresses of secure OS to be applied to SOSTC. An encryption key and an authentication key are self-encoded within kernel for increasing security before being loaded into kernel memory. The initial data are stored in their own file and these files can be protected with RBAC policy of SOS as Fig. 2. We assume that all the secure OSs which will use SOSTC have shared the IP addresses of secure OS, the algorithm and key for encryption and authentication in advance.

STDP(SOSTC Decision Part) determines whether SOSTC is applied to packet. It is different that the procedure of STDP in input and output processing of packet respectively. In the output processing, if the destination address of a sending packet is one of addresses of host loaded into kernel memory for applying SOSTC, that is, the destination of packet is SOS, the packet is passed to STPP for SOSTC output processing. In the input processing, if the 'next_protocol' field of IP header of a received packet is the value defined as SOSTC, the packet is passed to STPP for SOSTC input processing.

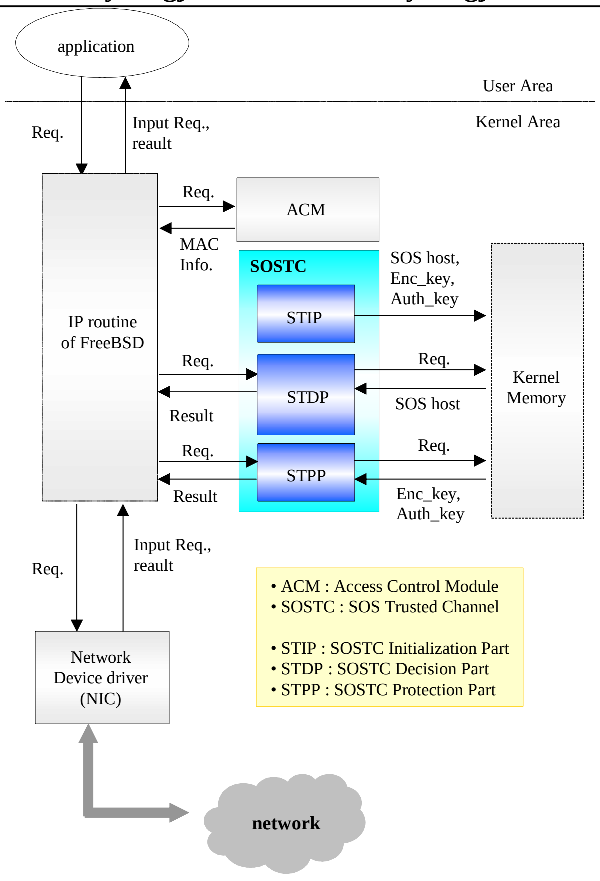
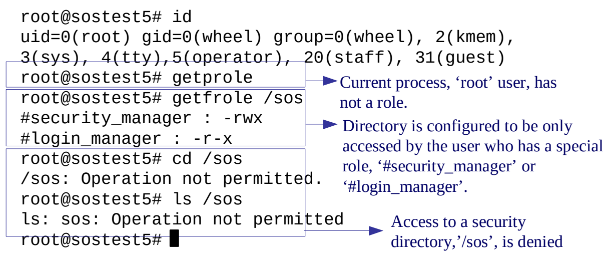

application
User Area
Kernel Area
Req.
Input Req., reault
IP routine of FreeBSD
Req.
ACM
MAC Info.
SOSTC
STIP
SOS host, Enc_key, Auth_key
Req.
STDP
Req.
SOS host
Result
Kernel Memory
Req.
STPP
Req.
Result
Enc_key, Auth_key
Req.
Input Req., reault
• ACM : Access Control Module
• SOSTC : SOS Trusted Channel
• STIP : SOSTC Initialization Part
• STDP : SOSTC Decision Part
• STPP : SOSTC Protection Part
Network Device driver (NIC)
network

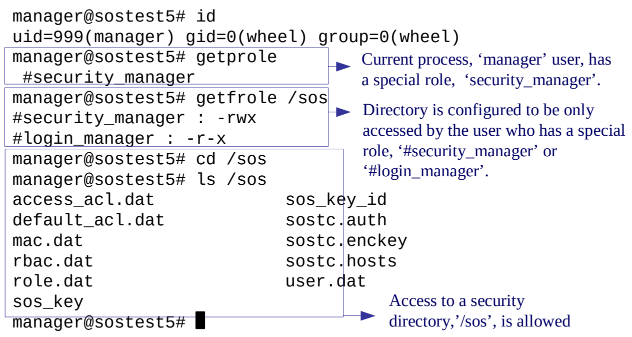
Fig. 1. The architecture of SOSTC.



```
root@sostest5# id
uid=0(root) gid=0(wheel) group=0(wheel), 2(kmem),
3(sys), 4(tty),5(operator), 20(staff), 31(guest)
root@sostest5# getprole
root@sostest5# getfrole /sos
#security_manager : -rwx
#login_manager : -r-x
root@sostest5# cd /sos
/sos: Operation not permitted.
root@sostest5# ls /sos
ls: sos: Operation not permitted
root@sostest5#
```

Current process, 'root' user, has not a role.
Directory is configured to be only accessed by the user who has a special role, '#security_manager' or '#login_manager'.
Access to a security directory,'/sos', is denied

(a) The case that directory is protected: In spite of root user, he cannot access to the '/sos' directory because he does not have the role, #security_manager, that is attached to the directory.



```
manager@sostest5# id
uid=999(manager) gid=0(wheel) group=0(wheel)
manager@sostest5# getprole
 #security_manager
manager@sostest5# getfrole /sos
#security_manager : -rwx
#login_manager : -r-x
manager@sostest5# cd /sos
manager@sostest5# ls /sos
access_acl.dat              sos_key_id
default_acl.dat             sostc.auth
mac.dat                     sostc.enckey
rbac.dat                    sostc.hosts
role.dat                    user.dat
sos_key
manager@sostest5#
```

Current process, 'manager' user, has a special role, 'security_manager'.
Directory is configured to be only accessed by the user who has a special role, '#security_manager' or '#login_manager'.
Access to a security directory,'/sos', is allowed

(b) The case that directory can be accessed: the manager can access to the '/sos' directory because he has the role, #security_manager, that is attached to the directory.

Fig 2. Example of protecting secure files in '/sos' directory from accessing by unauthorized users using RBAC policy.

STPP executes a SOSTC processing essentially. In the output processing, STPP makes a SOSTC packet with SOSTC header, executes an encryption of packet for confidentiality and computes an authentication data of packet for integrity. In the input processing, STPP checks the authentication data of packet for integrity and executes a decryption of packet and restores SOSTC packet to common IP packet by removing SOSTC header.

Fig. 3 shows the SOSTC header format for SOSTC communication and SOSTC packet. The shaded area indicates SOSTC header. SOSTC header consists of authentication data field for packet integrity, initial vector and padding length field for packet encryption, next protocol field for proper packet processing in the upper protocol level, and so on.

Especially, it has MAC information filed to deliver MAC class and category of process(user) who tries to send a packet. This can maintain the consistency of security information in a network environment consisted of secure OSs.

SOSTC packet will be described in the next section.

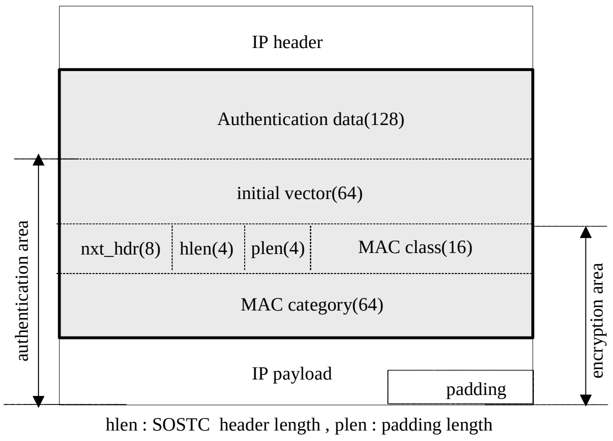


Fig 3. SOSTC Header and SOSTC packet

3.2 Packet Processing

The processing of SOSTC packet is done as follows in the input and output routine of IP layer respectively.

In the output processing, when the request of data sending from user process is issued, the relevant packet is passed to the IP output routine after processing in a upper protocol output routine, e.g. TCP output routine[6-7]. After packet processing except fragmentation is done in IP output routine, this packet is determined in STDP whether SOSTC is applied to it or not by comparing its destination IP address with secure OS IP address loaded into kernel memory at the boot-time. If SOSTC is needed, the packet is passed to STPP and then SOSTC packet is created with SOSTC header. After this, SOSTC packet is configured completely by encrypting an encryption area and computing an authentication data with authentication area as shown Fig. 3. After completing to configure a SOSTC packet, the remaining process of IP layer, namely fragmentation, is executed if needed. And then the packet is passed to the lower protocol output routine to be sent. If SOSTC is not applied, the packet to be sent is a common IP packet.

When packet is received, it is passed to the IP layer from the lower protocol layer, e.g. data link layer[6-7]. The processing of the packet is done for the most part and the packet is passed to STDP just before sending to the upper protocol input routine, i.e. just after reassembling of the packet in IP input routine. In STDP, this packet is determined whether SOSTC is applied to it or not by checking the next protocol field of IP header. If the field indicates SOSTC, the packet is passed to the STPP for SOSTC processing. In STPP, the integrity of the packet is checked with authentication data in SOSTC header. If the authentication data is invalid, packet is discarded. Only if the authentication data is valid, the process is continued. After checking the integrity, the packet is decrypted and restored to common IP packet by removing SOSTC header and padding. And then this packet is passed to the upper protocol input routine, e.g. TCP input routine. Fig. 4 shows the diagram of SOSTC packet processing in the input and output routine of IP layer.

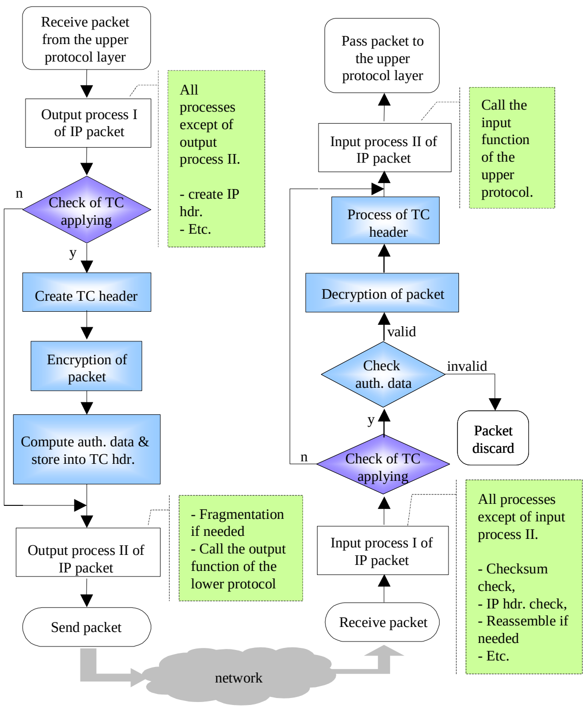

Receive packet from the upper protocol layer
Output process I of IP packet
All processes except of output process II.
- create IP hdr.
- Etc.
n
Check of TC applying
y
Create TC header
Encryption of packet
Compute auth. data & store into TC hdr.
Output process II of IP packet
- Fragmentation if needed
- Call the output function of the lower protocol
Send packet
network
Pass packet to the upper protocol layer
Input process II of IP packet
Call the input function of the upper protocol.
Process of TC header
Decryption of packet
valid
Check auth. data
invalid
Packet discard
y
n
Check of TC applying
Input process I of IP packet
All processes except of input process II.
- Checksum check,
- IP hdr. check,
- Reassemble if needed
- Etc.
Receive packet

Fig 4. The processing of SOSTC packet in the input and output routine of IP layer.

In STPP, it is used that blowfish encryption algorithm with CBC(Cipher Block Chaining) mode for confidentiality of packet and HMAC-MD5 authentication algorithm for integrity of packet. Blowfish algorithm has a better performance in speed than DES and IDEA that are used widely and can have a key of various length[8-9]. The key of 128-bit length is used in this paper.

4. PERFORMANCE

We can expect easily that the performance of network with SOSTC is not as good as that of network without SOSTC because when SOSTC is applied, additional processes related to encryption and authentication for packet are required essentially and these works may consume many computing resources.

System used for testing had a pentium IV 1.4GHz CPU and 256 MB main memory. The test of SOSTC was done on secure OSs introduced in [1] and that of non-SOSTC was done on FreeBSD 4.3 in the same system condition with secure OS. Fig. 5 shows the performance of file transferring with SOSTC and without SOSTC. The processing time is the mean value of three times of test that file is transferred to other system in respective file size shown in graph by FTP utility.

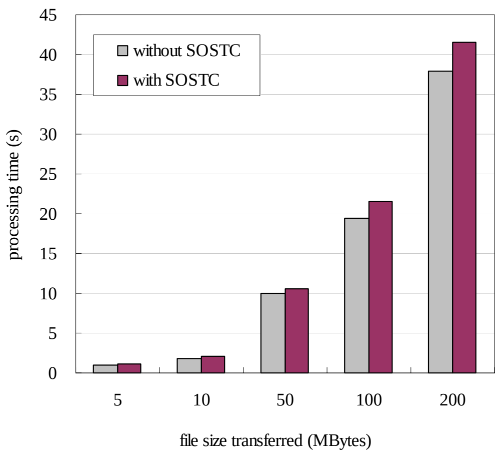


Fig 5. The performance of network with SOSTC and without SOSTC.

As expected, we can see the overhead of SOSTC in Fig. 5, yet the performance of network with SOSTC has 80-90% of that of network without SOSTC. Also the overhead is reduced gradually as file size is smaller. This is why the executions related to SOSTC are reduced gradually as file size is smaller.

Actually if it is considered that the size of most data transferred in common network environment is smaller than that of the tested data in this paper except multimedia data, SOSTC can be provided for confidentiality and integrity of packet without complaint. The main overhead of SOSTC is the processing of encryption and authentication for packet. This can be reduced by using the speed-optimized algorithm and/or hardware-based security processor.

5. CONCLUSION

The interest and demand of system security are being increased more and more in the situation that the damages caused by a drain of information through the internal intruder as well as the system attack through the external intruder are being increased. Accordingly many studies have been done on secure operating system using secure kernel that has various access control policies for system security. But access control policies are insufficient to protect network traffic. As system operation based on network environment is increased, data protection must be achieved on network traffic as well as on local host.

In this paper, we proposed a secure operating system trusted channel, SOSTC, that can provide secure communication to secure OSs. SOSTC can provide network traffic among secure OSs with integrity and confidentiality and can maintain consistency of security information by transferring security information of the subject. It is significant that SOSTC can be used to extend a security range of SOS to the network environment by providing communication channel that guarantees integrity and confidentiality for network traffic among secure OSs. Though the overheads arise from the processing related to SOSTC, it is small enough to use without inconvenience comparing with common network environment. Also, this can be reduced by using the speed-optimized algorithm and/or hardware-based algorithm.

But, it is necessary that the protection of configuration files that have SOSTC configuration data, that is, authentication key, encryption key and list of secure OSs in which SOSTC is running should be studied more deeply. Although this file can be protected by RBAC and/or MAC policies, it can be a potential security hole that these files are stored in local host.

REFERENCES

[1] J. G. Ko, J. N. Kim, & K. I. Jeong, “Access Control for Secure FreeBSD Operating System,” *Proc. of WISA2001, The Second International Workshop on Information Security Applications, 2001*, Vol. 2, pp.247-254, 2001.

[2] Bell, David Elliott, & Leonard J. La Padula, “Secure computer system: Unified exposition and multics interpretation,” *MITRE Technical Report 2997*, MITRE Corp, Bedford, MA, 1975.

[3] David F. Ferraiolo, Ravi Sandu, & Serban Gavrila, “A Proposed Standard for Role-Based Access Control,” *ACM transaction on Information and System Security*, VOL.4, NO.3, pp.224-274, Aug. 2001.

[4] “Class FTP: Trusted path/channels,” *Common Criteria for Information Technology Security Evaluation, Part 2: Security functional requirements*, Version 2.1, pp.167-170, 1999.

[5] S. Kent, R. Atkinson, “Security Architecture for the Internet Protocol,” *RFC 2401*, Nov. 1998.

[6] FreeBSD 4.3-RELEASE Source Code

[7] Behrouz A. Forouzan, Sophia Chung Fegan, *TCP/IP Protocol Suite*, Boston : McGraw-Hill, 2000.

[8] B. Schineier, “Description of a New Variable-Length Key, 64-Bit Block Cipher(Blowfish),” *Fast Software Encryption, Cambridge Security Workshop Proceedings*, Springer-Verlag, 1994.

[9] B. Schineier, *Applied Cryptography*, John Wiley & Sons, pp. 336-339, 1996.