Design of a High-Performance ATM Firewall

JUN XU and MUKESH SINGHAL
The Ohio State University

A router-based packet-filtering firewall is an effective way of protecting an enterprise network from unauthorized access. However, it will not work efficiently in an ATM network because it requires the termination of end-to-end ATM connections at a packet-filtering router, which incurs huge overhead of SAR (Segmentation and Reassembly). Very few approaches to this problem have been proposed in the literature, and none is completely satisfactory. In this paper we present the hardware design of a high-speed ATM firewall that does not require the termination of an end-to-end connection in the middle. We propose a novel firewall design philosophy, called Quality of Firewalling (QoF), that applies security measures of different strength to traffic with different risk levels and show how it can be implemented in our firewall. Compared with the traditional firewalls, this ATM firewall performs exactly the same packet-level filtering without compromising the performance and has the same "look and feel" by sitting at the chokepoint between the trusted ATM LAN and untrusted ATM WAN. It is also easy to manage and flexible to use.

Categories and Subject Descriptors: C.4 [**Computer Systems Organization**]: Performance of Systems; C.2.0 [**Computer-Communication Networks**]: General; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Asynchronous Transfer Mode* (ATM); C.2.5 [**Computer-Communication Networks**]: Local and Wide-Area Networks; C.2.6 [**Computer-Communication Networks**]: Internetworking —*Routers*

General Terms: Design, Performance, Security, Theory

Additional Key Words and Phrases: Asynchronous Transfer Mode (ATM), Firewall, Packet filtering, TCP/IP, Switch architecture

This research was partially supported by NSA grant MDA904-96-1-0111. An early version of this paper appeared in the Proceedings of 5th ACM Conference on Computer and Communication Security, Nov. 1998, pp. 93–102.
Authors' address: Department of Computer and Information Science, The Ohio State University, Columbus, OH 43210; email: jun@cis.ohio-state.edu; singhal@cis.ohio-state.edu.

1. MOTIVATION AND PREVIOUS WORK

1.1 Motivation for ATM Firewalls

ATM is a promising cutting-edge technology aiming at integrating data, voice, and video services in the same underlying communication infrastructure. It is expected that legacy TCP/IP data traffic will be carried over ATM networks, thanks to the popularity of TCP/IP-based Internet applications. Therefore, ATM networks are subject to all security problems (and perhaps more) that exist in router-based TCP/IP networks. Because a packet-filtering firewall is a very effective way of preventing a TCP/IP network from unauthorized access, it is desirable to apply it to ATM networks. Unfortunately, a traditional router-based firewall will not be able to deliver the filtering throughput that is even close to the low-end ATM rate of OC-3c (155 Mbps) due to two factors. First, a packet-filtering router needs to terminate an end-to-end ATM connection in the middle in order to extract IP packets for inspection. This involves SAR (Segmentation and Reassembly), which incurs a huge overhead. Second, the filtering bandwidth of a traditional firewall is generally below 100 Mbps, which is much less than the typical ATM rate of OC-3c and OC-12c (622 Mbps).

The ATM Forum favors avoidance of packet filtering by exerting discretion at connection establishment time. Based on this principle, two alternative access control schemes have been proposed. Smith [1994] proposes that access control decisions be made at connection establishment time, based on higher layer information (e.g., source and destination IP addresses, ports, etc.) contained in the ATM *signaling message* as *information elements*. After the connection is established, the access control device "gets out of the way." Obviously, this is unacceptable because an intruder can always lie about the service he wants to access at connection establishment time, and there is no way to check the contents of a connection once the connection is established. Pierson and Tarman [1995] try to fix this problem by proposing that "a new SVC (Switched Virtual Connection) is requested when each new service is started." However, this requires the ATM layer be notified whenever a new socket is opened, which entails considerable change to the whole TCP/IP stack and existing applications [Hughes and Guha 1995]. Even worse, it requires a new SVC for each transport layer flow, which may lead to *VC explosion*. So replacing packet-level filtering with *call screening* will not solve the problem of access control in an ATM network.

Arguably, another alternative is to apply cryptographic measures end-to-end so that packet filtering in the middle can be avoided [Tarman 1999]. A centralized encrypting gateway, called *encrypting ATM firewall*, is employed between a trusted ATM LAN and an untrusted WAN to perform an encryption operation on behalf of all protected hosts [Secant Network Technologies Inc. 1997]. However, authentication and encryption do not automatically ensure proper access control. Even when a connection is authenticated, we may still want to look into its contents if the parties

involved do not trust each other completely. Moreover, many connections need to be established between parties who do not trust each other at all, for example, between an Internet surfer and an organization's HTTP server. In such cases an encrypting firewall will not solve the problem of access control.

Therefore, packet-level filtering is an indispensable access control scheme in an ATM network. Because traditional firewalls can no longer perform packet-level filtering in an ATM network, a new ATM firewall architecture is called for.

1.2 Existing Approaches

So far only one packet-filtering ATM firewall design is available, the ATLAS product developed by StorageTek [Hughes 1996]. ATLAS is a line filter that scans an ATM physical link to perform packet-level filtering at the rate of OC-3c (155 Mbps). StorageTek claims that the next-generation ATLAS is going to support a filtering rate of OC-12c (622 Mbps) in the near future. Two performance-boosting strategies are used in ATLAS, which correspond to the two factors that render traditional firewalls unsuitable for an ATM environment. First, to avoid SAR, for each packet it only checks the first cell, which contains the IP header, protocol, TCP/UDP ports, and TCP flags (if applicable), to determine whether or not the packet is "safe." If the packet is considered safe, all the following cells that belong to it are passed or otherwise dropped. Second, it uses a *policy cache architecture* [Kowalski 1996] to dramatically speed up the process of deciding whether or not a packet header is safe. The core unit of this architecture is a cache block, called *policy cache*. Each entry of the policy cache is a combination of VPI/VCI, source and destination IP addresses, and source and destination TCP/UDP ports considered "safe." When the first cell of a packet arrives, the information in its header is compared with each entry in the policy cache. If a cache hit occurs, the packet's cells are forwarded. Otherwise the first cell goes through a software-screening process and other cells are buffered in a queue. If the cell is found unsafe, the whole packet is dropped. Otherwise the packet is allowed to pass, and an entry that contains the header pattern of the packet is now added to the policy cache. The policy cache is implemented using CAM (Content Addressable Memory), which enables simultaneous searches of all memory locations to find a match with the pattern being searched for. Therefore, ATLAS can decide whether a packet header hits the policy cache very quickly. However, the ATLAS product does have its limitations and drawbacks. First, it does not accept IP packets with IP option fields because IP options can be as large as 40 bytes and may "push" the TCP headers to the second cell. This may become a severe limitation in future internetworking environments where certain IP option fields such as AH (Authentication Header) [Kent and Atkinson 1998a] are used frequently. Second, using Content Addressable Memory (CAM) to cache a safe header is not a scalable solution. The problem in this approach is that CAM can not scale to a large size due to technological

constraints and is extremely expensive. Commercially available 128-bit-wide CAM comes at the size of 1K entries and costs about $30 [MUSIC Semiconductors 1998]. When the traffic volume is high (like 1 Gbps), the firewall requires hundreds of thousands of CAM entries to achieve a decent hit ratio (like 90%) [Xu et al. 1999]. It is not only that we cannot afford this much CAM, but such a huge CAM is hard to build as well. Third, it is not friendly to those who have to manage and administer it. Whenever a new PVC (Permanent Virtual Connection) or SVC is established, TCP/IP rules for that VC will have to be manually configured. This is acceptable in an environment where the number of connections is small and most of them are PVC. However, in future ATM networks, a large number of SVCs will be established on-the-fly in a short period of time, making it impossible for a network manager to manually configure on demand TCP/IP rules for each of them.

1.3 Our ATM Firewall

In this paper we present a conceptual design of a high-performance switch-based ATM firewall architecture. It incorporates our novel firewall design concept, called *Quality of Firewalling* (QoF), which employs security measures of different strength on traffic associated with different risk levels in order to achieve a nice tradeoff between performance and security. In terms of performance, it achieves a higher throughput and lower latency than ATLAS. In the next section we present the design philosophy and logical structure of the proposed ATM firewall architecture. Sections 3, 4, and 5 present the logical and physical design of our ATM firewall. Section 6 concludes the paper.

2. PHILOSOPHY AND LOGICAL DESIGN OF OUR ATM FIREWALL

2.1 Quality of Firewalling (QoF)

We observe that in an ATM network, IP traffic in different connections is associated with different risk levels determined by the identities of source and destination parties and Internet services requested and/or rendered. Based on this observation, we propose the Quality of Firewalling (QoF) concept. Informally speaking, higher QoF will be applied to the more "dangerous" connections, typically at the cost of a longer processing time per packet. Four QoF classes are defined, namely, classes A, B, C, and D, which are ordered from the "safest" to "the most dangerous." Class A connections are exempt from any kind of inspection because they are considered absolutely safe. Class C connections are those that can be secured by packet-level filtering.[1] The risk level of class B connections is between those of classes A and C, secured by a scheme called *traffic monitoring*. This scheme is slightly different than packet filtering; packet filtering determines whether the packet is safe before forwarding it, while

[1]This includes stateful filtering as used in FTP.

traffic monitoring reverses this order. We show in Section 2.2.4 that it incurs much less latency than *packet filtering* and is nearly as safe. Traffic in class D connections is dangerous and involves complex protocols that are hard to secure merely through packet-level filtering. They are secured through proxying, a scheme used in traditional firewalls to secure complex protocols such as TALK [Chapman and Zwicky 1995].

This classification of QoF is not ad hoc; instead, it naturally corresponds to different types of data traffic in future networking environments. A class A connection typically involves a foreign party that is trusted and authenticated, for example, a host at another site at the same company. A class B connection typically involves an authenticated cooperating party that can be held responsible if it intentionally violates the security policy, for example, a host that belongs to a business partner. In such a case, monitoring and responding promptly is enough to track down the bad guy and guard against further attacks. A class C or D connection typically involves a foreign party that is untrusted, e.g., an arbitrary Internet surfer.

The motivation for classifying traffic into different QoF classes is to optimize the effort to make connections secure. Safer connections are screened using faster but less aggressive checking methods, while less safe connections are screened using more aggressive but more time-consuming checking methods. There are two challenges in implementing the QoF concept. The first is to design firewalling schemes to secure each QoF class (except class A) efficiently, while satisfying the security level required by that class. The other challenge is to design an architecture that integrates all these firewalling schemes. How we address these challenges in our ATM firewall is described in the following section.

2.2 Logical Design of Our ATM Firewall

2.2.1 *General Scenario*. Figure 1 depicts the logical design of our ATM firewall. It consists of five *security services* that interact with each other. Let us briefly explain its overall operation from the perspective of a connection's life cycle. When an ATM *signaling message* requesting a new SVC arrives at the firewall, the *call-screening service* decides whether it is safe to establish the connection by checking the identity (e.g., ATM address) and authentication information contained in the signaling message against the security policy governing call admission. If the connection is allowed, the *call-screening service* assigns a QoF to the connection on the basis of the security policy. If the QoF of the connection is class A, its contents will be exempt from packet-level inspection. Otherwise, depending upon whether its QoF class is B, C, or D, the connection will be screened by *traffic-monitoring service*, *packet-filtering service*, and *proxy service*, respectively. The call-screening service also provides necessary information (e.g., packet-filtering rules) to the security service, which corresponds to the connection's QoF. While the traffic in the connection is screened, its profile information (e.g., amount of traffic within a certain time interval) and the packets that violate security policy are recorded and sent to a *firewall*

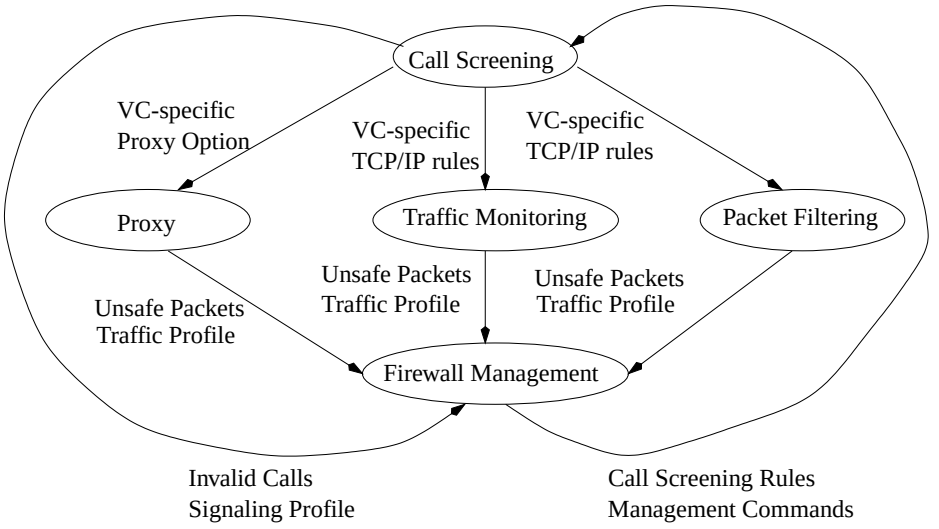

Fig. 1. Logical design of our ATM firewall.

management service, which controls and coordinates other security services. The details of these security services are presented in the following sections.

2.2.2 *Call-Screening Service*. A call-screening service inspects whether two communicating parties are allowed to establish a connection. This is checked when a signaling message arrives at the firewall. A signaling message contains fields about source and destination identity, higher layer information such as the port(s) to be accessed, and/or a digital signature that authenticates the origin of the message. The identity of an endpoint is typically its ATM address, however, it can also be the name of the end user. The firewall keeps a set of call-screening rules, each of which includes but is not limited to following five fields: (1) source identity, (2) destination identity, (3) authentication information (e.g., a digital signature), (4) QoF of the new connection to be established, and (5) information needed for packet-level inspection.

When a signaling message arrives at the ATM firewall, the call-screening service compares source and destination identities in the message with the first and second fields of call-screening rules for a match. If no match is found, the signaling message is blocked and access denied. The number of such rules may be large; but the search can be performed very fast using hashing techniques. The source ATM address of the Class A and B connections needs to be authenticated by a call-screening service using cryptographic measures, as discussed in Tarman [1999], as its spoofing may allow dangerous traffic flow through the firewall without being checked. The third field contains information, such as a digital signature, for authenticating the source end point. The signaling message is blocked if the authentication fails. If the connection is allowed, the fourth field will denote the QoF of the new connection.

If the QoF class of the connection is C, its contents are subject to packet-level filtering. In an ATM firewall, filtering rules for various connections are different and are generated on-the-fly, as follows. First, the ATM firewall determines source and destination IP addresses/prefixes, either from the higher layer IEs (information elements) contained in the authenticated signaling message, or from the source and destination ATM addresses by querying a preconfigured ATM address to an IP address/prefix mapping table. Then, from the fifth field of the security rule, the ATM firewall determines the source and the destination TCP ports. Finally, these two pieces of information are glued to generate a set of filtering rules. For example, if the source is a router for subnet 164.107.*.*, and the destination is a router for subnet 192.41.245.*, and the destination subnet only allows HTTP (port 80) and TELNET (port 23) from the source subnet, then the TCP/IP filtering rules are:

Src IP	Dest IP	Sp	DP	Action
164.107.*.*	192.41.245.*	>1023	80	Allow
164.107.*.*	192.41.245.*	>1023	23	Allow
(All other packets are blocked)				

Such automatic configuration of TCP/IP filtering rules is vital in future ATM environments where a large number of connections will be established within a short period of time. It has three major advantages:

—It can help guard against the spoofing of source IP addresses. Since only those IP addresses/prefixes that might be associated with the source ATM endpoint will be honored, packets that do not match these IP addresses/prefixes will be dropped.

—Network managers need not worry about setting up rules for each SVC when it is established. They just need to configure TCP/IP rules and cryptographic information for trusted outsiders.

—In an ATM connection the number of TCP/IP rules that needs to be checked in each connection is much less than in traditional firewalls because possible source and destination packet IP addresses/prefixes in a connection are limited to a small set.

Similarly, if the QoF class of a connection is B, the call-screening service generates the traffic-monitoring rules, which have the same format as packet-filtering rules, for the traffic-monitoring service. If a connection is of class D, a field in the signaling message will specify the options for the protocol proxied on that connection. For example, if the protocol to be proxied on a connection is FTP, one such option is to grant the source privileges of “put” and “mput,” but not “get” and “mget.”

2.2.3 *Packet-Filtering Service*. A packet-filtering service inspects the headers of IP packets to block “unsafe” packets, while allowing “safe” packets to pass. Like ATLAS [Kowalski 1996], our firewall achieves high

throughput by filtering on the first cell only (or first two cells in case of an IP option) and caching packet headers that are found safe. However, the similarity stops there. Unlike ATLAS, the cache in our firewall is built upon our novel cache architecture, which is much more scalable and cost-effective than CAM. In addition, we introduce a novel scheme, called "Last Cell Hostage" (LCH), into our ATM firewall to further reduce the latency incurred by packet filtering.

When a packet is fragmented, filtering the first packet only cannot guard against attacks that exploit fragmentation such as a *tiny fragment attack* and *overlapping fragment attack* [Ziemba et al. 1995]. So fragmented packets will not be allowed in the packet-filtering service. Since fragmentation can always be avoided by making the packet size no larger than MTU (Maximum Transport Unit) and less than 1% of the Internet traffic is fragmented, this should not cause any inconvenience.

As we have shown in Section 1.2, CAM is not a scalable solution for securing IP traffic at a very high rate (like 1 Gbps). We designed a layer-4 route cache architecture [Xu et al. 1999] that can achieve a high and stable hit ratio at a reasonable cost. The cache architecture is actually designed for an emerging routing technology called *layer-4 switching*, in which a forwarding decision is made not only on the basis of destination address, but also on source address, port numbers, protocol, and possibly some other fields such as TCP flags [Lakshman and Stiliadis 1998; Srinivasan et al. 1998]. Packet filtering is obviously a type of *layer-4 switching* in which the forwarding decision is either "Allow" or "Block." A high hit ratio over 90% can be achieved using only 4.8 Mbytes of RAM for a 1–Gbps ATM firewall. Such a high and stable hit ratio is achieved by using our novel cache management algorithm, called *near-LRU*, that best exploits the *locality behavior* exhibited by the Internet traffic at layer-4 [Claffy 1994; Jain and Routhier 1986; National Laboratory for Applied Network Research 1998]. Locality behavior of network traffic at layer-4 is characterized by the concept of "flow" in Claffy's Ph.D. thesis on Internet traffic characterization [Claffy 1994]. A flow is defined as a series of unidirectional packets that share the same ⟨src-IP, dst-IP, src-port, dst-port, protocol⟩ tuple, referred to as a *layer-4 address*. A flow is started when the first packet of the flow arrives, and is considered *expired* when there has been no activity for a timeout period D_{expire}. A flow is said to be *active* from the time it was created until the time it expires. Our *near-LRU* cache guarantees to cache every *active flow* during its lifetime. With near-LRU cache, there will be exactly one miss for each flow. So the miss ratio of *near-LRU* is $1/PF$, PF being the number of packets per flow. However, *near-LRU* is still a fully-associative cache management algorithm that is hard to implement precisely. Our cache architecture employs an *N-way dynamic set-associative scheme* that much better approximates *near-LRU* than traditional *N-way set-associative cache*, which would incur a large number of *collision misses*. We conducted a trace-driven simulation over a 5–Mbps corporate gateway trace using our cache architecture configured with 1500 entries of

cache and obtained a 92% hit ratio. We proved in Xu et al. [1999] that the amount of cache needed to achieve this hit ratio will increase at most as quickly as traffic volume. So with 300,000 entries (4.8 Mbytes as each cache entry is 16 bytes long), the cache can achieve the same high hit ratio over 1–Gbps traffic. If we use SRAM to store the cache entries, this scheme will cost about 250 dollars. By using the emerging Synchronous Link DRAM (SLDRAM) [SLDRAM Inc. 1998; IEEE Computer Society 1996] technology—a type of DRAM that delivers the same high throughput as SRAM for long sequential read/write using internal prefetching and pipelining—the cost of memory can be further reduced to about 15 dollars. A detailed description of our cache architecture can be found in Xu et al. [1999].

In ATLAS, if the first cell misses the policy cache, the whole packet has to be blocked to wait for a slow software inspection process to finish. In contrast, with our LCH scheme, all cells of a packet except the last one is allowed to pass even if a cache miss occurs. Only the last cell is kept as "hostage" before the software inspection is finished. The last cell is passed/dropped if the inspection determines that the packet is safe/unsafe. When the last cell is dropped, it is substituted with a pseudo last cell whose payload is generated randomly, so that CRC failure of the whole packet at the receiver is guaranteed. This prevents the dropping of one packet from affecting the following packet, which will otherwise be mixed with the previous packet and cause corruption. The introduction of LCH has the following two advantages:

—Even when a packet header misses the policy cache, if the packet is reasonably long the software-based filtering process can be finished before the last cell arrives, which results in no delay. We conducted a detailed performance analysis in Xu and Singhal [1999a] to show that LCH significantly reduces the average amount of delay incurred by a software-based filtering process. Let us explain this by an example. A recent survey on the packet size in WAN shows that the average packet size is around 348 bytes [National Laboratory for Applied Network Research 1998], which will occupy 8 cells if AAL5 is used [Ginsburg 1996]. If we assume that cells belonging to different packets interleave, in a T3 rate (45 Mbps) connection on an OC-3c line, the average lapse between arrival time of the first cell and the last cell of an average-size packet will be 22 cell times. On the other hand, today's software-based firewall technology can perform a header checking with 6 cell times (50,000 packets/second). In this case, there is a 99.8% probability (assuming exponential distribution for interarrival time between any two consecutive cells) that the inspection will finish before the last cell of the packet arrives. The size of a packet is going to increase due to improvement in WAN technology to accommodate large-size packets without fragmentation, which makes this scheme more attractive.

—LCH allows us to efficiently process IP packets with IP option fields. The technical challenge in filtering IP packets with IP option is that the decision process may not even start until the second cell arrives. ATLAS

does not process such packets because it would have to consider each such packet as missing the policy cache. When the percentage of such packets is high, the processing overhead in ATLAS becomes unbearable. However, with LCH, only the first cell is kept as the state information, and the software-based filtering process is postponed until the second cell arrives. Since all of the cells except the last one can still be passed without a delay, the IP packets with IP option can be processed as fast as those without.

Even though the LCH scheme only blocks the last cell of a packet while unconditionally allowing other cells to pass, it is still a safe approach. If an attacker sends a "bad" packet to an internal host, the packet will be discarded by the receiver since its last cell will have been "corrupted."

2.2.4 *Traffic-Monitoring Service*. LCH significantly reduces the amount of delay due to policy cache miss, but it cannot eliminate this delay entirely. In this section we show that the traffic-monitoring service is nearly as secure as the packet-filtering service for TCP traffic and introduces no latency even when a cache miss occurs. Traffic-monitoring service monitors the headers of IP packets contained in class B connections. When a packet in a class B connection arrives from an untrusted WAN, the traffic-monitoring service first checks whether the sender is spoofing an internal address by comparing the source IP with a list of internal IP addresses/prefixes. This operation can be performed fast using a small CAM. If the sender does not claim to be an internal host, the packet will be allowed to pass, but the first two cells that contain the TCP/IP header are duplicated and forwarded to the traffic-monitoring service. The traffic-monitoring service checks the packet headers against the traffic-monitoring rules (generally the same as the packet-filtering rules). When any attack attempt is detected, the traffic-monitoring service will immediately react to the attack, so that the "reply" (if any) from the receiver will be blocked. In addition, actions are taken to prevent the same source from initiating further attacks (e.g., terminating the connection and blacklisting the source).

The traffic-monitoring service achieves nearly the same level of security as the packet-filtering service for TCP traffic, as it effectively blocks the 3-way handshake for establishing unsafe sessions. Suppose a foreign host sent a "dangerous" SYN packet to an internal host. The packet will be passed because of the "after-the-fact" nature of the traffic-monitoring service. However, this service is guaranteed to block the ACK+SYN packet sent from the internal host to prevent the TCP connection from being established. Additionally, the traffic-monitoring service will tear down the connection to prevent the foreign host from making further attacks. However, there are still two loopholes to be addressed. The first is the *sequence number prediction attack* [Bellovin 1996]. By predicting the sequence number of the SYN+ACK packet sent from the internal host, the foreign host can forge the final ACK packet in the 3-way handshake without receiving the SYN+ACK packet from the internal host. The forged packet

is sent along with the SYN packet. If the prediction turns out to be correct, the receiver will assume that the TCP connection is established. This ACK packet may contain a BSD "r" command that requests the deletion of some files. As already explained, the traffic-monitoring service helps guard against this attack by detecting and blocking the spoofing of internal IP addresses. However, it is not effective in guarding against the spoofing of trusted external IP addresses. Other security measures such as a secure shell (SSH) or VPN (Virtual Private Network) cryptography needs to be used together with the firewall. The second loophole is *SYN floods*, in which an attacker floods the victim host with SYN packets. Maintaining state information for half-open connections as in CheckPoint Inc. [1996] can be incorporated into traffic monitoring to help defend from such attacks.

In order for the the traffic-monitoring service to be nearly as secure as packet filtering for TCP traffic, the inspection of a packet needs to be finished before its corresponding "response" comes back. The traffic-monitoring service is able to detect a malicious packet before the "response" from a protected host arrives because a round-trip delay of a packet between the firewall switch and the protected host is always larger than the queueing and service times. Today's firewalls can filter 50,000 packets per second in software [Keylabs Inc. 1998]. This is translated into an average of 20 microseconds for filtering each packet that misses the policy cache. Assuming a worst-case load of 90% and an M/M/1 queuing system [Nelson 1995], packet filtering in software should take no more than $20/(1 - 90\%) = 200$ microseconds, including queuing and service time. TCP implementation is notoriously inefficient, and there is a large body of literature [Watson and Mamrak 1987; Clark et al. 1989; Bjorkman and Gunningberg 1998; Rodrigues et al. 1997] that analyzes the reason behind this inefficiency. It was shown that operating system overhead such as memory management and timer management rather than protocol processing contributes to most of the processing delay [Clark et al. 1989]. Such processing delay used to be tens of milliseconds in the 1980s [Clark et al. 1989] and a couple of milliseconds in the mid-1990s [Bjorkman and Gunningberg 1998]. In a recent study, using high-performance Sun Ultra workstations and low latency LAN, a round-trip delay of 334 microseconds on fast Ethernet (contention-free dedicated link) and 542 microseconds on Myrinet for a minimum size IP packet was recorded [Rodrigues et al. 1997]. Even these numbers are much larger than 200 microseconds, the aforementioned queueing and processing time for a packet that misses the policy cache. So it is pretty safe for us to claim that the traffic-monitoring service can detect the intrusion before the "response" arrives.

On the other hand, we acknowledge that the security of the traffic-monitoring service should not rely on the high latency of TCP indefinitely. As the hardware and software technology keeps advancing, this latency will become smaller and smaller. Also, methods to reduce operating system overhead for processing TCP packets have been investigated for a long

time. Rodrigues has documented that using their user-level fast socket implementation [Rodrigues et al. 1997], a round-trip delay of about 70 microseconds for minimum-size TCP packet can be achieved. Fortunately, policy-based packet-filtering algorithms are also getting faster. Lakshman and Stiliadis [1998] report 1 microsecond worst-case search time with up to 10k filtering rules using a highly parallel hardware implementation. It is suitable for use in the ATM firewall to perform stateless filtering and monitoring. However, stateful inspection requires efficient storage and retrieval of per-flow (here a flow is a TCP connection) state information for a large number (say 1 million) of flows. We are investigating a scheme that can retrieve or store a flow in one memory access [Xu and Singhal 1999b] to be used for stateful packet filtering.

Since UDP does not require a 3-way handshake, the traffic-monitoring service cannot guarantee the security of UDP traffic in Class B connections. It may seem that the traffic-monitoring service can protect the UDP-based services such as NFS and SNMP because the information that an external hacker tries to steal is in the response (return) packet, which will be blocked if the corresponding request packet is found unsafe. However, if the request packet is to delete a file on an NFS server or to update the routing table (through an SNMP request), the damage has already been done before the traffic-monitoring service detects it. Therefore, to play safe, the firewall should block all UDP traffic in Class B connections. This can be performed very fast by checking the protocol field of the first cell of a packet in hardware. ICMP packets can be blocked in the same way. However, since ICMP is shown to be pretty safe [Bellovin 1989] and it may still be useful in debugging TCP connections, it will be selectively filtered in Class B connections.

In conclusion, traffic monitoring is nearly as safe as packet filtering for TCP traffic. Due to its after-the-fact nature, it does not incur any latency, while the packet-filtering service does when the packet size is small and its header misses the policy cache. Using the policy cache to boost performance, the throughput of traffic-monitoring service is no less than that of the packet-filtering service (1 Gbps). There is no technical difficulty in employing many such servers to perform the traffic-monitoring service in parallel.

2.2.5 *Proxy Service*. Proxy service acts as an application-level gateway for a number of Internet protocols. An application-level gateway (also called a proxy server) for a protocol acts as the middleman between the client and the server processes [Chapman and Zwicky 1995; Cheswick and Bellovin 1994]. It interprets certain steps of the client-server interaction and blocks unsafe steps. For example, an application-level gateway for FTP protocol can be configured to allow “get” and “mget,” but disallow “put” and “mput,” from inside to outside (letting a file in but not out). Unlike the packet-filtering service which looks only at the header of the packet, proxy service monitors the execution of the protocol and filters at the application level [Chapman and Zwicky 1995; Cheswick and Bellovin 1994]. However,

since proxy service may need to look into the contents of a packet to understand the protocol and requires SAR (Segmentation and Reassembly), it can only be performed at the rate of a traditional firewall. Fortunately, most protocols (including FTP) can be secured by stateful packet-level filtering protocol, proxying can thus be used sparingly.

Another use of proxy service is to "oversee" the execution of ISAKMP (Internet Security Association Key Management Protocol) [Maughan et al. 1998]. ISAKMP is used to exchange necessary *security association* information between two communication parties such as encryption algorithms, keys, and initialization vectors [Kent and Atkinson 1998c]. The security association will be used by ESP (IP Encapsulating Security Payload) [Kent and Atkinson 1998b] to provide confidentiality and/or by AH (Authentication Header) [Kent and Atkinson 1998a] to provide authentication. If two parties intend to communicate with each other using IP security protocols, they will indicate it in the signaling message. It can be arranged that the first several packets used to exchange security association information go through the proxy service. The proxy service will oversee the protocol steps to make sure that a valid security association has been successfully established. Once the proxy service finds out that the security association is successfully established, it can instruct the ATM firewall to establish a shortcut, so that cells in that connection will be passed without inspection. Afterwards, the firewall keeps on monitoring (in hardware) the SPI (Security Parameter Index) field contained in the packet header [Kent and Atkinson 1998c]. Once it indicates that the security association is terminated (e.g., by resetting to a default value such as 0), the firewall turns the connection back to the "proxying mode."

2.2.6 *Firewall Management Service*. The firewall management service controls and manages other security services in the ATM firewall and provides user-friendly administration tools to network managers. These tools allow network managers to perform operations such as specifying or updating call screening rules, monitoring abnormal user activities, and disabling connections that violate the security policy.

The firewall management system logs two types of events forwarded from other security services. The first type of event is the violation of security policy. This includes unsafe signaling messages detected by a call-screening service and unsafe packets detected by a packet-filtering service, traffic-monitoring service, and proxy service. Such information will be used to analyze what type of intrusion is involved in a violation instance. Security policy may need to be updated in response to the intrusions. The other type of event is the profile information on each connection, such as the identities of communicating parties, start and end times of the connection, and number of cells transported during its life cycle. With such information, an ATM firewall can expect access patterns from each source. This expectation allows it to identify abnormal activities from a certain source, such as an unusually large number of connection requests within a certain time window and an unusually large amount of data transported within a

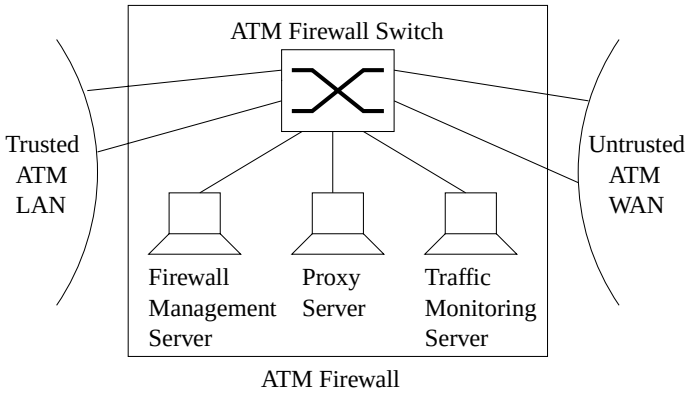

Fig. 2. Physical components of the ATM firewall.

certain time interval. Because such abnormal activities may indicate intrusion attempts, network managers should be alerted.

3. PHYSICAL COMPONENTS OF OUR ATM FIREWALL

In the previous section, we discussed the logical design of the ATM firewall, which consists of five security services. In this section we present the physical components of the firewall and show how the logical functions of these security services are mapped to physical components.

Figure 2 depicts the physical components of our ATM firewall. It consists of an *ATM firewall switch*, a *proxy server*, a *traffic-monitoring server*, and a *firewall-management server*. The proxy server, the traffic-monitoring server, and the firewall-management server are attached to the ATM firewall switch through high-speed links. The ATM firewall switch implements the call-screening service and the packet-filtering service, which are discussed in the next section. The proxy server implements the proxy service and the traffic-monitoring server implements the traffic-monitoring service; discussed in Section 5.1 and 5.2, respectively. The firewall-management server implements the firewall-management service. It is a general-purpose workstation equipped with the firewall-management software, the design of which is not in the scope of this paper.

4. ATM FIREWALL SWITCH

The ATM firewall switch is the most complicated and important component in our ATM firewall. Because the design of a firewall switch is based on a standard switch, in the following we first briefly present the logical components inside a standard ATM switch. Then we show how these components are modified in an ATM firewall switch to perform the packet-filtering function and provide support to the traffic-monitoring server and proxy server.

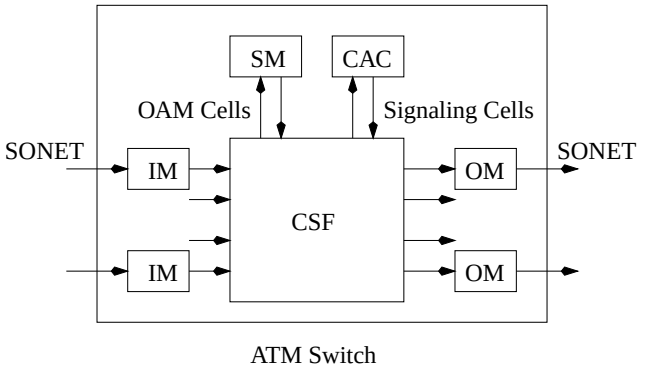

Fig. 3. Internal structure of an ATM switch.

4.1 The Structure of a Standard ATM Switch

A conceptual structure of a typical ATM switch is depicted in Figure 3 (adapted from Chen and Liu [1995]). An ATM switch consists of five functional modules, namely, input module (IM), output module (OM), cell-switching fabric (CSF), call admission control (CAC), and system management (SM). Note that these are just logical modules, actual partitioning of functions varies from implementation to implementation. For ease of discussion, we assume that SONET (Synchronous Optical Network) [Black 1996] is used as the physical layer.

IM extracts cells from the SONET payload envelope, determines the port or module each cell is destined for, and appends an internal tag to each cell. Three types of cells are identified by IM. These are user cells that carry user traffic, signaling cells that carry signaling messages, and OAM (Operation and Management) cells that carry management information. For each user cell, IM looks up its destined output port (or ports for multicast traffic) from the routing table and writes this information into the internal tags. For signaling cells and OAM cells, the internal tags appended by IM indicate that they should be forwarded to CAC and SM, respectively. Cell-switching fabric (CSF) routes the cells coming from IMs to their destinations designated in the internal tags. CAC assembles the signaling cells into a signaling message, processes the signaling message, decides whether or not the call should be accepted based on the availability of resources, or other considerations, and generates new signaling cells. SM processes the OAM cells, performs switch-management functions, and generates new OAM cells if necessary. User cells, new signaling cells generated by CAC, and new OAM cells generated by SM are forwarded to OM through CSF. OM processes and removes the internal tags appended to the cells, and puts the cells into the SONET payload envelopes for transmission. Compared to the organization of a standard switch shown in Figure 3, a firewall switch makes modifications to all five functional modules while keeping its overall structure intact. In the following, we explain the internal structures of these modules and discuss the modifications made to each of them.

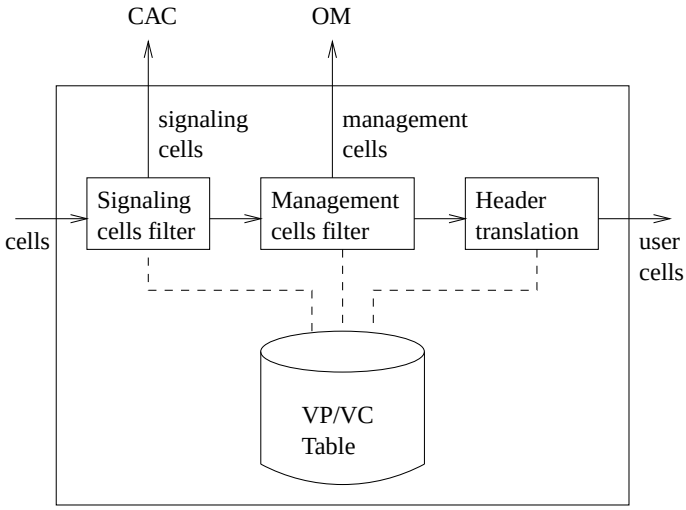

Fig. 4. Cell processing in a standard switch.

4.2 Input Module (IM)

The most important function that IM performs is to translate the VPI/VCI of each incoming cell to determine its output VPI/VCI/port. This is performed by a functional block, called *cell processing*. Since this block will be modified in a firewall switch to add firewall functions, its internal structure is explained in detail. IM also performs other functions such as cell delineation, SONET functions, and UPC/NPC; however, they are not relevant to our firewall design and are outside the scope of this paper.

Figure 4 (adapted from Chen and Liu [1995]) shows the internal structure of a cell-processing block inside a standard switch. It consists of a *signaling cells filter* block, a *management cells filter* block, a *header translation* block, and a *VP/VC table*. *Signaling cells filters* and *management cells filters* remove signaling cells and OAM cells from the cell stream and forward them to CAC and SM, respectively. The remaining two blocks handle the processing of user cells. Each entry in VP/VC table contains at least the following five fields: input VPI, input VCI, output port, output VPI, and output VCI. When a user cell arrives, the following three steps are performed by a header translation block:

(1) It looks up output VPI/VCI/port from the VP/VC table using the input VPI/VCI as the key.

(2) It updates the VPI/VCI in the cell with the output VPI/VCI found in the VP/VC table.

(3) It appends an internal tag to the cell, indicating which output port it should be routed to, whether it is a multicast cell, and other fields used for internal housekeeping.

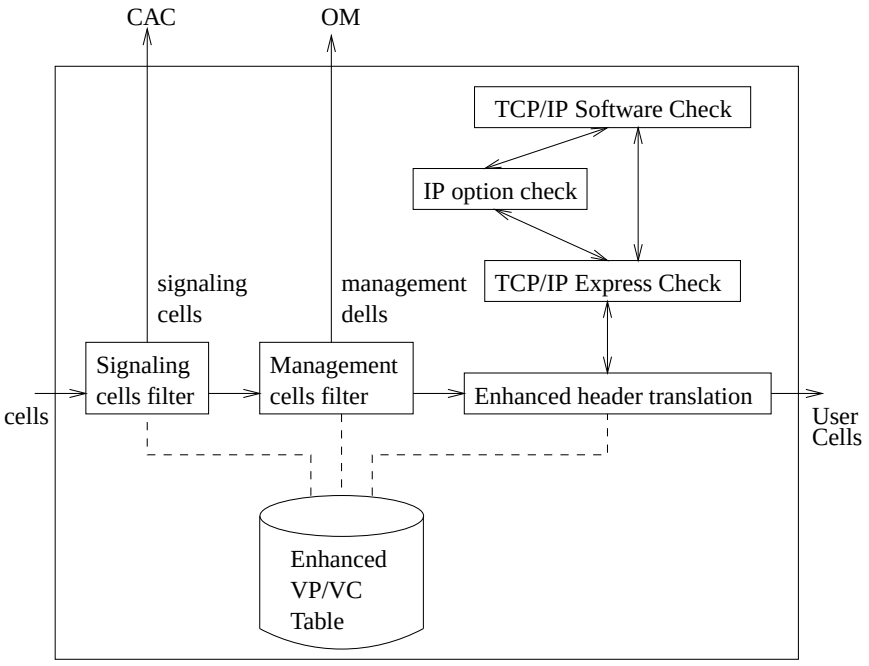

Fig. 5. Cell processing in our ATM firewall switch.

In a firewall switch, *cell processing* is modified to perform firewall functions, namely, distribution of traffic in different connections to different ATM firewall components based on their QoF values and filtering of the traffic in class C connections. Figure 5 presents the modified cell-processing block. Compared to its counterpart in a standard switch, it has enhanced the VP/VC table and the header translation block, and consists of three new functional blocks, namely, a *TCP/IP express check* (TEC) block, an *IP option check* block, and a *TCP/IP software check* block.

The enhanced VP/VC table in a firewall switch contains a number of firewall-related fields, including a connection's QoF, packet-level state information such as the decision on the current packet (pass, drop, or LCH), and cell-level state information such as whether the first or second cell of a packet is expected. The *enhanced header translation* (EHT) block in a firewall switch uses the VPI/VCI of the incoming cell to look up the value of such fields and to perform firewall functions on class B and C connections. With a class B connection, EHT will instruct the cell-switching fabric (CSF) to duplicate the first two cells of each packet in the connection and pass them to a traffic-monitoring server. The internal tag is extended to contain several firewall-related fields (see Appendix A), one of which is devoted to this purpose.

The major firewall function that EHT performs is to filter the traffic in class C connections. In a class C connection, when the first cell of a packet arrives at EHT, it is forwarded to the TCP/IP express check block, which decides whether the packet should be forwarded, dropped, or put into LCH.

In any case, the decision is sent back to EHT immediately, so that the cell will not be delayed. If the decision is to pass (drop) the packet, all cells of the packet are passed (dropped). If the decision is to put the packet into LCH (due to policy cache miss or the existence of IP option field in the packet header), EHT will instruct OM to keep the last cell of the packet as hostage (i.e., indicate that in the internal tag). All these operations are implemented using simple combinatorial logic (see Appendix A) and can be performed within less than a cell time.[2]

The TCP/IP express check (TEC) block employs a policy cache scheme based on our cache architecture [Xu et al. 1999]. When the first cell of a packet arrives at the TEC, it is checked against the policy cache for a match. If a match is found, the decision (forward or drop) is forwarded to EHT. Otherwise, if the packet contains an IP option or the packet misses the policy cache, TEC will instruct the EHT to put the packet into LCH. In the meantime, the cell(s) is forwarded either to the IP option check block or to the TCP/IP software check block. Once the decision whether the packet should be passed or dropped is available from either block, TEC will forward the decision to OM through CSF.

The IP option check block processes packets with IP option fields. It also employs a policy cache to speed up the filtering process. The detailed process is also explained in Appendix A. The TCP/IP software check block examines (in software) the header or possibly the data segment of a packet to see whether the packet is safe. This process is identical to what is used in a traditional firewall.

4.3 Output Module (OM)

In a standard switch, OM processes the internal tag of each cell for housekeeping purposes and updates output VPI/VCI for multicast cells. OM also contains a VP/VC table, which contains fields for housekeeping information such as number of cells transported on each connection and a field that stores output VPI/VCI for multicast connections.

In a firewall switch, OM is involved in implementing the LCH scheme. When the LCH scheme needs to be applied to a packet, IM will indicate that in the internal tag and forward the packet to OM. Decision on that packet will also be forwarded to OM from the TCP/IP express check block as soon as it is available. The responsibility of OM is to keep the last cell of the packet hostage until a decision on the packet arrives. The detailed process is explained in Appendix A.

4.4 Call Admission Control (CAC)

CAC in a standard switch decides whether an incoming signaling request can be accommodated on the basis of whether the requested QoS (Quality of Service) can be satisfied without compromising the QoS guarantees given to existing connections. In a firewall switch, CAC implements the call-

[2]A cell time is around 800ns in an OC-12c line.

screening service as discussed in Section 2.2.2. As specified by the ATM Forum, the identity of the source party is indicated in the signaling message and is authenticated by an unforgeable digital signature using either symmetric (secret) or asymmetric (public) key cryptography [Tarman 1999]. CAC is also equipped with fast cryptographic hardware to quickly authenticate the digital signature contained in the signaling message. Once identity is determined and authenticated, CAC compares the identities of the communicating parties contained in the signaling message with call screening rules to decide whether the connection is allowed to become established. If the connection is allowed, CAC will set up an entry in VP/VC table for the new connection and initiate the firewall-related fields such as QoF. If the QoF class of the new connection is C, CAC will generate the packet-filtering rules and send them to IM. Similar actions are taken if the QoF is B or D. CAC is also equipped with fast cryptographic hardware to quickly authenticate the digital signature contained in the signaling message. A call-screening process can be conducted in parallel with a normal call admission control process and will not introduce any extra delay.

4.5 System Management (SM)

In a standard switch, SM handles all the management plane functions, including fault management, performance management, configuration management, accounting management, security management, and traffic management [Chen and Liu 1995]. In a firewall switch, we need to add a management function, called firewall management, in order to manage the firewall functions that need to be performed by the switch. It performs the following functions:

—Maintaining firewall-related *managed objects* such as call screening rules.

—Executing the commands sent from a firewall management server such as updating call-screening rules.

—Monitoring the performance of a call-screening service and packet-filtering service, such as the throughput of a filtering operation at each port.

4.6 Cell-Switching Fabric (CSF)

CSF in a firewall switch needs to examine the T-MONITOR bit (see Appendix A) in the internal tag, duplicate a copy of each cell with this bit turned on, and pass the copy to the traffic-monitoring server. This can be achieved with little modification, no matter what architecture (e.g., shared memory) CSF is built on.

5. OTHER COMPONENTS OF OUR ATM FIREWALL

5.1 Traffic-Monitoring Server

The traffic-monitoring server is an ATM-attached workstation equipped with policy cache hardware to perform header checking at high speed. Its

implementation is almost identical to the implementation of the packet-filtering service. Many servers can run in parallel (each attached to the ATM firewall switch through a separate port) to increase the monitoring throughput, as explained in Section 2.2.4.

5.2 Proxy Server

The proxy server is a traditional proxy firewall equipped with ATM interface(s). When the ATM firewall switch decides that the QoF of a new connection is D, it will set up two ATM connections, one between the source and proxy server and the other between the proxy server and the destination. Two connections are concatenated in such a way that each packet received from one connection will be proxied by the proxy server and forwarded to the other if it is safe (and blocked otherwise).

6. SUMMARY

A router-based packet-filtering firewall is an effective way of protecting an enterprise network from unauthorized accesses. However, it will not work efficiently in an ATM network because it requires the termination of end-to-end ATM connections at a packet-filtering router, which incurs huge overhead in reassembling and disassembling packets. We offer a viable solution for high-speed packet-level filtering in ATM networks. We designed an ATM firewall that is based on novel concepts and has the following features: it is high in performance, easy to manage, and less restrictive in packet formats (allows IP option fields). Our ATM firewall incorporates our novel concept, QoF, which can potentially increase dramatically the amount of traffic that an ATM firewall can secure per unit time. We also introduced a "last cell hostage" (LCH) scheme to minimize the delay incurred when the cell misses the policy cache and to enable the ATM firewall process packets with an IP option. To demonstrate the feasibility of these schemes, we presented a physical design of the ATM firewall that is readily amenable to hardware implementation.

APPENDIX

A. DETAILED DESIGN OF IM AND OM

The enhanced VP/VC table in IM (input module) contains the following firewall-related fields:

(1) QoF: This field indicates the Quality of Firewalling (QoF) of the connection.

(2) SEL-NO: each packet in a class C connection is assigned a serial number, which is used by the OM in LCH to match filtering decisions with the packets arriving at OM.

(3) EXP-1st: indicates whether the first cell of a packet is expected in the connection.

(4) EXP-2nd: indicates whether the second cell of a packet is expected in the connection.

(5) DROP: indicates whether all cells of the current packet should be dropped.

(6) PASS: indicates whether all cells of the current packet should be passed.

(7) LCH: indicates whether the last cell hostage scheme (LCH) will be applied to the current packet.

(8) OMCHK (Output Module Check): when this field is turned on, OM should check each cell in this connection to see whether there is a queue for this connection at OM.

(9) SEL-NO-LAST-LCH: indicates the serial number of the last packet put into LCH.

In a firewall switch, the following firewall-related fields will be added to the internal tag:

(1) T-MONITOR: denotes whether the cell is among the first two cells of a packet in a class B connection. CSF will duplicate a copy of such cells to the traffic-monitoring server if this field is turned on.

(2) T-SEL-NO: the semantics and value of this field is the same as the SEL NO field in the VP/VC table. It is used by OM in the LCH scheme to match a packet put into LCH (actually the last cell) with the decision on it.

(3) T-LCH: denotes whether the last cell of the packet needs to be kept hostage at OM.

(4) T-OMCHK: the semantics and value of this field is the same as the OMCHK field in the VP/VC table.

The flow diagram of the enhanced header translation (EHT) is shown in Figure 6. When a cell arrives at this block, its VPI/VCI is used as the key to query the VP/VC table for the values of output VPI/VCI/port and firewall-related fields listed above. Different actions are taken depending on the QoF of the connection ("Branch on QoF").

If the cell belongs to a class A or D connection, header processing is performed in the same way as in a standard switch ("Normal Internal Tagging and Processing"). After that, the cell is forwarded to the CSF.If the cell belongs to a class B connection ("Class B traffic?"), the first two cells of the packet should be duplicated to the traffic-monitoring server. The first and second cells are identified by the Exp-1st and Exp-2nd fields in the VP/VC table, respectively. When the last cell of a packet is encountered ("Last Cell?"), which is identified by the PTI (payload type identifier) field [Ginsburg 1996] in the cell header, Exp-1st is turned on ("Exp-1st := TRUE"). When the first cell of a packet is encountered, Exp-2nd is turned

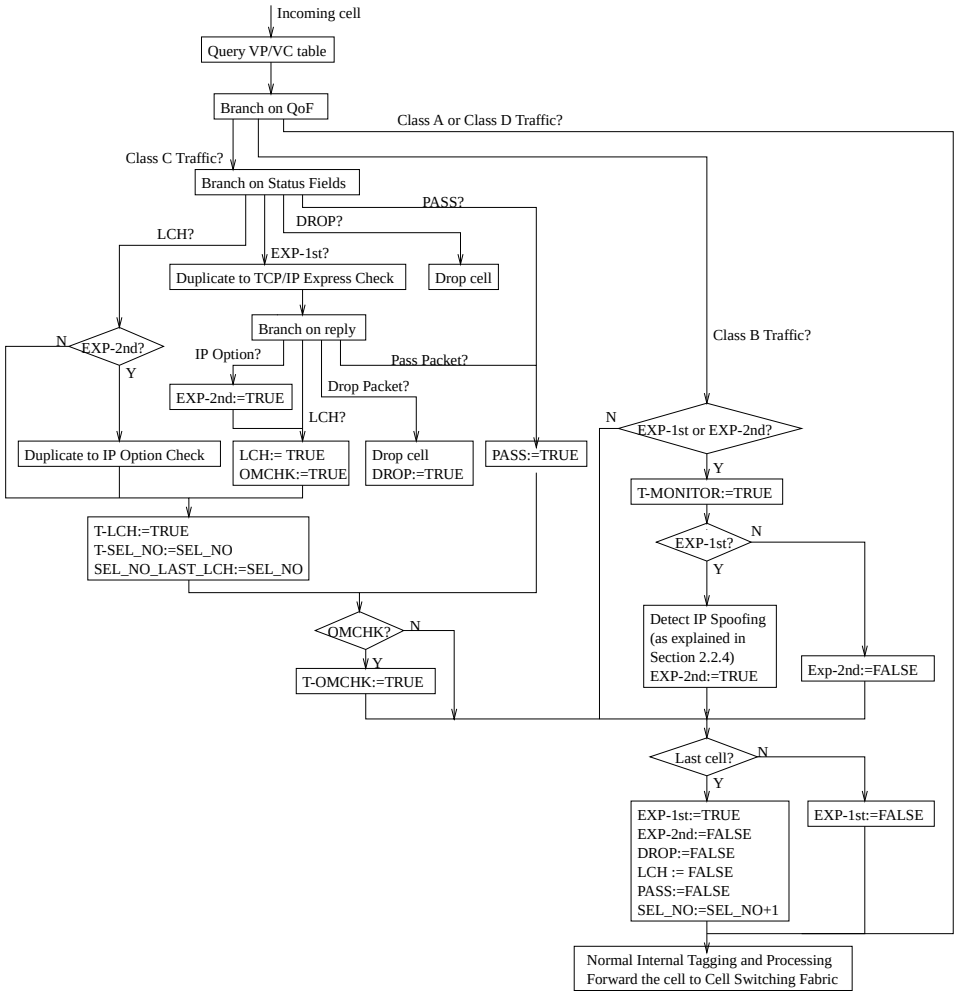

Fig. 6. Flow diagram of the enhanced header translation block.

on ("Exp-2nd := TRUE"). In the internal tags of the first two cells of a packet, the T-MONITOR bits are turned on. CSF will duplicate the cells with the T-MONITOR bit turned on to the traffic-monitoring server.

If the cell belongs to a class C connection that needs to be filtered by the firewall switch, one of four different actions will be taken, depending on the values of four fields ("LCH?," "Exp-1st?," "DROP?," and "PASS?") in the VP/VC table ("Branch on Status Fields"). If the cell is the first cell of a packet ("Exp-1st?"), it will be forwarded to the TCP/IP express check block for the decision. Four different actions are taken, depending on ("Branch on Reply") four possible decisions ("IP Option?," "LCH?," "Drop Packet?," and "Pass Packet?"). If the decision is to drop the packet ("Drop Packet?"), the DROP bit in the VP/VC table is turned on (DROP := TRUE) to indicate that all cells that belong to this packet should be dropped. If the decision is

to pass the cell ("Pass Packet?"), the PASS bit in VP/VC table is turned on (PASS := TRUE), to indicate that all cells in this packet should be passed. If the decision is that the last cell hostage scheme should be used, the LCH bit in the VP/VC table and the T-LCH bit in the internal tag of the cell are set, and the cell is forwarded to OM through CSF. If the decision indicates that the packet contains IP option ("IP Option?"), then the second cell is needed for the decision to be completed. In this case, the Exp-2nd bit in VP/VC table is turned on ("Exp-2nd := TRUE") to indicate that the second cell is expected for inspection. Since the last cell hostage scheme will also be used with this case, the LCH bit in VP/VC table should also be turned on. Whenever the LCH bit is turned on, the OMCHK bit in the VP/VC table should also be turned on and SEL-NO-LAST-LCH should be set to the serial number of the current packet. We will explain their use and rationale later.

On the other hand, if the incoming cell is a nonfirst cell of a packet, the action that needs to be taken on the cell depends on the decision the firewall switch has made on the first cell of the packet. As explained above, there are three possibilities, to drop (if DROP is TRUE), to pass (if PASS is TRUE), or to put it in "Last Cell Hostage" (if LCH is TRUE). The first two cases are straightforward. In the case of LCH, the T-LCH bit in the internal tag of the cell is turned on and the cell is passed to OM. If the Exp-2nd is TRUE, which means the incoming cell is the second cell of a packet and is needed by the IP option check block, a copy of the cell is forwarded to TCP/IP express check block.

In class B, C, and D connections, when the last cell of a packet arrives, the Drop, Pass, and LCH bits should be turned off, the Exp-1st bit should be turned on, and the SEL NO should be increased by one.

The flow diagram of the TCP/IP express check (TEC) block is depicted in Figure 7. When a cell is passed from the enhanced header processing block to the TEC, it must be either the first cell or the second cell of a packet. If it is the second cell ("Second cell?"), it must be expected by the IP option check block (explained immediately below) and is forwarded to it accordingly. Otherwise (the cell is the first cell of a packet), TCP/IP express check block looks to see whether the packet header contains an IP option field ("Contain IP option?"). If the answer is yes, TCP/IP check will advise the enhanced cell-processing block to put the packet into LCH. In the meantime, the cell will be forwarded to the IP option check block and will be buffered there. If the cell does not contain the IP option, TEC will try to match the packet header with the cached TCP/IP filtering rules. If there is a hit, the decision, which is either DROP or PASS, is given back to the VPI/VCI instantly. If a cache miss happens, TCP/IP check block will advise the enhanced cell-processing block to put the packet into LCH. In the meantime, the cell is forwarded to the TCP/IP software check block for further inspection.

When the IP Option Check block or the TCP/IP Software Check block has finished inspecting the packet header, it will forward the decision to the TEC, and then to the enhanced cell-processing block. The enhanced cell-

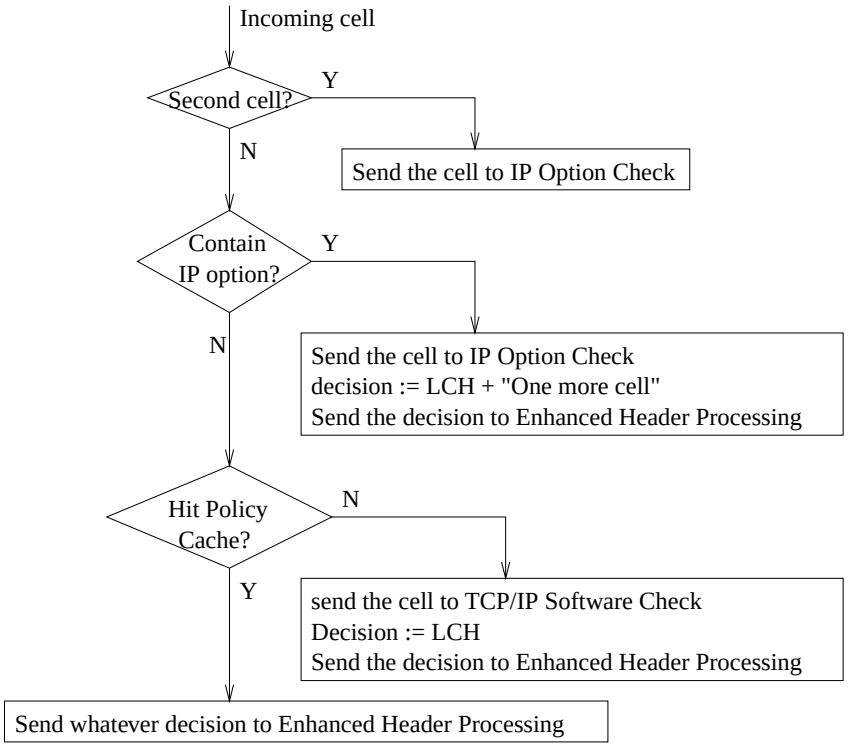

Fig. 7. Flow diagram of TCP/IP express check block.

processing block will forward the decision further to OM to either drop (if the decision is to drop) or forward (if the decision is to PASS) the cell that was put into LCH.

The IP option check block works in almost the same way as the TCP/IP express check block, except that it buffers the first cell of a packet that contains the IP option as the state information. When the first cell arrives, the offset of the TCP header can be precalculated from the IP option size field. When the second cell of a packet arrives, all the necessary TCP/IP fields will be retrieved from these two cells to match the TCP/IP filtering rules in the policy cache. If there is a hit, the decision is given back to TCP/IP express check block immediately. Otherwise the two cells are handed over to TCP/IP software check block. When the decision comes back from the TCP/IP software check block, it is in turn forwarded to the TEC. In addition, the decision will be cached into the policy cache so that later packets with similar header information may hit the cache.

In a firewall switch, OM (output module) is involved in implementing the LCH scheme in the packet-filtering service. When the LCH scheme needs to be applied to a packet, IM will indicate it in the T-LCH field of the internal tag and forward the packet to OM. A decision on that packet is also forwarded to OM as soon as it becomes available. The responsibility of OM is to hold the last cell of such a packet until the decision on the packet

arrives. The T-SEL-NO in the internal tag allows OM to match the last cell of a packet that was kept LCH with the decision on that packet, which contains a serial number equal to T-SEL-NO. While a decision on the LCH packet is pending, packets that follow the LCH packet in the same connection should also be buffered in order to preserve the cell order. This is taken care of by the T-OMCHK bit in the internal tag of a cell. As explained before, IM will turn the T-OMCHK bit on after a LCH packet occurs in the connection. When OM detects such a cell, OM will check whether there is a cell buffered in the same VC. If there is none, the cells should be allowed to pass; otherwise the cells need to be queued up.

ACKNOWLEDGMENTS

We thank Ravi Sandhu, the editor-in-chief of ACM TISSEC, and William Cheswick, the shepherd for the conference version of the paper, and the anonymous referees for their constructive comments, which helped to improve the quality of this paper.

REFERENCES

BELLOVIN, S. 1989. Security problems in the tcp/ip protocol suite. *SIGCOMM Comput. Commun. Rev. 19*, 2 (Apr.), 32–48.

BELLOVIN, S. 1996. Defending against sequence number attacks. RFC 1948.

BJÖRKMAN, M. AND GUNNINGBERG, P. 1998. Performance modeling of multiprocessor implementations of protocols. *IEEE/ACM Trans. Netw. 6*, 3, 262–273.

BLACK, U. 1996. *SONET and T1, Architectures for Digital Transport Networks*. Prentice Hall Press, Upper Saddle River, NJ.

CHAPMAN, B. AND ZWICKY, E. 1995. *Building Internet Firewalls*. O'Reilly and Associates.

CHECKPOINT INC. 1996. TCP SYN flooding attack and the firewall-1 SYNDefender. CheckPoint Inc..

CHEN, T. AND LIU, S. 1995. *ATM Switching Systems*. Artech House, Inc., Norwood, MA.

CHESWICK, W. R. AND BELLOVIN, S. M. 1994. *Firewalls and Internet Security: Repelling the Wily Hacker*. Addison-Wesley Professional Computing Series. Addison-Wesley Longman Publ. Co., Inc., Reading, MA.

CLAFFY, K. C. 1994. Internet traffic characterization. Ph.D. Dissertation. University of California at San Diego, La Jolla, CA.

CLARK, D., JACOBSON, V., ROMKEY, J., AND SALWEN, H. 1989. An analysis of tcp processing overhead. *IEEE Trans. Commun. 27*, 6 (June), 23–29.

GINSBURG, D. 1996. *ATM: Solutions for Enterprise Internetworking*. Addison-Wesley Longman Ltd., Essex, UK.

HUGHES, J. 1996. A high speed firewall architecture for atm/oc-3c. StorageTek Corp. http://www.network.com.

HUGHES, J. AND GUHA, A. 1995. Requirements for secure packet-level access over atm. ATM Forum/95-1126.

IEEE COMPUTER SOCIETY. 1996. Draft Standard for A High-Speed Memory Interface (SyncLink). IEEE Computer Society, New York, NY.

JAIN, R. AND ROUTHIER, S. 1986. Packet trains: Measurements and a new model for computer network traffic. *IEEE J. Sel. Areas Commun. 4*, 986–995.

KENT, S. AND ATKINSON, R. 1998a. IP authentications header. In *IPSEC Working Group*.

KENT, S. AND ATKINSON, R. 1998b. IP encapsulating security payload (ESP). In *IPSEC Working Group*.

KENT, S. AND ATKINSON, R. 1998c. Security architecture for the Internet protocol. In *IPSEC Working Group*.

KEYLABS INC. 1998. Test final report: Firewall shootout Networld+Interop. Keylabs Inc.

KOWALSKI, B. 1996. Atlas policy cache architecture. StorageTek Corp. http://www.network.com

LAKSHMAN, T. AND STILIADIS, D. 1998. High-speed policy-based packet forwarding using efficient multi-dimensional range matching. In *Proceedings of the ACM SIGCOMM '98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication* (SIGCOMM '98, Vancouver, B.C., Canada, Aug. 31–Sept. 4, 1998), M. Steenstrup, G. Neufeld, G. Delp, and J. Smith, Eds. ACM Press, New York, NY.

MAUGHAN, D. ET AL. 1998. Internet Security Association and Key Management Protocol. In *IPSEC Working Group*.

MUSIC SEMICONDUCTORS. 1998. CAM tutorial. MUSIC Semiconductors.

NATIONAL LAB. APPL. NETWORK RES. 1998. Flow statistics analysis for FIXWEST. Available at: http://www.nlanr.net/

NELSON, R. 1995. *Probability, Stochastic Processes, and Queueing Theory: The Mathematics of Computer Performance Modeling*. Springer-Verlag, New York, NY.

PIERSON, L. AND TARMAN, T. 1995. Requirements for security signalling. ATM Forum/95-0137.

RODRIGUES, S., ANDERSON, T., AND CULLER, D. 1993. High-performance local-area communication using fast sockets. In *Proceedings of the Conference on USENIX* (Anaheim, CA, Jan.).

SECANT NETWORK TECHNOLOGIES INC. 1997. Encrypting ATM firewall. http://www.secantnet.com

SLDRAM INC. 1998. 400 Mb/s/pin SLDRAM.

SMITH, T. 1994. Requirements and methodology for authenticated signalling. ATM Forum/94-1213.

SRINIVASAN, V., VARGHESE, G., SURI, S., AND WALDVOGEL, M. 1998. Fast and scalable layer four switching. In *Proceedings of the ACM SIGCOMM '98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication* (SIGCOMM '98, Vancouver, B.C., Canada, Aug. 31–Sept. 4, 1998), M. Steenstrup, G. Neufeld, G. Delp, and J. Smith, Eds. ACM Press, New York, NY.

TARMAN, T. 1999. ATM Security Specification Version 1.0. ATM Forum Technical Committee.

WATSON, R. W. AND MAMRAK, S. A. 1987. Gaining efficiency in transport services by appropriate design and implementation choices. *ACM Trans. Comput. Syst. 5*, 2 (May 1987), 97–120.

XU, J. AND SINGHAL, M. 1999a. Design and evaluation of an atm firewall switch and its applications. *IEEE J. Sel. Areas Commun. 17*, 6 (June), 1190–1200.

XU, J. AND SINGHAL, M. 1999b. Cost-effective flow table designs for high-speed Internet routers. Submitted to *ACM SIGMETRICS* (SIGMETRICS 2000). ACM Press, New York, NY.

XU, J., SINGHAL, M., AND DEGROAT, J. 1999. A novel cache architecture to support layer-four packet classification at memory access speeds. Submitted to *IEEE Infocom 2000*. IEEE Computer Society Press, Los Alamitos, CA.

ZIEMBA, G., REED, D., AND TRAINA, P. 1995. Security considerations for IP fragment filtering. RFC 1858.

Received: February 1999; accepted: July 1999