Measurement of IP Packet Flow and Evaluation with VINS

Nanjun Li
Hasso Plattner Institute at University of Potsdam
Potsdam, 14482, Germany
Nanjun.Li@hpi.uni-potsdam.de

***Abstract*—Due to enormous complexity of Internet topology and diversity of nodes' behavior, some important attributes of IP packet flows are not measurable in real world with existing technologies. However, by adequately simulating network nodes' behavior and evaluating their performance, new insights into IP packet flows can be gained. With a brief review on today's Internet architecture and recent development of the Visualized IP-based Network Simulator (VINS), this paper applies VINS to measure IP packet flows in a simulated network. We investigate the causes of packet loop and find the Time-To-Live (TTL) control may not work as expected in some circumstances. A slight modification of TTL control is proposed and tested with VINS, which may strengthen the capability of loop detection in a domain of nodes.**

Keywords- VINS IP Simulation Measurement

I. INTRODUCTION

Each individual node in IP-based networks must be able to handle packets arrive at non-deterministic time, whose source and destination addresses can also be random. A router node's functionalities include packet queuing, Time-to-Live (TTL) control, ICMP error control, routing table lookup and relaying. These nodes shall be organized properly. As a number of IETF RFCs required, the Classful Networks [1] and Classless Inter-Domain Routing (CIDR, [2]) schemes coexist as the architecture of today's Internet. Mis-addressed or mis-subnetted nodes and domains may not be functional as a part of the entire IP network.

However, these important addressing and subnetting schemes are not properly considered in many existing network simulation tools, e.g., NS-2 [3] and OPNET [4]. The Visualized IP-based Network Simulator (VINS) is proposed [5] aimed at validated IP-based network simulation, protocol stack behavioral analysis and performance evaluation. This paper briefly introduces the new progress made in VINS development and applies it in investigating some issues in IP packet flows in a simulated network.

Section 2 introduces packet loop analysis and measurement made in real systems as related work. Section 3 presents a review on Internet architecture required by RFCs. Section 4 introduces the flow level evaluation techniques employed in VINS. Verification is made in section 5 by comparing VINS' simulation results with mathematical expectations in a simple scenario. In section 6 we apply VINS to measure IP packet flows and discuss the causes of loop. In section 7 a new TTL control is proposed and tested with VINS, which may improve the loop detection. We conclude in section 8 and list a plan of future work.

II. BACKGROUND AND CONTRIBUTION

In today's Internet, inter-domain routing widely relies on the Border Gateway Protocol (BGP, [6]). Routers exchange BGP route announcements that consist of a network prefix and a list of nodes with its neighbors to update the new routing states. R. Dube [7] and J. Scudder [8] analyzed the probability of persistent loop that may be caused by a number of routers share the same network mask and prefix. U. Hengartner et al. made an analysis on transient routing loops caused by inconsistencies in routing state among a set of routers [9]. R. Mahajan et al. studied the persistent loops in real ISPs and the influences [10], and found that the "BGP misconfiguration" errors are pervasive in Internet and may cause connectivity disruption.

In this paper VINS is applied to measure some attributes of packet flow that have not been well-measured in real systems or other simulators. We discuss the causes of loop and propose an optimization of TTL control to improve the capability of loop detection in an autonomous domain.

III. MODELING AND ENCODING

Based on 4.4BSD's architecture [11] that the entire TCP/IP stack uses a single queue to enqueue all incoming packets' headers (`ipintrq`, with default capacity to 50), each node can be abstracted as a single-queue system with a routing table and a number of service routines. In VINS, a node is simulated with a software object [5], which is assigned with a class value {HOST, CLASS_A, CLASS_B, CLASS_C} as forwarding prefix length. Complying with the Classful Networking scheme, VINS enables inter-domain network simulation with complex topology.

Let a router node be named as '5' with class B, IP address 12.34.75.23, and its routing table records the neighboring nodes by name: {4, 6, 7}. This node owns a FIFO (First-In-First-Out) queue with capacity as 55, and the service time to each incoming packet has a mean value 25 ms, in Gaussian distributed with deviation 21.100, as shown in Figure 1:

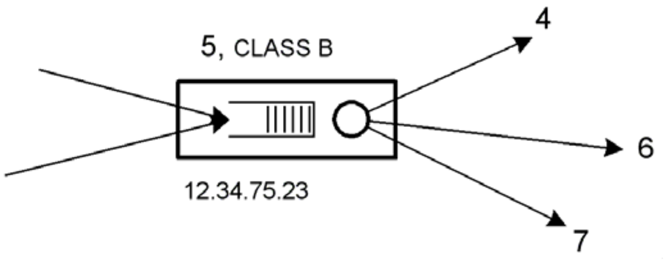


Figure 1. Single Queue Model of Node

This node can be encoded with XML syntax as presented in Figure 2:

```
<Node>
<Name>5</Name><Address>12.34.75.23</Address><
Type>CLASS_B</Type><Pos>433,452</Pos><Capacit
y>55</Capacity><Mean>25</Mean><Dev>21.100</De
v><PDF>GAUSSIAN</PDF>
<RtTab>4,6,7</RtTab>
</Node>
```

Figure 2. Node Encoding Sample

Currently a packet's travel time on a link between two nodes is 0 by default. For long distant links that latencies become considerable, it is recommended to use the following expression to encode the time to different directions (millisecond as unit):

```
<RtTab>4|10,6|15,7|20</RtTab>
```

This expression tells that the link latency from node 5 to node 4 takes 10 ms, to 6 takes 15 ms and to 7 takes 20 ms. When an IP packet goes through Node 5, its TTL is decremented by 1 on the node, and the latency of the link it traverses will be added to its "one trip time" (from its source to destination nodes).

VINS application objects can be mounted over host nodes using UDP or TCP to exchange data. An IP packet flow can be generated either with a pair of applications, or with two fictional nodes: SUPPLIER and CONSUMER. A CONSUMER only receives arrival IP packets without relaying; a SUPPLIER generates IP packets destined to its CONSUMER specified in the scenario file [5]. The interval between two successively generated packets is a stochastic number, distributed following SUPPLIER's PDF (Probability Density Function) option, deviation and mean value. VINS presently support four PDF options:

1) Uniform
2) Gaussian
3) Exponential
4) Deterministic

A router performs Longest Prefix Match (LPM, [12]) lookups to find the best matched neighbor. For a Classful network node, it firstly searches leaf-nodes (hosts) in the routing table for an exact match; if failed, it searches its sub-class nodes, then peer-class and super-class ones, till an adequate node is found to relay, or this packet shall be discarded due to un-deliverable. This routing behavior can be presented with a Petri-net diagram [13] as shown in Figure 3:

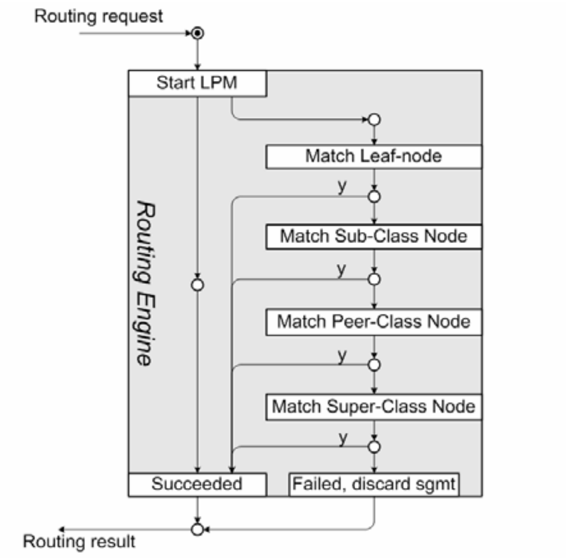


Figure 3. VINS Routing Engine

Figure 4 is a sample of a Classful network:

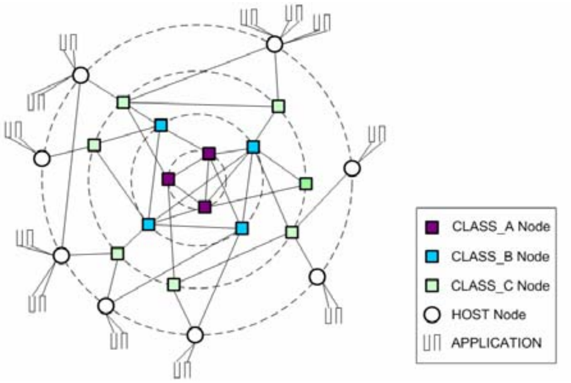


Figure 4. A Classful Network

IV. Flow Level Statistics

The transportation of an IP packet flow is a contribution by the nodes in the route. Packet flows can be isolated from each other if they dwell in exclusive routes, or meet at some nodes and engage in a competition for the services where they meet. As the crossed traffic may increase the queue length on the crossing node, there are two possible aftermaths: 1) the possibility of packet loss is increased; 2) the Mean One Trip Time (MOTT) of a packet is prolonged. To measure the properties of each flow in a network with arbitrary topology, VINS employs a number of counters for flow level statistics:

SENT counts the number of packets of a flow that have been injected into the network.

RCVD counts the number of packets of a flow that have arrived at the destination.

FS stands for Flow Size, counting the number of packets of a flow currently being relayed in the network instantly.

SFS stands for Smoothed Flow Size. As FS changes from observations, user can hardly estimate the average number of packets of this flow being relayed in the network. Instead, SFS might be a useful measurement, calculated as:

$$SFS = \frac{\sum_{i=0}^{N-1} \Delta t_i}{E} \qquad (1)$$

where Δt_i is the lifetime of packet i counted since its creation to vanish (being received, dropped or discarded); E is the Elapsed time of simulation; N is the number of packets of this flow that have been injected into the network.

MOTT is the Mean One-Trip-Time, the average time that a successfully delivered packet takes (unit: ms), calculated as:

$$MOTT = \frac{\sum_{i=0}^{N-1} \delta t_i}{N} \qquad (2)$$

where δt_i is the one trip time of an arrived packet i, N is the number of arrived packets of this flow. MOTT is not applicable to unreachable flows.

V. Mathematical Verification

In this section a sample scenario is built to study three elementary processes: M/D/1/K, M/M/1/K and M/G/1/K [14]. Mean values are calculated with Little's Law [15] and compared with VINS' simulative result, as a mathematical verification to VINS. The scenario can be found in [16] with name *MD1K-MM1K-MG1K.xml*. A screenshot is shown in Figure 5:

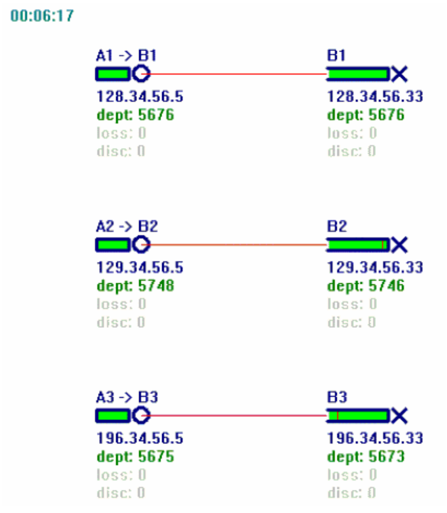
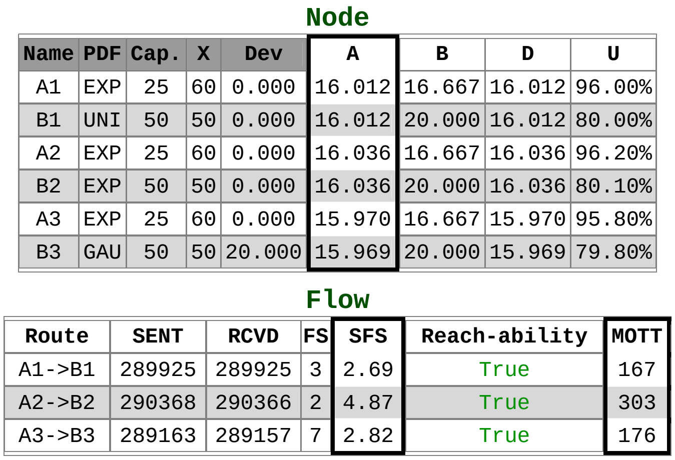


Figure 5. MD1K, MM1K and MG1K

Scenario description: SUPPLIER A1, A2 and A3 generate IP packets destined to B1, B2 and B3. The interval between two successive packets takes Exponential distribution with Mean values as 60 ms. CONSUMER B1 has deterministic service time as 50ms. CONSUMER B2 has Exponential distributed service time with Mean value as 50 ms. CONSUMER B3 takes Gaussian distributed service time with Mean value as 50ms and Deviation as 20.000. Thus, the processes on node C1 is M/D/1/K, on C2 is M/M/1/K and on C3 is M/G/1/K ("G" stands for General, which is set to Gaussian in this sample). Table 1 is partial VINS' system report:

TABLE I. System Report of MD1K-MM1K-MG1K.xml (partial)



Node

Name	PDF	Cap.	X	Dev	A	B	D	U
A1	EXP	25	60	0.000	16.012	16.667	16.012	96.00%
B1	UNI	50	50	0.000	16.012	20.000	16.012	80.00%
A2	EXP	25	60	0.000	16.036	16.667	16.036	96.20%
B2	EXP	50	50	0.000	16.036	20.000	16.036	80.10%
A3	EXP	25	60	0.000	15.970	16.667	15.970	95.80%
B3	GAU	50	50	20.000	15.969	20.000	15.969	79.80%

Flow

Route	SENT	RCVD	FS	SFS	Reach-ability	MOTT
A1->B1	289925	289925	3	2.69	True	167
A2->B2	290368	290366	2	4.87	True	303
A3->B3	289163	289157	7	2.82	True	176

Little's Law [15] tells that in a queuing system, the mean queue length N equals to system's mean response time R times its arrival rate A:

$$\mathbf{N = R \cdot A} \qquad (3)$$

Apply Little's Law in this sample: each flow's SFS equals to its CONSUMER's mean queue length N; flow's MOTT equals to its CONSUMER's response time R; each node's arrival rate A is printed in the column A. The LL's expectations of N on B1, B2 and B2 are:

$$N_{B1} = R_{B1} \cdot A_{B1} = 0.167 * 16.012 \approx 2.67 \text{ (pkts)}$$
$$N_{B2} = R_{B2} \cdot A_{B2} = 0.303 * 16.036 \approx 4.86 \text{ (pkts)}$$
$$N_{B3} = R_{B3} \cdot A_{B3} = 0.176 * 15.969 \approx 2.81 \text{ (pkts)}$$

Comparing LL's expectations on N with the values in column SFS {2.69, 4.87, 2.82}, the deviation has been controlled within 1%.

VI. Measurement of Flows

In this section a scenario (*flyingdutchman.xml* at [16]) is built as an example to study some properties of IP packet flows. We focus on the unreachable flows, especially the loop ones. A discussion is made on the so-called "misconfiguration", which shall be accounted for routers' limited knowledge (routing table) about the entire network's topology.

This scenario consists of 3 top-class (CLASS_B) routers G, M and T with same forwarding prefix 128.34.xx.xx. Their routing tables are:

G:	[C,H,D,T,M]
M:	[G,N,T,L]
T:	[M,G,H,C]

SUPPLIER nodes (K, S, A and B) initialize TTL of packets as 64 by default. The two isolated node R and Q are deliberately designed to let two flows be unreachable: `B->Q` and `S->R`. Figure 6 is a screenshot, in which the loop one is highlighted with moving bullets:

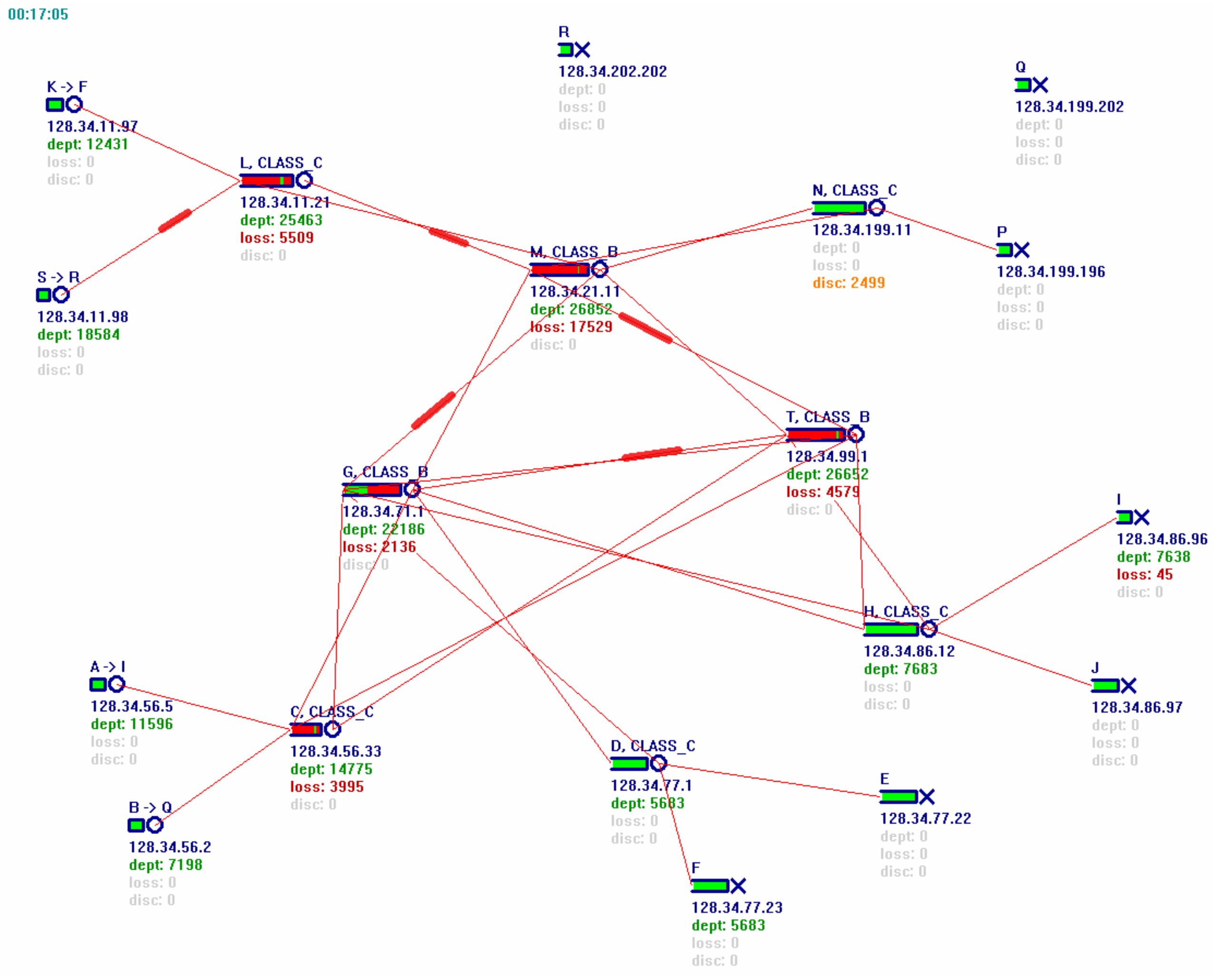


Figure 6. Scenario of Flyingdutchman.xml

A significant difference between two unreachable flows is that packets in B->Q are detected to be unreachable and thus discarded on node N; while S->R is trapped in a closed virtual circuit made up by G, M and T, and no one of them is aware of looping. Packets in S->R keep consuming network's service, slow down its performance and increase packet loss. From the screenshot we can find they are flooded out instead of being discarded as TTL expires. Table 2 is scenario's system report:

TABLE II. SYSTEM REPORT, TTL=64

Route	SENT	RCVD	FS	SFS	Reach-ability	MOTT
A->C->T->H->I	23057	16184	30	31.95	True	3478
B->C->T->M->N	14023	0	48	20.96	False	--
K->L->M->G->D->F	24498	12166	57	38.52	True	4758
S->L->M->G->T->M	39196	0	120	116.19	False	--

The delivery percentage (RCVD/SENT) of two reachable flows are:

A->I: 16184 / 23057 ≈ 70.20%

K->F: 12166 / 24498 ≈ 49.66%

A VINS Net-Pie illustrates the SFS of each packet flow and the network resource they possess:

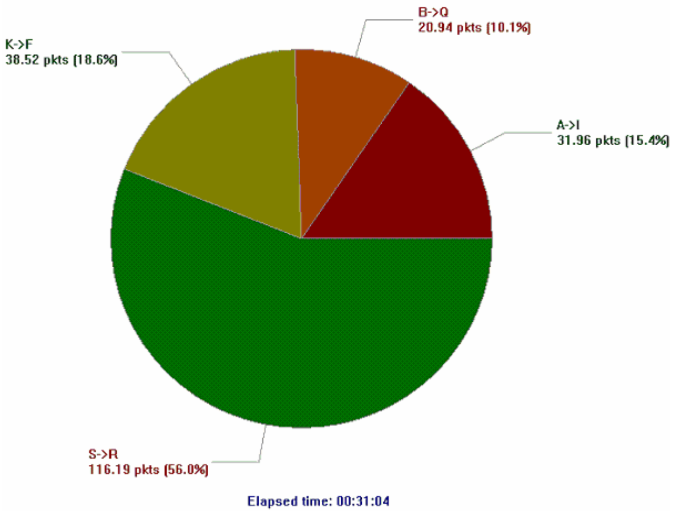


Figure 7. Net-Pie of Flyingdutchman.xml

In this sample the loop complies with the description in [6]: a number of routers share the same network mask and prefix. In advance, two conclusions can be made:

1. Packet loop can take chance in a domain if it consists of more than one top-class routers. The unreachable packets

might be relayed among these routers without being detected to be looping;

2. The long default TTL (64) may not work as expected in congested networks, because a looping packet might be flooded out before its TTL expires.

At the meantime, forwarding among peer-class nodes is indispensable for providing shortcuts to the packets. To detect the loop flows efficiently and reduce network's burden, an enhanced TTL control might be expected.

VII. NEW TTL CONTROL

Networks with multiple top-class routers are subject to an attack when remote machine(s) deliberately sending packets destined to a non-existing node, e.g., using UDP programs to flood. An improvement of TTL control can be made on the gateway nodes of a domain without changing current format of IP header:

1. for a packet going out of the domain, stamp its TTL as 64 or 256 so that it can go enough long distance (number of intermediate hops) before being discarded due to TTL expires;
2. for a packet coming in the domain, stamp its TTL to a small number (e.g., 4 or 8) according to longest non-loop route in this domain, so that the loop one can be efficiently discarded.

Figure 8 illustrates this idea: let G1 and G2 be gateways of a domain (an Autonomous System, AS). Outgoing packets' TTL are set to 64 while incoming ones are set to 4:

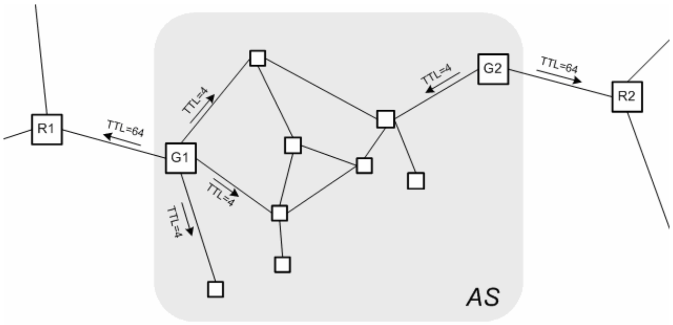


Figure 8. New TTL Control

A test of this idea is made in the same scenario as figure 6. Let K, S, A and B be gateways. TTL of incoming packets are set to 6. Figure 9 is a screenshot, which shows node G is able to discard the loop packets in flow `S->R`:

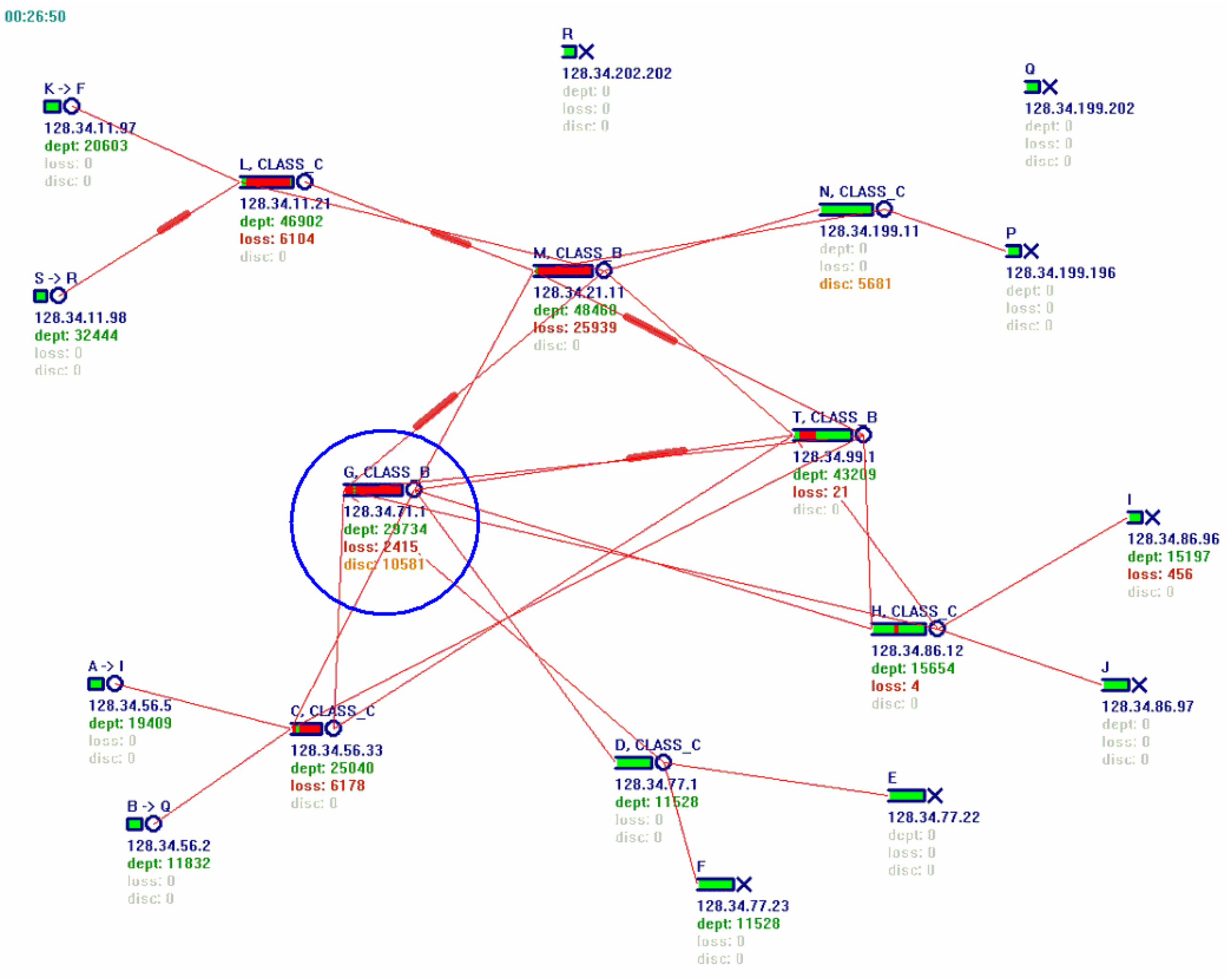


Figure 9. Scenario of Flyingdutchman.xml with New TTL Control

TABLE III. SYSTEM REPORT, NEW TTL CONTROL

Route	SENT	RCVD	FS	SFS	Reach-ability	MOTT
A->C->T->H->I	21771	17109	43	30.75	True	3179
B->C->T->M->N	13316	0	55	15.82	False	--
K->L->M->G->D->F	23290	12993	88	41.08	True	4834
S->L->M->G->T->M	36560	0	92	87.48	False	--

System report shows the delivery percentage of two reachable flows have been promoted:

A->I: 17109 / 21771 ≈ 78.59% (from 70.20%)
K->F: 12993 / 23290 ≈ 55.79% (from 49.66%)

Figure 10 is the Net-Pie under new TTL control, which shows the SFS of unreachable flow S->R is reduced from 116.19 pkts (56% of used network resource) to 87.7 pkts (49.9% of used network resource):

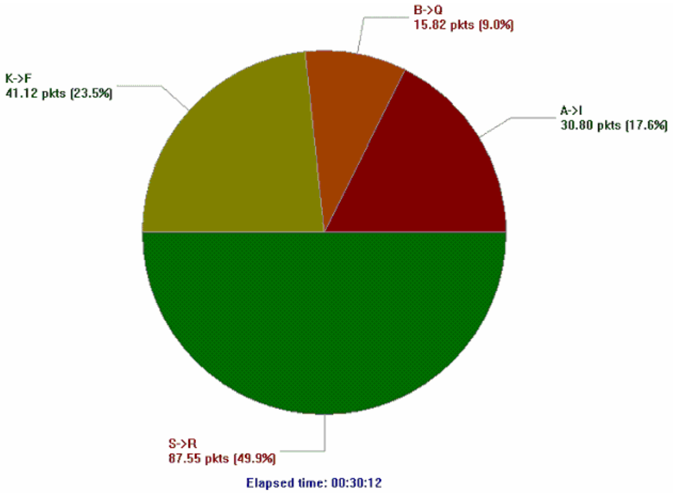


Figure 10. Net-Pie of Flyingdutchman.xml with New TTL Control

VIII. CONCLUSION AND FUTURE WORK

In this paper we apply VINS to study and measure IP packet flows. We investigate the causes of packet loop and conclude that the loops shall be accounted for the limitation of nodes' knowledge of the entire network's topology, instead of "misconfiguration". Packet loops may occur in a domain where there are more than one top-level routers coexist.

A new TTL control mechanism is proposed to protect a domain of nodes from being flooded by deliberately made or unintentional loop flows. This mechanism requires the recompilation of kernel (ip_input in 4.4BSD), and knowledge of the depth of the domain to be protected. We test this idea in a simulated network and it proves the network performance can be improved.

VINS development focuses on the following tasks: 1) re-implementation of the transport layer modules, especially to support more TCP variants; 2) more queue scheduling techniques, such as Random Early Detection (RED) [17]; 3) encoding time-event to support dynamic network topology and node status.

ACKNOWLEDGMENT

The author would like to thank Werner Zorn for his full support in the development of VINS, and TridentCom 2007's anonymous reviewers for their helpful comments.

REFERENCES

[1] Jon Postel, "Internet Protocol (Classful Networks)", RFC 791, 1981

[2] V. Fuller, "Classless Inter-Domain Routing, CIDR: an Address Assignment and Aggregation Strategy", RFC 1519, 1993

[3] Kevin Fall and Kannan Varadhan, "The ns Manual", April 14, 2002. NS-2 Home: http://www.isi.edu/nsnam/sn/

[4] OPNET home: http://www.opnet.com

[5] Nanjun Li, "Node-Oriented Modeling and Simulation of IP Networks", In proceedings of 14th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems, 2007

[6] Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP-4)". RFC 1771

[7] Rohit Dube, "A Comparison of Scaling Techniques for BGP", ACM SIGCOMM Computer Communication Review Volume 29 , Issue 3, July 1999

[8] John G. Scudder and Rohit Dube, "BGP Scaling Techniques Revisited", ACM SIGCOMM Computer Communication Review, 1999

[9] Urs Hengartner, Sue Moon, Richard Mortier, Christophe Diot, "Detection and analysis of routing loops in packet traces", 2nd ACM SIGCOMM Workshop on Internet measurement, 2002

[10] R. Mahajan, D. Wetherall, and T. Anderson. "Understanding bgp misconfiguration", in Proceedings of SIGCOMM, Pittsburgh, PA, August 2002.

[11] G. W. Wright and W. R. Stevens, "TCP/IP Illustrated Volume II - The Implementation", Addison Wesley, 1994

[12] A. S. Tanenbaum, "Computer Networks", third edition, Prentice Hall PTR, 1996

[13] Petri, C.A., "Kommunikation mit Automaten", Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2, 1962, Second Edition:, New York: Griffiss Air Force Base, Technical Report RADC-TR-65--377, Vol.1, 1966, Pages: Suppl. 1, English translation

[14] D. A. Menascé, V. A. F. Almeida and L. W. Dowdy, "Performance by Design: Computer Capacity Planning by Example", Prentice Hall, 2004, ISBN 0-13-090673-5

[15] Little, J. D. C, "A Proof of the Queuing Formula L = λ W", Operations Research, 9, 383-387, 1961

[16] VINS download package available: https://www-fgks.hpi.uni-potsdam.de/fileadmin/user_upload/Nanjun/VINS.zip

[17] S. Floyd., and Jacobson, V., "Random Early Detection gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking 1(4) 397-413, 1993