Topology Generation, Instrumentation, and Experimental Control Tools for Emulation Testbeds

Roman Chertov, Sonia Fahmy, Pankaj Kumar, David Bettis, Abdallah Khreishah and Ness B. Shroff

Center for Education and Research in Information Assurance and Security (CERIAS)

Purdue University

http://www.cs.purdue.edu/homes/fahmy/software/emist/

June 16th, 2006

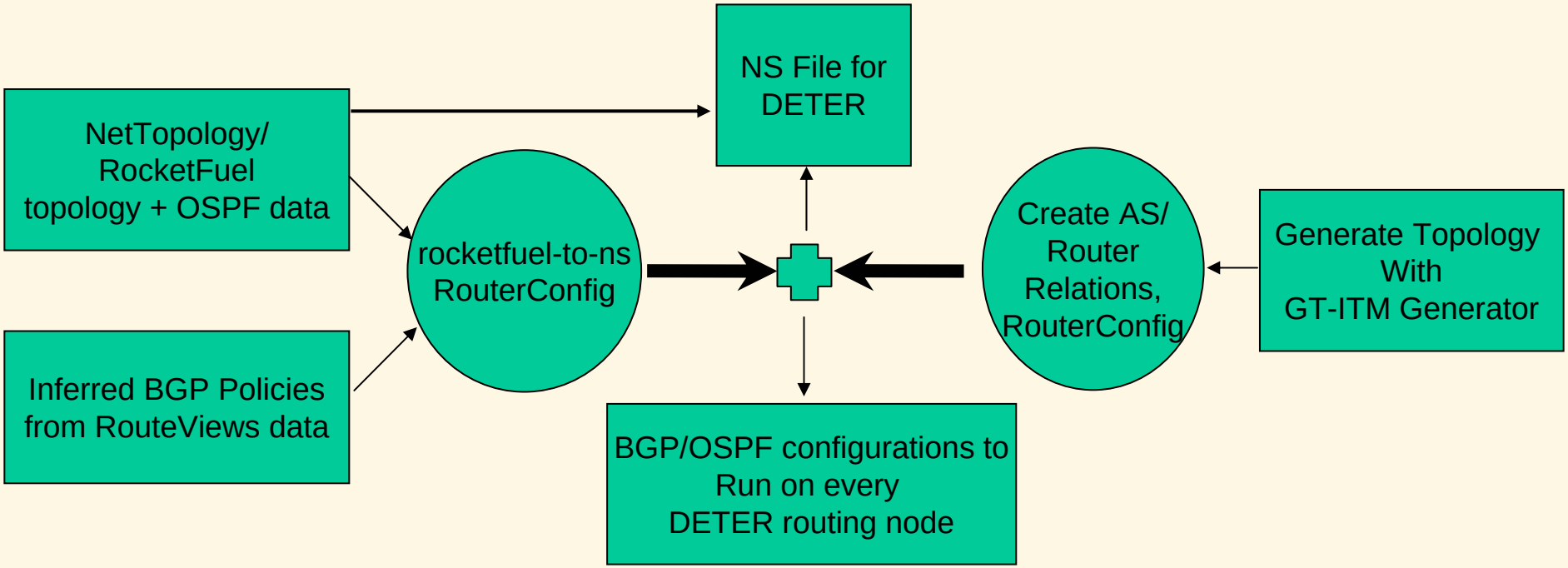
An Automated Experiment

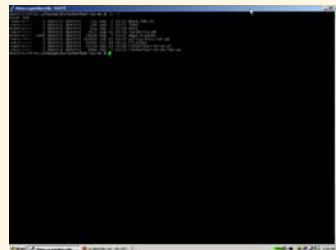
- Create a topology via a topology generator or based on available data. If applicable, create BGP/OSPF router configurations
- Create disk images on DETER with the desired tools and setup scripts
- Run an *event server* to which test nodes will attach
- Server triggers events, which are executed on test nodes. Loop capability allows easy iteration over experimental parameters
- Collected data is analyzed by scripts or can be interactively examined

Our Tools

- Can be found at http://www.cs.purdue.edu/homes/fahmy/software/emist/
- Topology generation and router configuration tools
  - From real data or generated topologies
- Scriptable Event System (SES)
  - Centralized experiment control
- Software link monitor
- Measurement and data integration tools, traffic generators, and other useful scripts

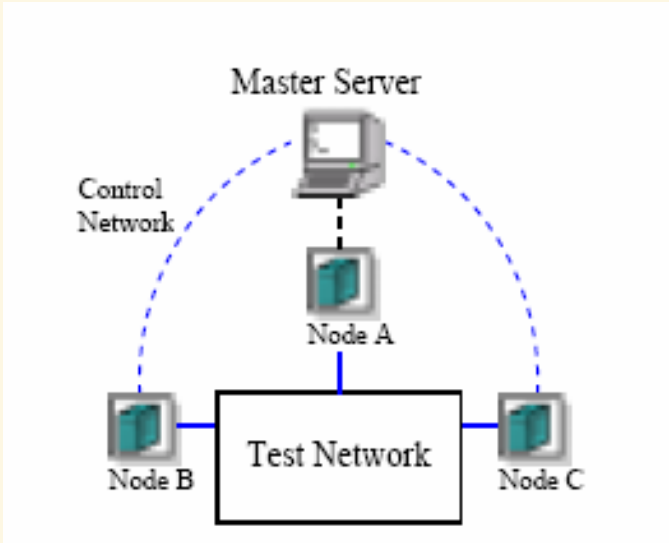
Topology/Routing Tools





Scriptable Event System

- A master application runs either on the main “users” account or on one of the test nodes.
- Communication between the master and experimental nodes is done via the control network that is free of experimental traffic.
- The server runs a script that defines the experiment:
  1. Starts measurements and configures software
  2. Launches attacks/benchmarks
  3. Once the benchmarks are complete, terminates the attack
  4. Copies local measurement logs to a central location

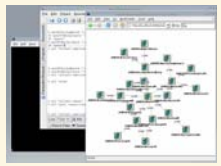


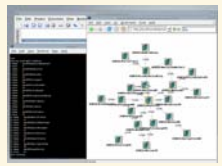
Sample Event Script

```
0 node0 node2 node3 r1 r2 "./tmeas -f /usr/local/tmeas.out"
1 "pause"
0 node2 "/usr/bin/ttcp -r > /dev/null"
1 "pause"
0 node0 node2 "rm /usr/local/dump.dmp"
1 "pause"
0 node0 node2 r1 r2 "sh /proj/DDoSImpact/exp/bell/scripts/dump.sh"
1 "pause"
5 node3 "./flood node1 -U -s10 -W160-1000 -D80000"
9 node0 "/usr/bin/ttcp -v -t node2 < /usr/local/f30m >/usr/local/ttcp.out!"
1 "pause"
0 node0 node1 node2 node3 r1 r2 "stop"
1 "pause"
0 node0 node2 r1 r2 "killall tcpdump"
1 "pause"
0 node0 "cp /usr/local/dump.dmp /proj/DDoSImpact/exp/bell/data/dump.node0"
0 node2 "cp /usr/local/dump.dmp /proj/DDoSImpact/exp/bell/data/dump.node2"
1 "pause"
0 node0 "cp /usr/local/ttcp.out /proj/DDoSImpact/exp/bell/data"
1 "pause"
0 node0 "cp /usr/local/tmeas.out /proj/DDoSImpact/exp/bell/data/tmeas.out.node0"
0 node3 "cp /usr/local/tmeas.out /proj/DDoSImpact/exp/bell/data/tmeas.out.node3"
0 node1 "cp /usr/local/tmeas.out /proj/DDoSImpact/exp/bell/data/tmeas.out.node1"
```

Measurement and Integration

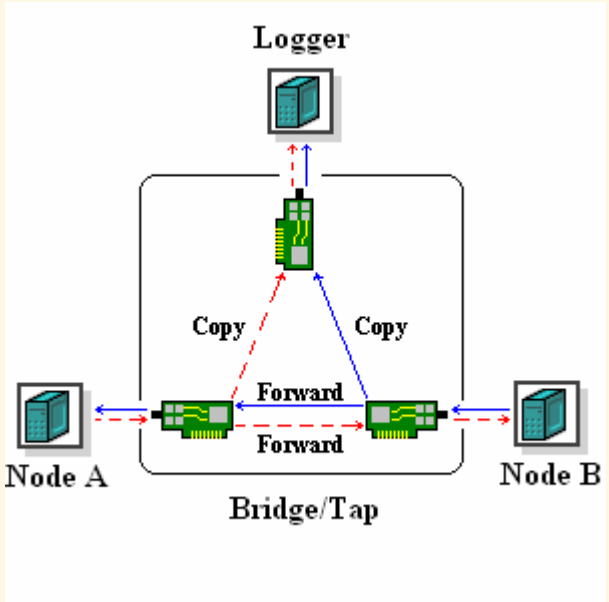
- Measurement of systems statistics at different points in the network can yield an understanding of what events are occurring in the entire network
- A tool based on a 1 second timer records CPU, PPSin, PPSout, BPSin, BPSout, RTO, Memory. The collected logs (plus routing logs) are aggregated and used to plot graphs via a collection of scripts
- TCP congestion window can be recorded on a per connection basis by reading the system files. Alternatively, it can be estimated from *tcpdump* files using *tcptrace*
- The data can also be displayed by the ESVT upon experiment completion, allowing easy graphical examination





Link Monitor

- An easy way to monitor links is to run tcpdump and drop counters on individual PCs
  - Tcpdump requires additional CPU processing
  - Drop counters are not always accurate, as they depend on the driver accuracy
- A software solution similar to a delay node can be placed on a link between two nodes
  - Two monitors can be used to find out what was dropped by comparing traffic in and traffic out
  - High traffic volumes require the tap/logger to be much faster than the test nodes
  - Extensive tests have shown that the tap can keep up with 148 Kpps, but tcpdump cannot



Future Work

- Short term:
  - Finalize and integrate tool suites
  - Add configuration options to tool suites
  - Further investigate ability to obtain/infer more complex BGP policies – perhaps through PREDICT
- Longer term:
  - Better understand scale down problem
  - Understand interaction between topology/routing policies and attacks/defenses