http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 10 (3): 626-638, 2011
ISSN 1812-5638 / DOI: 10.3923/itj.2011.626.638

Research on Internal Flow Control Mechanism of For CES Routers

Bin Zhuge, Cheng Yu, Kang-ping Liu and Wei-ming Wang
College of Information and Electronic Engineering, Zhejiang Gongshang University,
Hangzhou, 310018, People's Republic of China

Abstract: To improve the communication efficiency and security between Control Elements (CEs) and Forwarding Elements (FEs) of For CES routers, this study researches on internal flow control mechanism. Firstly, an evaluation model based on communication of Transport Mapping Layer (TML) is proposed, which is used to improve service performance of TML. Secondly, in order to prevent redirect channel interference with control channel, this study proposes a bandwidth allocation algorithm called Dynamic Probabilistic Priority Based on Rate and Buffer (DPPBRB). Lastly, to avoid the potential congestion on control channel between one CE and multiple FEs, this study puts forward an internal flow control mechanism. These internal flow control methods that this study proposed effectively prevents the DoS attack from redirect messages and improves reliability of For CES routers. Simulation and experiment results show the feasibility and effectiveness of these methods.

Key words: For CES, router, flow control, communication model, TML

INTRODUCTION

Forwarding and Control Element Separation (For CES), which is a working group under Routing Area of IETF, has made the most prominent achievements on open programmable networks. By separating Control Elements (CEs) from Forwarding Elements (FEs), the For CES router (For TER) greatly meet the demands of next generation network equipments (Takourout *et al*., 2006; Fu, 2008).

Recently, four core protocols of For CES have become standards RFC, which are Forwarding and Control Element Separation (For CES) Protocol Specification (Doria *et al*., 2010), SCTP-based Transport Mapping Layer (TML) for the Forwarding and Control Element (Salim and Ogawa, 2010) Forwarding and Control Element Separation (For CES) Forwarding Element Model (Halpern and Salim, 2010) and Forwarding and Control Element Separation (For CES) MIB (Haas, 2010).

The concept of For CES was investigated quite intensively in recent years. Haleplidis *et al*. (2009) propose Web Service-based CE architecture, in which Web Service interfaces is mapped to For CES messages for dynamic service deployment. Chrysoulas *et al*. (2006) presented modular node architecture and specify the description interface of nodes. Their aim is to make new services be easily added by inserting modules that have the appropriate functionality. In the For CES based distributed router, Hidell *et al*. (2005) presented measurements on the distribution of large routing tables in an experimental platform consisting of one CE and up to 16 Fes. Fu *et al*. (2009) specified a centralized routing scheme and a design of the centralized CE (Fu *et al*., 2009). The results showed that the centralized control scheme can provide faster routing convergence than link-state routing protocols. Yoon *et al*. (2006) designed and implemented the management functions of DiffServ-over-MPLS transit network in For CES architecture for QoS guaranteed broadband realtime multimedia service provisioning. Louati *et al*. (2008) presented the design, implementation and evaluation of an extensible, scalable and distributed heterogeneous cluster based For CES architecture with software component technologies to meet the requirements of the next generation software routers (Louati *et al*., 2008). In order to ensure secure communication of highly dynamic ad hoc radio networks, Hamigk *et al*. (2009) proposed an efficient solution based on For CES architecture.

With the growing maturity of research on For CES, the security of control channel has been taken into account. Increase on the number of FEs and a variety of network attacks will directly lead to control channel congestion of For TER, the loss of control message, or even system collapse. As key participants in the for CES work and authors of the For CES protocol, we set the motivation of the For TER implementation as the

Corresponding Author: Bin Zhuge, College of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou, 310018, People's Republic of China

evaluation of the For CES protocol (Wang *et al*., 2008). Therefore, designing an effective internal flow control mechanism is needed urgently.

Next Generation Network will become a high speed packet switched network which will provide various services to support a wide range of network applications (Struck and Weingarten, 2004; Wu and Xu, 2006). Due to network traffic with the feature of burst and fluctuation, even well designed, congestion occurs inevitably. Design of an effective flow control scheme is critical to the enhancement of the network performance. Long congestion will bring about network management difficulties and risks of failure to perform its functions, which result in complaints increasing and quality of service declining.

Regarding the urgent to reduce the seriousness and duration of the congestion, an effective internal flow control mechanism of For TER is a choice. Flow control dynamically adjusts input flow according to network payloads and efficiently allocates bandwidth to meet different demands. Thus, not only network congestion can be avoided, but also the utilization of network resources can be improved (Wei *et al*., 2006). In order to improve the Quality of Service and network throughput, select a suitable bandwidth allocation is particularly important (Gu and Luo, 2006). Bandwidth allocation algorithm includes queue management algorithm and scheduling algorithm. For queue management algorithm, Active Queue Management is the most classic. For scheduling algorithm, the Probabilistic Priority (Jiang *et al*., 2002), Weighted Fair Queuing (Parekh and Gallager, 1993), Weighted Round Robin (Katevenis *et al*., 1991), Deficit Round-Robin (Shreedhar and Varghese, 1996) and Core-Stateless Fair Queuing (Stoica *et al*., 1998) are the five most widely adopted disciplines. Under these disciplines, routers can allocate bandwidth fairly with an assigned priority.

One well-known problem in the communication model is that server resources often face great challenges. For example, in heavy-load situation, request from clients may be queued on the server and waiting for being processed. If the server resources are inadequate for the requirements of the arrival requests, the request will experience long queuing and processing delay. Therefore, the efficiency of model is mainly depending on a high-performance server.

To date, a lot of methods for server improvement have been proposed in the research literature or implemented in vendors' products. Among which, improvements on operating system performance come from reducing data movement (Pai *et al*., 1999), improve stability under overload (Mogul and Ramakrishnan, 1997) and improve server control mechanisms (Hu *et al*., 1997). Application designers have also faced this performance problem when thinking about how to make more efficient use of existing operating system. For example, recent servers use single-process model, which reduces context switching costs, while the early network servers created a single process for every connection. But most of designers paid attention on operating system and the complex scheduling disciplines.

In a For TER, the communication between CEs and FEs is realized on Transport Mapping Layer (TML), which is responsible for transportation of For CES protocol messages over variant transport media (like TCP, IP, ATM, Ethernet, etc.) (Wang *et al*., 2007a). In order to transfer messages efficiently, TML needs to allocate bandwidth between CEs and FEs. When the platform of CEs and FEs is changed, different transport media for realizing the set of TML services will be used. The TML that we researched uses TCP/IP based mediaand its communication model is Client/Server.

In this study, based on the application of Client/Server model in TML, we advance an evaluation model to analyze TML performance. By separating the communication process into two steps, we are able to get the key variables which effects performance obviously. Then we replace the process mechanism by thread mechanism in TML and separate sending and receiving threads, which improves performance of TML.

Because of the differences between TML and traditional router, it is improper to use traditional bandwidth allocation disciplines in TML. According to the bandwidth allocation requirements of TML, this paper proposes Dynamic Probabilistic Priority Based on Rate and Buffer (DPPBRB) algorithm. In this algorithm, we set an alterable parameter which will be modified when the arrival rate or buffer occupancy is changed. Then we use Probabilistic Priority(PP) discipline to change parameter. For the security of communications between one CE and multiple FEs, we propose a solution, which includes flow control Logical Function Block (LFB) entity in FEs that uses Token Bucket algorithm. Meanwhile, the token bucket parameters in FEs are dynamically calculated and re-configured by CE with the change of FE number and statue of all flow.

ARCHITECTURE OF FORTER

Figure 1 illustrates the architecture of For TER, which complies with the For CES Framework defined in RFC 3746. In For CES architecture, there has one primary CE or some redundant Ces for system high

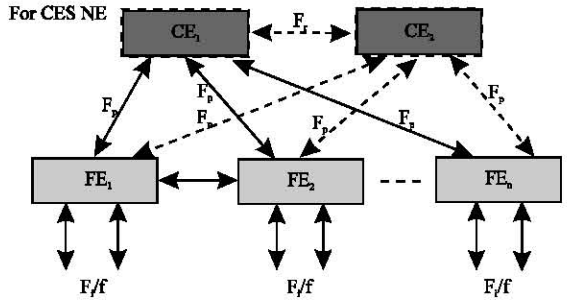

Fig. 1: Architecture of For TER

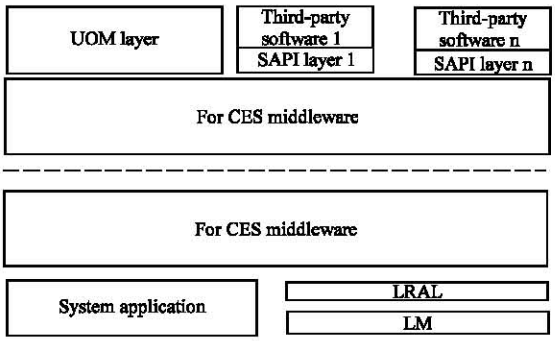

Fig. 2: Layered model of For TER

availability purpose. There are multiple FEs (may be up to hundreds of FEs for core routers), which are separated from CEs. The interface between CE and FE is to be standardized and the CE is responsible for the management of FEs by using the For CES protocol (Doria *et al*., 2010). The layered model of ForTER that we proposed is mentioned in Fig. 2 (Wang *et al*., 2008). The functions of each layer are described as follows.

UOM (User Operation Manager) Layer is set for the purpose of the whole CE management, which provides friendly interface to user for management of the whole For CES Network element.

Third-party software modules inside the CE contain independent software modules for CES applications. Typical application modules include routing protocol modules, like Zebra.

Service API (SAPI) Layer in CE is a bridge between third-party software and For CES Middleware. On the one hand, SAPIs in CE convert a service from third-party application software to a sequence of operation on LFBs. On the other hand, SAPIs in CE provides call back functions, which are interfaces to third-party software, for underlying function APIs in CE. In this way, SAPI Layer has hidden the feature of For CES for upper application and third-party software.

For CES Middleware executes encapsulation, decapsulation and transportation of For CES protocol messages between CEs and FEs. The main difference is that the For CES Middleware of CE contains function calls and data structures used for direct operation (i.e., configuration, inquiry, event report) on every attribute and capability of all LFBs, while in FE it registers all LFBs for path mapping.

LFB Resource Abstract Layer (LRAL) provides a standard interface and the management interface definition of operation.

LFB Middleware (LM) is gatherings of component with certain function.

COMMUNICATION OF MODEL OF TML

As Fig. 3 illustrates the logical model of TML, which contains two parts: CETML and FETML. CETML is as a server to handle request from different FETMLs. Usually, CETML adopts process mechanism to supply services for different FETMLs.

As Fig. 4 illustrates, TML lies under Protocol Layer (PL) and provides services for transporting protocol messages to PL. CE PL communicates with FE PL via CETML and FETML. When communication begins, PL delivers its For CES messages to its TML. Then the TML further delivers the connection messages to the destination peering TMLs. Then, on sender, TML delivers For CES messages to the destination peering TMLs on the channel which has been established; on receiver, TML delivers For CES messages to its PL.

There are two types of messages in For CES: Redirect messages and control messages. Redirect messages are the messages which contain data packets that are to or from outer networks via FEs, but need to be processed by CEs, including packets of routing protocol and SNMP protocol. These packets just are encapsulated directly in the redirect message and transport them between CEs and FEs. Control messages are all kinds of For CES protocol messages, which are produced by CEs or FEs and used to exchange control information like configure messages or query messages between CEs and FEs.

According to For CES requirements in RFC 3654, control messages should be transmitted reliably and its congestion is controllable, while redirect messages do not need to be reliably transmitted. In this study, TCP is applied to transmit For CES control messages and UDP is applied to transmit redirect messages (Wang *et al*., 2007b). In order to improve the TML system performance, this section we will describe the TML communication process. Then based on our evaluation model, we will explain the influences of variables on system performance and propose some improved methods.

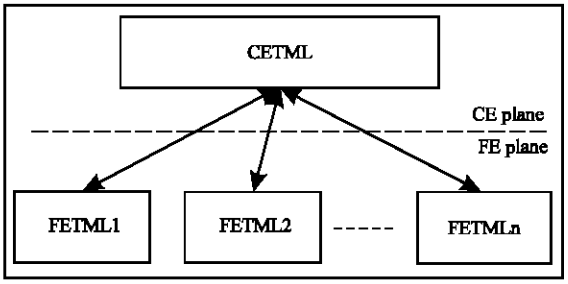

Fig. 3: Logical model of TML

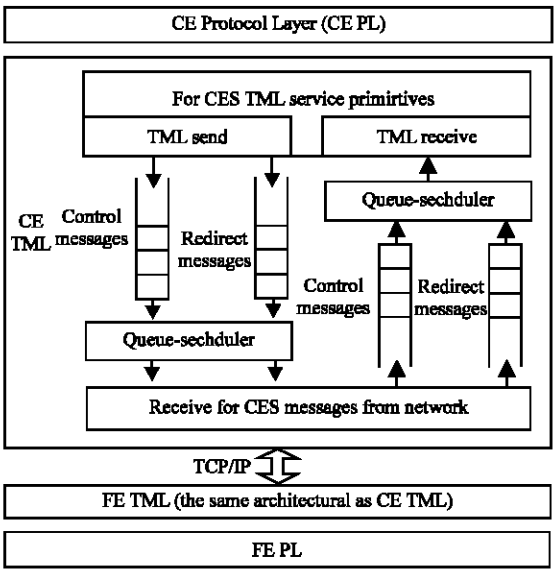

Fig. 4: Architecture of TML

Analysis of communication process in TML: By analyzing communication process in TML, we could find out the factors influencing system performance. The communication process of CETML and FETMLs describes as follows:

- CETML listens on a well known port and wait for FETML's request
- FETML sends request to CETML for connection
- CETML testifies the request information and handles it
- CETML sets up two message channels, which are used to transmit control and redirect messages, respectively
- CETML runs a watching process all the time. When messages that need to be transmitted on messages channels are detected, CETML will determine which process to deal with it

As described above, we know that CETML need to keep up a watching process. Once there is a connection request from FETML, it will answer and establish connection. In this process, security of communication messages is of great importance, for the majority messages in this process are type of TCP. But after connection setup, the dominant message on channel is changed. Because most of messages are from network and they are type of UDP. According with the communication process, we will create evaluation model of TML below.

Evaluation model of TML: Before our analysis, we normalize CETML resources as time. When the hardware is decided, we regard resources of CETML as T_{all}. Besides, CETML has a large number of session channel called logical channel in inter-communication process. Namely data transmission is all take place on logical channels. The following describes variables used in TML model.

- V_b: Packet handled per time
- T_s: Time of computation
- T_c: Time of communication
- T_p: Time of connection setup
- L_c: Length of data packet
- N_c: No. of processes
- N: No. of FETMLs
- M: No. of packets

It is well known that the time running a program is $T = T_s+T_c$. Let us consider N FETMLs communication with a single CETML and create N_c communication channels when processing with every FETML, the total time resource T_{all} is:

$$T_{all} = \sum_{i=1}^{N} T_i = \sum_{i=1}^{N} (T_{s_i} + T_{c_i}) \quad (1)$$

According to the traits of communication process, we could separate the inter-communication process into two steps: Connection-setup and data-transmission. In the connection setup step, CETML adopts multi-process to establish communication channel, whereas in data-transmission step, it sends and receives data on established channel. Thus, we define T_{cj} which is the communication time of channel j and can be described as below:

$$T_{cj} = T_{pj} + M_j * L_{pj}/V_{bj} \quad (2)$$

where, let$_u$ equal to $M_j * L_{pj}/V_{bj}$, which describes the time transmitting M_j packets in length of L_{pj} at the rate of V_{bj} on channel j.

So, T_{all} can be described as below:

$$T_{all} = \sum_{i=1}^{N}(T_{s_i} + \sum_{j=1}^{N_c}(T_{p_j} + M_j * L_{c_j} / V_{b_j})) \quad (3)$$

As T_{si} that describes the total time of computation mainly depends on the capability of computer hardware, so the key performance factor is:

$$\lambda = \sum_{i=1}^{N}\sum_{j=1}^{N_c}(T_{p_j} + M_j * L_{c_j} / V_{b_j}) \quad (4)$$

As described above, TCP messages play a dominant role in connection-setup step, which is expressed as T_{pj} in formula 4. In data-transmission step, UDP messages are in dominant, which is directly reflected on the factor. Furthermore, message transmission contains two actions: Sending and receiving. If the two actions are handled in single process, the switching time also becomes an important factor which is directly reflected on T_s.

Optimization of communication in TML: From what has been discussed above, we know that the number of FETML, the number of process, the computing time and the length of packet are the key variables which influence CETML performance. We will introduce some methods to improve the performance in the following.

Firstly, to enhance the load ability of CETML, we take thread mechanism instead of process mechanism. As known, in multi-thread model, all threads share CPU resources, which save CPU resources.

Secondly, to reduce the switching time, we separate sending and receiving into two threads. When sending and receiving share on one thread, it will take a lot of system time to wait for requests. While they are handled in separate thread, though it may increase the consumption of resource, it indeed improves communication efficiency.

In all, by using thread instead of process and separating sending and receiving threads, a lot of CPU resource can be saved and rate of data transmission can be increased.

INTERNAL FLOW CONTROL OF FORTER

From the last section we know that the optimization of TML can improve the performance of communication between CE and FE. In addition, we must consider internal flow control of ForTER. As is known, UDP is more aggressive than TCP, which leads to unfair bandwidth allocation and TCP congestion collapses when TCP and UDP are in the same link. In the ForTER, this will leave space for DoS attack from redirect messages. Thus a bandwidth allocation mechanism is needed to avoid the DoS attack from redirect messages, which includes one CE to one FE and one CE to multiple FEs. After analyzing the bandwidth allocation requirements, we discuss the scheduling discipline called DPPBRB for one CE to one FE and an internal flow control mechanism for one CE to multiple FEs.

The model of DPPBRB: In Fig. 5, each kind of messages has its own queue. Messages in the same queue are served in FIFO fashion. The control message queue is assigned an alterable parameter p_c ($0 = p_c = 1$)and the redirect message queue is assigned another alterable parameter p_d ($p_d = 1-p_c$). There is a scheduler at the end of the queue, a rate estimator at the head of the queue and a drop rate calculator in the middle.

At each service completion, the scheduler will use p_c to schedule messages between the two queues. As p_c increases, the bandwidth of the control channel will become larger. TML can reallocate bandwidth between control channel and redirect channel by adjusting p_c. The model of DPPBRB has a part of rate estimate and a part of drop rate calculate. As long as the arrival rate of control message flow is greater than its bandwidth or its buffer occupancy is less than a predefined threshold, the algorithm will increase the bandwidth of the control channel and reduce the bandwidth of the redirect channel by increasing parameter p_c. When the arrival rate of control message flow is less than its bandwidth, the algorithm will reduce the bandwidth of the control channel and increase the bandwidth of the redirect channel to enhance the transmission efficiency of TML. For redirect messages, the DPPBRB will also drop the message by arrival rate and buffer occupancy when its arrival rate is greater than its bandwidth.

Implement of DPPBRB: The packet enqueue and dequeue in TML should be modified by DPPBRB algorithm. When a message enters into a queue, firstly the algorithm estimates flow rate and calculate drop rate, then adjust p_c by these two parameters. The scheduler determines which queue is served in a service cycle by p_c.

Packet dequeue: If one queue is empty and the other is non-empty, the scheduler will transfer messages of the non-empty one. When both of the two queues are non-empty, the scheduler polls control message queue at each service circle firstly. The message at the head of the control message queue will be served by probability

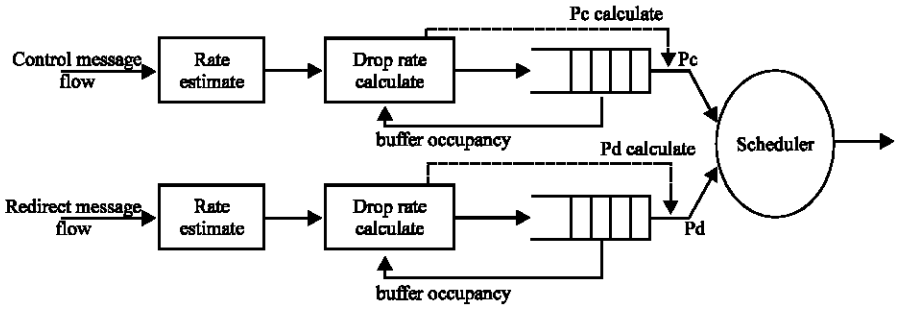

Fig. 5: The model of DPPBRB

p_c. At each service circle, if a control message is sent, the server will begin a new service circle. Otherwise, the server will send the message at the head of the redirect message queueand then begin a new service circle. In one service circle, set the probability of serving a control message equal to p_c and the probability of serving a redirect message equal to 1-p_c. Here, the bandwidths of the two message queues are:

$$B_c = p_c \times B, B_r = (1 - p_c) \times B \tag{5}$$

where, B is the total bandwidth, B_c is the bandwidth of the control channel and B_r is the bandwidth of the redirect channel. The control channel and redirect channel halve the total bandwidth when p_c = 0.5. The control messages won't be served if p_c = 0 while the redirect messages won't be served if p_c = 1.0. That is to say, the value of p_c decides the bandwidth allocation between control channel and redirect channel. In this study, the control and redirect channel halve the total bandwidth when the service begins. In the process of transmission, the parameter p_c will be adjusted in order to reallocate bandwidth between control channel and redirect channel.

Packet enqueue: First, set minimum buffer threshold Min_{thd} and maximum buffer threshold Max_{thd} for each message flow. Then according to message arrival rate, buffer occupancy and service rate of this flow to calculate the final drop probability.

Once there is a message needing to enqueue, first information of buffer occupancy is wantedand then compares the result to the setted threshold. If the length of queue has been greater than Max_{thd}, drop the message directly. If the length of message queue between Min_{thd} and Max_{thd}, packet enqueue module firstly calculate drop probability p_{q_dp} which was caused by change of queue length, then calculate drop probability p_{r_dp} which was incurred by change of message rate. At last, calculate final drop probability p_{dp} according to p_{q_dp} and p_{r_dp}.

The p_{q_dp} is defined as follows:

$$\begin{cases} p_{q\_dp} = Ax + B \\ Min_{thd} < x < Max_{thd} \end{cases} \tag{6}$$

where, A and B are constant and x is the length of current queue length.

The computation process of p_{r_dp} is as below:

Once there is a new packet arrives, we adopt exponential averaging to estimate the rate of the flow:

$$r_{new} = (1-\alpha)\frac{l_{k+1}}{T_{k+1}} + \alpha r_{old} \tag{7}$$

where,

$$\alpha = e^{\frac{t_{k+1}}{K}}$$

and K is a constant. $T_{k+1} = t_{k+1} - t_k$ which denotes the interval of the arrival of k+1th and kth packet.

Let B_i be the bandwidth of flow i and r_i be the rate of flow i. According to max-min fair bandwidth allocation, all messages of flow i should be forwarded when $r_i \le B_i$. Otherwise, a fraction $(r_i - B_i)/r_i$ will be refused, so it will have an output rate of exactly B_i. This suggests defining the p_{r_dp} as below:

$$p_{r\_dp} = \max\left(0, 1 - B_i / r_i\right) \tag{8}$$

Then, the final drop probability p_{dp} is defined as below:

$$p_{dp} = p_{q\_dp} \cdot p_{r\_dp} \tag{9}$$

When a redirect message comes, it will be dropped with the refusing probability p_{dp}. Otherwise, when a control message comes, p_c will be adjusted by p_{dp}. Let the threshold be p_{th} ($0 < p_{th} < 1$), p_c will be adjusted as follows:

If p_{dp} = 0and $B_c > B/2$, p_c is reduced as below:

$$p_c = \max\left(0.5, p_c \times \lambda_p\right), (0 < \lambda_p < 1) \quad (10)$$

If $p_{dp} > p_{th}$, p_c is increased as below:

$$p_c = \min\left(1, p_c \times \left(1 + p_{dp} - p_{th}\right)\right) \quad (11)$$

If control messages are too many to transfer, the buffer occupancy of control message will reduceand the refusing probability parameter p_{dp} will increase. This causes the parameter p_c to increase, thus the bandwidth of control channel increasesand the control messages are protected. If the bandwidth of control channel is greater than its arrival rate, reduce the bandwidth of control channel to improve the transmission rate and efficiency of redirect data by reducing p_c.

Design of internal flow control mechanism: The communication mechanism of multiple TMLs also needs to improve and research, so we propose an internal flow control mechanism. Regarding to the good extensibility of ForTER, the design of mechanism uses a flow control module in CE, which is as the brain of flow control mechanismand a flow control LFB that implements flow control. Figure 6 illustrates the block diagram of internal flow control mechanism.

Flow control LFB's abstraction and registration: XML file storage format endows LFB database with great flexibility, universal property and cross-platform portability. The tree structure of XML data is closely consistent with the requirements that CEs conveniently ask FEs to process LFB data, which may be the attributes, capabilities of LFBand etc. The abstraction of a LFB is a process of extracting the LFB's attributes, capabilities and other related informationand describing them in XML format. A LFB finishes registration to LFB database by parsing XML documents using the analytical tools. We preliminary abstract three attributes of flow control LFB: Flow Ctrl Flag, B_c(token bucket parameters) and B_e (token bucket parameters). FlowCtrlFlag is a flag used for judging whether to carry out the flow control strategy. Value 0, the default value, means unused and value 1 the opposite. All of the attributes are set the initial values, described in XML documents of class or instance. After LFB registration, the CE can configure the attributes of flow control LFB entities through For CES protocol.

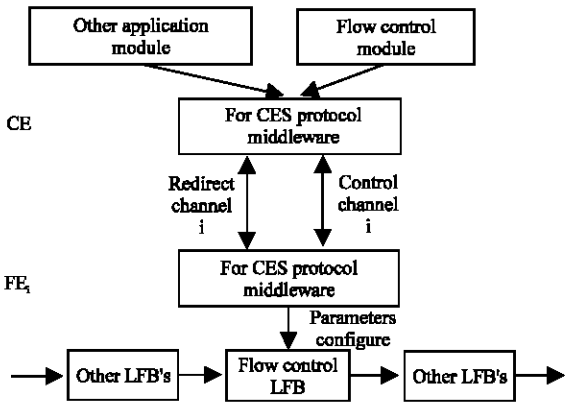

Fig. 6: Model for implementing flow control

Implementation of flow control LFB entity: First, we consider where to locate flow control LFB. Here we implement it when IP packets have been captured in Linux network layer and put into a custom queue. To capture packets from kernel, we register our own function in the NF_IP_PRE_ROUTING hook of Netfilter, which is the entry of network layer.

We use the return value NF_QUEUE to capture the packets required to be redirected to CE. Then we call the function that we register in nf_register_queue_ handler, in which we can do what we need to the captured packets. Here we insert the packets into a custom queue for further processing by the following LFBs.

The flow control LFB entity in this paper works by Linux soft interrupt mechanism (Bovet and Cesati, 2002). We define a soft interrupt by declaring like DECLARE_TASKLET (my_newtasklet, flowctrl, 0), among which flowctrl is the custom function where we can put our code for flow control (Pomerantz, 1998). Every time the soft interrupt is executed, the function we declared flowctrl will be executed. The tasks of Flow control LFB contain two steps. First, check Flow ctrl Flag to determine whether Token Bucket algorithm codes will be executed. Secondly, when CE asks FEs to execute flow control or update flow control parameters, Flow control LFB will process the configuration messages.

Task of flow control module in CE: The flow control module in CE works as follows: (1) compute flow control parameters B_c and B_e with the algorithm proposed in the study: (2) once the number of FEs alive changes, calculate these parameters again and re-configured them to FEs.

The proposed algorithm is aimed to settle the congestion problem that results from the increasing number of FE for a burst of redirect flow is more likely to cause congestion. This algorithm is chiefly contains following two equations:

$$B_c = T_s \frac{B_{total} - \sum_{i=1}^{N} B_{control_i}}{N} \quad (12)$$

where, B_{total} and $B_{controli}$ denote respectively the maximum communication bandwidth allowed by CE and the occupied bandwidth of control messages when communicate with FE i. N is the total number of FE. T_s is the inter-arrival time of Token. B_c/T_s is committed access rate:

$$\begin{cases} \dfrac{T_s \sum_{i=1}^{N} F\left(B_{redirect_i}(t)\right) - NB_c}{NB_c} \le \delta \\ F(x) = \begin{cases} x, x < B_e \\ B_e, x \ge B_e \end{cases} \end{cases} \quad (13)$$

$B_{redirect}(t)$ is the redirect flow of FE i at optional time t. is a factor of flow control quality. It's a non-negative constant and its value approaches zero. Computing parameter B_e is the key to the algorithm. It is a result by sample calculation with countless time, experience and estimation for specific network application environment. When:

$$\delta = 0, B_c \approx B_e$$

no extra burst is supported, that is, discard packets in the case there are not enough tokens. So control channel will not occur congestion at this time, meanwhile each FE is fairly treated. The deficiencies are a substantial waste of bandwidth and low efficient. The smaller δ is, the smaller B_e will be, accompanied by the less sudden flow allowed, the smaller probability congestion occurs. The managers can determine δ's value by the actual demand of quality themselves.

The value of T_s used in experiments is a round-trip time 1.5 sec. $B_{controli}$ is the occupied bandwidth of control messages which is estimated in certain circumstances when communicate with FE I.

SIMULATION AND EXPERIMENT RESULTS

Here, we will show you the simulations and experiment results.

Performance optimization test: In order to collect performance data, we deployed a Smart bits 600 flow generator in testing environment. Smart bits 600 is a high performance network data packet generator. We choose the type of packet by changing the parameters in SmartWindow which is analysis software cooperated with Smart bits 600. The other PC is used to run CETML and FETML program respectively. All the packets are generated from Smart bit 600, by handling at CETML or FETML which connect in Ethernet Switch. At last, the packets come back to Smart bits 600 again. During the process, the information of network is recorded in SmartWindow. For example, the number of packet handled in, the rate of packet, the type of packet. In this section, we will test our optimization strategy of communication in TML on the platform. Namely on the condition of transmission in single thread and double threads, we testing the load of CETML and the loss rate of packets. By testing, it verifies the right of our analysis and evaluation model of TML.

Load of CETML: The load of CETML directly reflects the condition that how many number of FETMLs CETML is able to handle. Here our test condition is as follows: Packets rate is 1000 packets per secondand the size of packet is 1024 bytes.

As mentioned in Fig. 7, the asterisk line shows the condition when CETML works in single thread, the maximum number of FETMLs it could handle is 11, whereas, if it works in double thread, the maximum number extends to 17. Furthermore, it obviously improves the performance when the number is small. Thus, it advances the stability of CETML.

Loss rate of packets: Here we test the loss rate of packets in two conditions. One is packets in length of 1024 bytes and transmitting in TCP.

As mentioned in Fig. 8, the asterisk line describes the performance of single threads while the round donut line describes the performance of double threads. When the packets to be handled are less than 400 sec^{-1}, both of them are without packet loss. But as the packets sending rate overcomes 400 sec^{-1}, compared with single thread, performance of double thread is better. Even sending rate up to 600 sec^{-1}, the loss rate is still zero.

The other is packets in length of 64 b. As mentioned in Fig. 9, the loss rate grows when the sending rate increases. The asterisk line and the round donut line describe the same the performance as Fig. 8. The difference between Fig. 8 and 9 is that no matter working in single thread or double threads, the loss rate of packet of 64 bytes is earlier than that of 1024 bytes.

The problem of the phenomenon is that the router handle IP data in packets, if the stream rate is a

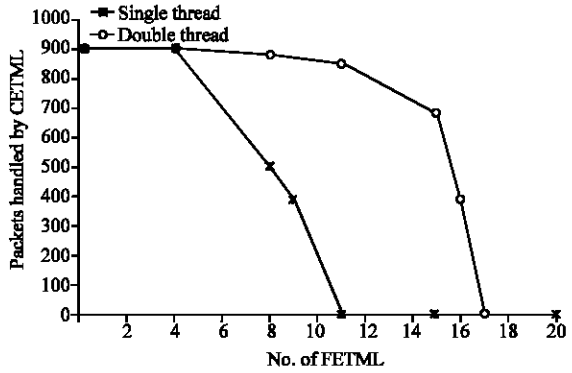

Fig. 7: Comparison of CE TML load abilities

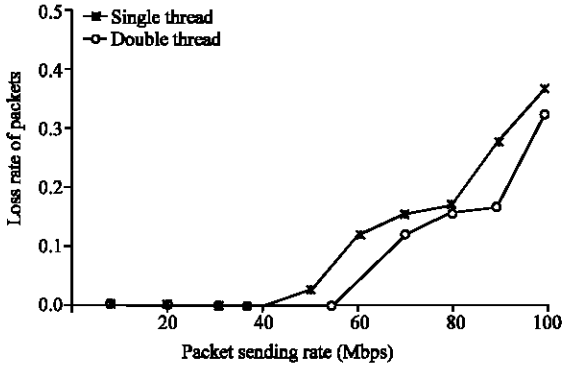

Fig. 8: Comparison of loss rate of packets (length of packet is 1024 bytes)

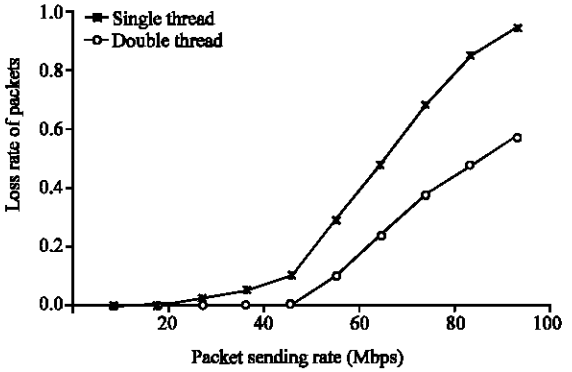

Fig. 9: Comparison of loss rate of packets (Length of packet is 64 bytes)

constant, the smaller size of packet is, a larger number of them. We suppose that by designing a method of combining small packets into a larger one is an effective way to improve forwarding rate.

Bandwidth allocation mechanism simulation: Here we simulated the performance of DPPBRB to confirm that it can avoid DoS effectively. We compared its performance with DRR scheme and DropTail scheme.

Table 1: Parameters of DPPBRB

Param	P_c	A	B	P_{th}	λ_p	Max_{thd}	Min_{thd}
Value	0.8	0.0127	0	0.011	0.995	2000075	25

All simulations are performed using ns-2.30. TCP and UDP traffic generation codes are available from the ns-2 distribution. We used the five-node topology shown in Fig. 10. There is a CE(0) and an FE(2) in simulations. The source (3) is used to generate rate adjustable TCP and UDP flows that are input to the FE. The TCP flow simulates For CES control message flow that to be sent from FE to CE via a TML. The UDP flow simulates For CES redirect message flow that to be delivered to the CE from the FE via the TML. There is another rate adjustable UDP flow which simulates the redirect message generated by DoS attacker (4). At the CE side, it is designed to monitor how TCP flow or UDP flow are transported from the FE to the CE, to see if some of them are blocked or not.

To show how DPPBRB performs with respect to DRR and DropTail, the link between the FE and the Router (1) adopts DPPBRB or DRR or DropTail in different simulations respectively. And the link has buffer occupancy of 100 packets. All of the links have a capacity of 10 Mbps and a latency of 0.1ms. In the simulations, we set the parameters of DPPBRB as Table 1. For DRR and DropTail, all the parameters used are default setting in ns-2.30.

Results of DPPBRB between one CE and one FE: In the first simulation, the Source begins to generate an increasing rate of TCP and UDP flows at 0 s. Both of the flows have the same rate (rate = time*0.5+2Mbps). At 1 s, the DoS attacker begins to generate UDP flow with an increasing rate (rate = time*2-1 Mbps). At 6 s, the Source and the DoS attacker stop sending packets.

We further contrast DPPBRB with DRR and DropTail by examining the flow rates are constant. The Source is set to generate a constant rate of TCP and UDP flows (the TCP's rate is 4 Mbps and the UDP's rate is 6 Mbps) from 0-6 sec. The DoS attacker takes the same act as the first simulation. Fig. 11 a-c and 12 a-c show the simulation results.

As the simulations show, when the link is not congested, all of the flows can be regularly transferred respectively. For DPPBRB, when the link is congested, the TCP flow can also be transferred by its own speed while the transmission speed of UDP flow decrease with time. While for DRR, the result is opposite and for DropTail, the result is worse.

The simulation result shows that DPPBRB discipline can avoid DoS attacks from redirect messages whether the sending rate is constant or not.

Results of DPPBRB between one CE and three FEs: The network topology we used is shown in Fig. 13. There is one CE (0) and three FEs 2, 3 and 4 in simulation platform. The S1(5), S2(6) and S3(7) are used to generate rate adjustable TCP and UDP flows that are input to the FEs. The TCP flow simulates For CES control message flows which are sent from the FE to the CE via a TML. The UDP flow simulates For CES redirect message flows which are delivered to the CE from the FE via the TML. There is another rate adjustable UDP flows which simulate the redirect message generated by DoS attacker (8). At the CE side, it is designed to monitor how the TCP flow and the UDP flow are transported from the FE to the CE.

To show how it performs, the links between the FE and the Router (1) adopts DPPBRB. And the link has buffer occupancy of 100 packets. All of the links have a capacity of 10 Mbps and latency with 0.1 m sec. We set the DPPBRB parameters as Table 1.

In the first simulation, every source is set to generate a constant rate of TCP and UDP flows from 0-6s. The rate of TCP flows is 1Mbps and the rate of UDP flows is 2Mbps. At the time of 1s, the Dos1 attacker begins to generate UDP flow with an increasing rate (rate = time*3-1 Mbps). At the time of 6s, the sources and Dos1 attacker stop sending packets. Then in the second simulation, we change the rate of TCP and UDP flows. The rate of TCP and UDP flows are both set to be 4 Mbps. And there's no Dos attacker in the simulation. The results are shown in Fig. 14 and 15.

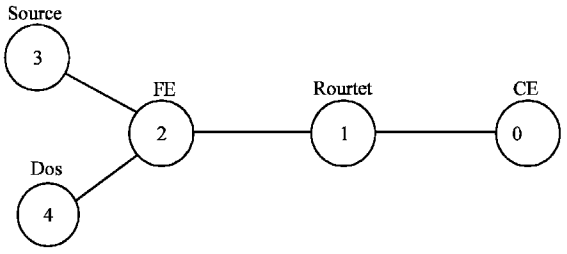

Fig. 10: Topology of bandwidth allocation simulation

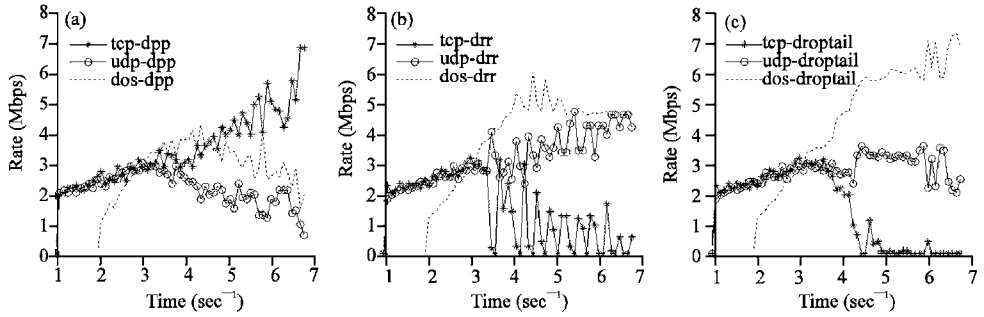

Fig. 11: Throughput status of the TCP and UDP flows when the flows' rate is increasing with time; (a) DPPBRB, (b) DRR and (c) drop tail

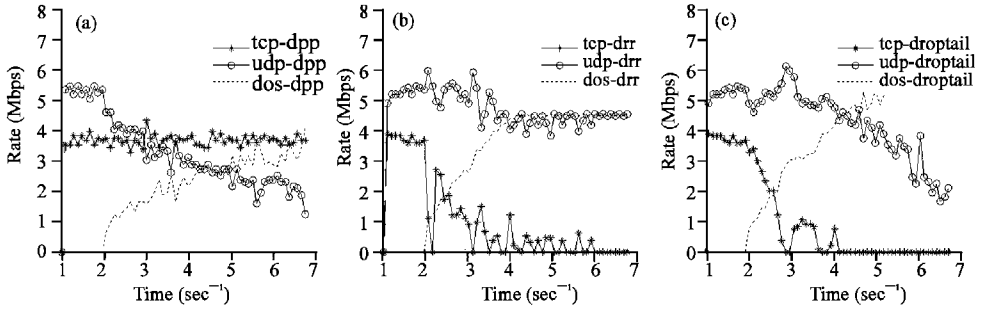

Fig. 12: Throughput status of the TCP and UDP flows when the flows' rate is constant; (a) DPPBRB, (b) DRR and (c) drop tail

In the above simulations, we can see that when the summation of three FE's flow rate is less than the link bandwidth of CE, all of the flows are transferred successfully. When the summation of three FE's flow rate is larger than link bandwidth of CE, even if the link between the FE and the Router adopts DPPBRB, every control channel suffers serious congestion which will cause ForTER's to collapse. So we need an internal flow control mechanism under multiple FEs.

Internal flow control test: The test platform consists of a switch, Smart bits 600 and our For TER, one CE and five FEs. Connecting CE and FEs are a 100Mbps Ethernet link. By adjusting the Smart bits 600 rate, we can simulate the control and redirect messages flows of different rates, respectively. We verify the feasibility of the internal flow control mechanism proposed in this paper by means of comparing the throughput status of control and redirect flow when the flow control mechanism is applied and not applied.

We simulate such a network environment where keep control messages sending traffic to every FE alway s with 5 Mbps, redirect messages sending traffic to two FEs keep with 5Mbpsand redirect messages sending traffic to the other Fes changes from 5 to 50 Mbps.

Figure 16 shows the total throughput status when no flow control system is applied. The throughput of the two flows will change with the change in redirect traffic. The throughput is represented by a percentage, indicating the relative throughput. In Fig. 16, we can clearly see the control throughput decreases along with the redirect traffic increases. This also means, the For CES control messages become harder and harder to be delivered when For CES redirect message traffic overwhelms, even resulting in ForTER's collapse.

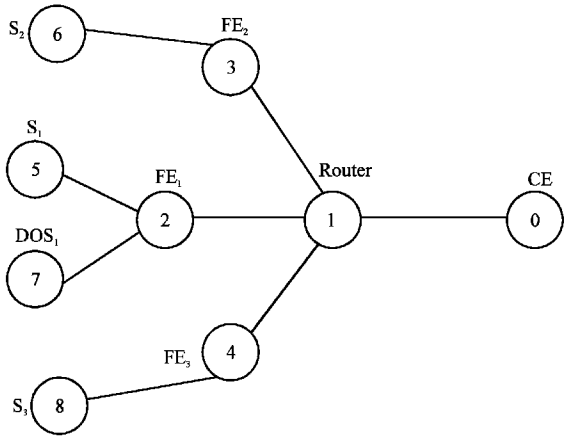

Fig. 13: The topology of one CE and three FEs

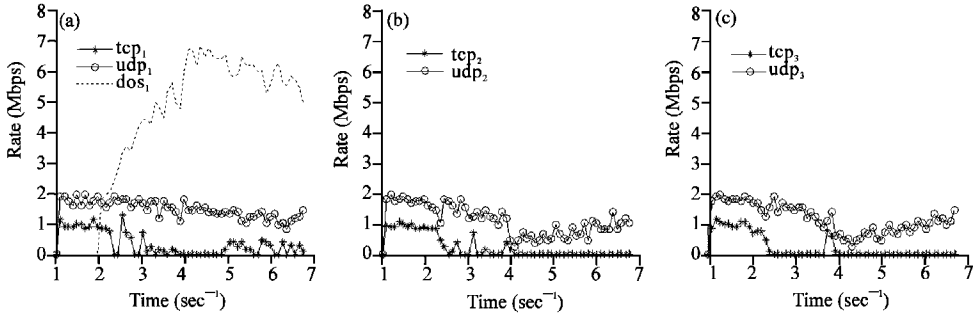

Fig. 14: Throughput status of the tcp and udp flows with Dos attacker; (a) FE_1, (b) FE_2 and FE_3

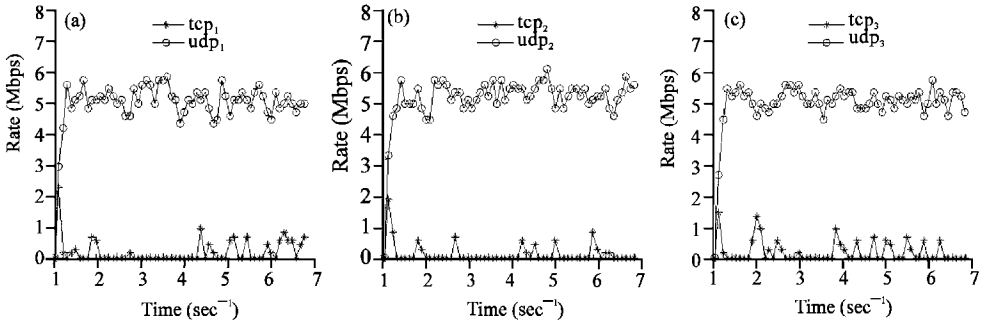

Fig. 15: Throughput status of the tcp and udp flows without Dos attacker ; (a) FE_1, (b) FE_2 and FE_3

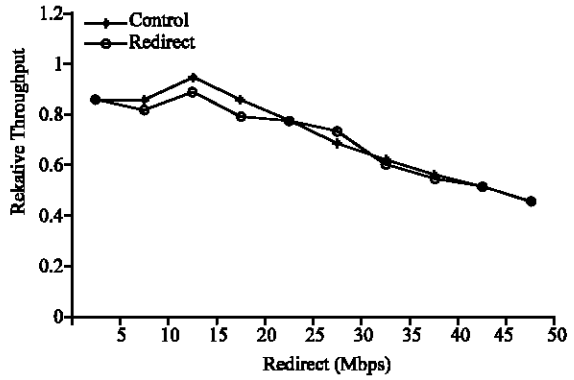

Fig. 16: Throughput status of no flow control

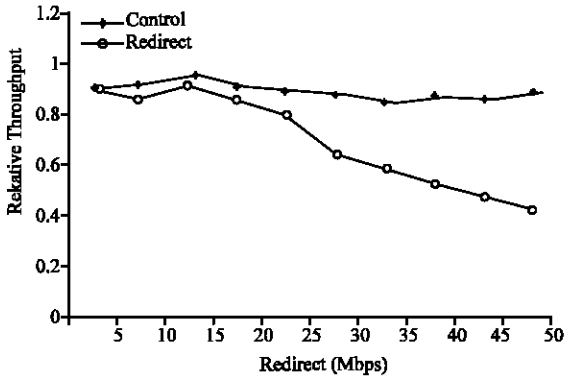

Fig. 17: Throughput status of adopting flow control

Figure 17 shows the different results when the flow control mechanism is applied. Now let equal to 1%, thus B_e is approximately computed 21.92 M, in the network environment depicted previously. What we can show in Fig. 17, as redirect traffic increases from 5 to 50 M, the relative throughput of control messages can roughly keep at 100% status, meaning the For CES control messages can be transferred normally. This also means attackers are unable to destroy the communications between CE and FEs, meanwhile a certain amount of burst is allowed. The internal flow control mechanism can effect to avoid DoS attacks from redirect messages in the case of multi-FEs.

CONCLUSION

To improve the communication efficiency between CE and FEs of For TER, this study focuses on the research of internal flow control mechanism. By analyzing an evaluation model of TML, We propose some methods to improve the performance of TML and verify its effectiveness. Then we present a scheduling discipline called DPPBRB after analyzing the bandwidth allocation requirements of TML. After the simulation we can see that the DPPBRB algorithm is proper for For CES router to avoid DoS attacks from redirect messages. But the simulation results show that we need the internal flow control under multi-FEs. In order to solve the problem, we propose an internal flow control mechanism and provide a good reference example.

IETF For CES working group is now working hard and will be further recognized by researchers and manufacturers. These algorithm and methods that this paper proposed effectively prevents the DoS attack from redirect messages and improves reliability of For CES routers. More importantly, the research of internal flow control mechanism will promote For CES applications and industrial. It is expected the algorithm and methods can contribute to IETF For CES work for its progress.

ACKNOWLEDGMENTS

This study was supported in part by a grant from the National High Technology Development 863 Program of China under Grant No. 2008AA01A323, 2009AA01A334, 2008AA01Z214, 2008AA01A325, the National Natural Science Foundation of China under Grant No.60903214, 60970126, Zhejiang Provincial NSF China No. Y1090452, Y200804981, Y1080078, Zhejiang Sci and Tech Project No. 2009C31066, 2009C11050.

REFERENCES

Bovet, D.P. and M. Cesati, 2002. Understanding the Linux Kernel. 2nd Edn., O'Reilly Publisher, New York, pp: 784.

Chrysoulas, C., E. Haleplidis, G. Kostopou-Los, S. Denazis and O. Koufopavlou, 2006. A distributed router's modeling and imple-mentation. ESRGroups Research Report, Dec.

Doria, A., J. Hadi-Salim, R. Haas, H. Khosravi and W.M. Wang, 2010. Forwarding and control element separation protocol specification. RFC 5810. http://tools.ietf.org/html//rfc5810.

Fu, J., 2008. On the design of next-generation routers and IP networks. Doctoral Thesis, KTH, Stockholm, Sweden.

Fu, J., P. Sjodin and G. Karlsson, 2009. Intra-domain routing convergence with centralized control. Comput. Networks Int. J. Comput. Telecommun. Networking, 53: 2985-2996.

Gu, G.Q. and J.Z. Luo, 2006. Some issues on computer networks: Architecture and key technologies. J. Comput. Sci. Technol., 21: 708-722.

Haas, R., 2010. Forwarding and control element separation (For CES). MIB, RFC 5813. http://www.faqs.org/rfcs/rfc5813.html.

Haleplidis, E., R. Haas, S. Denazis and O. Koufopavlou, 2009. A Web Service- and For CES-Based Programmable Router Architecture, Active and Programmable Networks. In: Lecture Notes in Computer Science, Hutchisea, D. *et al.* (Eds.). Springer, New York, pp: 108-120.

Halpern, J. and J. Hadi Salim, 2010. Forwarding and Control Element Separation (For CES) forwarding element model. RFC 5812. http://tools.ietf.org/search/ rfc 5812.

Hanigk, S., M. Kretzschmar and F. Eyermann, 2009. A distributed routing architecture for secure communication over highly dynamic radio networks. Proceedings of the 1st International Conference on Communication Systems and Networks Table of Contents, Jan. 05-10, Bangalore, India, pp: 234-242.

Hidell, M., P. Sjodin and O. Hagsand, 2005. Control and forwarding plane interaction in distributed routers. Proceedings of the 4th International IFIP-TC6 Networking Conference, (IIFIPNC'05), Waterloo, Canada, pp: 1339-1342.

Hu, J.C., I. Pyrali and D.C. Schmidt, 1997. Measuring the impact of event dispatching and concurrency models on web server performance over high-speed network. Proceedings of the 2nd Global Internet Conferenve IEEE, November 1997, Phoenix, AZ., pp: 1-2.

Jiang, Y., C.K. Tham and C.C. Ko, 2002. A probabilistic priority scheduling discipline for multi-service networks. Comput. Commun., 25: 1243-1254.

Katevenis, M., C. Sidiropoulos and C. Courcoubetis, 1991. Weighted round-robin cell multiplexing in a general purpose ATM switch chip. IEEE J. Selected Areas Commun., 9: 1265-1279.

Louati, W., I. Houidi, M. Kharrat, D. Zeghlache and H.M. Khosravi, 2008. Dynamic service deployment in a distributed heterogeneous cluster based router (DHCR). Cluster Comput., 11: 355-372.

Mogul, J. and K. Ramakrishnan, 1997. Eliminating receive livelock in an interrupt-driven kernel. ACM Trans. Comput. Syst., 15: 217-252.

Pai, V.S., P. Druschel and W. Zwaenepoel, 1999. A unified I/O buffering and caching system. Proceedings of the 3rd Symposia on Operating System Design and Implementation, Febrauary 1999, New York, pp: 1-15.

Parekh, A.K. and R.G. Gallager, 1993. A generalized processor sharing approach to flow control in integrated service network: The single node case. IEEE Trans. Network., 1: 344-357.

Pomerantz, O., 1998. Linux kernel module programming guide. http://www.linuxtopia. org/online_books /Linux_Kernel_Module_Programming_Guide/x1032. html.

Salim, J.H. and K. Ogawa, 2010. SCTP-based Transport Mapping Layer (TML) for the forwarding and control element. RFC 5811. http://www.faqs. org/rfcs/ rfc5811.html.

Shreedhar, M. and G. Varghese, 1996. Efficient fair queuing using deficit round robin. IEEE Trans. Network., 4: 375-385.

Stoica, I., S. Schenker and H. Zhang, 1998. Core-stateless fair queuing: Achieving approximately bandwidth allocations in high speed networks. ACM SIGCOMM Comput. Commun. Rev., 28: 118-130.

Struck, B. and M. Weingarten, 2004. Next gen switch/router design issues. Bus. Commun. Rev., 34: 44-50.

Takourout, R.N., S. Pierre and L. Marchand, 2006. Separation of the control plane and forwarding plane in next-generation routers. J. Comput. Sci., 2: 815-823.

Wang, W.M., J.H. Salim and A. Audu, 2007a. For CES Transport Mapping Layer (TML) service primitives. http://tools.ietf.org/html/draft-ietf-For CES-tmlsp-01.

Wang, W.M., L.G. Dong and B. Zhuge, 2007b. TCP and UDP based For CES protocol TML over IP networks. http://tools.ietf.org/html/draft-wang-For CES-iptml-02.

Wang, W.M., L.G. Dong and B. Zhuge, 2008. Analysis and implementation of an open programmable router based on forwarding and control elements separation. J. Comput. Sci. Technol., 23: 769-779.

Wei, D.X., C. Jin, S.H. Low and S. Hegde, 2004. FAST TCP: Motivation, architecture, algorithms, performance. I EEE/ACM Trans. Network., 14: 1246-1259.

Wu, J.P. and K. Xu, 2006. Research on next-generation internet architecture. J. Comput. Sci. Technol., 21: 723-731.

Yoon, S.H., D. Siradjev and Y.T. Kim, 2006. Management of Diffserv-over-MPLS Transit Networks with BFD/OAM in For CES Architecture. In: Lecture Notes in Computer Science, State, R. *et al.* (Eds.). Yeungnam University, Korea, pp: 136-148.