An Embedded Firewall Based on Network Processor

Quan Huang, Shengke Qiu
Research Institute of Information Technology (RIIT), Tsinghua University
Beijing 100084, China
{huangq03, qsk03}@mails.tsinghua.edu.cn

Shicun Qin, Cheng Cao
Beijing Extech Company,
Beijing 100084, China
{qinsc, caoc}@extech.com.cn

Abstract

A design of firewall system using Intel IXP425 network processor (NP) based embedded platform is proposed in this paper. Intel IXP425 Network Processor is a highly integrated processor. It has three network processor engines (NPEs), which offload most of the computational intensive network data operations, and one of its NPE is designed with hardware support for some of the widely used encryption algorithms and hash functions. These features make the embedded platform powerful enough to handle the network data to provide security protection. And the testing results of this embedded firewall show that the system is competent for Small Office/Home Office (SOHO) users, even for Small to Medium Enterprise (SME) users.

1. Introduction

With the popularization of personal computer, computers have spread all over the world. We use them in companies, in universities, even in families. And most of them are connected by the so called internet.

Unfortunately, internet is not that good as we expected. Attacks happen on the internet every day. The security issues come to every computer user. Some companies may buy firewalls to protect their LAN form outside malicious access. But, the traditional firewalls are too expensive for most of the other users, especially for family users. With the progress of embedded technology, kinds of low-cost and powerful embedded systems become competent for this job. And we choose Intel IXP425 network processor [1] based platform to complete the embedded firewall.

The embedded firewall achieves nearly all the functions of a traditional firewall, for example IPSec-VPN [2] and PPTP connections. And the firewall even features Graphic User Interface (GUI). So, for a firewall administrator, the embedded firewall can be used exactly the same as the traditional firewalls they are used to.

The remainder of this paper is organized as follows. Section 2 gives a brief introduction to the hardware architecture, including the overview of Intel IXP425 network processor. Section 3 details the design of the software and its logical architecture. Section 4 presents the performance testing results of the embedded firewall system. And finally, section 5 concludes the paper and discusses the future work based on the IXP425 based platform.

2. Hardware Architecture

Intel IXP425 network processor is a highly integrated, versatile single-chip processor. As illustrated in figure 11, it contains an XScale core and three NPEs (Network Processor Engines), which can run their instruction steams in parallel. The 533 MHz XScale core is compatible with ARM V5T Thumb instructions set and V5E DSP extensions. The three NPEs are designed to take over many

1 We only provide the parts we concern in figure 1 for the simplicity consideration.

computationally intensive data plane operations for XScale core (only NPE A and NPE B are used in the system, and Wan/Voice NPE will be used for future work), such as packet filtering and checksum computation. And two integrated 10/100-Mbps MAC with industry-standard Media Independent Interface (MII) are embedded in NPE A and NPE B. More importantly, NPE B is designed with the hardware acceleration of encryption (DES, 3DES, and AES) and hashing (MD5 and SHA1) functions, which brings great improvement to the performance of encryption and authentication related functions, such as IPSec-VPN. [3-6]

IXP425 also provides many other conveniences for the platform design, such as USB and UART interfaces [4]. That is the reason why we choose it for the platform. As in the figure above, in the hardware platform, NPE A and NPE B's MII interfaces are used for Ethernet access. A UART interface is used as console interface. And the expansion bus controller and SDRAM controller are used for 16M flash access and 64M SDRAM access respectively.

3. Software Architecture

In this section, the firewall software design architecture and logical function architecture are presented. The design architecture is the architecture we used in software developing, and the logical function-architecture describes the relations between the functions the system provides for firewall administrators.

3.1 Software Design Architecture

In the software design, we use a layered design philosophy based on the use input processing procedure. As illustrated in figure 2, the system can be divided into six layers. The arrays in the figure represent the user input flow directions between layers.

The first layer is User Interface Layer. It provides two kinds of interfaces for user. One is web-based GUI, using CGI (Common Gateway Interface) to handle user input operations from web pages, and the other is CLI (Command Line Interface). They provide user nearly the same function entries to manipulate the firewall, such as address configuration and policy configuration.

The second layer is Control Layer. It accepts user inputs form the first layer, most of which are system configurations (e.g., packet filtering rules), parses the inputs for validation, and then passes them to the next layer if they pass the validation or returns corresponding error numbers to the first layer according to the error types. And from the figure we can find that, the output of this layer has two paths, one to the Database Control Layer, and the other to the Execution Layer. The path to Database Control Layer is used only when we want to change the database, namely save current system configuration in database. The other path is used when we make any change to the system configuration, and the change takes effect immediately.

The third layer is Database Control Layer. This layer's functions can access the database directly, and add, delete, modify or read data form the database. It provides the Control Layer with APIs for handling database (e.g., query for a certain policy, modify an IP address, etc). In other words, the Control Layer will use the Database Control Layer APIs when it does database related operations.

The fourth layer is Database Layer. It stores the configuration data (e.g., network definition and packet filtering rules) for the system. And the saved data will be used as the default configuration when system boots up until new changes are made to the system and saved to the database.

The fifth layer is Execution Layer. It gets configurations from database or directly from Control Layer, and does corresponding works to perform the system functions (e.g., setups IPSec VPN, performs packet filtering according to the filter rules, etc).

The sixth layer is the kernel. We use standard linux kernel here. It servers the execution engines the same as it does for any other applications running in user space. After this procedure, some necessary results (e.g., the IPSec-VPN status and logs) will be returned to the user interface when user requires.

Except for the divided layers, we could also see from figure 2 that each layer is designed with its own APIs which can be classified into two kinds. One is a series of sub-APIs, the other is common APIs. Each sub-API

implements a certain function for the upper layer, and the common APIs provide some commonly used functions for the sub-function APIs. Through these divisions, the architecture provides the system a fine granularity while keeping system integrity. This feature makes the system very flexible. So each layer and layer's sub-function can be easily replaced with a more suitable one. Thus, it provides a convenient way for software update and substitution (e.g., we could use a more light-weighted database to replace the currently used one). Besides, it also provides a way of efficient memory usage. Because in most of the situations, not all the functions are used, we could store the files in zip format and only unzip them into the file system (in SDRAM) when needed.

3.2 Logical Function Architecture

Based on the relations between functions the system provides for user manipulating of the system, the system could also be divided into three layers. As illustrated in figure 3, the system is described as Definition Layer, Configuration Layer, and Function Layer. And the arrays between layers are the supporting relations between layers. More precisely, the array header points to the layer supported by the array-tailed layer.

The Definition Layer works on TCP layer and the layers below TCP. It gives some necessary definitions for Configuration Layer. Take Network Definition which works in IP layer for example, it defines different sub-networks or just a single IP address as independent network elements, and the network elements will be used for configuration in the Configuration Layer. Another example in TCP layer, Protocol Definition defines port numbers for different protocols (e.g., ftp, http, etc).

The Configuration Layer mainly works on the same layers as Definition Layer does. It uses the definitions from Definition Layer to create configuration files for the services in the upper layer, namely Function Layer. Take IPSec Configuration for example, it uses the networks and IPSec relevant definitions defined in the Definition Layer and creates IPSec configuration files, namely "ipsec.conf" and "ipsec.secrets".

The Function Layer is also the service layer. The services in this layer use the configuration files created by the Configuration Layer to provide services. Take IPSec-VPN service for example, it read the files "ipsec.conf" and "ipsec.secrets", and setup IPSec tunnels accordingly.

4. Performance Evaluation

The testing platform consists of a smartbits-600 (SMB600), two IXP425 based firewall, and a PC for firewall system configuration. The testing we performed focuses on three parts, namely packet forwarding, IPSec-VPN performance, and connection capability. In the packet forwarding testing, only one firewall is used. As illustrated in figure 4, SMB600 connects to the two Ethernet ports extended from NPEA and NPEB. The IPSec and connection testing use two embedded firewalls. The two firewalls setup an IPSec tunnel, and packets from SMB600 go through the tunnel to perform the testing as the arrays indicate.

In connection capability testing, the new connections setup rate can be 1400 transaction per second at first, and 300 sessions can be maintained. Then the setup rate fluctuates periodically, and finally comes to 830 new connections per second and maintains.

The IPSec tunnel mode ESP performance and packet forwarding performance are listed in table 1 and table 2 respectively.

5. Conclusion and Future Work

In this paper, we proposed the design of an embedded firewall system based on Intel IXP425 Network Processor based platform. The performance testing result shows that the system is powerful enough even for the SME users. And for the low-cost feature of the embedded firewall, the firewall may go into the family life in the company of home PCs.

But during the testing, we found that the system was not good at handling CGI requests. There is about one second delay. So we are going to do some optimization works in the future. Besides, IXP425 is such a versatile processor that it supports many other usages (e.g., audio codes), and we are going to

implement a SIP (Session Initialization Protocol) based secure VoIP system on it.

References

[1] Niraj Shah, "Understanding Network Processors", Version 1.0. September 4, 2001.

[2] William Stallings, "Network Security Essentials (Second Edition)", Prentice Hall PTR, 2003.

[3] Intel Corporation, "Intel IXP4XX Product Line Programmer's Guide (1.1)", February 2003.

[4] Intel Corporation, "Intel IXP425 Network Processor Family Technical Overview", 2002.

[5] Intel Corporation, "Intel IXP4XX Product Line and IXC1100 Control Plane Processors: Using the Intel LXT973 Ethernet Transceiver", July 2003.

[6] Intel Corporation, "Intel IXP42X Product Line of Network Processors and IXC1100 Control Plane Processor Hardware Design Guide", June 2004.

Appendix A

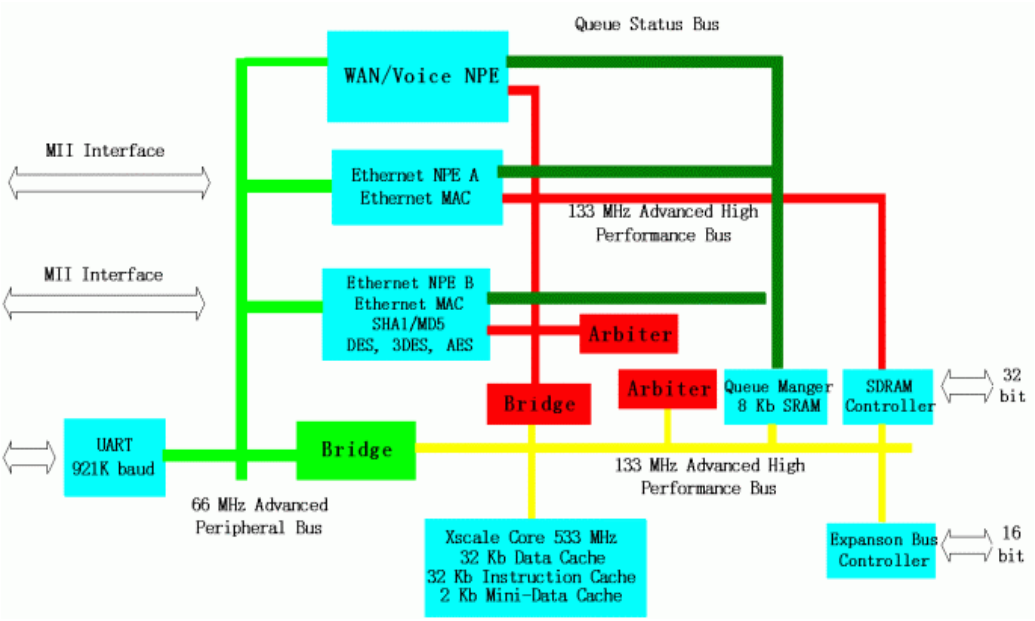

Figure 1. Intel IXP425 network processor architecture

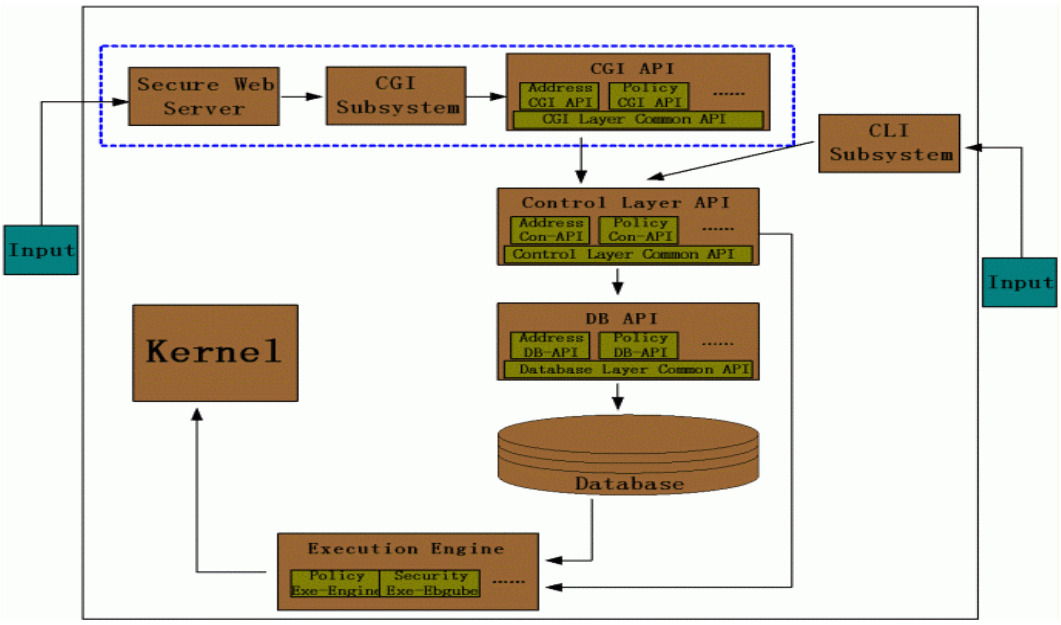

Figure 2. Software architecture

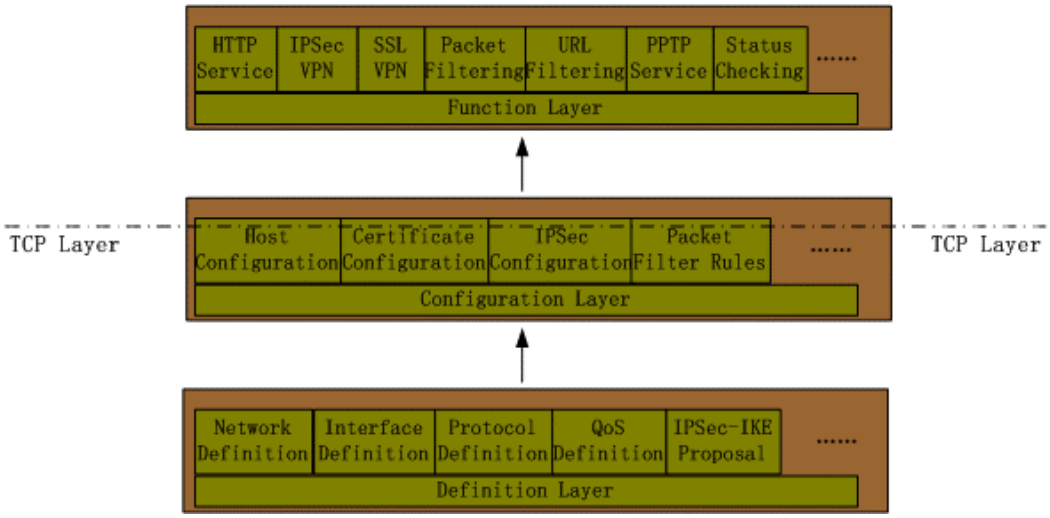

Figure 3. Function relation layers

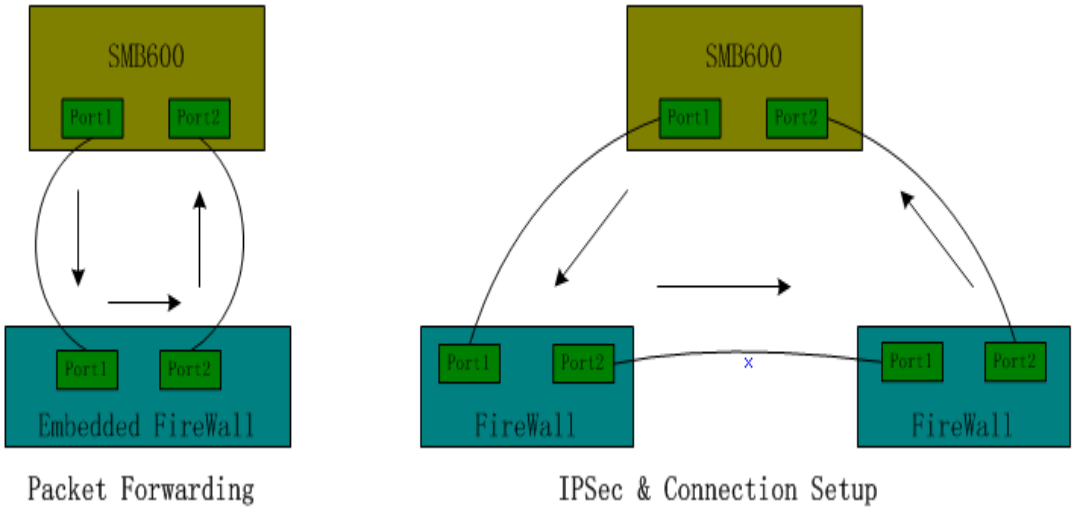

Figure 4. Performance test platform

Appendix B

Table 1. IPSec-VPN testing results[2]

Algorithm\Frame Size (Byte)	64	128	256	512	1024	1400
3DES-HMAC_MD5 (Mb)	5.1	9.2	18.1	33.3	57.1	69.0
AES128-HMAC_MD5 (Mb)	4.8	9.4	18.8	34.0	56.7	70.2
AES256-HMAC_MD5 (Mb)	4.9	9.9	18.8	34.1	56.7	70.2
3DES-HMAC_SHA1 (Mb)	4.9	9.5	19.0	34.0	58.0	69.8
AES128-HMAC_SHA1 (Mb)	5.0	9.9	19.0	33.8	56.8	69.9
AES256-HMAC_SHA1 (Mb)	5.0	10.0	18.7	34.0	57.1	69.8

Table 2. Packet forwarding testing results

Frame Size (Byte)	64	128	256	512	1024	1400
Packet Forwarding (Mb)	48	82	90	100	100	100

2 The testing results are single directional.