An FPGA-Based Coprocessor for ATM Firewalls

John T. McHenry, Patrick W. Dowd
Department of Defense
9800 Savage Road, Suite 6512
Fort Meade, MD 20755-6512

Frank A. Pellegrino, Todd M. Carrozzi, William B. Cocks
University of Maryland
Department of Electrical Engineering
College Park, MD 20742

ABSTRACT: An agile firewall coprocessor is described that is based on field programmable gate array (FPGA) technology. This implementation of the firewall enables a high degree of traffic selectability yet avoids the usual performance penalty associated with IP level firewalls. This approach is applicable to high-speed broadband networks, and Asynchronous Transfer Mode (ATM) networks are addressed in particular. Security management is achieved through a new technique of active connection management with authentication. Past approaches to network security involve firewalls providing selection based on packet filtering and application level proxy gateways. IP level firewalls were sufficient for traditional networks but cause a severe performance degradation in high speed broadband environments. The approach described in this paper discusses the use of an FPGA-based front end processor that filters relevant signaling information to the firewall while at the same time allowing friendly connections to proceed at line speed with no performance degradation.

Keywords: firewall, ATM, FPGA, networking security, broadband networks.

1.0 Introduction

Network security is an item of great concern, one that could stall the wide scale deployment of ATM in the public-private arena. This paper describes a new method for establishing network security that provides a strong degree of selectability without negatively impacting the performance advantages of high speed broadband networks.

One of the strengths of the approach described in this paper is its potential for an economically attractive implementation. It enables a significant degree of security at a remarkably low cost. This is very important in terms of its overall usefulness, since high speed networks will soon be widely deployed. Furthermore, even in lower speed environments such as with residential broadband networks (RBB), the approach described in this paper is still viable due to its low cost. The services delivered to the low end could range from 56 kilobits per second to 155 megabits per second.

For all of these deployment possibilities, one thing remains constant. Access outside also allows access inside. Users ranging from large corporations to individual home owners will need to put some protection between themselves and the outside world. The term firewall is used within this document to represent a device placed between an outside network and a protected enclave that prevents unauthorized communication between these two areas.

As shown in Figure 1, the firewall addressed in this paper is comprised of a Firewall Control Processor (FCP) and a Firewall Inline Processor (FIP). The FCP is a dedicated personal computer with a direct connection to the FIP. It provides the processing power for traditional computations and database functions performed by the firewall. A separate access list is used to determine whether a requested connection should be allowed between the outside network and the protected enclave. A more complete description of the FCP can be found in [1].

The FIP is a small card with an FPGA core that accepts or rejects connections between the Internal Network Port (INP) and the external network port (ONP). The FIP acts as a coprocessor by passing data to the FCP through the
control data port (CDP) and reacting to commands returned by the controller.

The purpose of this paper is to describe the functionality, structure, and benefits of the FPGA-based FIP. The paper begins with a brief overview of ATM networks with a concentration on the concept of active connection management that is proposed in this paper. The paper continues with an in-depth discussion of the capabilities that the FIP provides. This discussion briefly mentions the aspects of the FCP that interact with the FIP. A description of the FIP components is provided in Section 4, and the performance of the system is addressed in Section 5. The paper concludes with a summary of the results of this work and a discussion of future work that is planned in this arena.

2.0 ATM Firewall Overview

In order to understand the functionality of the FIP, some background information is required. This section provides a brief overview of ATM communication, a discussion of the concept of active connection management, and other work that has been performed relating to firewall technology for broadband networks.

2.1 ATM Communication

This discussion of ATM communication focuses on the aspects of the protocol components and traffic trends that pertain to the firewall implementation.

2.1.1 Data Flow and Signaling Structure

ATM is a cell-based communication protocol in which all of the data transmitted in the network is broken up into 53-byte cells. Each cell contains a five-byte header and forty-eight bytes of payload. The source and destination of a cell are not identified in its header. Instead, the cell headers only contain edge information. The header contains a 12-bit virtual path indicator (VPI) and a 16-bit virtual channel indicator (VCI) that assign the cell to a specific virtual data flow along a physical link between two switches.

ATM is circuit switched. A circuit or path must be formed between source and destination before any data can be transmitted. This path is formed by initiating cells that pass between switches on specific (VPI,VCI) pairs that are reserved solely for signaling. The payload of these cells contains information about the network service access points (NSAPs) for the source and destination of the connection that is requested. This information allows the switch to identify an appropriate route. The signaling messages used to create a connection are shown in Figure 2. The signaling channel is also used to pass status and state information between the switches.

Once the connection has been established, cells launched by the source are transported to the destination through intermediate ATM switches via hardware routing. The connection setup process sets up the route mapping (input port, input VPI, and input VCI) to (output port, output VPI, and output VCI) at each intermediate ATM switch.

Software routing is not performed at intermediate ATM switches since the route has already been constructed. This allows for fast switching, since the examination and routing is performed in hardware, but it makes it difficult to identify the source and destination of the traffic. Figure 3 shows a simple case where different VPI/VCI pairs are used along each edge to route a flow through the network. Although the same VPI/VCI pair is typically used for each direction of a bidirectional flow along a given edge, this is not required.

2.1.2 ATM Adaptation Layer

The ATM Adaptation Layer (AAL) maps the services provided by the ATM Layer into the upper layers of the protocol stack. AAL functions reside in two layers called the convergence sublayer (CS) and segmentation and reassembly (SAR). Segmentation provides a mapping of higher layer protocol data units (PDUs) into the information payload of the 53-byte ATM cells. Reassembly examines the payload information and builds the original PDU at the receiving end.

Several AAL service classes are available that specify the manner in which a PDU is mapped into ATM cells. They also provide varying degrees of service relating to timing, bit rate, and connection modes. AALS, for instance, is a popular adaptation layer for data communication and connectionless service.
2.2 Firewall Processing Techniques

Traditional networks usually install a firewall, of either the proxy or IP filter variety, to provide security. A firewall filter examines incoming packets and determines the source and destination IP addresses. It can also examine the specified port ID at the TCP level which may require additional reassembly above the IP layer. This filtration process is software intensive and often located within IP routers.

An ATM system cannot check cell by cell in an analogous fashion with the packet by packet checking of IP firewalls. Traditional IP firewalls could be performed on ATM networks, but each IP packet would have to be re-assembled and the IP header would have to be located within the PDU.

One possible method to improve performance is to avoid the adaptation layer re-assembly by detecting the start of an AAL5 PDU. Once the initial cell is identified, the IP header is often contained within its payload. This initial cell is examined while the remaining cells of the flow are allowed to pass until a proper policy for the flow is determined. One additional problem, however, is that ATM is a networking technology that enables integrated traffic. An effective ATM firewall needs to provide security for IP traffic and also for non-IP traffic such as voice, video, and data communication.

2.2.1 Active Connection Management

The problem with many firewalls is that they operate at the upper layers of the protocol stack. The approach proposed in this paper is far more broad and suitable to high speed networks where selection is made during initial connection negotiation.

The key to the proposed approach is active connection management. The idea is to examine the signaling traffic flowing across the firewall boundary. This represents a major change in the way protection is established. There are a few reasons why this is significant. The most important aspect, however, is that this is the only way to determine, with any sense of certainty, the source and destination of the connection.

There are three aspects to active connection management: participant verification, route validation, and endpoint authentication. Each of these aspects is used in concert to ensure a secure enclave.

Participant Verification

When a new flow is created, three essential messages are sent. These flows are illustrated in Figure 2. In the participant verification process, the sources and destinations of flows are identified by examining the signaling messages that set up the connection. Once the firewall verifies that the participants cited in the signaling message are acceptable, the firewall proceeds by performing authentication services.

Route Authentication

The discussion in this section is restricted to the Private Network-Network Interface (PNNI) since this interface will typically be used in the portion of the network associated with the firewall. This is because the firewall is expected to be used more at the edge of a the private enclave, where PNNI is prevalent, rather than within a public network [2].

PNNI is a hierarchical protocol that uses information hiding to provide scalability. Switches are divided into peer groups and communication outside a peer group is provided by the border node and the peer group leader (PGL). The border nodes are interconnected into another level of a peer group. They interact the same as in lower layers except that the routing and signaling are decoupled. This allows different signaling protocols to be used at different levels of the hierarchy.

Through the facilities of PNNI, a designated transit list (DTL) is used to specify the route through a peer-group. By examining the DTL, the firewall can ensure that not only are the source and destination NSAPs located on opposite sides of the firewall, but that the path of the signaling message is reasonable for their claimed locations.

When a connection is being created, the route of the connection is included within the signal flow. Crankback is a mechanism that uses this information to re-route an attempted connection that required link resources that were not available on the initially specified path. Information provided for the crankback mechanism can be used to deter spoofing since analysis can be performed on the route to determine if the path seems reasonable. This helps identify possible impersonations and misrepresentations.

The key to route authentication is that the firewall can learn expected flows between friendly endpoints. There is a sense of locality of reference in that a large fraction of the traffic comes from stable and well known locations. Within this class of friendly traffic, route authentication can be quickly applied. This allows the routes of frequent communicators to be quickly authenticated. For connection requests coming from sources that are unknown or not well characterized, additional resources can be applied to achieve route authentication. It is important to note that the number of such in-depth investigations is low due to locality of reference. This helps maintain the low cost of the firewall.

Endpoint Authentication

One point of concern is to enforce a consistent level of security within the enclave. This becomes increasingly difficult as numerous low-end machines are connected to a local network. Even when security procedures are applied in a consistent manner, they are sometimes disabled and circumvented when the protection impacts performance.

There can be some flexibility. The level of security required for communication within the enclave is usually not the same as for communication that crosses the enclave boundaries. The approach proposed here is a dual authentication process.
The idea is that while the signaling is flowing, trying to create a connection, the endpoints are authenticated with the firewall via in-band signaling. The use of in-band signaling was approved by the ATM Forum during the February 1997 meeting [3]. These authentication processes proceed in parallel in the two non-overlapping regions. As shown in Figure 4, one region is between the outside endpoint and the firewall, and the other region is between the firewall and the inside endpoint. This allows a degree of concurrency and also allows the authentication strategies to differ. This flexible solution is well suited to the dynamic nature of future computer systems that include mobile computing.

2.2.2 ATM Traffic Composition

A significant advantage with the method described here is that it is far more general and flexible than traditional approaches that rely on features that are specific to IP traffic. Native ATM applications are expected to become popular, so it is essential that non-IP traffic is supported. This section defines how specific traffic flows are handled. The discussion below describes processing that is performed in addition to the active connection management described in a previous section.

IP Traffic

With IP class traffic, many of the traditional IP approaches can be applied on top of the ATM selection methods. This is true as long as the first cell of the AAL5 PDU that is carrying IP is found.

The first cell of an AAL5 PDU contains the TCP and IP headers. Depending on the IP options used, the second and third cell of the PDU may also be required. Once this information is obtained, the firewall controller determines the suitability of the flow based on traditional IP packet filters such as the source and destination IP addresses and Port IDs.

Nothing is done when the packet is deemed acceptable. When the datagram is viewed to violate some filter rule, however, the controller instructs the firewall to terminate any additional cells associated with this PDU. This results in the tail end of the IP datagram being clipped. If the last portion of the AAL5 PDU is lost, the entire PDU is lost since the Segmentation and Reassembly (SAR) controller in the receiving ATM Network Interface Card (NIC) will discard it. Once the firewall decides to clip a given PDU, it can also decide to permanently disable the flow associated with that PDU or to allow future PDUs in that flow to proceed. The action that is taken depends on a predetermined policy used to configure the firewall. In addition to these approaches, methods such as contents analysis can also be applied to determine the suitability of an IP flow.

Non-IP Traffic

Non-IP type flows are generated by many different applications. These include traditional data communications that uses a native ATM API, video, and also telephony class applications. In the future, non-IP traffic may represent a large fraction of the total traffic. The magnitude in terms of volume of this class of traffic depends on many factors.

IP is well known due to the Internet, but some applications are not well served by it. This is true for important telephony applications, such as support for voice and video, and also some data applications.

Although there is much interest in the IP community to support video, IP is not the most appropriate protocol for video. It is not clear that IP will be used as the framework for its transport. It is possible to avoid the added overhead of IP by transporting video in a more native format. The same argument can be made with voice traffic. It is unlikely that voice will be used in the context of IP even though some internet phone applications are becoming popular.

The above observations imply that much of the non-IP traffic will have specific timing characteristics. The method described here would protect both voice and video in ways not possible with traditional IP firewalls. Furthermore, through traffic characterizations, it can determine if the flow is in fact video or voice and use this to compare with the negotiated values to determine if the flow is behaving as expected. This characterization helps deal with the inside threat, where a disgruntled employee may be tempted to cause an organization harm by releasing unauthorized information.

2.3 Related Work

There are several groups currently researching the topic of security in an ATM environment [3,4,5]. Much work is concentrated in authentication and authenticated signaling [3]. Secant Network Technologies is currently developing an encrypting ATM firewall [6]. This project addresses access control by requiring the establishment of a security context between communicating endpoints, Secant's cryptographic units, during call setup. The COAST
Laboratory at Purdue University has recently begun investigating firewalling and new security technologies for network perimeter defenses. Network Systems Corporation (NSC) is developing high-end ATM Firewalls [7]. Their approach uses low-level hardware to filter ATM traffic based on both VPI/VCI and on IP header information. Recently, NSC developed a method to cache policy information to speed up the verification [8].

3.0 FIP Functional Description

The FIP performs bit-level processing and routing tasks on high speed ATM data streams at real-time rates. The functional description of the FIP begins with a description of the interfaces between the FIP and its surroundings. It continues with a discussion of the Flow Status Indicator (FSI) table that is a central element in this high-performance, low-cost firewall implementation. The discussion concludes with a description of the runtime operation of the FIP. This includes a discussion of the procedures used to initialize the processing functions and the operations that it performs in steady state operation. This operation involves a description of the processing functions that are performed on the incoming data streams and the interaction that occurs between the FIP and the FCP.

3.1 FIP Interfaces

As shown in Figure 5, the FIP interacts with its surroundings via three ATM data ports. The protocols that are used to communicate over these ports, as well as the internal connections that are used to route data between these ports, are discussed in the following paragraphs.

The CDP could have been implemented using other approaches such as RS-232 or ethernet. ATM was used because it was a simpler, although slightly more expensive, alternative. Another possible implementation would be for the CDP port to be replaced by a PCI bus interface. In this case, the FIP would reside in one of the FCP's PCI slots.

Two types of communication occur on the CDP link. A separate bidirectional permanent virtual channel (PVC) is established on the CDP link for each of these distinct flows. The first type of communication is command flows between the FCP and the FIP that occur on the Command PVC (CPVC). This communication is transmitted using standard ATM cells with a single instruction contained in the payload of each cell. The other type of communication is the flow between the FCP and each of the FIP inline ports. The Incoming PVC (IPVC) contains flows from the INP to the FCP and from the FCP to the INP; the Outgoing PVC (OPVC) contains flows from the OPVC to the FCP and from the FCP to the OPVC.

Cells arriving on the IPVC and OPVC of the CDP are reformatted and transmitted on the INP and ONP respectively. Data in the IPVC and OPVC use the same format. The intent of these channels is to allow the FCP to transmit and receive complete ATM signaling cells from the inline ports. Cells traveling on the CDP link contain five bytes of routing information and a 48-byte payload. Unfortunately, complete 53-byte cells from the inline data streams must be packed into the 48-byte payload of cells in the CDP link. This means that more than one cell on the CDP link is required to hold cells traveling to or from the inline ports. Note that this is not a significant performance factor since only signaling traffic is typically transmitted on the CDP link, and it represents only a tiny fraction of the total traffic.

A two step tunneling process is used to transform cells from the standard ATM format used by cells in the inline data stream into the format used on the IPVC and OPVC channels. The first step is to add three bytes of padding to the ATM cell to increase its size to fifty-six bytes. The three padded bytes are added between the header and payload of the standard ATM cell. They allow the cell to be handled more easily by the FCP software. A typical modified cell is shown in Figure 6.
The next step is to pack these modified cells into 53-byte cells that are transmitted on the CDP link. The initial five bytes of every cell on the CDP link contain the standard ATM header information that indicates which channel the cell belongs to. The remaining forty-eight bytes of the standard ATM payload hold the contents of the modified cells. The first cell of a CDP link flow contains the first 48-bytes of the first inline cell being transferred. These 48 bytes contain the 5-byte header, the three bytes of cell padding, and the first forty bytes of the modified cell's payload. The second cell on the CDP link contains the remaining eight bytes from the payload of the first modified cell, followed by the five byte header, three bytes of padding, and first thirty-two bytes of payload of the second modified cell. This process continues until all of the data in the modified cells have been packed into the CDP cells. Padding is added to the fill out any remaining space in the final CDP cell. Figure 7 shows this packing process for a stream of three cells. It must be noted that a separate bit-pattern is be used for the end of message padding than for the three bytes of padding that is used to align the cells on an even word boundary. In this way, the receiver can determine if the payload remaining in the CDP cell after a complete modified cell boundary is part of a new cell or part of the padding at the end of a message.

![Typical CDP cell stream](image)

**FIGURE 7. Typical CDP cell stream.**

In general, one inline cell is packed into two cells transmitted by the FIP on the CDP link, with the remainder of the cargo of the second cell filled with padding. The rate of the signaling flow is many orders of magnitude less than the rate of data flows, so the inefficiency associated with packing one received cell into two outgoing cells is not significant. Communication from the FCP to the FIP, on the other hand, packs the cells that are heading to the inline stream into as few cells as possible on the CDP link. This is because the FIP typically does not know how many cells there are in a complete PDU when it receives a cell from the inline link. Thus, it must transmit each cell as it arrives without regard for whether it is only one cell from a complete PDU. The FCP, on the other hand, has knowledge about the longer PDU it is sending before it sends the first cell. Thus, it can pack all of the cells from a given PDU into the cargo of the consecutive cells transmitted on the CDP link.

The FIP identifies the end of a stream of traffic from the FCP by noting when there is a change in the VPI/VCI of cells on the CDP link. At the beginning of a stream, the encapsulated ATM header is found at the beginning of the payload. As long as the FIP continues to receive cells on the same VPI/VCI, it should find the next encapsulated cell at 56 bytes increments. Since dropped cells are a rarity, the header should appear in the expected place.

There is a possibility of errors occurring on the CDP link affecting the FIP. ATM cells are flowing on the CDP link at a maximum bit rate of 155.32 Mbps (OC-3c). Bit errors occurring on this fiber connection typically have an E-9 bit-error rate (BER). The last 8 bits of the ATM header is the HEC field which provides for error detection and the correction of a 1 bit error. ATM cells are dropped at the receiver if two or more bits in the 40-bit ATM header are received in error. In the case where a cell is dropped, the FIP must be able to resynchronize the encapsulated cells. Since the traffic flow is encapsulated, the encapsulated ATM cell headers are not scrutinized by the ATM receiver. The encapsulated ATM cell are subject to the BER of the link. At OC-3 line speeds we can expect a flipped bit in the payload on average of every 7 seconds.

The 3-byte padding in the encapsulated cells is used to indicate encapsulated cell boundaries. The FIP expects each byte of the padding to contain the same predetermined value. A three vote majority is performed on those bytes to determine whether or not a cell has been dropped. The FIP assumes that an encapsulated cell was lost if it encounters the wrong information in the added boundary fields. There is a risk of misdiagnosing encapsulated cells as lost due to bit errors in the boundary fields. By performing a three vote majority on three one byte words, however, the probability of misdiagnosing a lost encapsulated cell becomes very small.

In the case when an ATM cell is actually dropped, the FIP will be examining the payload portion of another encapsulated cell. This can be illustrated by examining the sequence of cells in Figure 7 on page 6. If the second CDP cell is not received, the FIP will examine the payload of the second encapsulated cell in the third CDP cell when it is actually attempting to locate the header of the second encapsulated cell in the payload of the second CDP cell.

### 3.1.2 FIP Internal Data Paths

When a cell is received at an input port it is stored in a buffer along with a status word that identifies the destination or destinations of the cell. Cells received on either of the inline ports can either be dropped by the FIP, output on the CDP, or output on both the CDP and the opposite inline port. The
mechanism used to determine appropriate destinations for the messages is described in Section 3.2.

A separate incoming pointer for each port identifies the location in the buffer where the next received cell from that port should be stored. Two additional pointers identify the last location where each of the two potential outgoing ports have read data from. Whenever an output port sees that one of its pointers in one of the incoming message buffers is less than the incoming pointer for that buffer, it examines the status word associated with the cell currently stored at the outgoing pointer to determine if it should be output on its port. If so, it reads the cell from the buffer, performs any necessary formatting, and transmits the formatted cell on its link. When the cell is output, or if the status message did not indicate that the cell should be output on the port, the contents of the outgoing pointer is incremented to point to the next message in the buffer. An example of this buffering structure for data arriving on the ONP is shown in Figure 8. A similar structure is used for the CDP and INP. It should be noted that each of the ports has a separate outgoing message pointer in each of the other two incoming ports. Thus, it must arbitrate between the outgoing message buffers it extracts data from. Also, it must extract a complete cell from one of the message buffers before it attempts to extract a cell from one of the other buffers.

3.2 Flow Status Indicator Table

A key element of the firewall operation is the flow status indicator (FSI) table that determines what action should be taken for a cell received on a given VPI/VCI pair. A two-tiered look up table algorithm is used to implement the FSI function. The first step uses a hash function based on the direction of the flow and the value of the VPI/VCI pair to map a given flow into one of 128K entries in the FSI table. Each entry of the table, in turn, is comprised of eight 32-bit subfields that distinguish between cases where two different VPI/VCI pairs map to the same entry in the FSI table. These subfields also indicate what action should be taken with a given flow. The procedures used for the mapping process are discussed in the following sections.

The first step in the table lookup is to determine which of the 128K entries in the FSI table is accessed by a given VPI/VCI pair. The entry in the table corresponding to a given VPI/VCI pair is assigned by a binary hash of the twelve bits in the VPI, the sixteen bits in the VCI, and one bit to indicate the direction that the message is flowing. The output of the hash function is a 17-bit value that is used to address an entry in the FSI table. The mathematical formula used for the hash is as follows:

\[
H(16) = \begin{cases} 
0 & \text{for flows from outside enclave} \\
1 & \text{for flows from inside enclave} 
\end{cases}
\]

\[
H(i) = \text{VCI}(i) \text{ for } 12 \leq i \leq 15 \\
H(I) = \text{VCI}(i) \oplus \text{VPI}(11 - i) \text{ for } 0 \leq i \leq 11
\]

where VCI(i) is bit 'i' of the VCI, H(J) is bit 'j' of the hash, and VPI(k) is bit 'k' of the VPI.

As can be seen from the hash function, many different VPI/VCI pairs map to the same hash value. It should be noted, though, that no two VCI's paired with a given VPI will collide. The structure of the FSI table allows it to distinguish between up to seven different VPI/VCI pairs that map to the same hash value. As shown in Table 1, each entry in the FSI table contains a status field and seven flow information fields (FIFs). Figure 9 shows the format of an FSI status field. The lower three bits contain a binary value that specifies how many active flows are currently held in the table. The flows, however, are not necessarily stored in order. Thus, the FSI status entry contains an additional seven bits that are set to '1' only when their corresponding FIF contains an entry for an approved flow.

| FSI Local | FSI Contents |
|-----------|--------------|
| 000       | Status field |
| 001       | Flow 1 information |
| 010       | Flow 2 information |
| 011       | Flow 3 information |
| 100       | Flow 4 information |
| 101       | Flow 5 information |
| 110       | Flow 6 information |
| 111       | Flow 7 information |

**TABLE 1. FSI entry format.**

| VALUE | 10 9 8 7 6 5 4 3 2 1 0 |
|-------|-------------------------|
| BIT POSITION | V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 |
| N1 = 10th Bit of the Binary Coded Number of Valid Entries |
| V1 = Marker Denoting Whether the 10th Entry is Valid |
| ☛ = Unused Bit |

**FIGURE 9. FSI status field format.**

For every approved flow, an entry is added to an FIF that
specifies the identity of the flow and the manner in which it should be handled by the firewall. The structure of an FIF is presented in Figure 10.

| VALUE | CTRL WORD | VPI | VCI |
|-------|-----------|-----|-----|
| BIT POSITION 31 | 28 | 27 | 16 | 15 | 0 |

**FIGURE 10. FSI flow field format.**

The lower twenty-eight bits of an FIF contain the complete values of the VPI/VCI of the stream. The upper four bits contain a control word that specifies how the stream should be handled. The meaning of the upper four FIF bits is shown in Table 2.

| Bit Number | Function                  |
|------------|---------------------------|
| 31         | x Reserved                |
| 30         | x Reserved                |
| 29         | 0 do not send cell to FCP | 1 send cell to FCP          |
| 28         | 0 drop cell from inline stream | 1 pass cell along inline stream |

**TABLE 2. FIF bit definitions.**

The addition and deletion of entries to the table is directed by commands from the FCP. Approved flows are added by updating the entries in the FSI. Whenever an approved flow is added, its corresponding entry in the FSI is determined by a hash on the VPI/VCI value associated with the flow. The count value in the FSI Status Field is incremented, and the validity bit corresponding to the FIF associated with the flow is set to ‘1’. Next, the value of the associated FIF is modified to contain appropriate entries for the flow.

### 3.3 Firewall Operation

The operation of the firewall is broken into two sections. The first section discusses the processes that execute during the initialization of the FIP. The second discusses the firewall’s steady state operation.

#### 3.3.1 Initialization

When the firewall is initially activated, the FIP sets the status field of all of the entries in the FSI to zero. The FIP also performs a number of other routines that configure it to interact with the data streams that it interfaces to. These parameters include setting the parity of the data busses, taking the communication ports out of the reset mode, and setting whether the FIP should operate in test mode or the normal data mode. Once these configuration words are completed, the FIP begins to act in the firewall mode.

#### 3.3.2 Steady State Operation

In steady state operation, the FIP monitors and reacts to the arrival of cells on its data ports. Cells arriving over the CPVC of the CDP contain commands that the FIP should execute. Typically, these commands will instruct the FIP to add or drop entries from its FSI table. The FCP can also command the FIP to completely reset the contents of the FSI table. Cells arriving on the IPVC and OPVC of the CDP will be reformatted and transmitted on the INP and ONP respectively. Whenever a cell arrives on either the INP or the ONP, the FIP uses the VPI/VCI pair of the arriving cell as an index into the FSI table. The FIP uses the data stored at this index to determine if it should allow the cell to continue along the inline link, pass the cell to the FCP, or drop the cell from the network.

The FCP provides the security aspect to the firewall by assigning the paths used to route traffic through the FIP. Upon initialization, the FIP blocks all traffic to the INP and ONP because all of its FSI table entries are empty. The FCP initiates its firewall functions by commanding the FIP to add the ATM signaling channels to the FSI table. The identity of the signaling channel is determined by the version of the software, such as SPANS [9] or UN1 [10], running on the ATM switches attached to the link. The signaling traffic will be allowed through the FIP, and a copy of it is sent to the FCP so that it can determine its suitability.

The FCP inspects the signaling messages it receives from the FCP and determines whether or not to allow a connection based on the source and destination addresses of the nodes that are attempting to create a connection. In the case where the connection is allowed, the FCP commands the FIP to pass cells on the particular VPI/VCI pair negotiated for that connection. Cells on this VPI/VCI pair will continue to transparently pass through the FIP until the connection is released. When the FCP notices that a connection release message has been sent, it commands the FIP to disallow further traffic on the associated VPI/VCI pair. The FCP continually examines all of the signaling cells passing through the inline link and commands the FIP to modify its FSI table to react to the creation and deletion of flows on the link.

### 4.0 FIP Component Description

The FIP, shown in Figure 11, is comprised of a small number of low cost components. These components include an FPGA processing core, SONET/SDH framing chips, ATM line interfaces, a 1M by 32-bit RAM, two 64K by 16-bit RAM, and oscillators and signal drivers to generate the system clocks that synchronize logic functions on the board. Each of these components is discussed in detail in the following paragraphs. The manufacturer and model number of the components used in the initial prototype are cited. It should be noted; however, that these citations are only intended to more fully define their expected functionality. Similar components that are available from a number of manufacturers could be used as
replacements for most of these parts.

FIGURE 11. Firewall inline processor board.

4.1 FPGA Processing Core

Due to speed requirements, there is no general purpose processor in the FIP design. Instead, processing is performed by a Lucent Technologies 2C40A FPGA [11]. This FPGA is comprised of an estimated 45,000 gates of reconfigurable logic. The configuration of this logic is programmed by the user before runtime and burned into a set of serial EEPROMs. The FPGA downloads its initial configuration from the EEPROM every time the device is powered on. The FPGA processing core is responsible for providing connectivity and control information to all of the memories and data ports on the FIP. An important aspect of this connectivity is the fact that the FPGA has a separate set of data and control lines for each component. This allows the FPGA to simultaneously perform processing and I/O functions for all of its connected components.

It is also important to note that the FPGA processing core is the only component that requires or allows the user to alter its functionality. All of the other components are dedicated, off-the-shelf devices that perform specific predefined functions. The FPGA controls the functionality of the FIP by defining the data flow and interaction between these dedicated components.

The fact that the processing functions on the hardware card are performed by an FPGA imposes specific limitations on the types of operations that are allowed. As a benefit, however, the FPGA processing core allows the firewall to rapidly identify and route incoming data streams. When the need to perform higher level functions arises, the FIP passes messages to the FCP and then acts on any commands sent back from the FCP. The timing involved with waiting for commands from the FCP is a key factor in determining the division of functionality between the FPGA processing core and the FCP host computer. The FPGA handles functions relating to the data flow where a rapid response is required. The controller handles more complex functions where latency can be tolerated. This two-level processing strategy achieves high performance since time critical operations are performed in hardware. It also is a low cost solution since a single FPGA performs all of the processing functions on the FIP and the controller is a standard personal computer.

4.2 SONET/SDH Framing Chips

The SONET/SDH framing chips provide an interface between the digital control and data flow logic on the FIP and the ATM line interfaces. The prototype design uses PMC-5346 chips to perform this interface function. The receiving side of the interface removes SONET/SDH framing from the received data stream and delivers ATM cells contained in the incoming data stream to the FPGA control logic. The transmitting side of the interface accepts complete cells from the FPGA control logic and adds the necessary SONET/SDH framing before transmitting them on the link. This is the typical way that network interface cards and ATM switch interface modules operate. In addition to the data flow connections, the FPGA control logic is connected to a status port on the framing chips that allows the control logic to configure the framing chips and obtain statistics about data transmission on the link.

4.3 ATM Line Interface

The physical transmission of data to and from the FIP is performed by HFBR-5205 optical transceiver pairs. All communication and connectivity on the FIP to and from these transceiver pairs is achieved through the SONET/SDH framing chips. A separate transceiver pair is used for each of the three communication ports on the card.

4.4 FSI RAM

The FSI lookup table is implemented in a 1M by 32-bit static RAM. Each memory location in the RAM is used to store a single subfield entry in the FSI table. The result of the FSI hash function is used as the upper seventeen bits of the 20-bit address used to access the FSI memory. The lower three bits of the RAM address are used to determine which subfield in a given entry is accessed. The implementation of the RAM control circuitry allows it to read or write a single entry of the RAM on every tick of the 25 MHz system clock.

4.5 Message Buffer RAMs

The FIP encounters situations where it is desirable to simultaneously transmit two different data streams on the same link. This situation occurs when an information flow is traveling through the FIP over the inline stream at the same time that the FCP attempts to insert cells into the inline stream. To handle these times of contention, message buffer RAMs are inserted into the system that can temporarily store cells until the inline port is prepared to
transmit them. Both of the message buffer RAMs can hold over sixty-four thousand 16-bit entries. As with the FSI RAM, a new data word can be written or read on every tick of the system clock.

4.6 Clock Network

The synchronous network is based on two fundamental clock frequencies. The first clock, generated by a 19.88 MHz oscillator, is used as a reference clock for the line side of the SONET framing chips. The primary clock frequency for the digital logic on the FIP is a 25 MHz clock with a 50% duty cycle that is derived from a 50 MHz oscillator. This clock controls all of the synchronous logic on the FPGA controller as well as the internal control logic on the processor side of the SONET framing chips. The framing chips have internal buffers that synchronize data transfers between their internal control logic that operates at 25 MHz and the ATM line interfaces that operate at 19.88 MHz.

5.0 FIP Performance

A key element in the development of an ATM firewall is the ability to monitor and react to ATM cells at real-time rates. At the 155 megabit per second transmission rate of the ATM link, a 53-byte ATM cell can arrive on a port every 2.74 microseconds. In order to process the cells in real-time, the FIP must perform all of its required processing in less than this amount of time.

For every cell that enters the FIP on a given inline port, a maximum of ten cycles of the FIP logic clock is required to access the FSI table and determine where the cell should be routed. An additional fifty-six cycles is then required to store the data in the message buffer RAM. Thus, a total of sixty-six clock cycles are required to examine and store each incoming cell. Since the system clock rate is 25 MHz, the total amount of time required for this operation is 2.64 microseconds.

The FSI table is not accessed by a port during the 56 clock cycles in which it is storing the cell. This makes it possible for the other ports to access the table during this time. Since the other inline port requires a maximum of ten clock cycles for it to determine what to do with its data, a total of forty-six out of every sixty-six clock cycles is available for the command port to modify the contents of the FSI table when instructed by the FCP.

The message buffer RAMs allow data to be both read and written at an average rate of one byte per clock tick. Thus, the processes for storing and retrieving data from the message buffers can be pipelined. During the fifty-six clock ticks where data is being stored in the message buffer, a previously stored message can be extracted from the buffer and transmitted on an outgoing link. Since there is a separate buffer for each of the outgoing links, data from the two inline ports are not in contention and do not affect each other's performance. This data flow structure allows the FIP to process both of the inline data ports at the maximum line speed.

6.0 Conclusions

This paper describes a new method for achieving low cost protection for a broadband network environment. Low cost is essential so that protection is available to a larger class of users, rather than only to large corporations. The method described in here is easy to manufacture and may also have the potential to be folded into the design of ATM switches. The level of protection possible with this approach is very high due to the extreme degree of flexibility associated with the design. Traditional methods may be applied to IP class traffic. The security device proposed here, however, also enables the selection of non-IP traffic such as voice and video. Future work will examine the use of this hybrid inline and control processing to perform additional security features and second-level trend analysis.

7.0 References

1. P.W. Dowd, et al., “A Method for High Performance Low Cost Network Security for IP and Non-IP Traffic in an ATM Environment,” Technical Report, United States Department of Defense, 1997.

2. “Private Network-Network Interface Specification Version 1.0,” ATM Forum, af-pnni0055.000, March 1996.

3. ATM Forum Technical Committee, Security Working Group, ATM Forum BTD-SECURITY-01.021, Apr. 1997.

4. P.W. Dowd, T.M. Carrozzi, F.A. Pellegrino, A. Chen, “Native ATM Application Programmer Interface Test-bed for Cluster-based Computing,” Proc. IEEE International Parallel Processing Symposium (IPPS'96), Apr. 1996.

5. J. Brian Lyles and Christopher L. Schuba, “A Reference Model for Firewall Technology and its Implications for Connection Signaling”, Proceedings Open Signaling Workshop, Columbia University, Oct. 1996.

6. D. Winkelstien, “Encrypting ATM Firewall”, Secant Network Technologies, Sept. 1996.

7. J. Hughes, “A High Speed Firewall Architecture for ATM/OC-3c”, INTEROP, 1996.

8. B.J. Kowalski, “ATLAS (R) Policy Cache(TM) Architecture”, Technical Report, Research and Advanced Development Group, Network Systems Corporation, Nov. 1996.

9. “Simple Protocol for ATM Network Signaling,” FORE Systems, Inc., 1992.

10. The ATM Forum, “User-Network Interface Specification, Version 3.0,” Prentice-Hall, 1993.

11. Lucent Technologies, “Field Programmable Gate Array Data Book,” October, 1996.