Proceedings 2nd ACM Conference on Computer and Communications Security, Fairfax, Virgina, Nov. 2-4, 1994.

Application Access Control at Network Level

Refik Molva, Erich Rütsche*
Institut EURECOM, 06904 Sophia-Antipolis, France
{molva, ruetsche}@eurecom.fr
Ph: +33 93 00 26 12 Fax: +33 93 00 26 27

Abstract

This paper describes an access control mechanism that enforces at the network level an access control decision that is taken at the application level. The mechanism is based on the pre-computation of encrypted counters called tickets. An access enforcement device verifies the existence of a valid ticket in each packet that is subject to access control and kills unauthorized packets. Tickets are not computed as a function of the user data. Due to the timing constraints of shared media LANs the presence of a valid ticket in a packet proves that the operation implied by the user data has been authorized. The access control mechanism is elaborated for Internet protocols over Ethernet and we discuss its properties for internetworking and multicasting.

1. Introduction

An important security requirement in distributed systems is the control of accesses between users and applications. In an attempt to formalize the various actors intervening in a distributed access operation, the active entities that actually perform an access can be named **subjects** as opposed to the resources that are accessed that can be encompassed by the concept of an **object**. A subject can be a user, a process, a program or a client in a typical client-server environment. An object can be a file, a program or a service managed by a server in case of client-server environments. The list of operations a subject is authorized to perform on an object are the **access rights** of this subjects on the corresponding object. The definition of access rights for each possible pair of subject and object of a system should be based on a set of well defined rules that state the **security policy**. As means of implementing the security policy, access control at the application level thus has a direct relationship with the management of the security in a given organization.

Another important aspect of access control is the dichotomy between the **access control decision** function that generates the access control decisions pertaining to each specific operation based on the security policy and the **access control enforcement** function that implements the access control decisions in an operational environment by preventing unauthorized operations.

In a typical enterprise, the main objective of security is the protection of application layer specific entities like files and service or application programs rather than lower layer objects like nodes, terminals, connections or packets. The protection of lower layer resources is considered only when it is a mandatory prerequisite for the security of application resources. Even though most application programs include built-in access control schemes, a global access control system is still missing due to the lack of an integrated management scheme encompassing the diversity of applications. Retrofitting a global access control scheme at the application layer requires substantial changes to the existing application packages. Most efforts aiming at transparency for existing applications integrate access control at the lowest layers of the distributed system by implementing access control enforcement at the network layer or in the distributed file system. Despite the advantage of being transparent to application programs, the major shortcoming of these schemes is the lack of relationship between the layer where the access control decisions should be taken (application) and the layer where the access control is enforced. It is a major problem to establish a relation between application layer objects and objects that are manipulated in lower layers, where access control enforcement functions are located at the network components such as bridges and routers. The mapping of network level addresses that are observed by access control monitors to application level names that are used in access

*The work of Erich Rütsche was funded by the *Schweizer Nationalfonds*.

control decisions is particularly hard or infeasible in most systems. The visa scheme introduced by Estrin and Tsudik [1] solves this problem in the context of internet access control. This solution does however not suit local area networks because of its requirement to perform cryptographic operations within the transmission time of each packet.

The design proposed in this paper allows for the enforcement of access control at the network layer based on application layer specific access control decisions. In this design the network traffic is analyzed by a network monitor that performs an on-line verification of cryptographic authorization certificates called tickets. Packets corresponding to unauthorized access operations are killed. The generation and verification of tickets takes into account application level access control decisions so that a strong relationship between the decision and the enforcement of access control is maintained. Because of stringent timing constraints concerning on-line verification of tickets, the design is only valid for LANs but a version of the design that applies to internet access control providing weak security is also presented in the paper. Its application is discussed for multimedia multicast streams.

The general principles of the suggested access control system are presented in Section 2. Section 3 describes the implementation of this design on Internet protocols over Ethernet. An extension of the design for access control over internet and for multicast streams is depicted in Section 4.

2. The Principles of the Access Control Mechanism

We assume that the distributed system consists of hosts interconnected via a single-hop shared medium network (Figure 1). Each host contains a set of application programs and one or several communication subsystems. The communication subsystems map an application layer **communication requests** onto one or several network packets. The communication requests bear (at least) the names of the source and destination application entities that correspond to the identification of the subjects and objects in terms of access control. The shared media network allows each host to receive all the traffic that circulates on the network. A typical example of a shared media network is Ethernet. The network is assumed to be single-hop in the sense that there is no network layer packet forwarding like in the Internet Protocol or in the layer 3 of the OSI model.

The access control mechanism consists of the following components:

- **The access control server (ACS):** This component implements the **access control decision** function on behalf of the system administration. The ACS decides whether a subject is authorized to perform an operation on an object over the network.
- **The access control filter (ACF):** The function of the access control filter is the **enforcement** of access control at the network layer. The ACF can check whether each packet transmitted over the network contains a valid authorization ticket and kills packets that are not authorized.
- **The secure protocol stack (SPS)**: For each communication request that might be subject to access control the application entity needs to use the secure protocol stack. The SPS is responsible for consulting the ACS to check whether the request is authorized. For each request that is authorized the SPS obtains a set of tickets from the ACS. The tickets are assigned to the network packets that result from that request.

An application program (playing the role of a subject) that needs to communicate with another application program (playing the role of an object) that is protected (subject to access control) must use a secure communication subsystem. Communications with a resource that is not protected can go over standard (non-secure) communication subsystems. From the point of view of the application programs the presence of the SPS is transparent since the SPS provides a standard communication interface. The choice between a secure communication subsystem and a non-secure one can also be hidden from the application programs either by substituting all communication subsystems by a secure version or by encapsulating the secure and non-secure communication subsystems with a thin layer in charge of dispatching communication requests to the most suitable subsystem depending on the degree of protection required by the destination objects.

In a typical scenario an access attempt between a subject application program and an application object located on a remote host is shown in Figure 1. First, a communication request is issued to the communication subsystem (1). The subject and the object are identified by the application level source and destination names that are present in the communication request. If the destination object is protected, the SPS issues an access control decision request to the ACS (2). This request contains the names of the source

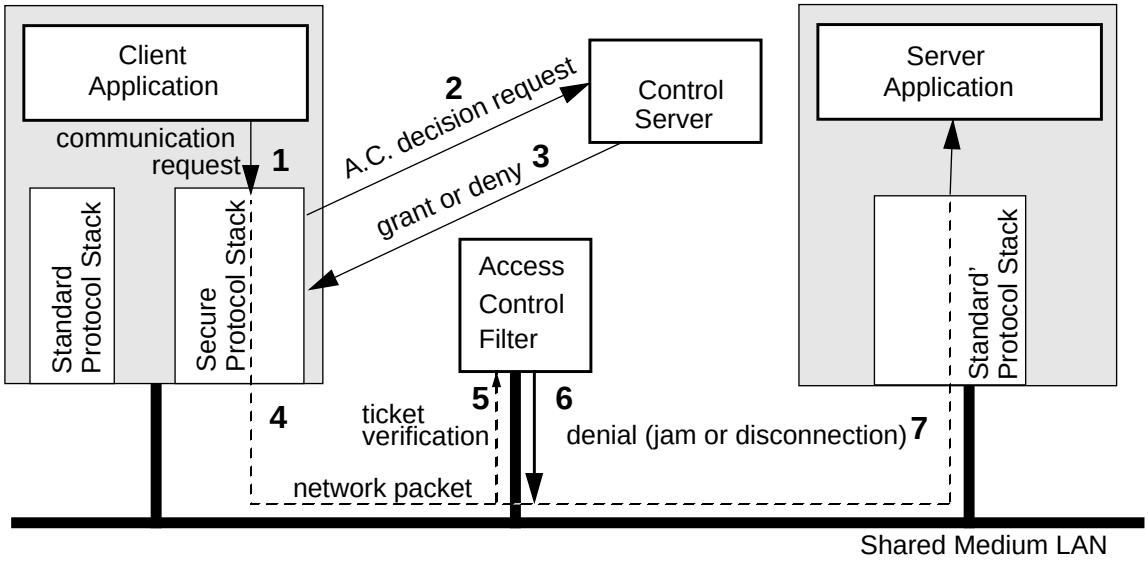


Figure 1. Access Control Mechanism

and destination entities which the ACS uses to check whether the access operation can be authorized. Depending on the result of this decision the ACS can send back a reply carrying a *grant* or *deny* message (3). If the communication request should be granted the right to access, the SPS continues with the normal processing of the request until the transmission of the first network packet resulting from its processing. In order to prove that the user data they carry corresponds to an authorized access operation, each network packet resulting from the communication request is assigned with a cryptographic check called **ticket** (4). The ACF enforces access control by checking the value of the ticket with respect to a set of pre-computed tickets (5). Since this verification can be performed in a very short time frame thanks to the particular verification technique used in this design and to the protocol filter hardware, the result of the verification can be obtained before the packet transmission time elapses and in case the verification fails, the ACF can kill the packet by jamming the signal of the shared media (6). Because the ACF watches all packets on the network only verified packets can access a server (7). The design of the protocol filter hardware and the details of the on-line verification will be addressed in Section 3.

The validity of this design thus relies on the tickets, their computation and verification as will be analyzed in the remaining paragraphs of this section.

The protocol stack on the server might require small changes to accept the added ticket and to parse past it.

2.1. Tickets

A ticket is an encrypted message that proves that the application data contained in a network packet pertains to an application level operation that was authorized by the ACS.

A ticket is unique per packet and can only be used once. Tickets must be introduced in the lowest possible layer of the protocol stack to be useful with a maximum set of protocols. Typically tickets will be placed into the header of a network layer protocol because the network layer packets can directly be monitored through the shared media. It is also possible to assign tickets to data link layer (layer 2 of the OSI model) frames instead of network layer packets.

Computation of tickets:

At system set-up time each host H_i gets a secret key K_i [1]. Each ticket has a sequence number n. A ticket for the n^{th} outgoing packet of the host H_i is calculated by

$$ticket(n) = E_{K_i}(n)$$

E is a one-way hash function that can be implemented using DES ECB [2] mode encryption as depicted by the following expression:

$$ticket(n) = DES_{K_i}(n)$$

whereby K_i is the encryption key. It is assumed in that expression that n is a 64 bit number that fits in a complete

1.K_i can be distributed using one of several alternative methods; secure key distribution by a server, id-based key generation, etc.

DES encryption block. A more efficient version of E that avoids using an encryption function can be obtained based on a native hash function like MD5 [3] as presented by the following expression (see [4] for a justification of this expression):

$$ticket(n) = MD5\left(K_i \middle| n \middle| K_i\right)$$

whereby | denotes the concatenation of messages. Since the value resulting from both versions of function E might excess the security field that is available in network packets, a shorter substring can be derived from the original result of E using a simple choice operation.

On-line Verification of Tickets:

The on-line verification of tickets is performed by the ACF. The ACF consists of two processes:

- **The ticket computation process** that computes expected ticket values to be used by each host in future network packets. The computation is based on the same principle as for the generation of the tickets by the SPS of each host as described in the previous paragraph. This process thus knows the master secret key K_i of each host H_i. The tickets corresponding to each host are pre-computed and the resulting sequence of tickets are kept in separate lists associated with each host. Each list should be sufficiently long that the ticket checking process can handle possible network bursts before new expected ticket values are computed. The pre-computation of tickets for each host can be regulated by a more or less sophisticated algorithm that is out of the scope of this paper.
- **The ticket checking process** that captures all network packets and performs on-line verification of tickets contained therein. This process does not perform any ticket computation. Its role is to check if a packet is subject to access control using the addresses and the control field of the packet. In the simplest case this decision can be taken based on the destination network layer address but the mechanism provides also for checking of higher layer addresses. For each packet that is subject to access control this process identifies the origin host of the packet. The identity of the origin host then allows this process to look up the expected value of the next ticket in the list of pre-computed tickets corresponding to the origin host. The actual verification of the ticket value contained in the packet is only a bit-wise comparison with the expected value thus retrieved from the list. Each ticket value of the pre-computed ticket list that has been observed in a network packet is erased from the list (the pointer to the next ticket is incremented) in order to detect the re-play of a network packet replayed or sent by intruders. In the hypothesis of packet loss, the ticket value received in a packet can be compared against a limited number of expected ticket values in the list. In this case, all the expected ticket values in the list till the one matching the ticket value received in the packet are deleted from the pre-computed ticket list. A network packet is accepted as a result of an authorized access request if its ticket value matches one of the expected ticket values in the pre-computed ticket list associated with the origin host. Alternatively, the sequence number used in the ticket computation by the SPS can be sent in the packet header in order to allow a direct access to the corresponding expected ticket value in the list. Nonetheless the exposure of value n and its encrypted value would be vulnerable to *known-plaintext attacks* aiming at the discovery of the value of K_i. These attacks can be avoided by using a more complex expression and a sequence number that includes a secret offset.

These two processes run in parallel on the ACF. If the ACF is based on a monoprocessor hardware and a multitasking operating system, then the ticket checking process should have the highest priority whereas the computation process can be a low-priority background task.

Access Denial by the ACF:

If a network packet is not accepted by the on-line verification function, the ACF can deny the access by using one of the following alternative techniques:

- The ACF can generate noise on the physical network before the transmission of the packet being checked is terminated; this alternative is possible because the packet verification is a very quick operation based on simple table look-up and bit-wise comparison and because the data used for packet verification is contained in the header of each packet. Packet verification can thus be performed in parallel with the transmission of the packet bits subsequent to the header and the ACF can react with noise generation before the end of the transmission of the whole packet. The details of this denial procedure will be illustrated in Section 3.
- The ACF can generate control packets that force the origin network entity to disconnect the virtual connection to which the packet belongs or to shutdown the network operation of this entity using lower layer messages (data

link layer error or disconnect messages, etc.). This alternative does not require any special device since the timing constraints are much looser than in the previous one.

2.2. Evaluation of the Design

An important feature of the design is that the tickets do not bind the user data contained in the network packet. Thus neither the privacy, nor the integrity of the user data, nor the relationship between the header of a packet and its payload are assured by the ticket itself. This is the major difference between this design and other solutions based on **visas** [1] or **message authentication codes**.

The tickets are not a function of the whole packet, yet the overall design accomplishes the objectives stated at the introduction of this paper, that is, when a network packet is accepted by the ACF the data contained therein is necessarily a result of an application operation that is authorized by the ACS.

The informal proof of this property relies on the following assumptions stated earlier:

- A_1: The SPS is trusted in that the SPS only generates network packets for communication requests that were authorized by the ACS or for communication requests that are not subject to access control. Trusted SPS should not be confused with trusted clients since the clients are not trusted at all, in other words a client that needs to perform an operation on a sensitive resource (that is subject to access control) better issues its call using a trusted SPS instead of a standard protocol stack. However the security of the access control scheme is not based on the client's using the SPS, since in case the client uses a standard stack to issue an operation subject to access control, the operation will be "killed" by the ACF due to the absence of a valid ticket.
- A_2: Each master key K_i is known only by the SPS of host H_i and the ACF.

The proof is based on the following properties:

Property 1 (packet origin authentication): A valid ticket is the proof of the origin of a network packet. I.e., if a network packet destined to a protected host address bearing the address of host H as the source address is accepted by the ACF, then the actual origin of this packet is host H.

This property is true because in the context of single-hop shared media networks, neither replay of a valid packet nor its on-line modification by an intruder can succeed.

If a packet including a valid ticket is replayed by an intruder (or even by H itself), the ticket of the replayed packet will not be accepted because, since the valid packet has already successfully been transmitted, it must have been received by the ACF and the ticket must have been erased from the expected ticket list of the host H according to the verification process.

What is the possibility of the packet (header or user data) being modified during its first transmission? In a shared media network (like Ethernet), as defined in the introduction, an intruder cannot write to the shared media simultaneously with the legitimate transmitter (H) of the packet without killing the packet at all. On the other hand the transmission of a legitimate packet after its receipt and tampering of its header (source or destination addresses) or of its user data by the intruder will be detected by the ACF since the ticket of the tampered packet will not be valid any more.

Property 2 (trusted SPS operation): If a SPS sends a network packet this packet necessarily results from an operation that is authorized at the application layer.

This property is a logical conclusion of the trust assumption (A_1).

Property 1 and Property 2 imply that if a packet is accepted by the ACF then it necessarily results from an operation that has been authorized by the ACS. If a packet is not accepted then it will be killed and the access denied for the operation from which it originates.

The objective of the application layer specific access control will thus be reached since authorized access operations will be carried over the network and the unauthorized ones denied by the ACF.

3. Implementation of the Access Control Mechanism for Internet Protocols

We develop in this section the implementation of the access control mechanism for Internet protocols over Ethernet. In the Internet protocol stack each host is identified by its IP (Internet Protocol) network layer address. On top of IP the transport protocols TCP and UDP provide sockets. Sockets are the end-points of a transport connection and provide an interface to an application. The conventions to assign sockets to applications are partly standardized in [5]. These conventions can be used as a base for an access policy.

Figure 2. The Access Control Filter

Ticket

The ticket must be introduced into the header of IP as in the visa scheme. For the calculation of the ticket we use a standard method like DES. To simplify ticket checking we transmit the sequence number *n* in plain text with the ticket. To avoid the known-plaintext attack we use a secret offset S_C to calculate the ticket. The offset S_C is a secret 64 bit value that designates the client *C*. The available space in the ticket table of the ACF (see below) is limited. Therefore we truncate the calculated ticket to 32 bit. The actual ticket of a packet *n* is:

$$ticket_C(n) = DES_{K_C}\{S_C \oplus n\}\big|_{32}$$

Only the 8 least significant bits of the sequence number *n* are written to the actual network ticket to reduce its size. Thus, the network ticket consumes 40 bits: 8 bit of the sequence number plus the 32 bit ticket.

3.1. Client Implementation

In each host the networking part of the operating system must be enhanced with a part implementing the SPS. The socket API must be changed such that each request to open a connection is first redirected to the SPS. The SPS looks up the request in its local cache of the ACS access matrix and grants or denies a connection to an application. Requests that cannot be resolved locally are forwarded to the ACS. For a granted connection requiring access control, the SPS processes the protocols. It generates a ticket locally, writes it to the option field of the IP header, and sends the packet to the net. The TCP implementation sends segments that are small enough to avoid IP segmentation. For every outgoing packet a new ticket is issued such that eventual retransmissions of TCP result in identical packets being sent with different tickets.

For applications without access control restrictions, the SPS grants the use of the standard protocol stack.

3.2. Access Control Filter

The ACF is the most critical part of the ACM. It is responsible to enforce the access decision in real-time. The ACF must parse the protocol header and analyze the addresses and the ticket of each packet in real-time. To achieve the real-time processing capability the ACF relies on dedicated hardware.

The ACF consist of two parts: a real-time part where the connection and ticket analysis is performed in hardware and a control part with the control processor and the DES unit, that implements the DES algorithm. The architecture of the ACF is shown in Figure 2. The real-time part is based on the protocol filter explained below.

Protocol Filter

The protocol filter (PF) is a central device of the ACF that is needed to analyze the protocol headers in real-time. It scans each incoming packet header and extracts the information defining a connection. If a known connection is found, the protocol type and the connection number are

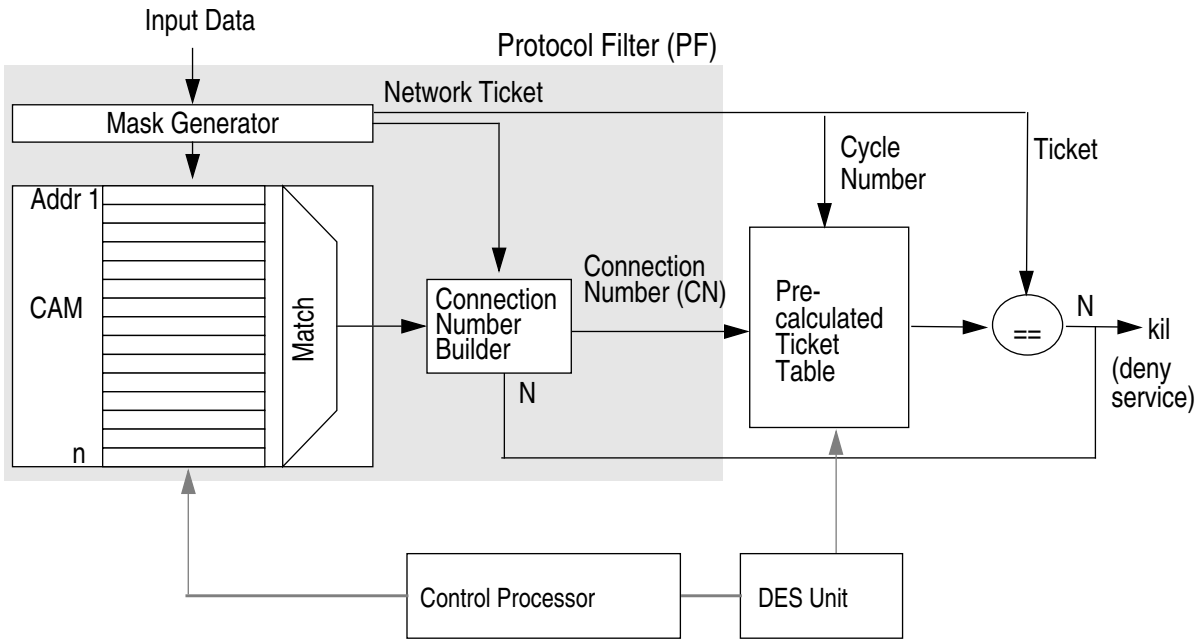

Input Data
Protocol Filter (PF)
Network Ticket
Mask Generator
Addr 1
CAM
n
Match
Connection
Number
Builder
Connection
Number (CN)
N
Cycle
Number
Ticket
Pre-
calculated
Ticket
Table
==
N
kil
(deny
service)
Control Processor
DES Unit

Figure 3. Protocol Address Tree Structure (Internet Protocols)

returned. The connection number (CN) is a unique number that defines the address information of a connection.

The addresses and protocol specific information used in the various layers of a protocol stack form a tree. A connection is defined by the path through the tree (see Figure 3). This path is stored in a content addressable memory (CAM). A CAM row contains the information of the tree level, the protocol type and the address information. For each branch of the tree, the CAM returns the address of the matched word. These addresses are concatenated to form a unique connection number.

The architecture of the PF is shown in Figure 2. The PF filter consists of two hardware state machines built around the central CAM. The Mask Generator extracts the relevant header fields from the protocol header and generates a mask that is compared in the CAM. The Connection Number Builder reads the addresses given by the CAM and builds a unique CN. As the connection detection works in O(1) time, the PF can be built to run at network speed. A detailed description of the PF is given in [6].

3.3. Ticket Checking

The mask generator of the PF analyzes the IP option field and extracts a present network ticket. If the PF cannot generate a CN, a negative message is given to the physical network interface, to kill a packet. A packet is killed by sending a jamming signal to the network to physically destroy the packet. If the ticket could be extracted from the IP header, the CN given by the PF is used as a pointer to a table of precalculated tickets. The CN points to a range of 128 tickets[1], and the cycle number is used as an offset to this range. The ticket at this address is read and compared with the ticket of the packet. If they are different the packet is killed. A used ticket is invalidated by resetting its value to zero.

Figure 4 shows the timing of the ACF. After the last relevant header field has passed by the ACF, the time t_D to react must be as short as possible to kill even very short packets. The delay in the Protocol Filter is determined by the delay of the Mask Generator and the Connection Number Builder that can be implemented in a FPGA (Field Programmable Gate Array). The CAM device (AMD Am99C10) has a cycle time of 100 ns. We estimate the delay in the PF to be about 300 ns. The reading of the ticket (two read operations), the comparison of ticket values (one operation) and the generation of the jamming signal are very fast. We assume the cost of these operation to add up to about 400 ns[2]. The delay t_D will therefore be in the range of 0.7 us what is less then the 3.2 us a 32-bit word needs to fly by the ACM. Thus, t_D is small enough to kill even very small packets. [3]

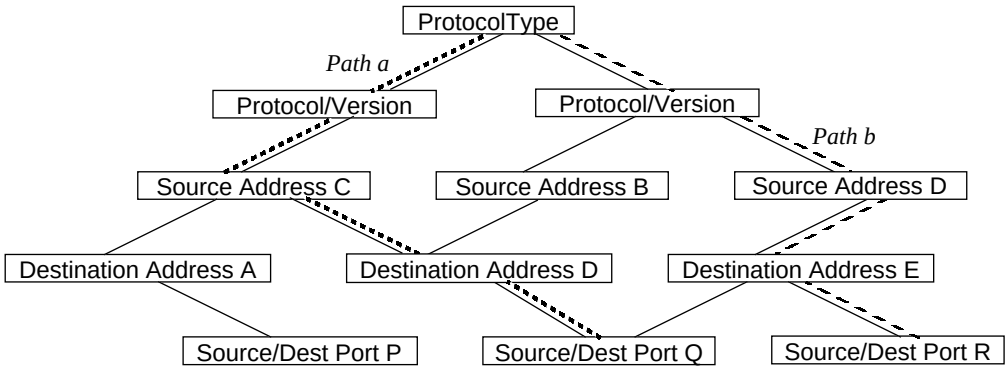
The control processor of the ACF replaces used tickets in the table with the new tickets for the next cycle. It triggers the DES unit to calculate tickets of the required sequence numbers. These operations run in parallel to the real-time ticket checking. Therefore relatively slow DES implementations are sufficient to keep the ticket table up to data. If we assume a fully loaded Ethernet of 10000 packets/s of 100 bytes then the DES algorithm must be able to process 10000 64-bit encryption operations per second, what is very well possible with today's devices [7].

Access denial could also be performed by the ACF sending disconnect messages to the sender of an unauthorized

1.In the actual implementation the ticket range is derived from the CN by a indirection operation in a look-up table.

2.These are conservative assumptions. We assume 100 ns per operation. Today's technology provides read cycles of less than 100 ns and logical operations of less than 10 ns.

3.The same mechanism is also applicable to higher speed networks. In a 100Mb/s network 70 bit would pass before the network signal would be jammed.


ProtocolType
Path a
Protocol/Version
Protocol/Version
Path b
Source Address C
Source Address B
Source Address D
Destination Address A
Destination Address D
Destination Address E
Source/Dest Port P
Source/Dest Port Q
Source/Dest Port R

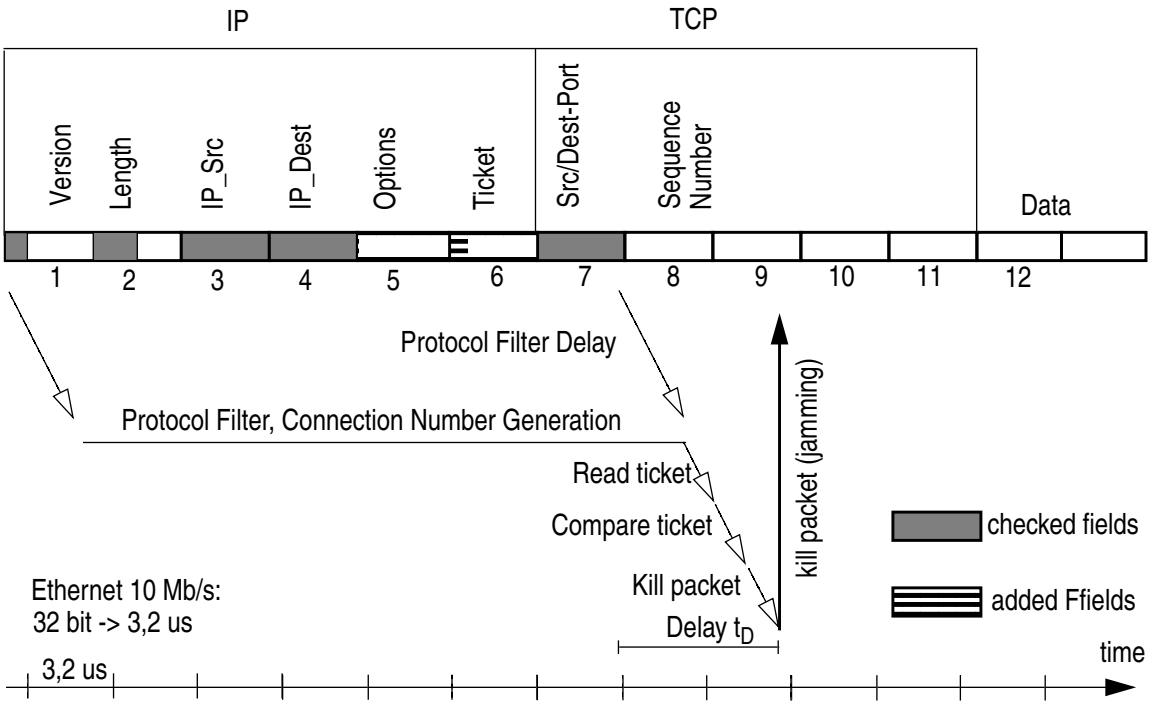


Figure 4. Access Control Timing

packet. If the ACF detects that a host on the network is a serious security problem, it could insulate that host by jamming all traffic that is coming from or going to that host.

4. Application Access Control and Internetworking

The access control design can also be used for access control between networks. The ACF would then be used to analyze all incoming packets in a relay system, e.g., a router or a gateway. Faulty packets would not be forwarded but removed from the memory. However, the access control mechanism looses part of its power because a common knowledge about the packets on the network cannot be assumed any more. Packets could be stored and changed by a malicious relay system. This problem can be partly solved by creating a cryptographic association between tickets and the source *and* the destination address of a connection, where an address can be a layer 4 address. This can be done by assigning a unique application number $S_{X/Y}$ to each application connection from client X to server Y. The application numbers are globally assigned by an ACS and distributed to each SPS and each ACF. A ticket will be calculated by

$$ticket_X(n) = DES_{K_X}\{S_{X/Y} \oplus n\}\big|_{32}$$

The mechanisms of access decision and enforcement are the same as in the case of a shared medium network. However, the ticket cannot guarantee anything else than the source address, destination address, and the packet number. All other information in the header and the payload could be corrupted by a malicious relay device.

For internetworking, the ACM can provide at least the security functions of an application gateway or of a firewall [8][9]. The ACM controls the access rights of an application connection *and* the number of network packets sent. The secure gateway or router could be replaced by a relay system that is enhanced with an ACF. The ACF can replace the filter functions that are mostly implemented in software in today's relay devices. As the ACF runs in real-time it can improve the performance of a secure relay device and also offer more flexibility. A policy can be enforced that defines which hosts, subnetworks, or services must be protected by tickets. E.g., all application connections going over a vulnerable link or to a critical server might be protected by the ACM.

The ACM could be implemented in a relay device as shown in Figure 5. It analyses all incoming network packets and extracts the CN of the packets. The CN could be used as a pointer to the routing table only if the packet it authorized.

Bridged Networks

The access control mechanism is also well suited for bridged networks. Existing access control mechanisms

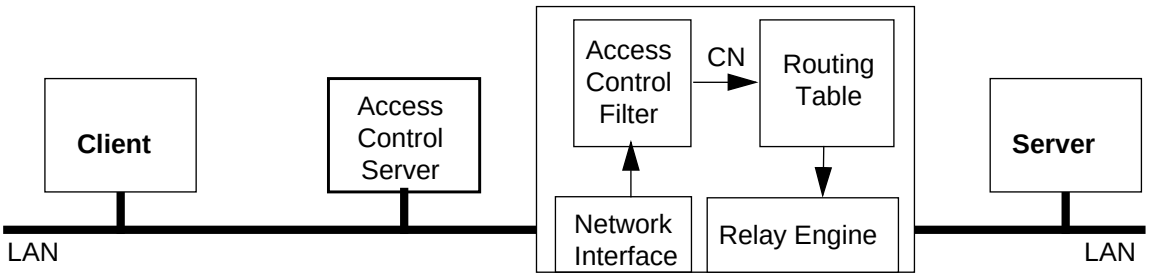


Figure 5. Access Control Filter in a Realy Device

implemented by bridges are based on the analysis of the link level addresses regardless of the application specific data contained in the packets. This type of access control provides only a course granularity protection since there can only be a single authorization for all the application entities located on the same network station. The implementation of the ACM in bridged networks can enhance the granularity of the access control. Access control enforcement can be based on application tickets and on addressing information up to the transport layer. With an ACF a bridge can build a firewall and provide the same level of security as a router or a gateway. The ACM can thus be viewed as a general concept for improving the security of networks that rely on layer 2 addresses, e.g., Netbios [10].

Multicast Streams

The security of multicast streams [11] is a problem that becomes more and more important with the increasing usage of these services in the Internet. Multicast streams are mostly used for audio and video conferences. Traditional security measures like encryption are not suitable because the additional delay due to encryption would conflict with quality of service requirements of multimedia streams.

A more stringent requirement that is the integrity of the multicast channel can be fulfilled using the ACM. The integrity of the multicast channel consists of preventing unauthorized sources from broadcasting information on the multicast channel. The integrity of the multicast channel needs to be protected even when data privacy is not needed or cannot be assured in order to prevent intruders from flooding multicast channels of the Internet with garbage messages.

In order to prevent illegal flooding of domains through multicast channels, the ACM implemented in relay devices can provide access control and filtering capabilities. Each party that wants to become a source on a multicast connection can request the corresponding right from an ACS. If the access is granted, the sender should include into each multicast packet a ticket that authenticates the packet. Access control filters in relay devices can then check these tickets and allow only authorized packets into a trusted domain.

If there were no trusted domains then the ACF function would be required in each end-node of a multicast tree. Research in applying our scheme on general multimedia multiparty authentication and access control is currently under way.

5. Conclusion

This paper presented an access control mechanism applied to shared medium LANs. The main advantage of this mechanism is its transparency with respect to existing application programs. The originality of the scheme is due to the pre-computation technique that provides a message origin authentication capability avoiding the computation of message authentication codes yet equivalent to them in terms of replay and modification detection power. We have shown that our access control scheme guarantees data integrity in a shared medium LAN even though the ticket is not computed as a function of the application data contained in the packet.

The scheme can also be applied in internetworking to improve security of firewalls by checking not only addresses but also application tickets. For bridged networks the scheme provides the functionality of a secure network or application layer gateway. However in internetworking the access control mechanism does not provide for data integrity. The scheme is especially useful to enhance the security of multimedia multicast communication where the quality of service requirements of the applications permits no encryption.

6. References

[1] Estrin, D., Mogul, J.C., Tsudik, G., Anand, K., "Visa Protocols for Controlling Inter-Organizational Datagram Flow: Extended Description," USC TR 88-50, 1988.

[2] "Data Encryption Standard", FIPS 46, NBS, Jan 77.

[3] Rivest, R., "The MD5 Message Digest Algorithm Draft," July 1991.

[4] Tsudik, G., "Message Authentication with One-way Hash Functions", Proceedings of IEEE INFOCOM'92, May 1992.

[5] Reynolds, J., Postel, J., *"Assigned Numbers"*, Network Working Group, Request for Comments: 1340, July 1992.

[6] Rütsche, E., "Multimedia Communication Subsystems: Architectures, Interfaces and Implementation," Ph.D. Thesis ETH Zürich, Nr. 10228, VDI Verlag, Reihe 10, Nr. 257, Düsseldorf 1993, pp. 129 -134.

[7] Cryptech, "DES PROCESSOR," Cryptech, DOC 00552008E-ED01, 1989.

[8] Cheswick, B., "Design of a Secure Internet Gateway", Proceedings of the USENIX Summer 1990 Conference, Anaheim, CA, June 1990, pp. 233-237.

[9] Treese, G.W., Wolman, A., "X Through the Firewall and Other Application Relays," DEC CRL, 93/10, May 1993.

[10] Sinah, A., Path, R., "An Introduction to Network Programming Using the Netbios Interface", Microsoft Systems Journal, Mar, 1992

[11] Deering, S., Cheriton, D., "Multicasting in Datagram Internetworks and Extended LANS," ACM Transaction on Computer Systems, Vol 8, May 1990