International Journal of Research

International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 09, September 2015
Available at http://internationaljournalofresearch.org

Implementation of Network monitoring system using Instinctive Check Packet Generation

Hari Kishan Chapala[#]& Murali Gudipati[*],
# PG Scholar, Dept of CSE, St.Mary's group of Institutions, Chebrol, AP
*Principal, St.Mary's group of Institutions, Chebrol, AP

ABSTRACT

Nowadays networks are growing wide and more complex. However administrators use tools like ping and trace route to debug problems. Hence we implemented an automatic and Methodical approach for testing and debugging networks called Automatic Test Packet Generation (ATPG)[1]. This approach gets router configurations and generates a device-independent model. ATPG generate a few set of test packets to find every link in the network. Test packets are forwarded frequently and it detect failures to localize the fault. ATPG can detect both functional and performance (throughput, latency) problems. We found, less number of test packets is enough to test all rules in networks. For example, 4000 packets can cover all rules in Stanford backbone network, while 53 are much enough to cover all links. We implemented an automated and systematic approach for testing and debugging networks by name "Instinctive Check Packet Generation" (ICPG). ICPG reads router configurations and generates a device-independent model. The model is used to generate a minimum set of test packets to (minimally) exercise every link in the network or (maximally) exercise every rule in the network. Test packets are sent periodically, and detected failures trigger a separate mechanism to localize the fault. ICPG can detect both functional (e.g., incorrect firewall rule) and performance problems (e.g., congested queue). ICPG complements but goes beyond earlier work in static checking (which cannot detect liveness or performance faults) or fault localization (which only localize faults given liveness results).

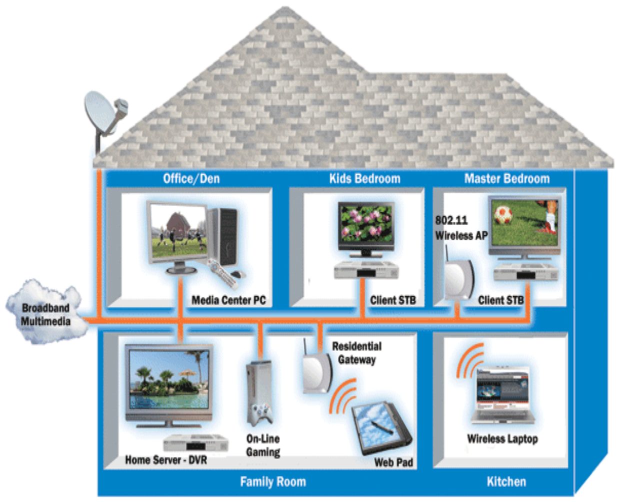
Keywords: Fault Localization; Test Packet Selection; Network Debugging; Automatic Test packet Generation; "Instinctive Check Packet Generation" (ICPG)

I. INTRODUCTION

Whenever networking comes into picture, questions that we come across are about "How to secure your network? Is my network secure? What do I need to do make network secure?" But network security does not limit only by implementing new firewall optimizing techniques or to secure the information, rather it also includes monitoring the packets, forwarding entries etc. Now, this would raise the question of how this would help to secure the network. The answer to this is, the security could be easily breached by tampering the rules and exploiting the errors. Until now it is the network administrator's problem to tackle with such issues. Troubleshooting a network is difficult for three reasons. First, the forwarding state is distributed across multiple routers and firewalls and is defined by their forwarding tables, filter rules, and other configuration parameters [2]. Second, the forwarding state is hard to observe because it typically requires manually logging into every box in the network. Third, there are many different programs, protocols, and humans updating the forwarding state simultaneously.

Characteristics of Networking The following characteristics ought to be thought of in network style and in progress maintenance:

- **Availability** is often measured in a very share supported the amount of minutes that exist in a very year. Therefore, period of time would be the amount of minutes the network is out there divided by the amount of minutes in a very year.



- **Cost** includes the price of the network elements, their installation, and maintenance.
- **Reliability** defines the responsibility of the network elements and therefore the property between them. Mean solar time between failures (MTBF) is often accustomed live responsibility.
- **Security** includes the protection of the network elements and therefore the information they contain and/or the info transmitted between them.
- **Speed** includes how briskly information is transmitted between network finish points (the information rate) [3].
- **Scalability** defines however well the network will adapt to new growth, together with new users, applications, and network elements.

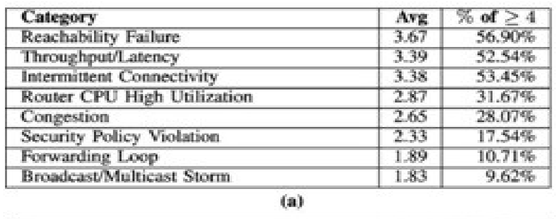
Advantages of Networking:

- **Simple Communication:** It is terribly simple to speak through a network. Individuals will communicate expeditiously employing a network with a bunch of individuals. They will relish the good thing about emails, instant electronic communication, telephony, video conferencing, chat rooms, etc.
- **Ability to Share Files, information and Information:** This is one amongst the key blessings of networking computers. Individuals will realize and share info and information owing to networking. This can be helpful for big organizations to keep up their information in associate organized manner and facilitate access for desired individuals [5].
- **Sharing Hardware:** Another vital advantage of networking is that the ability to share hardware. For example, a printer is often shared among the users in a network so there is no need to have individual printers for each and every pc within the company. This can considerably cut back the price of buying hardware.
- **Sharing Software:** Users will share package inside the network simply. Networkable versions of package square measure accessible at significant savings

compared to separately licensed version of constant package. So giant corporations will cut back the price of shopping for package by networking their computers.

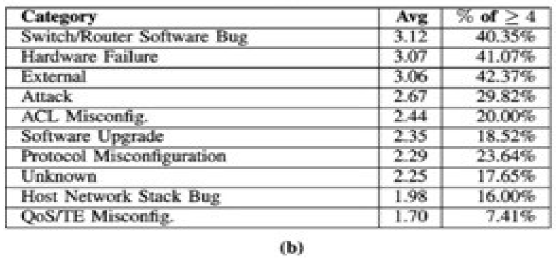
- **Security**: Sensitive files and programs on a network are often Arcanum protected. Then those files will solely be accessed by the approved users. This can be another vital advantage of networking once there square measure issues concerning security problems. Conjointly every and each user has their own set of privileges to forestall those accessing restricted files and programs.
- **Speed:** Sharing and transferring files inside networks is extremely speedy, looking on the sort of network. This can save time whereas maintaining the integrity of files [4, 6].

- The following was the study of Network failures



Category	Avg	% of ≥ 4
Reachability Failure	3.67	56.90%
Throughput/Latency	3.39	52.54%
Intermittent Connectivity	3.38	53.45%
Router CPU High Utilization	2.87	31.67%
Congestion	2.65	28.07%
Security Policy Violation	2.33	17.54%
Forwarding Loop	1.89	10.71%
Broadcast/Multicast Storm	1.83	9.62%

(a)



Category	Avg	% of ≥ 4
Switch/Router Software Bug	3.12	40.35%
Hardware Failure	3.07	41.07%
External	3.06	42.37%
Attack	2.67	29.82%
ACL Misconfig.	2.44	20.00%
Software Upgrade	2.35	18.52%
Protocol Misconfiguration	2.29	23.64%
Unknown	2.25	17.65%
Host Network Stack Bug	1.98	16.00%
QoS/TE Misconfig.	1.70	7.41%

(b)

a. Symptoms of Network failure
b. Causes of Network failure

II. RELATED WORK

Testing animateness of a network could be an elementary downside for ISPs and enormous knowledge center operators. Causation probes between each combine of edge ports are neither thoroughgoing nor climbable. It suffices to seek out a lowest set of end-to-end packets that traverse every link. However, doing this needs the way of abstracting across device specific configuration files, generating headers and also the links they reach, and at last deciding a minimum set of check packets (Min-Set-Cover) to check implementing consistency between policy and also the configuration [7].

1. Packets: A packet is taken by number of the packet's places in the network at any time and locations instant; every physical node in the network is assigned a same id number.

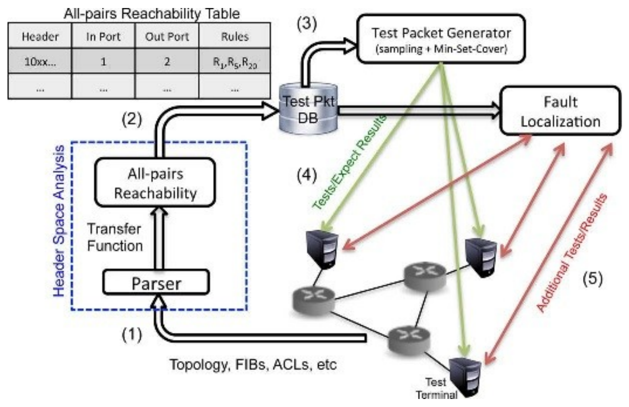
2. Switches: A switch transfer function models is taken a network device such as a switch Each network device contains a set of forwarding conditions that determine packets are processed An arriving packet is associated with different rule by matching it against each rule in descending order in different locations and is dropped if no rule matches in different models [8].

3. Rules: A rule generates a list of one or more output packets corresponding to the output port to which the packet is sent and defines how packet fields are changed the rule abstraction models all real world conditions we know including IP forwarding and ACLs Essentially a rule defines how a region of header space at the ingress is transformed into regions of header space at the egress.

4. Rule History: Take any node each packet has conditions history an ordered list of rules the packet matched and traveling the network. Rule histories are fundamental problems provided provide the basic raw material

5. Topology: The topology different function models the network models by specifying which

pairs of ports and links are rules that forward packets from to without changing. If no models rules match an input port the port is an edge port and the packet has reached its destination.
The following is the block diagram of ATPG.



The system first collects all the forwarding state from the network (step 1). This usually involves reading the FIBs, ACLs, and config files, as well as obtaining the topology. ATPG uses Header Space Analysis [6] to compute reachability between all the test terminals (step 2). The result is then used by the test packet selection al-gorithm to compute a minimal set of test packets that can test all rules (step 3). These packets will be sent periodically by the test terminals (step 4). If an error is detected, the fault localization algorithm is invoked to narrow down the cause of the error (step 5).

Test Packet Generation: Set of test nodes are considered in the network model to send and receive test packets. It is to generate a set of test packets and exercise every rule in every switch object so that every fault will be observed by at least one test packet. This is scanner type software test models that try to test every possible branch models. The broader goal can be limited to testing every link or every queue. When generating test packets ATPG considered two key constraints [9].

(1) Port: ATPG was forced to use test terminals that are available

(2) Header: ATPG used headers that each test terminal is permitted to send for the network administrator may only allow using a specific model of VLAN.

Test packets are taken into the network in which every rule is covered directly from the data plane with different locations treats links like normal forwarding conditions. Its full coverage provides testing of every link in the network model It can also be specialized to form a minimal no of of packets that obviously test every link for network likeness At least one basic form we would feel that ICPG similar technique is fundamental to networks Instead of reacting to failures many network operators such as Internet2 proactively check the health of their network using pings between all pairs of sources all-pairs of network provide testing of all links and has been found for large networks model such as Planet Lab [10, 11].

Limitations of ATPG:

Dynamic Boxes: ATPG cannot model boxes whose internal state can be changed by test packets. **Invisible Rules:** A failed rule can make a backup rule active, and as a result, no changes may be observed by the test packets.

Sampling: ATPG can miss match faults since the error is not uniform across all matching headers.

Transient network states: ATPG cannot uncover errors whose lifetime is shorter than the time between each round of tests.

III. PROPOSED SYSTEM

Contender framework generates minimum no of packets automatically to debug the false occurring in the network model This tool could automatically generate packets for checking performance assertions such as like packet loss finds and determines errors by independently testing all forwarding entries any packet processing rules and security models in network test packets are generated algorithmically from device configuration files and from FIBs which requires minimum number of packets for complete coverage Test packets are fed into the network in which that every rule is covered directly from the data plane Since treats links like normal forwarding conditions its full coverage provides testing of every link in the network model It can also best specialized to form a minimal set of packets that obviously test every link for network likeness At least in this basic form, we would feel that some different technique is fundamental to networks Instead of reacting to failures many network operators.

Self-generated check Packet Generation (ICPG) framework that mechanically generates a lowest set of packets to check the animateness of the underlying topology and also the harmoniousness between knowledge plane state and configuration specifications. The tool also can mechanically generate packets to check performance assertions like packet latency. It also can be specialized to get a lowest set of packets that simply check each link for network animateness.

Advantages of Proposed System: A survey of network operators revealing common failures and root causes, A check packet generation algorithmic rule, A fault localization algorithmic rule to isolate faulty devices and rules. ICPG use cases are useful for both functional and performance testing.

Implementation:

A. Check Packet Generation:

Algorithm: It was assumed that a set of test terminals in the network can send and receive test packets. The goal is to generate a set of test packets to exercise every rule in every switch function, so that any fault will be observed by at least one test packet. This is analogous to software test suites that try to test every possible branch in a program. The broader goal can be limited to testing every link or every queue. When generating test packets, ICPG must respect two key constraints First Port (ICPG must only use test terminals that are available) and Header (ICPG must only use headers that each test terminal is permitted to send).

B. Generate All-Pairs Reachability Table: ICPG starts by computing the complete set of packet headers that can be sent from each test terminal to every other test terminal. For each such header, ICPG finds the complete set of rules it exercises along the path. To do so, ICPG applies the all-pairs reachability algorithm described. On every terminal port, an all- header (a header that has all wild carded bits) is applied to the transfer function of the first switch connected to each test terminal. Header constraints are applied here.

C. ICPG Tool generates the minimal number of test packets so that every forwarding rule in the network is exercised and covered by at least one test packet. When an error is detected, ICPG uses a fault localization algorithm to determine the failing rules or links.

D. Fault Localization:

Algorithm: ICPG periodically sends a set of test packets. If test packets fail, ICPG pinpoints the fault(s) that caused the problem. A rule fails if its observed behavior differs from its expected behavior. ICPG keeps track of where rules fail using a result function “Success” and “failure” depend on the nature of the rule: A forwarding rule fails if a test packet is not delivered to the intended output port, whereas a drop rule behaves correctly when packets are dropped. Similarly, a link failure is a failure of a forwarding rule in the topology function. On the other hand, if an output link is congested, failure is captured by the latency of a test packet going above a threshold.

IV.CONCLUSION

In current System it uses a method that is neither exhaustive nor scalable. Though it reaches all pairs of edge nodes it could not detect faults in liveness properties. ICPG goes much further than liveness testing with same framework. ICPG could test for reachability policy (by checking all rules including drop rules) and performance measure (by associating performance measures such as latency and loss of test packets). Our implementation also enlarges testing with simple fault localization scheme also build using header space framework. Testing liveness of a network is a fundamental problem for ISPs and large data center operators. Sending probes between every pair of edge ports is neither exhaustive nor scalable. It suffices to find a minimal set of end-to-end packets that traverse each link. However, doing this requires a way of abstracting across device specific configuration files (e.g., header space), generating headers and the links they reach (e.g., all-pairs reachability), and finally determining a minimum set of test packets (Min-Set-Cover). Even the fundamental problem of automatically generating test packets for efficient liveness testing requires techniques akin to ICPG. ICPG, however, goes much further than liveness testing with the same framework. ICPG can test for reachability policy (by testing all rules including drop rules) and performance health (by associating performance measures such as latency and loss with test packets). Our implementation also augments testing with a simple fault localization scheme also constructed using the header space framework. As in software testing, the formal model helps maximize test coverage while minimizing test packets. Our results show that all forwarding rules in Stanford backbone or Internet2 can be exercised by a surprisingly small number of test packets.

REFERENCES

[1] Hongyi Zeng, Peyman Kazemian, George Varghese, and Nick McKeown, “Automatic Test Packet Generation”, IEEE/ACM Transactions on Networking, Vol. 22, No. 2, pp. 554-566, April 2014.

[2] M. Jain, 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.

[3] Kompella, R. R., Greenberg, A., Rexford, J., Snoeren, A. C., Yates, J. Cross-layer Visibility as a Service", In Proc. Of fourth workshop on Hot Topics in Networks (HotNet-IV) (2005)

[4] D. Maltz, G. Xie, J. Zhan, H. Zhang, G. Hjalmtysson, A. Greenberg,"Routing design in operational networks: A look from the inside", In Proc. ACM SIGCOMM, 2004.

[5] Mark Stemm, Randy Katz, Srinivasan Seshan,“A network measurement architecture for adaptive applications”, In Proceedings of the nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 285 - 294, 2000.

[6] Verma, D.,"Simplifying Network Administration using Policy based Management". [3] Kompella, R. R., Greenberg, A., Rexford, J., Snoeren, A. C., and Yates, J. Cross-layer Visibility as a Service. In Proc. Of fourth workshop on Hot Topics in Networks (HotNet-IV) (2005).

[7] D. Maltz, G. Xie, J. Zhan, H. Zhang, G. Hjalmtysson, and A. Greenberg. Routing design in operational networks: A look from the inside. In Proc. ACM SIGCOMM, 2004.

[8] Mark Stemm, Randy Katz, Srinivasan Seshan, “A network measurement architecture for adaptive applications”, In Proceedings of the nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 285 - 294, 2000.

[9] 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.

[10] H. Weatherspoon, “All-pairs ping service for PlanetLab ceased,” 2005 [Online]. Available: http://lists.planetlab. org/pipermail/users/2005-July/001518.html

[11] M.Reitblatt,N.Foster, J. Rexford, C. Schlesinger, andD.Walker, “Abstractions for network update,” in Proc. ACM SIGCOMM, 2012, pp.323–334.

Hari Kishan Chapala PG scholar in dept of CSE in St.Mary’s group of Institutions which is affiliated to JNTUK, Kakinda University. His research interests include Computer networks, Information Security and Cloud Computing.

Murali Gudipati received M.Tech in CSE from JNTUK, Kakinda University and Ph.D in Computer Science and Engineering from Acharya Nagarjuna University, Guntur. His research interests include Bandwidth estimation in Computer Networks, Network Security and Data Mining. He is working as Principal in St.Mary’s group of Institutions. He published several research papers in various reputed National and International Journals. He presented several research papers in various conferences.