International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2015): 6.391

Network Testing: An Overview of Approaches

Suchita Navpute[1], Sanjeev J. Wagh[2]

[1, 2] K. J. College of Engineering and Management Research Pune, India

Abstract: ***Todays networks are very large in size as well as complex by design. Validating today's networks against any problem is really cumbersome task with traditional approaches like ping, traceroute, tcpdump etc. Networks are generally susceptible to problems like loop, black-hole, drop, software bugs and physical failures. Protecting networks from these kind of problems using traditional approaches is complicated and time consuming which is not at all affordable in any enterprise scale networks. To avoid loss caused due to network problems, one should have tools which can validate network against above said problems in less time as soon as they are introduced in network. Researchers have developed many state of the art tools like HSA, ATPG, Veriow, Anteater, NetPlumber. These tools are capable of detecting forwarding state anomalies with minimum human intervention. Approaches used in above tools can be broadly classified into two categories i.e. static validation and dynamic validation. Static validation is capable of detecting loop, drop and black-hole along with these, dynamic validation can detect problem caused due to software bug and physical failure. In this paper we will study general network problems, approaches to tackle them and some of their implementations in detail.***

Keywords: Network verification, dynamic validation, static validation, Software Define Network

1. Introduction

In the beginning, the work of networking devices was quite simple. Work of networking devices was just to see the forwarding table entries and on the basis of destination address decide where to send packet next. Number of hosts previously were also less so it was not so hard to handle network. On the other hand today's situation of network has changed. Network grew in all dimensions like size, complexity, number of hosts, size of data which we want to transfer on network and many more.

Hosts grew exponentially which leads in crossing the address limit of IPv4. To overcome this problem middle boxes like NAT, firewall comes in picture. Consequence of which is routing become little bit complex. Solution on this is new mechanisms invented like VLAN, MPLS which help to make routing flexible. Detecting errors like reachability problem, loop and drop in such large and complicated network with old tools is not efficient as well as time consuming process and need to fix problems manually.

Testing small networks manually is not an issue but if we have to test large and complex network like data center manually then it is quite hard. It is time consuming to deal with such network which has thousands of switches and millions of rule to check. It is headache for administrators so there is need of an automated tool for validating networks.

There are some typical problems which generally occur in networks like causing packet to loop indefinitely, packet drop before it is reached to destination, fall into black-hole, error occurred due to faulty line card, packet drop due to buffer overflow, etc. As we have no choice for large and complex network then it is must to face the problems and finding solution. For voluminous network errors are unavoidable.

Tracking down failure in network is hard because of the following reasons. First, forwarding states are updated simultaneously by many different protocols, programs and humans. Second, for observing forwarding state administrator has to logging manually into each network device separately.

There are mainly two types of network validation

1) **Static Validation**: It validates network using networks forwarding states snapshot representation. It assumes forwarding state snapshot is consistent with underneath network. We will see more details about static validation in section III.
2) **Dynamic Validation**: It validates network using actual underneath network because of which it could detect problems cause due to physical failure and software bug. We will see more details about dynamic validation in section III.

Trying new analyzing mechanism or protocol on network is quite hard and take more time to settle down the network because it need to manage two things at a time that are control functionality and data transfer functionality of network devices. Software Define Network is start of new era of computer network industry. It separates control plane and data plane so controlling the network devices become easy. Due to SDN, network devices like switch, router have become simple data transfer entities and these entities are controlled by SDN controller centrally.

Some common network problems are discussed in section II whereas section III represents approaches for validating network

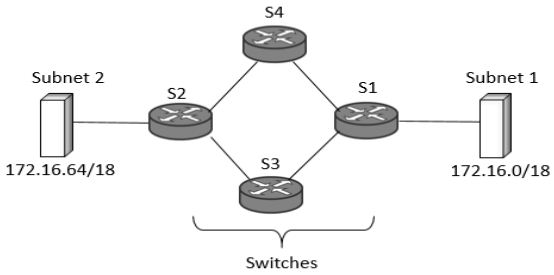

Figure 1: Example Network

2. Network Problems

Network problems arises mainly due to two causes, first is forwarding state inconsistencies and second is hardware failure. As there is no any central entity controlling and configuring network devices, network administrator has to configure each network entity separately using its own communication mechanism. Forwarding state of network is a cumulative, cooperative entity among all network data forwarding entities. Configuring thousands of routers and switches at their individual level could possibly cause conflicting entries across different forwarding entities which is main cause of forwarding state inconsistency. Hardware failures mainly include network card failures and network line failures.

Error in forwarding table causes packet to loop indefinitely, packet drop before it reach to destination, packet fall in to black-hole. Identifying these type of errors in complex network is hard because in such network, administrator has to deal with thousands of switches and millions rule.

Consider small network example shown in fig1. This network has 4 switches and 2 subnets. Our example network uses multipath routing. The node having more than one outgoing edges represent multipath routing in the network. It has 8 rules as shown in fig2. Packet arriving at S1 from subnet 172.16.0/18 has two paths S3 and S4 to reach to the destination address 172.16.64/18.

Rules:
S1: 172.16.64/18 → S3, S4
172.16.0/18 → direct
172.16.0/18 → S3, S4
172.16.64/18 → direct
S3: 172.16.64/18 → S2
172.16.0/18 → S1
S4: 172.16.64/18 → S2
172.16.0/18 →S1

Graphical representation of example network is shown fig 2. Network behavior of subnet can be best represented using directed forwarding graph.

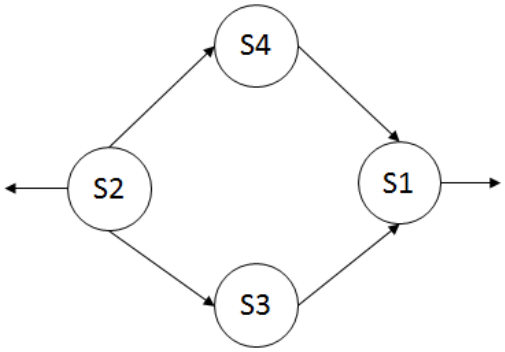

Figure 2: Normal Network

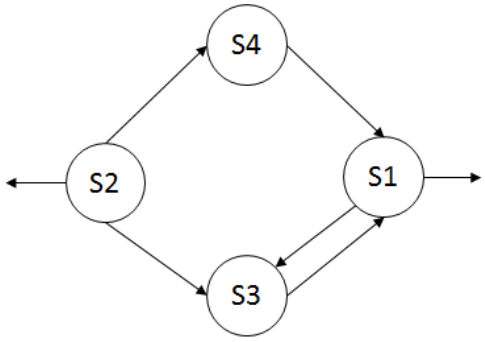

Figure 3: Loop in Network

e.g. In Fig. 2 arrow from S2 to S4 means packets from 172.16.64/18 be forwarded from S2 to S4.

- **Loop:** Fig. 3 shows how an error in forwarding table can lead to loop in the network.

172.16.64/18 → S3, S4
172.16.0/18 → S3

Consider above forwarding entries at switch S1. From the forwarding state at S1 it is clear that switch S1 will forward packet destined to 172.16.0/18 to switch S3 causing loop as shown in fig4. Network has one loop S1S3-S1 and packets for 172.16.0/18 will never reach to their intended destination. It will keep infinitely looping between S1 and S3.

- **Black-holes:** Fig.7 shows black-hole situation in network. Consider that entry 172.16.0/18 → S1 direct is dropped. In this scenario packet destined to subnet 172.16.0/18 will be dropped at S1 as it has no forwarding entry to decide its outgoing path.

Detecting hardware failure in large network is hard because network devices are typical black-boxes i.e. switches and routers. To find location of the failure administrator has to login into each device manually and check all configuration of the device. Detecting simple line-card failure in large network take time because administrator has to decide manually which ping packet to send and after observing forwarding states they find clue of failure.

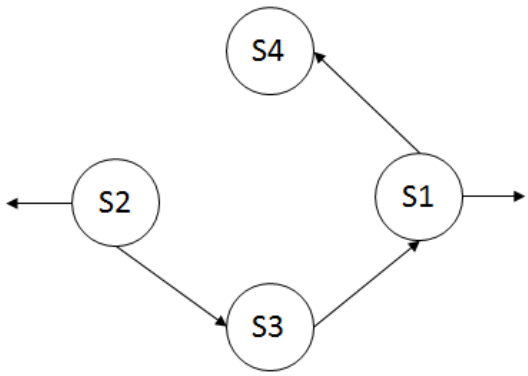

Figure 4: Black-Hole in Network

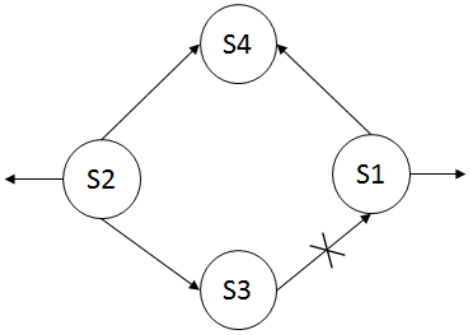

Figure 5: Physical Failure in Network

- **Drop due to physical failure:** Consider that physical link from S1 to S4 is failed due to from physical breakage of network line is shown in Fig. 5. Packet destined to 172.16.0/18 from S2 will be spread across S3 and S4. Packets following route from S2 to S4 will be dropped due to physical failure of line.

3. Approaches

Network error should be caught before too much traffic is lost and security is in danger. To analyze network many ideas come forward, among them only few stand in performance test. Though they succeed in analyzing network as well as finding errors, some of them are failed in test of time consumption, some are failed in performance. Need varies as per type of network and also as per user. But most commonly, expectations from analyzing tool are that it should analyze network correctly and it should catch error in less time as it possible. To satisfy these requirement many tools are implemented but sadly some have succeeded in analyzing network correctly but simultaneously time consuming. Some have shown result in very less time but are not guaranteed about correctness.

Fig. 6 shows general network scenario it shows data plane and control plane. Forwarding state required by data plane is written by control plane. Control plane can be local in case of traditional or non SDN network whereas control plane will be remote in case of SDN [1]. Control plane should correctly implement network policies into data plane.

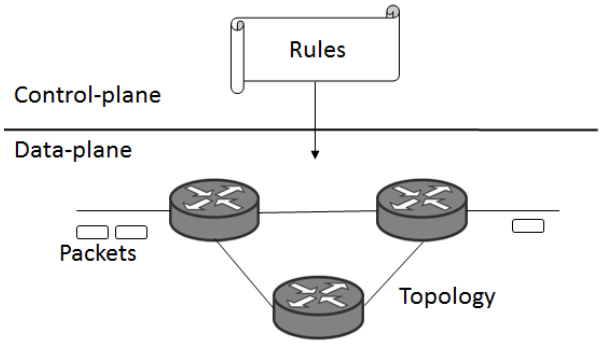

Figure 6: Simplified View of Network

Controller in control plane compiles policy into device specific configuration file. Forwarding behavior of network is decided by configuration files. To work network properly all three aspects i.e. policy, configuration files, forwarding behavior should be in consistent state with each other. In addition to this sufficient physical links and nodes should be in working state. Basically there are two types of validation

3.1 Static Validation

It validates network using networks forwarding states snapshot representation. Static validation tool calculates representational model (snapshot) e.g. HSA [1] from forwarding state of network. As these snapshots are static representation of network forwarding state, tools based on analysis of such static snapshot are referred as static validation tools. Static validation tools are capable of confirming consistency between network policy and configuration files. Most of the static validation tools work offline on built snapshot, they may miss any consistencies caused in underline network after calculation of snapshot. Some of the examples of static validation tools are HSA [1], Anteater [4], Netplumber [5].

As static validation tools validates network based on representational model, they are not capable of detecting problem caused due to physical failure or software bugs. And also not able to confirm consistency between forwarding configuration and actual forwarding behavior in data plane.

3.2 Dynamic Validation

It validates network using actual underneath network because of which it could detect problems cause due to physical failure and software bug. Dynamic validation may use snapshot approach to help tuning and reducing overhead while validating network. As actual network is test bed in dynamic validation, keeping validation overhead small is an important factor in dynamic validation otherwise validation itself could hamper performance of network. Example of dynamic validation include ATPG[2].

4. Previous Implementation

To caught routing error before too much data loss and security breached there is of need fast and scalable tool. Many tools have been proposed to fulfill networks need like Netplumber[5], HSA[1], Anteater[4], Veriflow[3], ATPG[2].

These all tools use different model for analyzing the network. Most of them do static validation. We will see one by one these tool and their approaches.

A.HSA

It validates network statically. As we discussed above static validation is nothing but by studying representational model of actual network detecting the failure. To represent network HSA use geometric model i.e. it consider a packet as a point in the 0,1L space where L is the length of the header. And packet will be transferred by the network devices from one point to other point in the space.

HSA detect some class of failure like reachability failure, loop in the network, detecting slice isolation and detect leakage.

There are some tools were available before HSA which analyzes network but the all were are protocol dependent e.g.

Firmato[8], Fang[9], a toolkit for firewall modeling and analysis[7]. HSA is protocol independent model. It do not worry about what protocol is running in the network.

There are some limitations of HSA as it doesn't tell the cause of error. It will show the location of error like mistake in configuration line or forwarding table inconsistency but it not able to tell how and why the error occurred. As it is static checker it not able to recognize problem in running network or it can be said it will not possible to caught the failure before it happened. It is not possible to HSA to predict problem by looking the forwarding table inconsistency.

B. Netplumbr

It extends the work of HSA. Netplumber [5] is based on HSA but it is real time checker. Netplumber take leverage of Software Define Network (SDN). As we discussed SDN provide facility to control network centrally. Netplumber seats in-line with controller. Netplumber uses graphical model to represent the network. Real time checking of updates and flexible way to add new complex policy queries without writing new ad hoc code for each policy check are advantages over HSA.

As Netplumber agent seats in between the controller and network devices it get each update like installation or deletion of rule, link up and link down. So as update acquire Netplumber can updates its own internal model and if violation occurs it can inform to administrator. The internal model of Netplumber is graph based in which directed edges of graph represent next hop and nodes represents rules in the network.

Netplumber take more processing time to verifying link updates so it is not suitable for network which has high rate of link up and down. This is the greater limitation of Netplumber as well it is HAS like static checker so it can't possible to save network before failure.

C.Anteater

It also follow static validation approach but difference is that it analyses data plane. As anteater statically checks data plane it is possible to catch bugs which are invisible at the level configuration files. Anteater uses Boolean satisfiability problem (SAT) to localize an error. Anteater can check forwarding loops, packet loss, and consistency. Anteater translate high level network invariants into instances of Boolean satisfiability problems check these problems against network state using SAT solver and report if violations have been found.

Anteater uses boolean function as representation model. Along with forwarding table entries, anteater also considers network topology to build a boolean function. Anterator is given with any invariant to be checked against network like packet reachability, loops in network, inconsistency among forwarding entries across the routers, etc. It combines invariant factor and network's data plane state into instances of boolean satisfiability to analyze network. If any state in the network breaks any invariant, anteator detects such entities causing problem like packet header, forwarding table entries and path. There are two limitations of anteater first, it is unable to find bugs in network devices such as router which interprets and act on configuration files. Second it can focus on validating single protocol like BGP or firewall.

D.VeriFlow

It is layer between SDN controller and the data plane. VeriFlow is much faster than all above approaches. It is also basically static checker but it perform real-time validating network in the context of SDN. It take leverage of SDN. As it seats in between the controller and the network devices it get each and every update from SDN before it forwarded to the data plane. VeriFlow uses trie data structure for its internal representation. The main task of VeriFolw is tracking each forwarding state change event as is passes by SDN, apply that new update on its internal model check if any violation. If it found any problem due to update it can alarm so that network will saved from violation so it is called as real-time checker.

Advantages of VeriFlow are first, it is quite faster than earlier tools not because of its data structure but its methodology it check only that subpart of a trie which will going to affect due to updating. Second it can prevent system before any failure.

Limitation of VeriFlow: As it is static checker so it is unable to prevent network from hardware failure e.g. physical link down.

E. ATPG

ATPG stand for Automatic Test Packet Generation. It is one of the example dynamic validation. ATPGs process contains 5 stages.

Stage1: It collect all information about network like forwarding state, configuration file, topology etc. so that it create its internal representational model.
Stage2: with the help of HSA it check all reachability between test terminals.
Stage3: ATPG create minimal set of test packet.
Stage4: These set of packets are sent to the test terminals and check for failure.
Stage5: If error detected, the fault localization took placed.

As ATPG is able to detect physical failure and it can check network online, it can be said dynamic checker. Detecting physical failure is one of its big advantage.

5. Conclusion

In this paper we discuss different approaches to troubleshoot the network. As we seen there are few tools which analyses network close to accurate but they are static checker like HSA, Anteater, Netplumber and VeriFlow. Though these tools are validating network more accurately but as they check it offline, it is unable to prevent bug before they harm the system. As well as static checkers can't find physical failure of the network. This limitation is overcome by ATPG, as it is a tool which validates the network by sending test packets on actual network. By tracing the test operator could find the physical failure in the network. ATPG tried good approach to validats the network but there are some networks typical mechanisms like NAT which dynamically

gives port no. to the packet this process could confuse to the analyzer as the same test packet will give different response. The SDN is growing fast and gives many control to the operator to manage the network. It will definitely help to researcher to try new ideas very flexibly.

References

[1] Peyman Kazemian, George Varghese, and Nick McKeown. 2012. Header space analysis: static checking for networks. In Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation (NSDI'12). USENIX Association, Berkeley, CA, USA, 9-9.

[2] Hongyi Zeng, Peyman Kazemian, George Varghese, and Nick McKeown. 2012. Automatic test packet generation. In Proceedings of the 8th international conference on Emerging networking experiments and technologies (CoNEXT '12). ACM, New York, NY, USA, 241-252. DOI=http://dx.doi.org/10.1145/2413176.2413205

[3] Ahmed Khurshid, Xuan Zou, Wenxuan Zhou, Matthew Caesar, and P. Brighten Godfrey. 2013. VeriFlow: verifying network-wide invariants in real time. In Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation (nsdi'13), Nick Feamster and Jeff Mogul (Eds.). USENIX Association, Berkeley, CA, USA, 15-28.

[4] Haohui Mai, Ahmed Khurshid, Rachit Agarwal, Matthew Caesar, P. Brighten Godfrey, and Samuel Talmadge King. 2011. Debugging the data plane with anteater. In Proceedings of the ACM SIGCOMM 2011 conference (SIGCOMM '11). ACM, New York, NY, USA, 290-301. DOI=http://dx.doi.org/10.1145/2018436.2018470

[5] Peyman Kazemian and Michael Chang and Hongyi Zeng and George Varghese and Nick McKeown and Scott Whyte. 2013. Real Time Network Policy Checking Using Header Space Analysis. Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13).

[6] S.Shenker,The future of networking, and the past of protocols,2011 [Online].Available:http://opennetsummit.org/archives/oct 11/shenkertue.pdf

[7] L.Yuan,J.Mai,Z.Su,H.Chen,C-N.Chuah,andP.Mohapatra,FIREMAN: A Toolkit for Firewall Modeling and Analysis, 2006 IEEE Symposium on Security and Privacy, pp. 213-228

[8] Y. Bartal, A. J. Mayer, K. Nissim, and A. Wool. Firmato:A novel rewall management toolkit, Proc. 20th IEEE Symposium on Security and Privacy, 1999.

[9] A.Mayer,A.Wool,andE.Ziskind,Fang: A firewall analysis engine, Proc. IEEE Symposium on Security and Privacy, 2000.