International Journal of Computational Intelligence Systems, Vol. 5, No. 5 (September, 2012), 851-859

A Credibility-based Congestion Control Scheme and its Performance Evaluation*

Jin-xin Zhang, Man-gui Liang+
Institute of Information Science, Beijing Jiaotong University
Beijing, 100044, China
E-mail: {07112081,mgliang}@bjtu.edu.cn

Shu-juan Wang
Columbia University, 1300 Seeley W. Mudd
New York,10027 , USA
E-mail: shujuan@ee.columbia.edu

Received 20 November 2011
Accepted 15 June 2012

Abstract

A congestion control scheme based on credibility is proposed. In this scheme, the whole network is divided into several independent domains, and each domain contains a congestion control server (CCS) and several control modules (CM). The CCS is used to collect credit information and make punishment decisions based on the credit information, while the CM creates signaling packets to calculate responsibility mark. Violators will be punished according to their responsibility mark. The effectiveness of this scheme is also analyzed. We provide simulation results to demonstrate that the proposed scheme can process congestion and provide better performance gains.

Keywords: congestion control; credibility-based; effectiveness; low cost

1. Introduction

Although computer networks have overspread largely in recent years, it only provides single class of "best effort" service, there is no admission control and the network offers no assurance about when, or even if, packets will be delivered [1]. The main reason is that the simple design of Internet makes it a great success, but it also brings a lot of problems. Congestion is one of the most severe problems.

In addition, network security is very important for its stable operation. Denial-of-Service (DoS) is one of the most popular attack fashions. It makes network resource unavailable to its intended users, and causes congestion. Thus, congestion control can alleviate the loss of DoS attack, but it can not avoid being attacked.

Generically, congestion control algorithm can be modeled as a feedback system where the input is congestion information and the output is the adjustment sending rate of the end system; in turn, the sending rates of end systems affect the state of congestion in the network [2].

Congestion can not be solved by simply increasing network resource, for example, large buffer space, high-speed links and high-speed processors [3]. The design objectives of congestion control algorithm contain: low overhead, fair, distributed and efficiency.

* This work was supported by Ministry of Science and Technology of China (Grant No 2007AA01Z203).
+ The corresponding author.

In this paper, we propose a credibility-based congestion control scheme [4] and it works at network layer. For this scheme, only source is responsible for traffic shaping, other elements in network don't need to do this work, so the cost of network is decreased. The whole network is divided into several domains, so the risk among network is isolated effectively.

The remainder of this paper is organized as follows: section 2 introduces the related works; in section 3, the proposed scheme is explained in detail; the effectiveness of the proposed scheme is explained in section 4; numerical simulations are given in section 5; section 6 concludes this paper up.

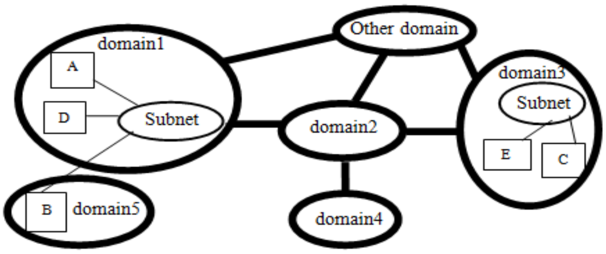

Fig. 1 Illustration of domain

2. Related Works

In the last few years, TCP congestion control policy has been extensively studied in the literatures [5]-[9]. The primary methods regulate the congestion window size maintained by each TCP sender [5]. The coming approaches for network congestion control cover a broad range of techniques, including source quench [6], slow start, schedule-based control [7], binary feedback [8] and rate-based control [9]. Moreover, Low [10] proposed an optimization framework based on utility function, and designed a series of new congestion control schemes that can achieve relative balance of objectives: using network effectively, allocating resource fairly and low queuing delay.

The TCP congestion control and the congestion control scheme based on optimization all work at the transport layer, so it may cause significant delay between congestion occurring and taking control action. If the length of information sequence is very short, the feedback may be arrived after the source sent all data. Therefore, a credibility-based congestion control scheme is proposed in this paper. It works at the network layer, and can take control action immediately. The advantages of this scheme contain: don't need to allocate resource at network layer, only increase the complexity of the shared resource (switch, etc.); can decrease the cost of whole network.

The proposed scheme is similar to computational intelligence, and it is self-adaptive. In the beginning, there are a number of violators. When violator realizes the serous punishment, it will restrict its behavior. The most serious violator will be prohibited from using network. Finally, there are only several violators so that the entire network will be more safe and reliable.

3. Credibility-based Congestion Control Scheme

3.1. *Definition*

Domains are usually divided into several subnets in the light of regions or other purposes. Fig. 1 shows a network example. The thick circles denote subnets, and it contains smaller subnet (thin circles) or terminal nodes (rectangles).

We define a domain as a subnet which has congestion control schemes and runs independently. A domain consists of one congestion control server (CCS) and several port controllers (PC). CCS is responsible for storing and collecting general information, PCs control communication with other parts in network. The basic idea is to compute each CCS's responsibility mark according to congestion state, and then use this information to give punishment to CCS related domains. The entire network's responsibility is providing each node's responsibility mark to the decision center, so that the punishment can be decided.

3.2. *Congestion Control Model*

The congestion control model is shown in Fig. 2. Data are sent from right to left. The right irregular circle denotes the source subnet, the left one is the destination subnet, and the middle circle is the domain named A where congestion occurs. A connected with source and destination subnets through ports which include congestion control modules (CM). We call the entrance module in port connects source subnet and A as InM, and the outlet module in port connects A and destination subnet as OutM. S and D is source and destination, the diamond shows congestion point.

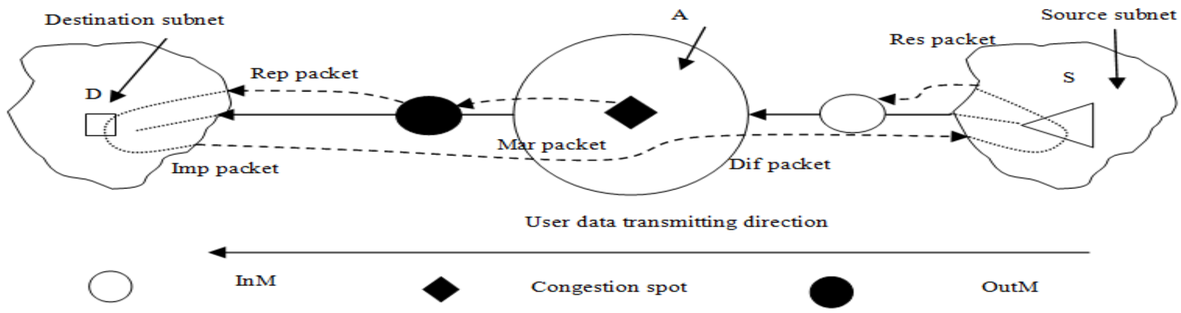

Fig. 2. Congestion control model.

The congestion control process will create five kinds of signaling packets: Mar packet, Rep packet, Imp packet, Dif packet and Res packet. Their specifications are given as follows:

- Mar packet: when congestion happens, the corresponding switching equipment samples some data packets, and changes these data packets into Mar packets by altering the data packets' header.
- Rep packet: OutMs create Rep packets base on the number of Mar packets, and recover Mar packets into data packets.
- Imp packet: destinations create Imp packets to inform sources that congestion happens.
- Dif packet: InMs change Imp packets into Dif packets to inform sources the congestion.
- Res packet: sources' responses after received Dif packets, carry acknowledgement or clarification information.
- Acknowledgement information: sources admit violation after self-check, and account for taking responsibility.
- Clarification information: sources declare innocence.

We suppose the diamond rectangle is the congestion spot (named as G). G sends Mar packets to OutM. OutM then creates Rep packets, reverting Mar packets into data packets, and sends them to the destination subnet. After receives Rep packets, sources are informed that congestion happens in the routes, then sources send back Imp packets to OutM. OutM forwards Imp packets into network, and there will be 3 results for the packets:

(i) Reach at the right source;

(ii) Reach at wrong sources;

(iii) Don't reach any source or end equipment.

Source will send back packets carrying acknowledgement or clarification information in case (i). In case (ii) or (iii), end equipments or related intermediate equipments will send messages to alarm the system that network failure occurs and InM won't get Res packets. The counters will be triggered when signaling passes through OutM and InM, as Fig. 3 shows.

Counter variables specifications are given as follows:

- Crep: count for report, stores in OutM, port granularity.
- Cim: count for impeachment, stores in OutM, port granularity.
- Cdi: count for diffuse impeachment, stores in InM, aggregation paths granularity.
- Cac: count for acknowledgement, stores in InM, aggregation paths granularity.
- Ccl: count for clarification, stores in InM, aggregation paths granularity.

Take all paths from InM to OutM in domain as one aggregation path, we can compute three sampling values, *Sl* for not-impeach sampling, *Sr* for not-response sampling, and *Sa* for acknowledgement sampling.

In domain's views, congestion CMs are the

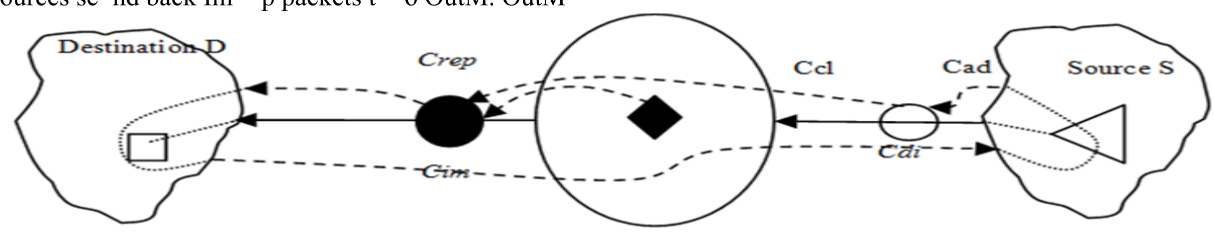

Fig. 3. The generation of counter variables.

responsibility holders. Numbering the congestion CMs, we can get the serial numbers of aggregation paths. For example, if there are N modules on the edge of the domain, numbered by $\{1, 2, ..., N\}$, the aggregation path from i to j marked as $[i, j]$, where i is InM's number and j is OutM's number. All parameters in OutM form a one-dimension array with subscript $[j]$; all parameters in InM form a two-dimension array with subscript $[i, j]$. Responsibility mark is computed as follows: once a counting time (pre-defined) finished, OutM use $Crep[j]$ and $Cim[j]$ to get $Sl[j]$, InM use $Cdi[i, j]$, $Ccl[i, j]$ and $Cac[i, j]$ to get $Sa[i, j]$ and $Sr[i, j]$ respectively, the values are computed by the following formulas:

(a) Not-impeach sampling $Sl[j]$: if destination can be trusted, then $Crep[j] = Cim[j]$. Otherwise $Crep[j] > Cim[j]$; $Crep[j] < Cim[j]$ means that terrible problems occur and congestion CMs should alarm the system. Number j module's not-impeach mark:

$$Sl[j] = \frac{Crep[j] - Cim[j]}{Crep[j]} \quad (1)$$

If $Crep[j] = 0$, set $Sl[j] = 0$.

(b) Not-response sampling $Sr[i, j]$: if source can be trusted, then $Cdi[i, j] = Ccl[i, j] + Cac[i, j]$. Otherwise $Cdi[i, j] > Ccl[i, j] + Cac[i, j]$; $Cdi[i, j] < Ccl[i, j] + Cac[i, j]$ means that terrible problems occur and congestion CMs should alarm the system. Aggregation path $[i, j]$'s not response mark:

$$Sr[i, j] = \frac{Cdi[i, j] - (Ccl[i, j] + Cad[i, j])}{Cdi[i, j]} \quad (2)$$

If $Cdi[i, j] = 0$, set $Sr[i, j] = 0$.

(c) Acknowledgement sampling $Sa[i, j]$: a source needs to take on some responsibility when it acknowledges violation. The calculating principle is that in one counting time, if one congestion CM acknowledges violation, other modules can avoid punishment, only the one acknowledges be scored. Situation may exits that several modules acknowledge violation, and then all modules would be scored one mark. In all above situations, $\sum_i \sum_j Cad[i, j] \neq 0$.

If $Cad[i, j] > 0$, $Sa[i, j] = 1$; otherwise $Sa[i, j] = 0$.

All modules' $Cad[i, j]$ in the same domain should be take n into account to determine each module's $Sa[i, j]$. After several counting times, all $Cad[i, j]$ will be sent to CCS to be computed. If no congestion CM acknowledges violation, which makes $\sum_i \sum_j Cad[i, j] = 0$. Set $\Delta = \sum_i \sum_j Cdi[i, j]$, if $\Delta = 0$, then $Sa[i, j] = 0$.

We set a period of time to implement this method. Take R as an object's responsibility mark in one counting time, which concerns the whole network effect. The total score in the set period of time can be calculated as follows:

$$U[n] = \lambda U[n-1] + (1-\lambda) R[n] \quad (3)$$

$U[n]$ is the total score after n counting times, $R[n]$ is the responsibility mark including other modules' effect in counting time $[n]$, $0 \le \lambda \le 1$, $\lambda = 1/M$, where M is the equivalent time length of counting times. Similarly, we can get formulas for Ul, Ua and Ur as follows:

$$Ul[n] = \lambda Ul[n-1] + (1-\lambda) Rl[n] \quad (4)$$

$$Ua[n] = \lambda Ua[n-1] + (1-\lambda) Ra[n] \quad (5)$$

$$Ur[n] = \lambda Ur[n-1] + (1-\lambda) Rr[n] \quad (6)$$

$Rl[n]$, $Ra[n]$ and $Rr[n]$ is left-responsibility mark, acknowledgement score and right-responsibility mark in number n counting time. $Ul[n]$, $Ua[n]$ and $Ur[n]$ is left-total score, ack-total score and right-total score after n counting times. Sources are responsible for Ua and Ur in InM, and destinations are responsible for Ul in OutM. Ul, Ur and Ua are computed in each congestion CM, and be sent to the CCS after a given period of time to get the quantification punishment.

4. Effectiveness of Credibility-based Congestion Control Scheme

Credibility means that all nodes are well-reputed except violators. Thus, according to the number of violators and the reputation of violators, the congestion can be divided into five cases in a domain.

4.1. *Single domain*

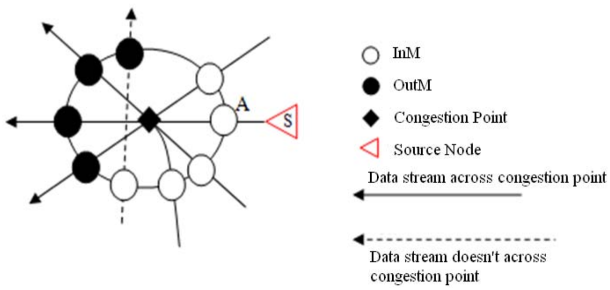

Fig. 4. The violation of a trustful source node in a domain.

(i) The violation of one trustful source node

As shown in Fig. 4, there is a trustful source node S violating the rules, which causes congestion in a domain, so congestion control scheme is triggered. After receiving a Dif packet belongs to S, it sends back a Res packet which carries acknowledgement information to InM A, and accounts for taking responsibility. The punishment result is that the violator S and its connected InM A have direct acknowledgement marks, but other CMs don't have marks.

(ii) The violation of one distrustful source node

As shown in Fig. 5, suppose the source node S_2 violates the rules, which causes congestion, so congestion control scheme is triggered. Then all InMs (A、B、C and D) send Dif packet to their sources in the domain respectively. After receiving the Dif packet, these sources send back Res packets. Res packet contains two kinds of information: acknowledgement information and clarification information.

In this case, source nodes S_1、S_3 and S_4 will send back Res packets carrying acknowledgement information, because none of them violates the rules. As the source node S_2 is the violator and is distrustful, it will also send back Res packet carrying acknowledgement information in order to avoid responsibility. All source nodes send back Res packets carrying acknowledgement information, so the CCS can not identify which one is the violator. Therefore, that congestion should be in charge of responsibility by all four InMs (A、B、C and D) on average. Their responsibility marks are calculated based on their traffic level. The InM forwarding heavy traffic will undertake main responsibility, because the heavy traffic has high possibility causing congestion.

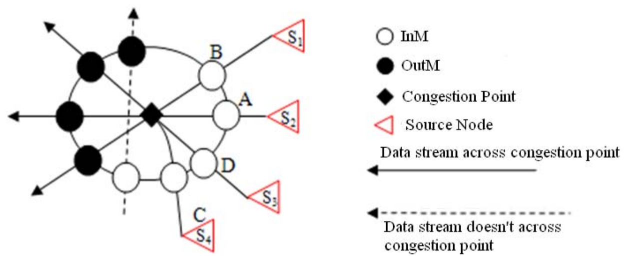

Fig. 5. The violation of a distrustful source node in a domain.

There are two methods to identify the violator. 1) when CCS receiving alarm, it starts monitoring equipment which is an arbitration equipment to identify the violator; 2) if there is no monitoring equipment, the innocent nodes may tolerant until reach their threshold. Then the innocent nodes may install measuring module that will incurring some costs. Finally, all nodes install measuring modules except the violator, so it is easy to identify the violator in that case.

After identified violator, it will be punished heavily and be set low repute level. The punishment is calculated by time interval from the beginning of congestion to identifying violator multiply the maximum cost per time.

(iii) The violation of multiple trustful source nodes

As shown in Fig. 5, suppose the source nodes S_1 and S_2 violate the rules causing congestion but the source nodes S_3 and S_4 do not violate the rules, so congestion control scheme is triggered. As all source nodes are well-reputed, S_1 and S_2 will send back Res packets carrying acknowledgement information to A and B respectively; S_3 and S_4 will send back Res packets carrying clarification information. Their responsibility marks are undertaken by S_1、S_2、A and B, and the mark of other InMs and source nodes will not be reduced.

(iv) The violation of multiple source nodes and only one distrustful source node

Suppose multiple source nodes violate the rules causing congestion but only one distrustful source node. In that case, the violator may not be punished, but it will generate heavy traffic again violating the rules owing to the fluke mind. Therefore, the violator will be identified sooner or later according to the case (ii).

(v) The violation of multiple distrustful source nodes

Suppose multiple source nodes violate the rules causing congestion and none of them is trustful node. That case is similar to the case (iv), so the violator will be identified ultimately according to the case (ii).

4.2. *Multiple domains*

(i) The violation of one trustful source node

As shown in Fig. 6, there is a trustful source node S_1 communicating with destination D, but it violates the rules that causing several congestion points in multiple domains, so congestion control scheme is triggered. Then all InMs located at the same domain as congestion points will identify the violator. As S_1 is well-reputed, it will send back Res packet carrying acknowledgement information to its InM, and S_2 will send back Res packet carrying clarification information. The responsibility mark is only undertaken by S_1, and it is calculated by the number of congestion points.

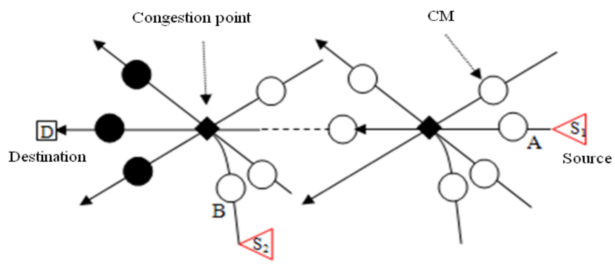

Fig. 6. The violation of a trustful source node in multiple domains.

(ii) The violation of one distrustful source node

Suppose there is a trustful source violating the rules, which causes n congestion points in multiple domains, so n*x node will be treated unjustly, where x is the number of traffic across congestion points. Their responsibility mark y is scaled by the sampling value of Cac. Therefore, the acknowledge responsibility mark of the violator is n*y.

(iii) The violation of multiple trustful source nodes

Suppose multiple trustful source nodes violate the rules. As all source nodes are inter-independent, the interferences among them are very low. Therefore, that case is similar to the case (i).

(iv) The violation of multiple source nodes and only one distrustful source node

Suppose multiple trustful source nodes violate the rules and only one distrustful source node. As all source nodes are inter-independent, the interferences among them are very low. Therefore, that case is similar to the case (ii).

(v) The violation of multiple distrustful source nodes

Suppose multiple distrustful source nodes violate the rules. As all source nodes are inter-independent, the interferences among them are very low. Therefore, that case is similar to the case (ii).

5. Numerical Simulations

According to the effectiveness theory of credibility-based congestion control scheme in section 4, we only need to consider the case having one violator which is enough to validate the feasibility and effectiveness of the proposed congestion control scheme.

5.1. *Simulation topology*

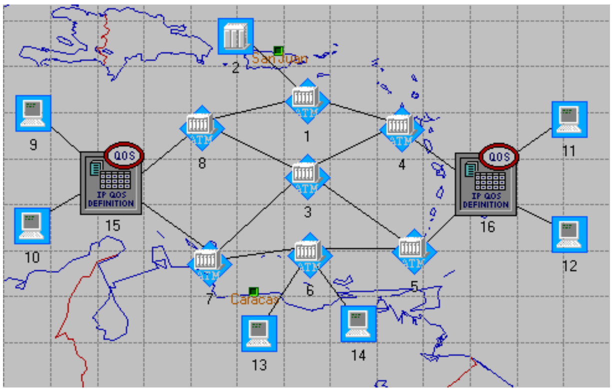

Fig. 7. The simulation network topology.

The simulation network topology is shown in Fig. 7. Node 11 and 12 are source nodes, node 9 and 10 are destinations, node 16 is InM, node 15 is OutM, node 2 is router which collects link information and calculate path for terminal nodes. Suppose node 11 will send packets to node 10, and node 12 will send packets to node 9.

To compare the proposed congestion control scheme with common network, we also run simulation in the comparison topology which substitutes InM and OutM by ordinary switch nodes.

5.2. *Establishment of node model*

There are four kinds of node models in the network including: terminal node, switch node, InM/OutM and router. The terminal node is used to generate traffic and receive traffic and it does not need to modify. The router refers to routing algorithm and some signaling protocols, so it is out of our consideration. Thus, only InM/OutM and switch node model need to design and implement in OPNET [11].

In order to coordinate with the proposed scheme, only the queue module within switch node needs to be modified. The main functions of queue module are: cache packets and schedule.

The processing flowchart of queue is very important for the proposed congestion scheme. As shown in Fig. 8, when the lengths of queue reach 50%, the low priority packet may be dropped to guarantee the success transmission of high priority packet.

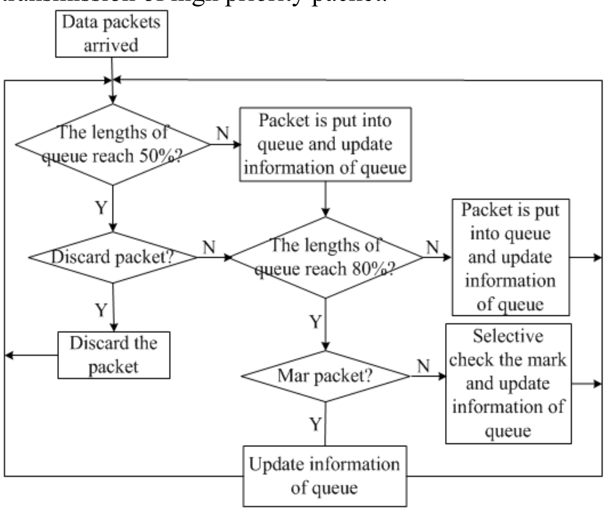

Fig. 8. The flowchart of queue module.

The InM/OutM node located at the edge of domain and edge of domain is implemented by switch node, so the InM/OutM node model should be a switch node model. Their difference is that the CM should be plugged into the InM/OutM. The flowchart of CM is shown in Fig. 9.

5.3. *Validation of the proposed scheme*

In the case that violation of one source node, we set node 12 is the violator, which means that node 12 uses more bandwidth than it applied for, and then generate a congestion point at node 3.

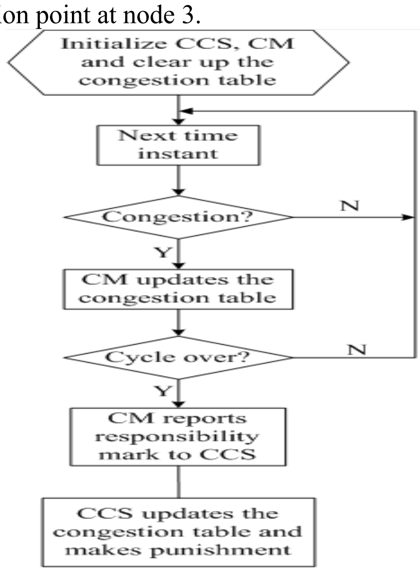

Fig. 9. The flowchart of congestion controller and congestion control server.

The parameter values used in the simulation are set as follows. The simulation time is set to 150s. The node 11 and 12 start sending packet at time 100s. The queue capacity of node 3 is set to 9600bit. If the size of packets sent from node is 1024bit, that is not a violation; if the node send packet with size of 4096bit or above, it will violate the rules. The working period of InM/OutM is set to 150s.

After running the proposed congestion control scheme, if InM/OutM can identify the violator 12, that can prove the proposed scheme is feasible.

The neighbor relationships of OutM 15 and InM 16 are given below. In outM 15, node 9、node 10、node 8 and node 7 are connected by port 1、port 2、port 3 and port 4 respectively. In InM 16, node 4、node 5、node 11 and node 12 are connected by port 1、port 2、port 3 and port 4 respectively.

The simulation results of OutM 15 and InM 16 are shown in Fig. 10 and Fig. 11 respectively.

	port1	port2	port3	port4
Crep	10	7	0	0
Ccl	0	0	0	0
Cac	0	0	0	0
Cim	10	7	0	0
Cdi	0	0	0	0

Fig. 10. The simulation result of node 15.

As shown in Fig. 10, it is seen that only column port 1 and port 2 have non-zero value, because terminal nodes are connected by port 1 and port 2. The OutM 15 received 10 and 7 Crep packets from node 9 and 10 respectively, and also received 10 and 7 Cim packets from node 9 and 10 respectively. The reason is: when congestion happen at node 3, the OutM 15 sends Rep packets based on Mar packet which created by node 3 to terminal node 9 and 10 respectively. Then node 9 and 10 send Imp packets to InM 16. After InM 16 received Imp packets, it sends Dif packets to source node 11 and 12 respectively. Finally, the source node 11 and 12 send Res packets to OutM 15.

	port1	port2	port3	port4
Ccl	0	0	17	0
Cac	0	0	0	17
Cim	0	0	0	0
Cdi	0	0	17	17

Fig. 11. The simulation result of node 16

As shown in Fig. 11, it can be seen that only column port 3 and port 4 have non-zero value, because source nodes are connected by port 3 and port 4. The InM 16 received 17 Ccl and 17 Cdi packets from node 11, and received 17 Cac and Cdi packets from node 12. Apparently, node 11 does not violate rules because InM received clarification information from node 11, but node 12 violates rules causing congestion because InM received acknowledgement information from node 12. It can be concluded that the proposed credibility-based congestion control scheme can identify the violators accurately.

5.4. *Performance Evaluation*

The performance of network is mainly described by packet drop ratio, throughput, transmission delay, transmission jitter, and so on. In this simulation, throughput and end-to-end delay are used to illustrate the effect of the proposed congestion control scheme comparing to the case of without congestion control.

The parameter values used in the simulation are set as follows. The simulation end time is set to 150s. The node 11 and 12 start sending packet at time 10s. The queue capacity of switch nodes is set to 750bit. The length of data packet is set to 30bit. The packet inter-arrival time subjects to Poisson distribution, and its mean value is set to 1s.

(i) Throughput

Network throughput is the average rate of successful packet delivery over a link. Fig. 12 shows the comparison results of throughput under the case of having congestion control and without congestion control. In this simulation, data packets are sent according to Poisson distribution, which means that traffic is added with time elapsing.

It is seen that throughput is increased with traffic increasing, but when traffic reach at some value, throughput reach its maximum value. If additional packets are sent to network continually, throughput is not increased because of congestion. The threshold of queue length is set ahead. When processing time larger than packet inter-arrival time, some packets may be dropped due to the queue reaching its limit. Thus, on account of limit of queue length, when there is heavy traffic on network, throughput tends to balance, but not decreases sharply.

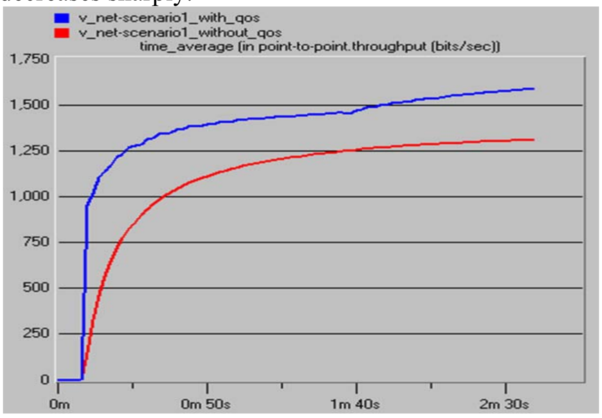

Fig. 12. The comparison result of throughput

Under the condition of light traffic, throughput of network with and without congestion control scheme is almost same. However, the throughput of network without congestion control scheme keeps at lower value than congestion control scheme when the same traffic is employed. The reason is that many packets are dropped due to congestion which causes low throughput, but congestion control scheme can avoid congestion to some extent.

(ii) End-to-end delay

End-to-end delay refers to the time taken for a packet to be transmitted across a network form source to destination. Fig. 13 shows the comparison results of end-to-end delay with and without congestion control.

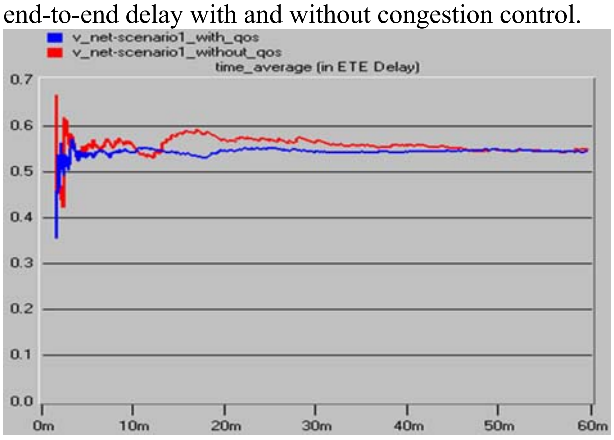

Fig. 13. The comparison result of the end-to-end delay

It is seen that end-to-end delays both tend to balance after jitter period under the two cases. The end-to-end delay equals about 0.55s under the case of without congestion control, whereas that value is about 0.54s with congestion control.

In general, end-to-end delay is composed of transmission delay, propagation delay and queue delay. It is defined as follows.

$$d_{end-end} = N \cdot (d_{trans} + d_{prop} + d_{que}) \quad (7)$$

where N is the number of links.

In our simulation, since the distances between nodes are close and bandwidth of link is 1Gbps, transmission delay and propagation delay can be omitted. Equation (7) can be simplified to

$$d_{end-end} \approx N \cdot d_{que} \quad (8)$$

Therefore, the number of links from source to destination can be obtained by equation (8). The number of links with congestion control equals 0.54/0.09=6, and it can be validated from Fig. 7.

Through above simulations, the proposed credibility-based congestion control scheme is proved that it can work accurately. The performance of throughput and delay can be improved by the proposed scheme.

6. Conclusion

This paper proposed a credibility-based congestion control method at network layer. It divided the whole network into several domains, thus facilitated the question. It relied on sources' self-restrict to avoid congestion, traffic shaping is only needed to be done in ports and great resources would be saved. By collecting and computing responsibility mark of each element, the congestion control server would make decisions according to the mark, and made appropriate punishment to violators. Moreover, the effectiveness of this scheme is analyzed. We provided extensive simulation results to demonstrate that the proposed scheme can process congestion and provided better performance gains (throughput and delay) than without congestion control.

References

1. Scott Shenker, Fundamental design issues for the future internet, *IEEE Journal on Selected Areas in Communications*, 13(7) (1995) 1176–1188.
2. R. J. Gibbens and F. P. Kelly, Resource pricing and the evolution of congestion control, *ELSEVIER Automatica*, 35(12) (1999) 1969–1985.
3. R. Jain, Congestion control in computer networks: issues and trends, *IEEE Network*, 4(3) (1990) 24–30.
4. Shujuan Wang and Mangui Liang, A credibility-based congestion control method, *in Proc. Wicom* (IEEE, Beijing, 2009).
5. Van Jacobson, Congestion avoidance and control, *in Proc. SIGCOMM* (ACM, NY, 1988), pp. 157–187.

6. W. Prue , and J. Postel, Something a host could do with source quench, *RFC-1016*, (Network Working Group, 1987).
7. U. Mukherii, A schedule-based approach for flow-control in data communication networks, *Ph.d. Thesis*, (Massachusetts Institute of Technology, 1986).
8. K. Ramakrishnan and R. Jain, A binary feedback scheme for congestion avoidance in computer networks, *ACM Trans. on Computer Systems*, 18(2) (1990) 158–181.
9. D. Comer and R. Yavatkar, A rate-based congestion avoidance and control scheme for packet switched networks, *in Proc. ICDCS*, (IEEE, 1990), pp. 390–397.
10. A. Bharathan and J. McNair, An OPNET modeler study of the visa mechanism for multi-network authentication, *in Proc. OPNETWORK*, (Washington, D.C., 2003).
11. S. Low and D. E. Lapsley, Optimization flow control I: basic algorithm and convergence, IEEE Trans. Networking, 7(6) (1999) 861-874.