Performance Oriented Management System for Reconfigurable Network Appliances

Hiroki Matsutani†, Ryuji Wakikawa‡, Koshiro Mitsuya‡ and Jun Murai†

Faculty of Environmental Information, Keio University†
Graduate School of Media and Governance, Keio University‡
5322 Endo, Fujisawa-shi, Kanagawa 252-8520, Japan
Tel +81-466-49-1100, Fax +81-466-49-1395, E-mail nn@sfc.wide.ad.jp

ABSTRACT

In this research, a performance oriented management system for network appliances is proposed to optimize the utilization of its hardware resources according to its environment at any time. This system provides two mechanisms: one is to change its hardware feature dynamically and another is to trigger the hardware change owing to adjust the hardware feature with optimum condition. A trigger is occurred at when the function that has been called most frequently is replaced by another one in same system. After the trigger is occurred, suitable hardware configuration is chosen among many candidates and then downloaded from the Internet to install it to the target. We implemented an IPsec hardware/software co-system supporting multiple cryptographic algorithms. This IPsec system successfully reconfigured its hardware with the cryptographic algorithm that has been called most frequently according to the applications. Consequently, we showed that the limited hardware resource is successfully assigned proper function with less penalty by this system.

Keywords: *network appliance, networked sensor, reconfigurable hardware, IPsec/IPv6 application*

1 INTRODUCTION

Due to the progress and wide diffusion of network technology, sensors and appliances with network interfaces stand in the center of the spotlight. These network appliances are considered for use with nursing and health care and are considered to play an important role in our living in the near future.

In recent years, functional requirements toward a network appliance are becoming more sophisticated. For example, appliances which use motion pictures and audio data for user interface are increasing, and consideration for a secure communication and insuring privacy are becoming indispensable. Since these functions require high CPU load for processing, it is reasonable to add proper hardware accelerator onto network appliances with limited resources. However, because many of these devices are embedded in our living, it is often difficult to add or update both new and existing features according to user's usage at run-time. This problem may end up limiting the application range of these devices which contain application-specific hardware.

In this paper, we describe the existing technologies of network appliances and point out the present problems at Section 2. In order to solve the problems, a performance oriented management system is proposed at Section 3. We explain an implementation of IPsec[1] stack as an example of our system at Section 4. This IPsec stack is autonomously configured its hardware with the cryptographic algorithm that has been called most frequently according to the applications. Finally, we show the evaluation of this system at Section 5 and give conclusion at Section 6.

2 NETWORK APPLIANCES

We primarily target network appliances supporting the Internet connectivity. The network appliances are used for specific tasks, but not for a general purpose. These are based on Low Cost Network Appliances[2] discussed in the Internet Engineering Task Force (IETF).

2.1 Hardware updating technologies

Network appliances naturally have several functional capability and run in long life time. However it is impossible to predict which functions are required later or to provide all possible functional capability. As a result, it is common to update or add new features by software.

Many recent works have focused on the reconfigurable computing[3]. The reconfigurable hardware represented by Field Programmable Gate Array (FPGA) can be used as commodity parts in consumer products. A bit-stream called "configuration data" is written into each devices to build an arbitrary logic.

Dynamic reconfiguration

Many reconfigurable hardware can be reprogrammed, so the run-time reconfiguration is also possible. The run-time or on-demand reconfiguration systems successfully perform the function that requires larger hardware resource than the actual one they could support at one time.

The Adaptive Cryptographic Engine[4] is an FPGA-based architecture and has a capability of adapting any cryptographic algorithms by its reconfiguration. The configuration data of various cryptographic algorithms is compressed and stored in memory as "cryptographic library".

Configuration delivery framework

The reprogramming is needed to update or add new features for the network appliances after their manufacture. It is expensive to do so for each device, because many network appliances will be embedded in our living.

A system design methodology that enables updating and adding new features to the target hardware across the Internet represented by Internet Reconfigurable Logic (IRL)[5] has already come into practical use. IRL provides two way, Push and Pull, to deliver new configuration data to the target. In the Push, control process in a network server sends new configuration data to the targets. In the Pull, the targets contact a network server and download new configuration data if it is available.

2.2 Configuration data size problem

Reconfigurable technologies mentioned at Section 2.1 are reasonable solutions to accelerate any functions on network appliances under the resource limitation.

However, memory space to store sets of configuration data requires Mbit order area per one data and it is crucial for resource limited devices. To solve this problem, data compression has been applied, but only decreasing data size is possible.

Actually, it is difficult to predict the end-user environments in sophisticated network appliances. Suppose, for instance, there is a network appliance which supports audio data decoding and secure communication such as IPsec. In one environment, sets of hardware acceleration of various cryptographic algorithm might be required, but in the other environment, sets of hardware acceleration of audio decoding might be more important than any cryptographic algorithms.

Therefore, supporting several functional capability causes either the waste of memory space to store configuration data or the limitation of the application range.

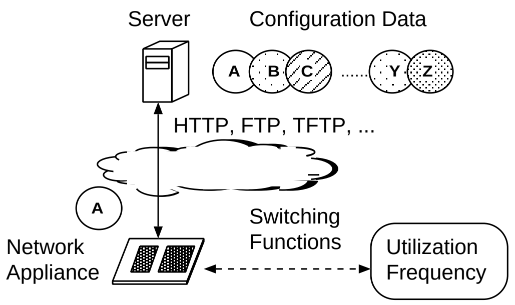

Figure 1: Overview of the system.

3 PERFORMANCE ORIENTED MANAGEMENT SYSTEM

In order to solve the problems mentioned at Section 2.2, network appliances are required to update their hardware features autonomously adopted to their environment under the resource limitation. To achieve this objective, a performance oriented management system which has following features is proposed in this research.

As a concept of this system, high loaded functions are assigned to the hardware and the rests of functions are executed as software. During a reprogramming of the hardware, all task is executed as software. This is reasonable to accelerate specific tasks with less penalty of reconfigurations.

Configuration data management at network server

The more variation in these configuration data provides the more variation in available hardware features and network appliances are able to be optimized to more suitable composition to their environment. However, network appliances cannot keep large amount of configuration data due to the limitation of available memory space. To solve this resource limitation, sets of configuration data are managed on the Internet and are downloaded on-demand to network appliances.

Figure 1 shows the overview of this system. Server manages many configuration data and network appliance is able to download it by using arbitrary file transfer protocols as a client. Network appliances should cache frequently used data into their local memory to eliminate download time over the Internet.

Selecting suitable configuration data

Each processing which has capability to be executed as both hardware and software is defined as "task" in this system. As illustrated in Figure 1, the suitable configuration data is chosen based on the utilization frequency that is the number of times each task is called during a timeslice. To

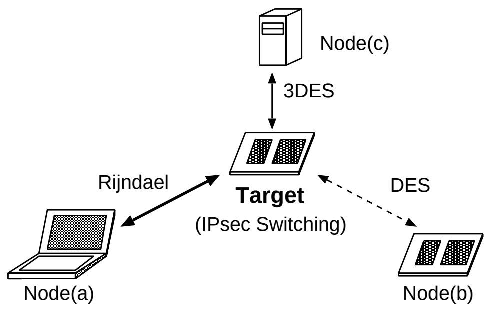

Figure 2: Environment for IPsec Switching.

Figure 3: Appearance of prototype board.

calculate utilization frequency of each task, they have the *importance* counter. Whenever each task is called, its *importance* is added by the *weight* value which is peculiar to each task. The most important task is periodically searched at fixed *interval* and then the most important task is created on the hardware by reconfiguring it. If the most important task already had been created on the hardware, the reprogramming is not performed.

When the task that has been called most frequently is replaced by another one, new configuration data suitable for the new environment is downloaded from the servers and then applied to update the hardware as shown in Figure 1.

Automatic adjustment for switching frequency

It is difficult to keep running continuously during reprogramming the hardware, because time for several hundreds of milliseconds is required to reprogram for the general FPGAs. Therefore, it is necessary to hold down the frequency of reprogramming and then this system performs following algorithms.

The number of times to change its hardware per a day (Changes Per a Day: *cpd*) is defined as hardware switching frequency. In order to hold down the *cpd*, the value of *interval* for searching the most important task is controlled. The short *interval* makes *cpd* increase, because increasing of the number of times to search the most important task requires more frequently reconfiguration. On the other hand, the long *interval* requires less reconfiguration, but it takes the long time for adapting new environment when a change in the environment arises. The maximum value of *cpd* (*max_cpd*) is defined in each system. If the value of *cpd* in a system currently running exceeds *max_cpd*, the *interval* is doubled to decrease the *cpd*.

In order to hold down the hardware switching with little or no effect, the several percent of *allowance* is applied to ignore the trivial difference among their importance.

Security issue

Because configuration data is downloaded over the Internet, network appliances are exposed to threats such as receiving malicious configuration data which physically destroys devices[6]. To prevent these attacks, secure protocols which provide integrity and data origin authentication are required (e.g., IPsec AH).

4 AN EXAMPLE IMPLEMENTATION — IPSEC SWITCHING

IPsec supports multiple cryptographic algorithms as illustrated at Figure 2, so we chose IPsec as an example that performs some heavy tasks simultaneously. In order to show the effectiveness of this system, IPsec hardware/software co-system was implemented. This IPsec stack is autonomously configured its hardware with the cryptographic algorithm that has been called most frequently according to the applications.

Table 1 shows the detail of the hardware and software spec of IPsec Switching. This system consists of the “Control Part” and the “Reconfigurable Part”. The utilization frequency of each cryptographic algorithm performed is monitored

Table 1: Hardware and Software spec.

Control Part	
CPU	16bit microcontroller @ 20MHz
I/F	10Base-T Ethernet
RTOS	micro ITRON4.0 compatible
	Scheduler for reconfiguration
net	IPv6/IPsec, TFTP
	DES(SW),3DES(SW),Rijndael(SW)
Reconfigurable Part	
FPGA	Xilinx Spartan-IIe (300KGates)
config	DES(HW),3DES(HW),Rijndael(HW)

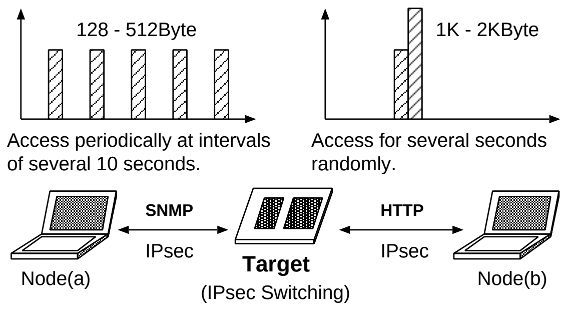
128 - 512Byte
Access periodically at intervals of several 10 seconds.
1K - 2KByte
Access for several seconds randomly.
SNMP
IPsec
HTTP
IPsec
Node(a)
Target
(IPsec Switching)
Node(b)

Figure 4: Assumed simulation environment.

at the Control part. The cryptographic algorithm that has been called most frequently is created as a hardware on the Reconfigurable part at runtime. The suitable configuration data is downloaded from the Internet using Trivial File Transfer Protocol (TFTP) over IPv6. Figure 3 shows appearance of a prototype board.

5 EVALUATION

The objective in this research is to optimize the utilization of hardware resources according to their environment at any time. To achieve this objective, conformance test was conducted to show the effectiveness of two features: one is to control the hardware switching frequency to requisite minimum and another is to adjust its hardware feature to the most optimized one.

The conformance test is required to show that our system is always effective on the assumed environment of network appliances. The effectiveness of IPsec Switching, an example application, would depend on the traffic pattern such as volume or frequency in the real environments. It is difficult to investigate with various traffic patterns in real environments. In order to evaluate this system in full detail, simulation program was used.

Figure 4 shows the assumed environment for the simulation. There are two kinds of nodes. Some nodes use SNMP to access the target and the other nodes use HTTP instead. Those packets are encrypted but each node uses different cryptographic algorithm.

The traffic patterns used by the simulation program is the following three types. (1)SNMP type is a pattern generated by SNMP when monitoring sensors periodically. Small packets, 128 - 512Byte data, are generated periodically at intervals of several 10 seconds. (2)HTTP type is a pattern generated by HTTP when one operates network appliances by using a Web browser. Large packets, 1K - 2KByte data, are generated for several seconds randomly. (3)Hybrid type is a pattern that is mixed SNMP type and HTTP type.

In the simulation, the following two values are investigated while increasing the value of *interval* from 10 to 86,400 seconds.

- *cpd* — number of times the hardware switching arises per a day
- *hit*[%] — utilization ratio of function that is currently created on the hardware (not software)

The simulation was performed under the following conditions. Only one algorithm can be performed on the hardware at the same time and the rests of algorithms are run as software. The block size of each algorithm is 64bits. The *weight* for importance of each algorithm is 1. The *allowance* to hold down the changes with little effect is 2.5%.

For instance, when 16Byte data is encrypted, then its *importance* increases 2, because the algorithm whose block size is 64bits is called twice.

5.1 Control of the switching frequency

Three type traffic patterns mentioned above were tested on the simulation to investigate the value of *cpd* and *hit*.

(1)Case of 3 SNMP Traffics

In this case, set of 3 SNMP traffics that uses different algorithm each other was simulated. Figure 5 shows that the value of *cpd* is set to about 0 if the *interval* is made into 60 seconds or more. This system is effective against the regular traffic pattern such as SNMP type, because the candidate function to be created on the hardware is chosen based on the utilization frequency information already calculated.

(2)Case of 3 HTTP Traffics

In this case, set of 3 HTTP traffics that uses different algorithm each other was simulated. Figure 6 shows that the value of *cpd* is set to about 6 if the *interval* is made into 3,000 seconds. Long interval is required to stabilize the hardware switching in case of HTTP type compared with SNMP type.

Nevertheless the traffic regularity such as "algorithm(x) and (y) are used at a rate of 7:3" is found, if it is seen in long enough time. Therefore, the proper *interval* which makes *cpd* below *max_cpd* is automatically adjusted. Thanks to the automatic adjustment of the *interval*, hardware switching frequency is controlled to requisite minimum.

(3)Case of 3 Hybrid Traffics

The result of this case was close to HTTP type.

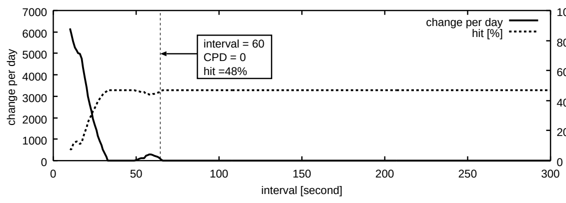

Figure 5: Case of 3 SNMP traffics.

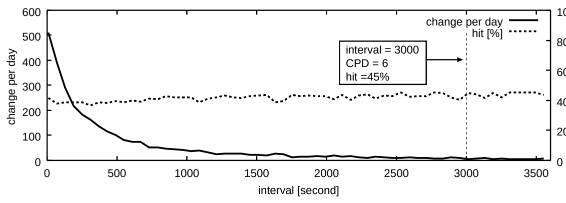

Figure 6: Case of 3 HTTP traffics.

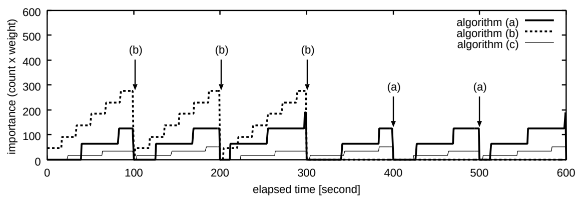

Figure 7: Importance transition of 3 algorithms.

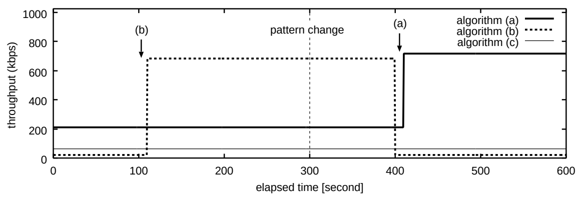

Figure 8: Throughput transition of 3 algorithms.

5.2 Adaptation for new environment

In order to verify the automatic adjustment for each network appliances to the most optimized hardware configuration, traffic of the algorithm that has been called most frequently is stopped on purpose. Then this simulation shows another algorithm is created on the hardware correctly.

Figure 7 shows the *importance* transition of 3 algorithms. The traffic of algorithm(b) that has been called most frequently is stopped at time of 300 seconds on purpose. Figure 8 shows the cryptographic throughput transition of 3 algorithms. More than 100 seconds after stopping the algorithm(b), algorithm(a) is created on the hardware and its throughput is increasing immediately.

In case of this simulation, algorithm(a), (b) and (c) are assumed as Rijndael, 3DES and DES respectively. These constants such as the hardware/software throughput of each algorithm and the reconfiguration overheads are based on the actual value measured on Table 1.

Consequently, this system has capability of detecting the environmental changes and then applying suitable hardware feature for new environment.

6 CONCLUSION

The objective in this research is to optimize the utilization of hardware resources according to the environment for each network appliances.

To show the effectiveness of this system, IPsec stack which is configured its hardware with the cryptographic algorithm that has been called most frequently according to the applications was implemented. As the evaluation, conformance test verified that it is capable of controlling the hardware switching frequency to requisite minimum and its limited hardware resource is successfully assigned proper function by this system.

From this research, applying this system to the sophisticated network appliances is effective to adapt for their environment at any time.

ACKNOWLEDGMENT

We thank members of Murai lab. and Amano lab. at Keio University for effective advice.

REFERENCES

[1] S. Kent and R. Atkinson, Security Architecture for the Internet Protocol, 1998. RFC 2401.

[2] N. Okabe, S. Sakane, A. Inoue, M. Ishiyama, and H. Esaki, Host Requirements of IPv6 for Low Cost Network Appliances, 2002. Work in Progress, draft-okabe-ipv6-lcna-minreq-02.txt.

[3] F. Vahid, "The Softening of Hardware," IEEE Computer, vol.36, no.4, pp.27–34, 2003.

[4] A. Dandalis, V.K. Prasanna, and J.D.P. Rolim, "An Adaptive Cryptographic Engine for IPSec Architectures," Proc. 2000 IEEE Symposium on Field-Programmable Custom Computing Machines, pp.132–144, 2000.

[5] Xilinx, Architecting Systems for Upgradability with IRL (Internet Reconfigurable Logic), 2001. XAPP412 (v1.0).

[6] I. Hadzic, S. Udani, and J.M. Smith, "FPGA Viruses," Proc. 9th International Workshop on Field-Programmable Logic and Applications, pp.291–300, 1999.