AUTOMATIC TEST PACKET GENERATION IN NETWORK

Dipty Trimbak Survase[1], Aparna Prakash Chandgude[2], Surekha Babasaheb Kerulkar[3], Pranali Prakash Vichare[4], Mohan V. Pawar[5]

[1]*Computer Engineering, JSCOE, Maharashtra, India*
[2]*Computer Engineering, JSPMCOE, Maharashtra, India*
[3]*Computer Engineering, JSCOE, Maharashtra, India*
[4]*Computer Engineering, JSCOE, Maharashtra, India*
[5]*Faculty of Computer Engineering, JSCOE, Maharashtra, India*

Abstract

Now a day's we see that networks are widely distributed so administrators depends on various tools such as ping and traceroute to rectify the problem in the network. We proposed an automated and systematic approach for testing and debugging network called "Automatic Test Packet Generation"(ATPG). Initially ATPG reads router configuration and then generates a model which is device freelance. The model is used to generate the minimum number of test packets to cover every link and rule in network. ATPG is capable for detecting both functional and performance problems. Test packets are sent at regular intervals and special technique is used to localize faults.

Keywords: *Test Packet Generation Algorithm; Network Troubleshooting; Data Plane Analysis.*

----------***----------

1. INTRODUCTION

It is not an easy task to debug a network. The network engineers face problems like router misconfigurations, Fiber cut, mislabeled cables, software bug, Faulty interfaces etc. Network engineers try to solve these issues using mostly used tools such as ping and trace route. Debugging networks is getting more and more complex as not only size of networks but also their level of complexity [32] is increasing day by day. ATPG produces a model which is not dependent on devices after reading configuration from routers. Another advantage of ATPG system is that it covers each link and every rule in network with minimum number of test packets. Uniformly the test packets are send, and if any fault is detected, it is triggered by separate mechanism namely fault localization. ATPG can cover both functional and performance fault.

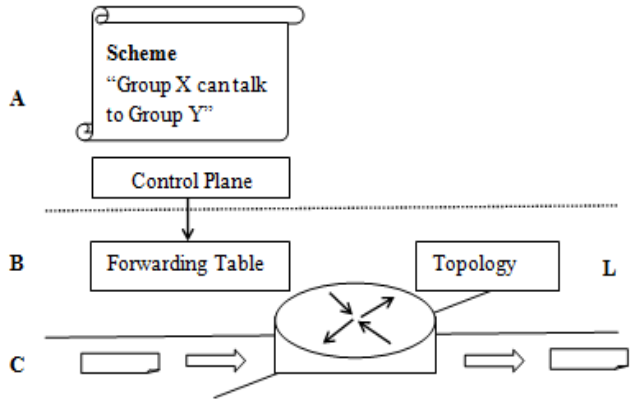
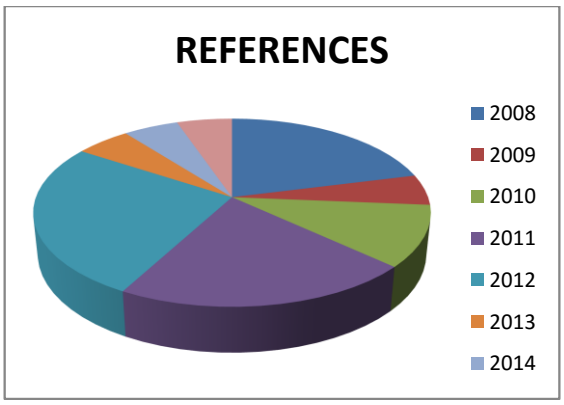


Fig1:- Network State

2. Table No: 1

YEAR OF PAPER	*REFERNCES*
2008	A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, Y.Ganjali, and C. Diot[26], N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner[27], A. Mahimkar, J. Yates, Y. Zhang, A. Shaikh, J.Wang, Z. Ge, and C. T. Ee[24], C. Cadar, D. Dunbar, and D. Engler[6]
2009	F. Le, S. Lee, T. Wong, H. S. Kim, and D. Newcomb[21]
2010	D. Turner, K. Levchenko, A. C. Snoeren, and S. Savage[34], B. Lantz, B. Heller, and N. McKeown[20]
2011	H. Mai, A. Khurshid, R. Agarwal, M. Caesar, P. B. Godfrey, and S. T.King[25], S. Shenker[32], A. Mahimkar, Z. Ge, J. Wang, J. Yates, Y. Zhang, J. Emmons, B.Huntley, and M. Stockert[23], P. Gill, N. Jain, and N. Nagappan[12]
2012	M.Reitblatt,N.Foster, J. Rexford, C. Schlesinger, andD.Walker[31], M. Kuzniar, P. Peresini, M. Canini, D. Venzano, and D. Kostic[18], P. Kazemian, G. Varghese, and N. McKeown[16], M. Canini,D.Venzano, P. Peresini,D.Kostic, and J. Rexford[7],
2013	*I.Pomeranz,S.M.Reddy,J.Rajski,Kwang-Ting Cheng,J.A.Abraham[3]*
2014	Hongyi Zeng ,Peyman Kazemian, George Varghese, *ACM* and Nick McKeown*[2]*
2015	Dipty survase,Pranali Vichare,Mohan V. Pawar[1]



3. ALGORITHMS AND TECHNIQUES USED

1) Algorithm: we have a tendency to assume a collection of take a look at terminals within the network will send and receive take a look at packets. Our goal is to come up with a collection of take a look at packets to exercise each rule each switch perform, in order that any fault are determined by a minimum of one take a look at packet. This can be analogous to software package take a look at suites that attempt to take a look at each potential branch in a very program. The broader goal will be restricted to testing each link or each queue. Once generating take a look at packets, ATPG should respect 2 key constraints: a) Port: ATPG should solely use take a look at terminals that square measure available;

b) Header: ATPG should solely use headers that every take a look at terminal is allowable to transfer. for instance, the network administrator might solely enable employing a specific set of VLANs.

2) Algorithm: Our algorithmic program for pinpointing faulty rules assumes that a take a look at packet can succeed as long as it succeeds at each hop. For intuition, a ping succeeds provided that all the forwarding rules on the trail behave properly. In a same way, if a queue is full, associate degreey packets that travel through it'll incur higher latency and will fail an end-to-end take a look at. Formally, we've got the subsequent. Assumption one (Fault Propagation)R(pk)=1: if and as long as For all r Є(p,k).history,R(r,k)=1, ATPG is use to pinpoint a faulty rule by initial computing the stripped set of probably faulty rules. Formally, we've got drawback a pair of. drawback a pair of (Fault Localization): Given an inventory of (pk0,(R(pk0),(pk1,R(pk1)….tuples, notice all r that satisfies ∃pki,R(pki,r)=0.

4. ATPG SYSTEM

ATPG is one in every of the systematic approach uses for debugging network. ATPG generates stripped variety of take a look at packets in order that each forwarding rule the network is exercised and coated by a minimum of 1 take a look at packet.A fault localization algorithmic program is use by ATPG to see the failing rules or failing links.

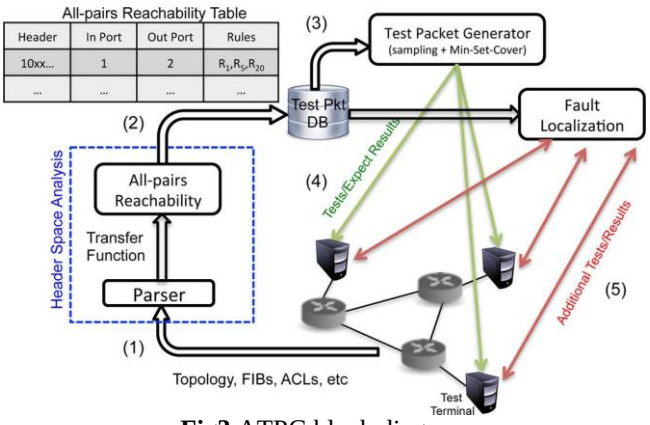
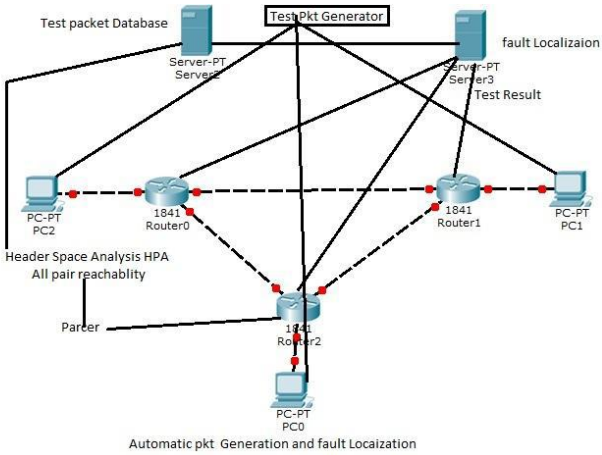


Fig3.ATPG block diagram

5. PROPOSED SYSTEM

Automatic take a look at Packet Generation (ATPG) structure that consequently produces a negligible arrangement of bundles to check the basic's livener's topology and also the coinciding between data plane state and style determinations. The equipment will likewise naturally produce bundles to check execution affirmations, for instance, parcel dormancy. It will likewise be specific to provide a negligible arrangement of parcels that solely take a look at every association for system liveners.

- A survey of network operators revealing common failures and route causes take a look at packet generation algorithmic program.
- A fault localization algorithmic program to isolate faulty device and rules.
- ATPG use cases for useful & performance testing
- Evaluation of an example ATPG system victimisation rule sets collected from the Stanford and internet2 backbones.

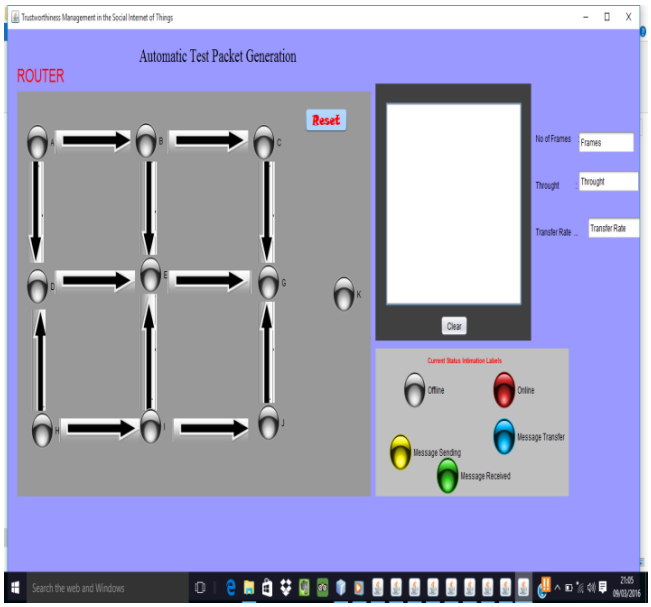


6. FUTURE SCOPE AND RELATED WORK

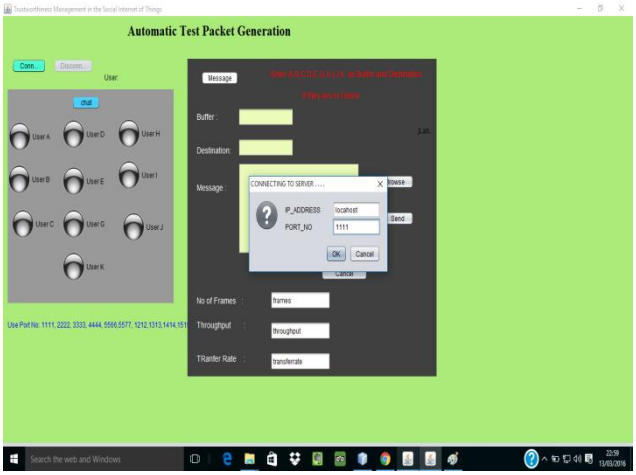
Many of the newest techniques used for mechanically generating take a look at packets square measure given. nearest technologies illustrious square measure few offline tools. These offline tools square measure use to envision invarients in network. It additionally supports Automatic take a look at Packet Generation one in every of the logged

off utensils that square measure utilised for modify take a look at parcels automatically up to the mark plane is nice. Header house analysis [16] uses geometric model to envision reachability, discover loops and verify slicing. SOFT was projected [18] to verify consistency between completely different openflow agent implementations that square measure to blame for bridging management and information planes in SDN context. in service with discharge stream arrangement somebody should be urged to impact difficulties like expansive place of switch state, large place of elbow grease bundle, tremendous flexibility of occasion requesting and then on to beat these difficulties NICE is of unimaginable utilization. operating of NICE is incontestible. good somebody brings to the table controller program close topology of framework that joins state of switches and hosts. The somebody has the independence to connect inquiry approach that is needed by him. At long last NICE [7]offers the hints of benefits contradiction or property to be up to the imprint with their evidences as yield.The instrument NICE chips away at high of things plane equally within the data plane there is another disconnected from cyber web equipment which will be utilised significantly Anteater[25]. Insect consumption animal accumulates the setup and causation data bases (FIBs) of methodology, and depict them as scientist capacities. At that time a screw up to be checked is set by administrator against the system, such lapses are consistency of causation tenets among switches[12], reachability or circle free forward. Insect consumption animal makes the mix of those slips and proselytes them into tests of scientist satisfiability downside (SAT), and makes utilization of a weekday issue problem solver to execute study.Actually our work is expounded to figure in programming languages and symbolic debugging.

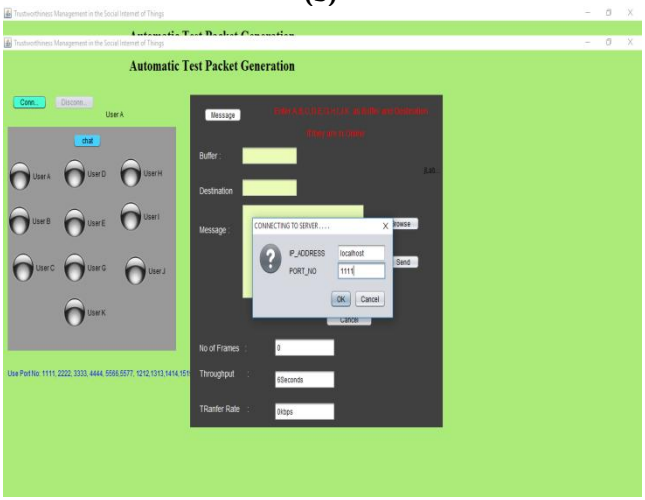
7. RESULT



(a)



(b)



(c)

CONCLUSION

Nowadays network engineers depend upon recent tools like ping and traceroute to right the system. However networks are becoming larger and additional advanced, so that they would like additional refined instrument for right system. because of vast network access administrator face some problems in testing animateness of system. to beat such variety of problems we have a tendency to developed ATPG. By testing all tips comprehensive in any respect drop rules ATPG has capability to check reachability methodology. ATPG employments easy issue restriction strategy developed with the help of header house investigation to confine deficiencies. Customary model of ATPG framework serves to hide most extreme connections or standards in a very system with least variety of take a look at bundles.

REFERENCES

[1]. A Survey on Automatic Test Packet Generation in Network.Ms.Survase Dipty1,Ms.Pranali Vichare2, Prof.M.V.Pawar3[2015].

[2]. "Automatic Test Packet Generation",Hongyi Zeng, Member, IEEE, Peyman Kazemian, Member, IEEE, George Varghese, Member, IEEE, Fellow, ACM, and Nick McKeown, Fellow, IEEE, ACM,2014.

[3]. "Automatic Test Pattern Generation," 2013 [Online]. Available,http://en.wikipedia.org/wiki/Automatic_test_pattern_generation.
[4]. "Beacon," [Online]. Available: http://www.beaconcontroller.net/
[5]. Y. Bejerano and R. Rastogi, "Robust monitoring of link delays and faults in IP networks," IEEE/ACM Trans. Netw., vol. 14, no. 5, pp.1092–1103, Oct. 2006.
[6]. C. Cadar, D. Dunbar, and D. Engler, "Klee: Unassisted and Automatic generation of high-coverage tests for complex systems Programs," in Proc. OSDI, Berkeley, CA, USA, 2008, pp. 209–224.
[7]. M. Canini, D.Venzano, P. Peresini, D.Kostic, and J. Rexford, "A NICE Way to test OpenFlow applications," in Proc. NSDI, 2012, pp. 10–10.
[8]. Dhamdhere, R. Teixeira, C. Dovrolis, and C. Diot, "Netdiagnoser: Troubleshooting network unreachabilities using end-to-end probes and routing data," in Proc. ACM CoNEXT, 2007, pp. 18:1 18:12.
[9]. P. Yalagandula, P. Sharma, S. Banerjee, S. Basu, and S.-J. Lee, "S3: A scalable sensing service for monitoring large networked systems," in Proc. INM, 2006, pp. 71–76.
[10]. N. Duffield, F. L. Presti, V. Paxson, and D. Towsley, "Inferring link loss using striped unicast probes," in Proc. IEEE INFOCOM, 2001,vol. 2, pp. 915–923.
[11]. N. G. Duffield and M. Grossglauser, "Trajectory sampling for direct traffic observation," IEEE/ACM Trans. Netw., vol. 9, no. 3, pp. 280–292, Jun. 2001.
[12]. P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: Measurement, analysis, and implications," in Proc. ACM SIGCOMM, 2011, pp. 350–361.
[13]. "Hassel, the Header Space Library," [Online]. Available: https://bitbucket. Org/peymank/hassel-public.
[14]. Internet2, Ann Arbor, MI, USA, "The Internet2 observatory data collections," [Online]. Available: http://www.internet2.edu/observatory/Archive/data-collections.html.
[15]. M. Jain and C. Dovrolis, "End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput," IEEE/ACM Trans. Netw., vol. 11, no. 4, pp. 537–549, Aug. 2003.
[16]. [16] P. Kazemian, G. Varghese, and N. McKeown, "Header space analysis: Static checking for networks," in Proc. NSDI, 2012, pp. 9–9.
[17]. [17] R. R. Kompella, J. Yates, A. Greenberg, and A. C. Snoeren, "IP fault localization via risk modeling," in Proc. NSDI, Berkeley, CA, USA,2005, vol. 2, pp. 57–70.
[18]. M. Kuzniar, P. Peresini, M. Canini, D. Venzano, and D. Kostic, "A SOFT way for OpenFlow switch interoperability testing," in Proc.ACM CoNEXT, 2012, pp. 265–276.
[19]. K. Lai and M. Baker, "Nettimer: A tool for measuring bottleneck link, bandwidth," in Proc. USITS, Berkeley, CA, USA, 2001, vol. 3, pp.11–11.
[20]. B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in Proc. Hotnets, 2010, pp. 19:1–19:6.
[21]. F. Le, S. Lee, T. Wong, H. S. Kim, and D. Newcomb, "Detecting network-wide and router-specific misconfigurations through data mining," IEEE/ACM Trans. Netw., vol. 17, no. 1, pp. 66–79, Feb.2009.
[22]. H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "iplane: An information plane for distributed services," in Proc. OSDI, Berkeley, CA, USA, 2006, pp. 367–380.
[23]. Mahimkar, Z. Ge, J. Wang, J. Yates, Y. Zhang, J. Emmons, B. Huntley, and M. Stockert, "Rapid detection of maintenance induced Changes in service performance," in Proc. ACM CoNEXT, 2011, pp.13:1–13:12.
[24]. Mahimkar, J. Yates, Y. Zhang, A. Shaikh, J.Wang, Z. Ge, and C. T. Ee, "Troubleshooting chronic conditions in large IP networks," in Proc. ACM CoNEXT, 2008, pp. 2:1–2:12.
[25]. H.Mai, A. Khurshid, R. Agarwal, M. Caesar, P. B. Godfrey, and S. T. King, "Debugging the data plane with Anteater," Comput.Commun. Rev., vol. 41, no. 4, pp. 290–301, Aug. 2011.
[26]. A.Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, Y. Ganjali, and C. Diot, "Characterization of failures in an operational ip backbone network," IEEE/ACM Trans. Netw., vol. 16, no. 4, pp. 749–762, Aug. 2008.
[27]. N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," Comput.Commun. Rev., vol. 38, pp. 69–74, Mar. 2008.
[28]. "OnTimeMeasure," [Online]. Available: http://ontime.oar.net.
[29]. "Open vSwitch," [Online]. Available: http://openvswitch.org/.
[30]. H. Weatherspoon, "All-pairs ping service for PlanetLab ceased," 2005 [Online]. Available: http://lists.planet lab.org/pipermail/users/2005 July/001518.html
[31]. M.Reitblatt, N.Foster, J. Rexford, C. Schlesinger, and D.Walker, "Abstractions for network update," in Proc. ACM SIGCOMM, 2012, pp. 323–334.
[32]. S.Shenker, "The future of networking, and the past of protocols," 2011 [Online].Available: http://opennetsummit.org/archives/oct11/shenkertue.
[33]. "Troubleshooting the network survey," 2012 [Online]. Available: http://eastzone.github.com/atpg/docs/NetDebugSurvey.pdf
[34]. D. Turner, K. Levchenko, A. C. Snoeren, and S. Savage, "California fault lines: Understanding the causes and impact of network failures," Comput.Commun. Rev., vol. 41, no. 4, pp. 315–326, Aug. 2010.