**TEST PACKET GENERATION TO THE NETWORK**

**ABSTRACT:**

Systems are getting bigger and more intricate, yet chairmen depend on simple apparatuses, for example, and to troubleshoot issues. We propose a robotized and efficient approach for testing and troubleshooting systems called “Automatic Test Packet Generation” (ATPG). ATPG peruses switch setups and produces a gadget autonomous model. The model is utilized to create a base arrangement of test parcels to (insignificantly) practice each connection in the system or (maximally) practice each manage in the system. We depict our model ATPG execution and results on two certifiable informational collections: Stanford University's spine system and Internet2. We find that few test bundles gets the job done to test all guidelines in these systems: For instance, 4000 parcels can cover all principles in Stanford spine organize, while 54 are sufficient to cover all connections. Sending 4000 test bundles 10 times each second expends under 1% of connection limit. ATPG code and the datasets are freely accessible.

**EXISTING SYSTEM:**

* Testing liveness of a system is a principal issue for ISPs and extensive server farm administrators. Sending tests between each match of edge ports is neither comprehensive nor versatile .
* To check implementing consistency amongst strategy and the setup

**DISADVANTAGES OF EXISTING SYSTEM:**

* Not designed to identify liveness failures, bugs router hardware or software, or performance problems.
* The two most common causes of network failure are hardware failures and software bugs, and that problems manifest themselves both as reachability failures and throughput/latency degradation.

**PROPOSED SYSTEM:**

* Automatic Test Packet Generation (ATPG) structure that naturally creates a negligible arrangement of bundles to test the liveness of the fundamental topology and the consistency between information plane state and design determinations
* It can likewise be specific to create an insignificant arrangement of parcels that only test each connection for organize liveness.

**ADVANTAGES OF PROPOSED SYSTEM:**

* A survey of network operators revealing common failures and root causes.
* A test packet generation algorithm.
* A fault localization algorithm to isolate faulty devices and rules.
* ATPG use cases for functional and performance testing.
* Evaluation of a prototype ATPG system using rule sets collected from the Stanford and Internet2 backbones.

**SYSTEM ARCHITECTURE:**

****

**MODULES:**

* Test Packet Generation
* Generate All-Pairs Reachability Table
* ATPG Tool
* Fault Localization

**MODULES DESCRIPTION:**

**TEST PACKET GENERATION:**

* We accept an arrangement of test terminals in the system can send and get test bundles. We will likely produce an arrangement of test parcels to practice each govern in each switch work, with the goal that any fault will be seen by no less than one test package.
* This is similar to programming test suites that endeavor to test each conceivable branch in a program.

**GENERATE ALL-PAIRS REACHABILITY TABLE:**

* ATPG starts by computing the complete set of packet headers that can be sent from each test terminal to every other test terminal.
* 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 constraint are applied here.

**ATPG TOOL:**

* ATPG 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, ATPG uses a fault localization algorithm to determine the failing rules or links.

**FAULT LOCALIZATION:**

* ATPG periodically sends a set of test packets. If test packets fail, ATPG pinpoints the error(s) that caused the problem.
* ATPG 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.

**SYSTEM REQUIREMENTS:**

**HARDWARE REQUIREMENTS:**

* System : Pentium IV 2.4 GHz.
* Hard Disk : 500 GB.
* Monitor : 15 VGA Colour.
* Mouse : Logitech.
* Ram : 1GB.

**SOFTWARE REQUIREMENTS:**

* Operating system : Windows 7.
* Coding Language : JAVA/J2EE
* IDE : Netbeans 7.2
* Database : MYSQL