



A Fault Localization Algorithm To Isolate Faulty Devices and Rules

¹R.Sowndarya,²M.Vamsi Krishna

¹Final MTech Student,²Professor & HOD

¹²Dept of Computer Science and Engineering, Chaitanya Institute of Science & Technology, Madhavapatnam, Kakinada, E.G Dist, Andhra Pradesh, India

Abstract:

Network engineers pursue down bugs by means of the simplest tools and track down root causes using a grouping of mount upinsight and perception. Debugging networks is only fetching harder as networks are getting bigger. Modern data centres may surround 10 000 switches, a campus network may serve 50 000 users, a 100-Gb/s long-haul link may carry 100 000 flows and are getting more thorny with over 6000 RFCs, router software is based on millions of lines of source code, and network chips repeatedly contain billionsof gates. It is a mall conjecture that network engineers have beenlabelled “masters of complexity”.

Keywords: Data plane analysis, network troubleshooting, testpacket generation.

I. Introduction:

Organizations can modify ATPG to get together their needs. For exemplar, they can decide to just ensure for network liveliness (link cover) or check every rule (rule cover) to make sure refuge strategy. ATPG can be adapted to check only for reach ability for presentation as well. ATPG can get a feel for to limitation such as necessitate test packets from only a few places in the network or by means of special routers to produce test packets from every port. ATPG can also be harmony to apportion more test packets to employ more important rules. For case in point, a healthcare network may contribute more test packets to Firewall rules to guarantee HIPPA obedience. We tested our method on two real-world data sets—the backbone networks of Stanford University, Stanford, CA, USA, and Internet2, representative an venture network and a nationwide ISP.

II. Related Work:

The approach is balancing to these proposals by integrate input and port constraints, ATPG can produce test packets and inoculation points using existing deployment of dimension devices. Our employment is intimately connected to work in programming languages and representative debugging. We complete a preliminary attempt to use KLEE and establish it to be 10 times slower than even the unoptimized header space structure. We guess that this is basically because in our framework we directly simulate the forward path of a

packet in its place of resolver constraint using an SMT solver. However, more work is requisite to know the disparity and probable occasion.

I. Literature Survey:

THE AUTHOR, Ajay Mahimkar (ET .AL), AIM IN [1], Chronic network conditions are reason by recital harm events that take place sporadically over a comprehensive epoch of time. Such conditions can cause recurring performance poverty to customers, and every now and then can even turn into sombre hard failures. It is consequently vital to troubleshoot and revamp chronic network conditions in a judicious fashion in order to make sure all dependability and routine in large IP networks. Today, troubleshooting persistent conditions is habitually performed manually, making it a monotonous, time unbearable and error-prone method.

THE AUTHOR, Jennifer Yates (ET .AL) AIM IN [2], we present NICE (Network-wide Information Correlation and Exploration), a novel communications that allows the troubleshooting of chronic network conditions by notice and analyzing arithmetical association crossways manifold data sources. NICE uses a novel round permutation test to decide the statistical meaning of correlation. It also lets supplement study at various spatial granularities e.g., link, router, network level, etc. We confirm NICE using real extent data collected at a tier-1 ISP network. The results are rather positive. We then be appropriate NICE to troubleshoot real network issues in the tier-1 ISP network. In all three case studies behaviour so far, NICE productively uncovers before unknown chronic network conditions, resultant in better network operations.

III. Problem Definition:

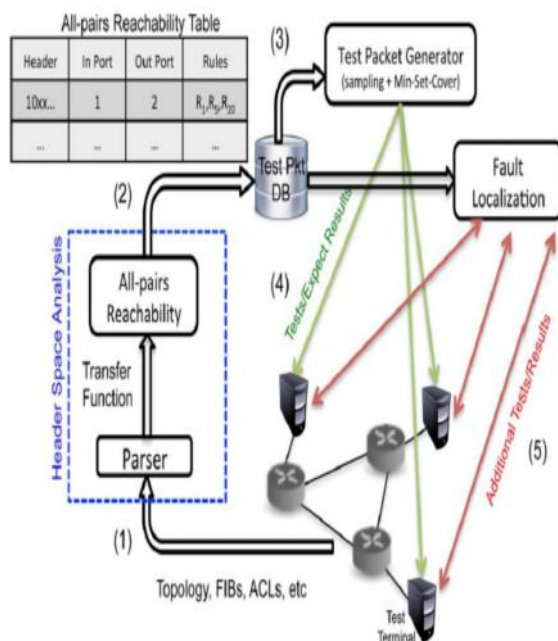
The two most frequent causes of network malfunction are hardware failures and software bugs, and those problems evident themselves both as reach ability failures and throughput/latency squalor. Testing liveness of a system is a primary problem for ISPs and large data centre operators. Distribution probes among every pair of edge ports are neither thorough nor scalable. It is sufficient to discover a minimal set of end-to-end packets that cross each link. Though, doing this needs a way of nonfigurative across machine specific configuration files, generating headers and the links they reach,

and at length formative a minimum set of test packets (Min-Set-Cover). It is to ensure enforcing steadiness between policy and the configuration. Not intended to recognize liveness failures, bugs router hardware or software, or act problems.

IV. PROPOSED APPROACH:

A survey of network operators discloses common failures and root causes. A fault localization algorithm is to cut off faulty devices and rules. ATPG use cases for purposeful and presentation testing. Assessment of a prototype ATPG system by means of rule sets collected from the Stanford and Internet2 backbones. Automatic Test Packet Generation (ATPG) framework by design produces a smallest set of packets to test the liveness of the causal topology and the equivalence between data plane state and construction stipulation. The tool can also repeatedly create packets to examination routine allegations such as packet latency. It can also be specific to produce a minimal set of packets that only test every link for network liveness.

V. System Architecture:



VI. Proposed Methodology:

Generate All-Pairs Reachability Table:

ATPG begin in on arranging unquestionably the arrangement of bundle headers that can be pass on from every test lethal to each other test terminal. For each such header, ATPG figures out the complete arrangement of tenets it practices along the way. To do as such, ATPG be appropriate the all-sets achieve capacity algorithm clarified.

Test Packet Generation:

ATPG must approval two key imperatives First Port ATPG should just involve test terminals that are reachable and Header ATPG should just utilize headers that every test serious is true blue to convey. We underestimate an arrangement of test terminals in the system can put and be given test parcels. Our

motivation is to create an arrangement of test parcels to utilize all guideline in each switch capacity, so that any mistake will be exact by at scarcest one test bundle. This is reporter to programming test gathering that push to test every potential branch in a system. The more extensive try can be unfinished to testing each connection or each line up.

Atpg Tool:

ATPG amasses the unimportant number of test parcels so that every sending principle in the system is comprehend and encased by no less than one test bundle. At the point when a shortcoming is remark, ATPG utilizes a error limitation calculation to close the coming up short principles or connections.

Fault Localization:

A pretentious standard miss the mark if a test packet is not pass on to the future yield port, while a drop guideline accomplishes effectively when packets are go down. In like manner, a connection breakdown is a glitch of a sending guideline in the topology rule. Then again, if a creation connection is congested, disappointment is detained by the inactivity of a test packet takeoff over an edge. ATPG consistently sends off an arrangement of test parcels. In the event that test packet miss the mark, ATPG perceive the fault(s) that premise the unpredictability.

VII. Algorithm:

Fault Localization Algorithm:

INPUT: N1, N2, N3, R1, R2, R3, ATPG TOOL

START:

STEP1: Packet PK arrives at a network port P.

STEP2: The switch function that T contains the input port PK.P

STEP3: Produce a list of packets.

STEP4: If packet reaches destination it is recorded.

else

Topology function invokes switch function containing new port.

STEP5: Process repeats until packet reaches or dropped to destination.

END

OUTPUT: Packets reached status

Rate Control Algorithm:

On arrival of BF packet p from egress router e

if (p.asynchronous == FALSE)

e = cur_time - p.timestamp;

if (e.currentRTT < e.baseRTT)

e.baseRTT = e.currentRTT;

deltaRTT = e.currentRTT - e.baseRTT;

RTTElapsed = (CurrentTime -

LastFeedbacktime) / currentTime;

for each flow f listed in p

f.mrc = min (MSS / e.currentRTT, f.egress_rate / MF);

if (f.phase == SLOW_START)

if (deltaRTT * f.ingress_rate < MSS * e.hopcount)

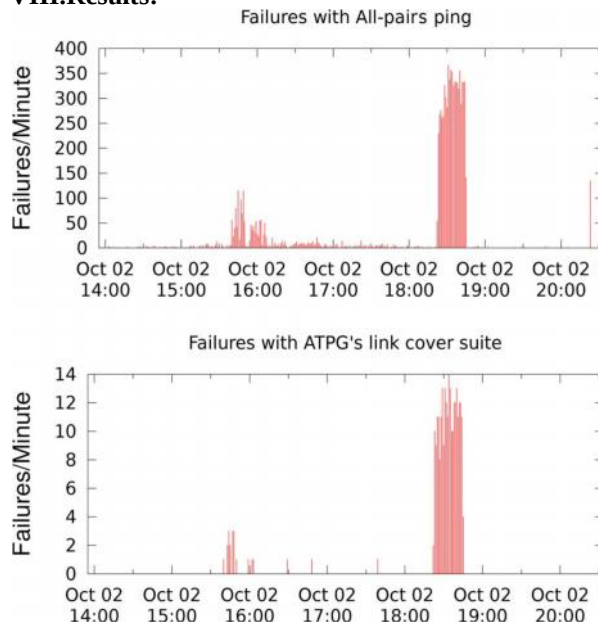
f.ingress_rate = f.ingress_rate * 2^RTTElapsed;

else

f.ingress_rate = f.egress_rate - f.mrc;

The exchange of feedback between routers at the borders of a network in order to detect and restrict unresponsive traffic flows before they enter the network, thereby preventing congestion within the network.

VIII.Results:



It explains the number of unsuccessful test cases throughout that period. At the same time as both all-pairs ping and ATPG's certain test suite rightly detected the outage, ATPG uses drastically less test packets. In fact, ATPG makes use of only 28 test packets per round compared to 2756 packets in all-pairs ping, a 100x reduction. It is effortless to see that the decline is from quadratic overhead (for all-pairs testing between 53 terminals) to linear overhead (for a set cover of the 30 links between switches). We memo that whereas the set cover in this experimentation is so undemanding that it could be worked out by hand, other networks will have Layer-3 rules and more multipart topologies necessitate the ATPG minimum set cover algorithm.

IX.Enhancement:

To overcome congested line issues in switches proposing rate control algorithm in switches. Absence of undelivered groups avoids over-weight as a result of re-transmission. Sensible circulation of information exchange limit is ensured.

X.Conclusion:

Network managers nowadays make use of prehistoric tools such as ping and trace out. Our review results point out that they are enthusiastic for more sophisticated tools. Other fields of engineering point towards that these desires are not irrational. For case, both the ASIC and software design industries are reinforced by billion-dollar tool businesses that bring in techniques for both static (e.g., design rule) and dynamic (e.g., timing) corroboration. Infact, numerous months after we make and name our

system, we exposed to our revelation that ATPG was a familiar contraction in hardware chip testing, where it stands for Automatic Test Pattern Generation. We expect network ATPG will be uniformly of use for automated dynamic testing of production networks.

XI.Future Work:

Systems are more eccentric and having particular sorts of models. Future examination bearing on upgrade execution of ATPG device and add more convenience to recognize directing attacks and execution issues.

XII.References:

- [1] "ATPG code repository," [Online]. Available: <http://eastzone.github.com/atpg/>
- [2] "Automatic Test Pattern Generation," 2013 [Online]. Available: http://en.wikipedia.org/wiki/Automatic_test_pattern_generation
- [3] P. Barford, N. Duffield, A. Ron, and J. Sommers, "Network performance anomaly detection and localization," in *Proc. IEEE INFOCOM*, Apr. , pp. 1377–1385.
- [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] A. 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] N. Duffield, "Network tomography of binary network performance characteristics," *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5373–5388, Dec. 2006.
- [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.



M VAMSI KRISHNA

received the M Tech CS in Allahabad University, M.Tech (AI &R) degree in Andhra University, and Ph.D from Centurion University ,Odisha. Currently he is working as Professor & HOD in Department of Computer Science and Engineering. He has 15 years of experience in teaching. His research interests include Artificial intelligence, computer networks, image processing.



R.Sowndarya is a student of Chaitanya Institute of Science & technology, Madhavapatnam, Kakinada, E.GDist, Andhra Pradesh, India. Presently she is pursuing her M.Tech in Computer Science and Engineering in this college and she received her B.Tech from Aditya

Engineering College Surampalem in Peddapuram, affiliated to JNT University, Kakinada in the year 2013. Her areas of Interest is Computer Networks, Software engineering Virtualization technologies and all current trends and techniques in computer science.