Security Policy Communication in a Distributed Network Element*

Mahesh V. Tripunitara and Gene Spafford
CERIAS, Purdue University, West Lafayette, IN 47907
{tripunit,spaf}@cerias.purdue.edu

CERIAS TR-99/01

Abstract – This paper discusses the distributed network element concept as it pertains to the communication of security policies. In the context of this paper, a security policy specifies access control rules on network traffic. We introduce the problem and give a software architecture to solve it. The solution has four components that form the distributed network element: the controller, adaptor, driver and the network element. We also discuss the application of the solution to two cases: one in which the network element is an ATM switch and the other in which the network element is an IP switch.

Keywords – Distributed Network Element, Security Policy, Network Element, Controller, Adaptor, Driver, ATM, IP Switching.

1 Introduction

A *distributed network element* has four components: a *network element*, an *adaptor*, a *driver* and a *controller*. A network element is a switch. The controller communicates switching *policies* to the network element. The adaptor and driver mediate communication between the controller and the network element. These are explained in section 3 in more detail.

One of the objectives of the distributed network element project at AT&T - Geoplex labs[1][Org98a, Mih98] and the P1520 IEEE standardization effort[BLH+98] is isolation of the entity that (decides and) communicates QoS and flow separation policies from the switch that implements them. The advantage with this approach is that switches export a standard interface and the controller is not tied to a single switch. This promotes interoperability.

This paper examines the same objective, but in the context of security policies. A security policy, in this context, specifies access control rules on network traffic. We consider communication with TCP/IP that uses distributed network elements. The specific setting we discuss our technique in, is described in section 2.

In this paper, we:

- Motivate the problem of isolating the controller from the network element in the context of security policies,
- Describe a (software) architecture to achieve such isolation,
- Discuss the language the controller uses to communicate with the network element, and,
- Give two examples of the working of this setup: one in which the network element is an Asynchronous Transfer Mode (ATM) switch and the other in which the network element is an IP switch.

2 The Setting and Motivation

We consider a community of users that uses a TCP/IP cloud to communicate. The protocol stack at each user's host is shown in figure 1. This is consistent with the architecture proposed in [Org98a, Mih98].

In our context, the cloud is a provider of transport, QoS and security services for the network traffic. This cloud can be a corporate intranet or an Internet Service Provider's (ISP's) network. Figure 1 illustrates the use of a cloud by users for communication.

Gates route traffic through a cloud like a traditional router: IP packets flow up to the IP layer where a routing decision is made based on a table. A super gate, on the other hand, is capable of *cut-through routing*.

*Portions of this work were supported by sponsors of CERIAS. A version of this paper appears in the Proceedings of the International Conference on Advanced Communication Technology (ICACT'99).

[1] AT&T and Geoplex are registered trademarks.

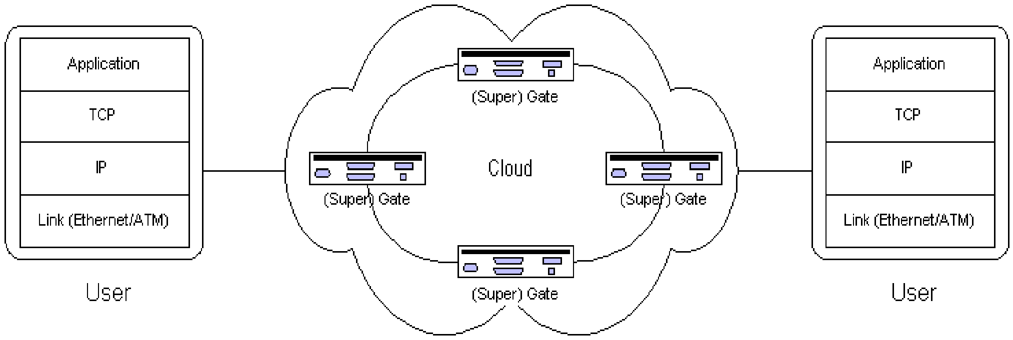

Figure 1: Communication using a cloud based on TCP/IP

Cut-through routing is performed to speed up network traffic. In the context of the protocol stack, this means that at each super gate, packets do not flow up to the IP layer before they are routed, but are switched at the link layer. Figure 2 illustrates this scenario.

Gates and super gates impose access checks on network traffic. In the context of this paper, this refers to a pass or drop action on each IP packet, based on information contained in its IP header. A pass action allows for the packet to be routed to the next gate on the route. A drop action drops the packet at this gate.

A third action is also possible, that results in the packet being forwarded up the protocol stack to a transport or application layer proxy. But we ignore that possibility because the action of the proxy will ultimately result in either a pass or drop action on the packet.

When cut-through routing is in effect, packets do not flow up to the IP layer, but are switched at the link layer. To perform access checks on traffic based on IP header information, the super gate must map IP header information to link layer connections or sessions, or frames or cells.

In this paper, we assume that each network element has its own, built-in security capabilities. For instance, if the network element is an ATM switch, we assume that it implements the ATM Forum Security Specification 1.0 [Tar97] security services.

Our problem then, is the mapping of access checks based on IP header information to the security services based on link layer information. That is, we need to communicate security policies to the network element. This is achieved within the distributed network element framework, as discussed in the remainder of this paper.

3 The Distributed Network Element in the context of Security Policies

The distributed network element [Mih98] is part of each super gate and consists of four components: the network element, the driver, the adaptor and the controller (see figure 3.)

The network element is a switch. The controller communicates (security) policies to the switch. The adaptor is associated with each switching technology, for example, ATM switching or IP switching. The driver is associated with each (vendor's) switch. The driver and the adaptor translate the security policy communicated by the controller to the security services available at the switch.

We employ a Distributed Network Element Control Language (DNECL) and a DNECL Adaptation Layer (DNECL-AL) for communication between the controller and the driver. The DNECL-AL translates the DNECL policy to the security services available in the particular network element. The DNECL-AL is contained within the adaptor in figure 3.

The driver makes the necessary invocations in the network element to enforce the policy. We assume that the network element provides the security services necessary for enforcing the policy.

The DNECL and DNECL-AL are described in [Org98b]. We provide DNECL snippets and discussions on the DNECL-AL capabilities in section 4, which discusses two examples of the use of our setup.

Implicit in the DNECL-AL is the mapping of security policies between different types of networks. IP is a packet oriented, connectionless protocol. An example of the link layer is ATM, which is a packet- and connection-oriented protocol. Classical IP over ATM (CIPoA) [SKS94] is a technique to integrate the two, with ATM at the link layer. The DNECL-AL must be able to translate access rules which are enforced at

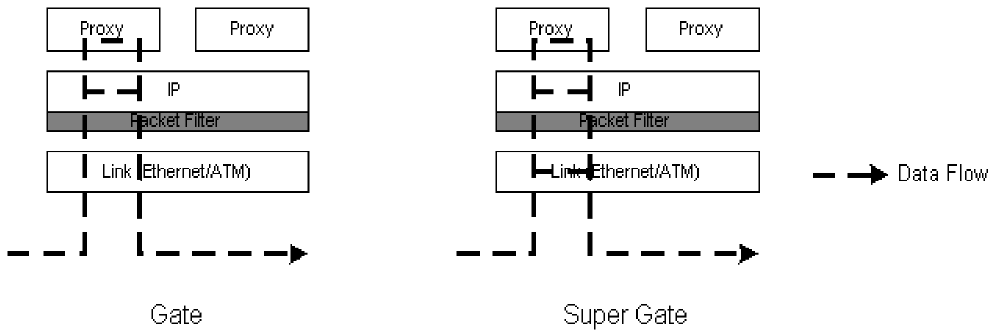

Figure 2: Gate vs. Super Gate: Gates route at the IP layer, Super Gates can switch at the link layer

the IP layer in a gate, to access rules based on ATM connections and cells at the link layer in a super gate.

4 Examples

In this section, we discuss two examples of the application of the architecture from section 3 . The first example considers the use of CIPoA. That is, ATM at the link layer with the interaction between IP and ATM as specified in [SKS94]. The second example considers an IP switch.

The difference between the two scenarios is that in the case of an IP switch, the IP layer is contained in the network element. In CIPoA, connection setup happens before the first IP packet is forwarded. With an IP switch, a flow is set up only after a flow has been identified. Flow identification is outside the scope of this paper. Flow identification is controlled using a protocol such as the Generic Switch Management Protocol [NEH+96a]. Flow establishment is achieved using a protocol such as the Ipsilon Flow Management Protocol [NEH+96b].

4.1 A super gate that uses Classical IP over ATM

CIPoA is a technique by which IP can use ATM at the link layer. We assume that the security services described in the ATM Forum Security Specification 1.0 [Tar97] are available at the ATM layer.

[Tar97] describes four types of security services:

• *Confidentiality* refers to protection of control and data traffic from unauthorized disclosure.
• *Integrity* refers to the ability to detect unauthorized change in control and data traffic in transit.
• *Authentication* refers to the validation of the identity of the sender and receiver of control and data traffic.
• *Access Control* refers to disallowing unauthorized usage of network bandwidth.

Specific security mechanisms to realize the four security services are also described in [Tar97]. These include private and public key encryption techniques such as the ones described in [RSA78, Nat93, Bru96], and keyless and keyed hashing techniques such as those described in [Pre93, Bru96]. Techniques for certificate and key exchange are also discussed in [Tar97].

In this paper, only access control from the perspective of IP is discussed. Such access control requires authentication, integrity and access control services from the ATM switch.

Figure 4 illustrates the situation with an ATM switch at the link layer. Consider a case in which the DNECL specifies a simple rule that calls for all traffic between two pairs of IP addresses and ports to pass:

Src-ip Dst-ip Src-port Dst-port pass

The DNECL-AL now has to translate this to an access policy for the ATM switch. We assume that a user's host is bound to single IP and ATM addresses. The IP to ATM address translation and vice versa is achieved using ATM ARP [SKS94]. We assume that the binding between the IP and ATM addresses is strong[2].

The DNECL-AL translates the above access control rule to the following invocations that are processed by the driver. All parameters are call by value. The return value is obvious from the call.

Src-ATM := *GetATMFromIP(Src-ip)*

AccessCheckATM(Src-ATM)

[2] In general, this is a poor assumption, because the binding between an IP address and an ATM address is weak. But we can use techniques such as those described in [BE98] to address this issue.

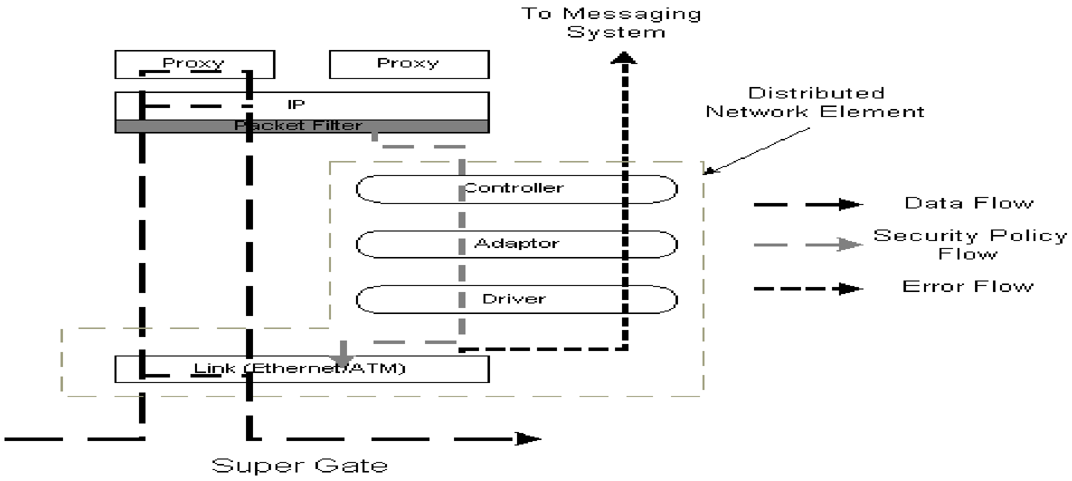

Figure 3: Components of the Super Gate and the Distributed Network Element

Dst-ATM := *GetATMFromIP(Dst-ip)*

AccessCheckATM(Dst-ATM)

Signaling(AUTHENTICATION, Src-ATM, Dst-ATM)

Data(AUTHENTICATION | INTEGRITY, Src-ATM, Dst-ATM)

The *GetATMFromIP( )* invocation performs the ATM ARP operations to map the source and destination IP addresses to ATM addresses. The *AccessCheckATM( )* invocations perform access control based on the respective ATM addresses.

The *Signaling( )* invocation specifies constraints to be enforced at call setup by this network element (ATM switch.) The first parameter is a flag that specifies which of the security services from [Tar97] are to be employed. In this case, we request the driver to employ authenticated signaling [Sch97]. Authenticated signaling is call setup and tear-down enhanced with authentication of both the initiator and the target of the call.

The *Data( )* invocation is similar to the *Signaling( )* invocation, but specifies constraints on the data traffic (as opposed to the control traffic.) The first parameter is again a flag. In this case, we request the driver to enforce both authentication and integrity constraints on the data traffic. Note that in this case, the integrity service is employed per ATM Adaptation Layer (AAL) Protocol Data Unit (PDU) or ATM cell.

Two properties implicit in our mapping of access control based on IP header information to ATM security services are:

• We do not assume that filtering on data can be performed at the ATM layer. If we could filter on data, we could simply look for the data that corresponds to the IP header and filter on that.

• We are unable to translate the filtering based on ports directly to the ATM switch. Therefore, the DNECL-AL maps port-based filtering to only authentication and integrity of the end-hosts. Again, if filtering on data were available, we would be able to map the filtering of ports directly in the ATM switch.

The DNECL-AL for ATM at the link layer also supports a few other options and invocations. For instance, we can perform authentication of only the initiator or the target(s) of the IP session. We can also allow for the pass action at the DNECL to be associated with an option to encrypt the data. This would be translated directly to a *CONFIDENTIALITY* flag for the ATM traffic associated with that connection.

4.2 A Super Gate that uses an IP Switch

An IP switch routes packets to the destination until a flow is identified. Once a flow is identified, the switch employs cut-through routing to switch packets to the (intermediate) destination. Unlike an ATM switch, an IP switch does not establish a connection (or a flow) before it transfers data. Also, a flow is not necessarily end-to-end. Along a route, between the source of the packets and the destination, only a few adjacent switches could choose to establish a flow.

Protocols such as GSMP [NEH+96a] are used to configure flow identification mechanisms in an IP switch. A protocol such as IFMP [NEH+96b] is used by switches to communicate with adjacent switches about establishing a flow between them.

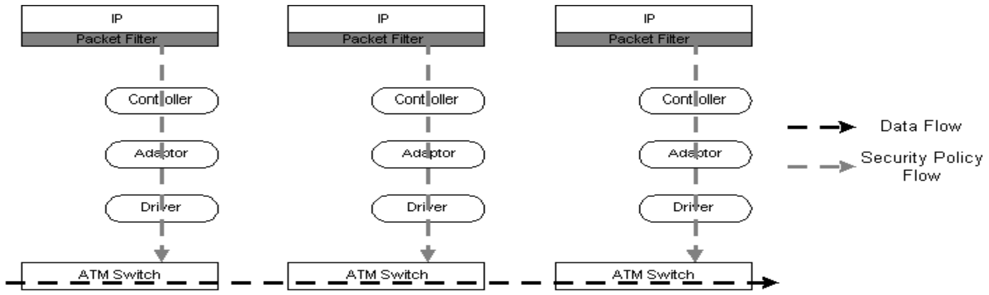

Figure 4: Data and Security Policy Flows when ATM switches are used

Figure 5 illustrates the case when an IP switch is used. We again consider the simple example for a DNECL specification that calls for a pass action for packets between a pair of IP addresses and ports.

Until the time a flow is established, the IP switch functions as a gate would: it filters IP packets at the IP layer. Thus, the DNECL-AL for an IP switch communicates the above specification to the driver as-is. Once a flow is established, data traffic is subjected to the same authentication and integrity security services as in the case of an ATM switch.

But, before establishing a flow, we require revalidation of the sender and receiver. This is similar to performing authenticated signaling in the case of an ATM switch. Our revalidation mechanism is built into the IFMP protocol[3].

We are unable to perform revalidation of the end hosts that are involved in the communication. This is because, as we mentioned earlier, a flow is not necessarily established end-to-end. Thus, our revalidation mechanism only extends up to the switches where the flow begins and ends. The semantics of our implementation is that the onus of revalidating the sending and receiving hosts is then pushed to those switches.

The switches at the ends of the flow may choose to perform the revalidation immediately or when they establish a flow that includes the switch immediately after the sending host and immediately before the receiving host in the route. The second option implies that cut-through routing will be performed by the switches between which a flow has been established while the sending and receiving hosts remain unvalidated. But, given that the switches not involved in the flow continue to filter the packets at the IP layer, and that the components of a cloud cooperate with each other in providing security services to the end-user, the second option is not necessarily weaker, from a security stand-point, than the first.

But, the second option implies that we do not protect the switches from Byzantine failures.

[3]Thus we have an "Enhanced IFMP" or eIFMP.

5 Concluding Remarks

This paper discusses the communication of security policies for IP packets to the link layer, so they can be applied when cut-through routing is employed. As of the writing of this paper, the architecture and language of communication are still under development. We discuss two examples, which are taken from our development environment, for which we are trying to realize the distributed network element architecture with support for security services. The architecture promotes interoperability by defining a language and an architecture that can be used to communicate security policies to link layer switches.

6 Acknowledgements

Dr. Nelu Mihai provided the author the support and inspiration to work on the security aspects of the distributed network element project. Srikar Rajamani helped the author get introduced to the distributed network element setup as used for flow separation. Laci Boloni developed the first prototype of a distributed network element that has support for flow separation policies.

References

[BE98] Carsten Benecke and Uwe Ellermann. Securing Classical IP over ATM Networks. In *Seventh USENIX Security Symposium*, January 1998.

[BLH+98] Jit Biswas, Aurel A. Lazar, Jean-Francois Huard, Koonseng Lim, Semir Mahjoub, Louis-Francois Pau, Masaaki Suzuki, Soren Torstensson, Weiguo Wang, and Stephen Weinstein. The IEEE P1520 Standards Initiative for Programmable Network In-

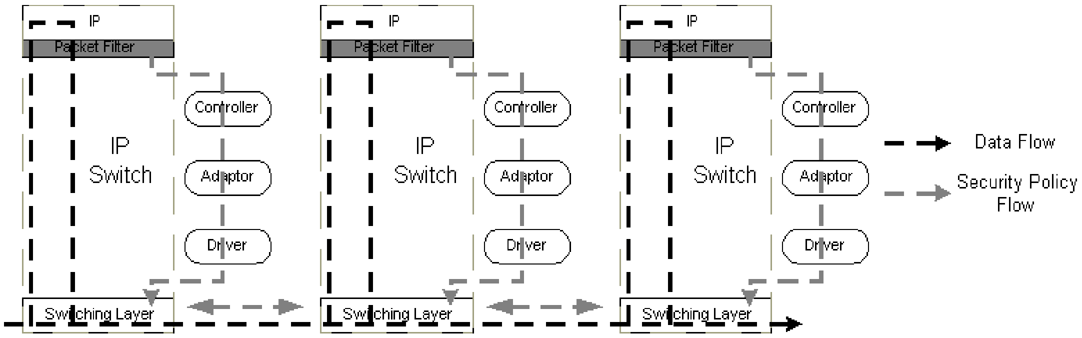

Figure 5: Data and Security Policy Flows when IP switches are used

terfaces. *IEEE Communications Magazine*, October 1998.

[Bru96] Bruce Schneier. *Applied Cryptography.* John Wiley and Sons, Inc., second edition, 1996.

[Mih98] Nelu Mihai. Geoplex - an Open Service Platform. In *IEEE Open Signaling Workshop (keynote address)*, October 1998.

[Nat93] National Institute of Standards and Technology. *NIST FIPS PUB 46-2, Data Encryption Standard (DES).* U.S. Department of Commerce, December 1993.

[NEH+96a] P. Newman, W. Edwards, R. Hinden, E. Hoffman, F. Ching Liaw, T. Lyon, and G. Minshall. *RFC-1987 Ipsilon's General Switch Management Protocol Specification.* Network Working Group, August 1996.

[NEH+96b] P. Newman, W. Edwards, R. Hinden, E. Hoffman, F. Ching Liaw, T. Lyon, and G. Minshall. *RFC-1953 Ipsilon's Flow Management Protocol Specification for IPv4.* Network Working Group, May 1996.

[Org98a] AT&T Internet Platforms Organization. 21st Century Advanced Network Services Platform Technology Overview. *White Paper*, April 1998.

[Org98b] AT&T Internet Platforms Organization. A Distributed Network Element Control Language and Adaptation Layers. *White Paper, in preparation*, December 1998.

[Pre93] B. Preneel. *Analysis and Design of Cryptographic Hash Functions.* Doctoral Dissertation, Katholieke Universiteit Leuven, 1993.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A Method of Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21, February 1978.

[Sch97] Christoph L. Schuba. *On the Modeling, Design and Implementation of Firewall Technology.* PhD Thesis, Department of Computer Sciences, Purdue University, December 1997.

[SKS94] Christoph Schuba, Berry Kercheval, and E. H. Spafford. *Classical IP and ARP over ATM, Technical Report CSD-TR-94-029.* Department of Computer Sciences, Purdue University, September 1994.

[Tar97] Thomas D. Tarman, editor. *ATM Security Specification Version 1.0 (Draft).* ATM Forum Technical Committee, September 1997.