A secure method for transferring active packets

YOUNGSOO KIM, JUNGCHAN NA, SEUNGWON SOHN
Network Security Department, Information Security Technology Division, ETRI
161 Kajong-Dong, Yusong-Gu, Taejon 305-600
KOREA

Abstract: - Active networks are obviously more complex than traditional networks and raise considerable security issues. A packet that carries executable code can potentially change the state of a node. Nodes (routers, switches, etc.) are public resources and are essential to the proper and correct running of many important systems. Therefore, security requirements placed upon the computational environment where the code of packets will be executed must be very strict. On the other hand, it is one of the critical security issues to execute active packets in intermediate nodes securely. Because active packets are executed in intermediate nodes as well as end nodes, some conventional cryptographic solutions are not available. In this paper, we propose a new scheme that can transfer active packets to all neighboring active nodes securely, and execute executable code included in those packets in each active node. We use both public key cryptosystem and asymmetric key cryptosystem in our scheme.

Key-Words: - Active networks, Active packets, Active nodes, Executable code, Cryptosystem

1 Introduction

Traditionally, the function of a network has been to deliver packets from one endpoint to another. There was a distinct boundary between what is done within the network and what is done by the users. Processing within the network was limited basically to routing, congestion control and quality of service schemes. This kind of a network can be regarded as 'passive'. Several problems with 'passive' networks have been identified: the difficulty of integrating new technologies and standards into the shared network infrastructure, poor performance due to redundant operations at several protocol layers, and difficulty accommodating new services in the existing architectural model. An additional shortcoming is that, recently, applications which sometimes require computations within the network have emerged, such as firewalls, web proxies, multicast routers, and mobile proxies. In the absence of architectural support for doing so, these applications have adopted a variety of ad hoc services for performing user-driven computations at nodes within the network. A need was felt to replace the numerous ad hoc approaches to network-based computation, with a generic capability that allows the users to program their networks. This innovative idea of imparting the user the ability to program the network is called active networking [1].

Active networks represent a new approach to network architecture. These networks are 'active' in two ways: routers and switches within the network can perform computations on user data flowing through them; And users can 'program' the network, by supplying their own programs to perform these computations.

On the other hand, since active networks are much more flexible than passive, the number of security issues are tremendously increased [2]. By safety we mean reducing the risk of mistakes or unintended behavior. By security we mean the usual concept of protecting privacy, integrity, and availability in the face of malicious attack. A packet that carries executable code can potentially change the state of a node. Nodes (routers, switches, etc.) are public resources and are essential to the proper and correct running of many important systems. Therefore, security requirements placed upon the computational environment where the code of packets will be executed must be very strict.

In the current Internet, the only resource consumed by a packet at a node is the memory needed to temporarily store it and the CPU cycles necessary to find the correct route. In such an environment, strict resource control in the intermediate nodes was considered non-critical. However, an active packet

may consume not only many more resources but also at a faster rate.

In an active network, active packets may misuse active nodes, network resources, and other active packet in various ways. Also, active nodes may misuse active packets. But, protecting the nodes and the packets in a flexible environment such as active networks is not an easy task. Some techniques that may be used to protect both of them are suggested [3] (e.g., monitoring/control, authentication of active packets, and limitation techniques, and so on). However, because these techniques are still in their infancy, there is much to be done before definite results are reached.

On the other hand, it is one of the critical security issues to execute active packets in intermediate nodes securely. Because active packets are executed in intermediate nodes as well as end nodes (e.g., origin node and destination node), some conventional cryptographic solutions are not available [4].

In this paper, we propose a new scheme that can transfer active packets to all neighboring active nodes securely, and execute program included in those packets in each active node. We use both public key cryptosystem and asymmetric key cryptosystem (i.e., conventional cryptosystem) in our scheme.

The rest of the paper is organized as follows: Section 2 presents some assumptions and settings needed to apply our scheme. And, in section 3 we describe our scheme in detail. Finally, we conclude our findings in section 4.

2 Network setups and an active packet flow

In this section, we consist a network and show how an active packet flows in our network briefly.

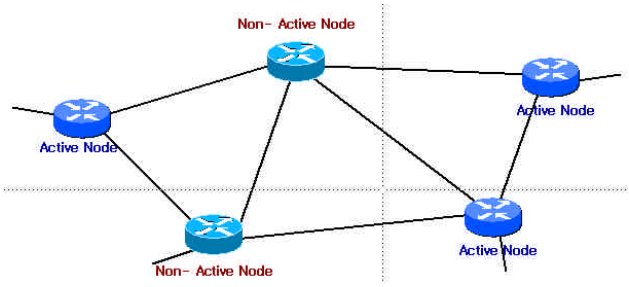

Fig.1 Network setups

Fig.1 shows network configurations for our idea. There exist active nodes and non-active nodes (passive nodes) together. If a passive node receives an active packet, it ignores and simply forwards it to neighboring nodes.

If an active node(Sending node) has executable code which needs to be delivered to the next active node securely, he/she generates an active packet and encrypts payload of it using symmetric key cryptosystem (Triple DES [5], Rijndael [6], etc.). An active packet consists of IP header, originator's IP address, active packet header and payload. Payload contains target executable code. Fig.2 shows active packet format. Originator of this active packet broadcasts it to neighboring active nodes, because he/she does not know addresses of them. If passive nodes receive this active packet, they ignore and forward it simply. If a neighboring active node(Receiving node) receives this encrypted active packet, he/she generates a new active packet contains his/her public key(in public key cryptosystem) and an request message for getting secret key(in symmetric key cryptosystem). Since he/she is able to get origin's IP address from the received active packet header, he/she sends the generated active packet there. If sending node receives this active packet, he/she checks header and payload of it, and encrypts secret key with the receiving node's public key using public key cryptosystem(RSA [7], etc.). This public key is included in the payload of received active packet. Sending node generates an active packet includes the encrypted secret key and sends it to the receiving node. If the receiving node receives that packet from the sending node, he/she decrypts encrypted secret key using his/her private key(in public key cryptosystem) and gets clear executable code from the encrypted active packet by decrypting it with the secret key(in symmetric key cryptosystem). Finally, after receiving node executes that program, he/she generates a new encrypted active packet and broadcasts it to neighboring active nodes. In this case, the receiving node becomes the sending node.

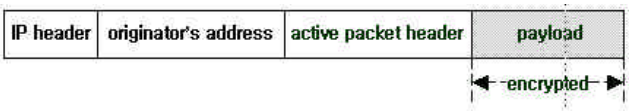

Fig.2 Active packet format

3 A proposed scheme

In this section, at first, we make some basic assumptions to clear our idea and describe some notations to explain operation processes easily. And we explain our operation processes in detail.

3.1 Assumptions

To clear our idea, we need the following assumptions:

- An active node does not know IP addresses of the neighboring active nodes
- Each domain has a CA(Certification Authority [8]) and CA is a TTP(Trusted Third Party [9]). In this case, domain means network or subnetwork.
- Each active node in a domain is already registered in CA.
- Each active node has his/her own secret key and public-key/private-key pairs.
- CA issues certificates, digitally signed with CA's key, for each active node at registration. A digital signature scheme used is a heuristically existentially unforgeable signature scheme such as a Schnorr[10] signature or an RSA signature with an appropriate hash function before signing.
- At all intermediate active node, located in between end nodes, executable code in active packets is executed.

3.2 Notations

Some notations used in our scheme are as follows:

- KE_A: Active node A's public key (in public key cryptosystem)
- KD_A: Active node A's private key (in public key cryptosystem)
- KS_A: Active node A's secret key (in asymmetric key cryptosystem)
- CERT_A: Active node A's certificate (in public key cryptosystem)
- PGM: Executable code included in an active packet
- $ENC_X(Y)$: Encrypting Y with key X using public key cryptosystem
- $DEC_X(Y)$: Decrypting Y with key X using public key cryptosystem
- $E_X(Y)$: Encrypting Y with key X using asymmetric key cryptosystem
- $D_X(Y)$: Decrypting Y with key X using asymmetric key cryptosystem
- $SIG_X(Y)$: Signing Y with key X using digital signature scheme
- VER(S): Verifying signature S
- CA: Certification authority
- K_CA: CA's key for issuing certificates
- INFO: Some informations included in a certificate, for example, a cryptographic algorithm used, expiry date, issue date, etc.
- REQ(Y): Request message for getting Y

3.3 Operation processes

Fig.3 depicts operation processes between active node A (Sending node) and active node B (Receiving node).

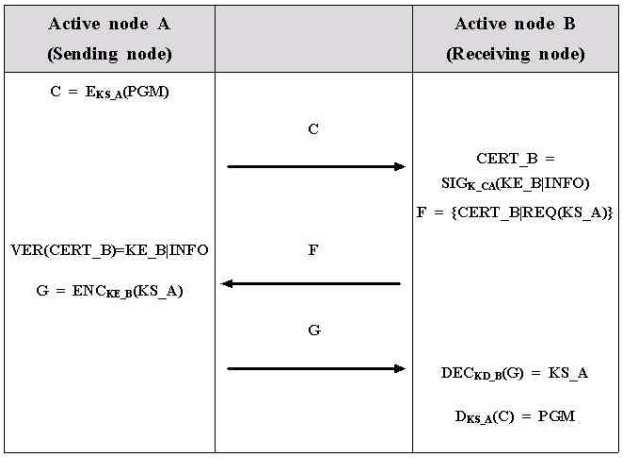

Fig.3 Operation processes

Active node A has executable code which he/she received from other active node or generated himself/herself. It must be delivered to the next active node securely. He/She wants to send that executable code to neighboring active nodes and have all intermediate active node execute them. Operation processes are as follows:

1. Active node A (Sending node) generates an active packet. Target executable code is included in payload of this active packet. Sending node encrypts payload of it using symmetric key cryptosystem.
   $$C = E_{KS\_A}(PGM)$$
2. Active node A broadcasts it to neighboring active nodes, because he/she does not know IP addresses of them.
3. If neighboring active node B (Receiving node) receives this encrypted active packet(=C), he/she checks if it is active packet through active packet header. And he/she records this active packet's IP address and C value in his/her table.
4. Since active node B needs active node A's secret key KS_A to decrypt C, he/she has to request it to active node A. Therefore, at first,

he/she prepares his/her certificate CERT_B, which is digitally signed with CA's own key K_CA using a digital signature scheme. CERT_B contains active node B's public key KE_B and some informations INFO, which have a cryptographic algorithm used, expiry data, issue date, etc.

$$CERT\_B = SIG_{K\_CA}(KE\_B|INFO)$$

5. Active node B generates a new active packet (=F) contains CERT_B and a request message for getting KS_A, REQ(KS_A). And he/she gets active node A's IP address from his/her table and sends F to that IP address.

$$F = \{CERT\_B|REQ(KS\_A)\}$$

6. If active node A receives F from active node B, he/she checks active packet header and payload of it, verifies CERT_B, and gets KE_B and INFO from it. In this case, the verification phase depends on the digital signature scheme used.

$$VER(CERT\_B) = KE\_B|INFO$$

7. Active node A records this active packet's IP address, KE_B, CERT_B, and F value in his/her table.
8. Active node A encrypts his/her secret key KS_A with the receiving node's public key KD_B using a public key cryptographic algorithm. And he/she gets active node B's IP address from his/her table and sends this value G to that IP address.

$$G = ENC_{KE\_B}(KS\_A)$$

9. If active node B receives G from active node A, he/she decrypts encrypted secret key G using his/her private key KD_B and gets active node A's secret key KS_A. And he/she also decrypts encrypted active packet C with KS_A and gets target executable code as a clear form.

$$DEC_{KD\_B}(G) = KS\_A$$
$$D_{KS\_A}(C) = PGM$$

10. Receiving node B executes that ptorgram and records results in his/her table. And he/she repeats above steps for some results or the received executable code to deliver securely to neighboring active nodes.

4 Conclusion

We have proposed a new scheme that would allow active nodes transferring active packets to all neighboring active nodes securely, and executing executable code included in those packets in each active node. At first, we delved the concept of active network and reviewed security problems of it. We also setup networks and describe an active packet flow briefly. Finally, we have presented a new scheme that can transfer executable code between active nodes securely and execute it at all intermediate active nodes.

References:

[1] D.L.Tennenhouse, J.M.Smith, W.D. Sincoskie, D.J.Wetherall, and G.J.Minden, A survey of active network research, *IEEE Communications Magazine*, Vol.35, No.1, 1997, pp80-86

[2] K.Psounis, Active networks: Applications, Security, Safety, and Architectures, *IEEE Communications Surveys*, First Quarter 1999

[3] M.S.Greenberg, J.C.Byington, and D.G.Harper, Mobile Agents and Security, *IEEE Communications Magazine*, Vol.36, No.7, July 1998

[4] T.Sander and C.F.Tschudin, Towards Mobile Cryptography, *TR 97-049*, ICSI, 1997

[5] ANSI X9.17 (Revised), American National Standard for financial Institution Key Management (Wholesale), *American Bankers Association*, 1985

[6] J. Daemen and V. Rijmen, Rijndael, the advanced encryption standard, *Dr. Dobb's Journal*, Vol.26, No.3, pp.137-139, March 2001

[7] R.L.Rivest, A.Shamir, and L.Adleman, A method for obtaining digital signatures and public key cryptosystems, *Communications of the Association for Computing Machinery*, Vol.22, No.2, pp.120-126, 1978

[8] B.Schneier, *Applied Cryptography: Second Edition*, Wiley, pp.185-187, 1996

[9] A.J.Menezes, P.C.Oorschot, and S.A.Vanstone, *Handbook of Applied Cryptography*, CRC Press, pp.547-550, 1997

[10] C.P.Schnorr, Efficient signature generation for smart cards, *Proc.of Crypto'89*, Springer-verlag, LNCS Vol.435, pp.239-252, 1990