ToMaTo - Topology Management Tool

ToMaTo

Origin

- ToMaTo is being developed as part of the experimental facility of the German-Lab project
- Open-Source since version 2.0

What it is

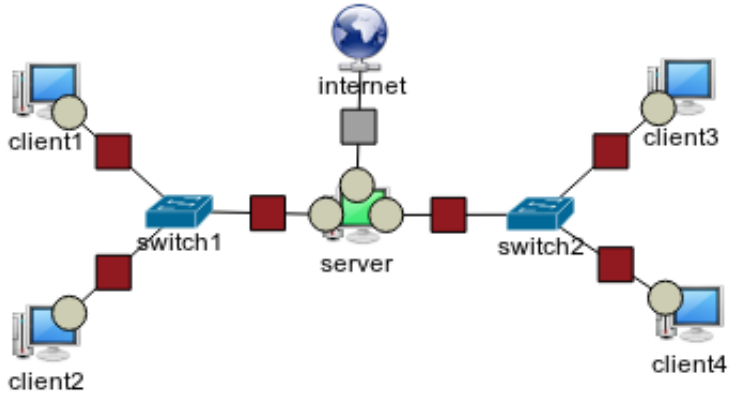
- A topology-oriented testbed software
- An easy-to-use feature-rich tool for networking experiments

What it is NOT

- A management tool for physical network topologies — Topologies are virtual
- A tool to manage virtualized production networks — ToMaTo is still under development
- A virtualization solution or virtualization frontend — ToMaTo is more

Topology elements

ToMaTo is topology-oriented, i.e. users build topologies for their experiments.



Devices

- produce and consume data
- can run software

Connectors

- forward and manipulate data
- connect devices

Three kinds of devices

- KVM devices
- OpenVZ devices
- Programmable devices

Two kinds of connectors

- VPN networks
- External networks

KVM devices

Full virtualization (KVM)

- Emulated hardware
- Nearly all x86 operating systems supported

Pros

- KVM devices can run any x86 operating system
  - Windows
  - Linux
  - BSD
- Linux inside KVM can run custom kernels

Cons

- KVM virtual machines use lots of resources
- Access to the VM is complicated
  - Login needed
  - VM is hard to control with scripts
- Interfaces cannot be configured by ToMaTo

OpenVZ devices

Container virtualization (OpenVZ)

- Guest operating systems share the Linux kernel of the host system
- Only userspace (users, files, processes, etc.) is virtualized
- Restricted access to kernel-space (hardware, filesystems, kernel modules, kernel variables, etc.)
- Nearly all Linux distributions can be run

Pros

- OpenVZ can run nearly any Linux operating system
- Easy to access remotely
  - Completely scriptable
- Efficient resource usage
- Network interfaces fully configurable
  - also using the ToMaTo editor

Cons

- Only Linux systems
- Limited kernel space access
  - No kernel modules
  - Restricted access to some kernel calls

Programmable devices

Sandboxed python scripts (Repy)

- Raw network packets can be processed with python scripts
- Sandboxing environment disallows some language elements (also most python libraries)
- ToMaTo contains library of network protocols and server implementations

Pros

- Very lightweight, uses nearly no resourcesy
- Completely free to define packet processing
- Extensive library ready to use
- Parameters can be defined by ToMaTo

Cons

- Cannot run any standard software
- Most python libraries not available

Device features

Console access

- Direct access to device console from webfrontend
- Remote control of the device via mouse and keyboard (only OpenVZ and KVM)
- Optional access via VNC (RFB protocol) using standard clients

Image upload/download

- Download and upload the virtual harddisk of the device
- Exchange the script of a programmable device
- Can be used for backups or to clone an existing device
- VMs can be imported into ToMaTo

Device templates

- Pre-installed standard operating systems
  - Various Linux distributions: Debian, Ubuntu, ...
  - Windows (only KVM, not public due to legal restrictions)
  - FreeBSD (only KVM)
- Lots of programmable device scripts: DNS server, DHCP server, Pingable node, ...
- Templates can be managed by admins

Connectors

Connectors forward and manipulate networking packets on their way between devices.

VPN networks

- Based on Tinc peer-to-peer VPN
- Configurable semantics: Hub, Switch or Router (needs additional configuration)
- Network is a virtual ethernet segment (not wit router semantics)
- Network is completely contained, no packets enter, no packets escape
- VPN can work between different sites
- Users can enter the VPN using VTun

External networks

- Connect devices to an external network, accessible on the host system
- Defined openings in the experiment topology
- Mosty used for Internet connections
- Experimental research networks can be accessed
- Can be used to include other resources into the topology (like Planet-Lab slices)

Connecting with devices

Devices can be used instead of connectors to provide special forwarding or manipulation schemes. Programmable devices offer an easy way to implement complex forwarding and packet manipulation algorithms.

VPN connector extra features

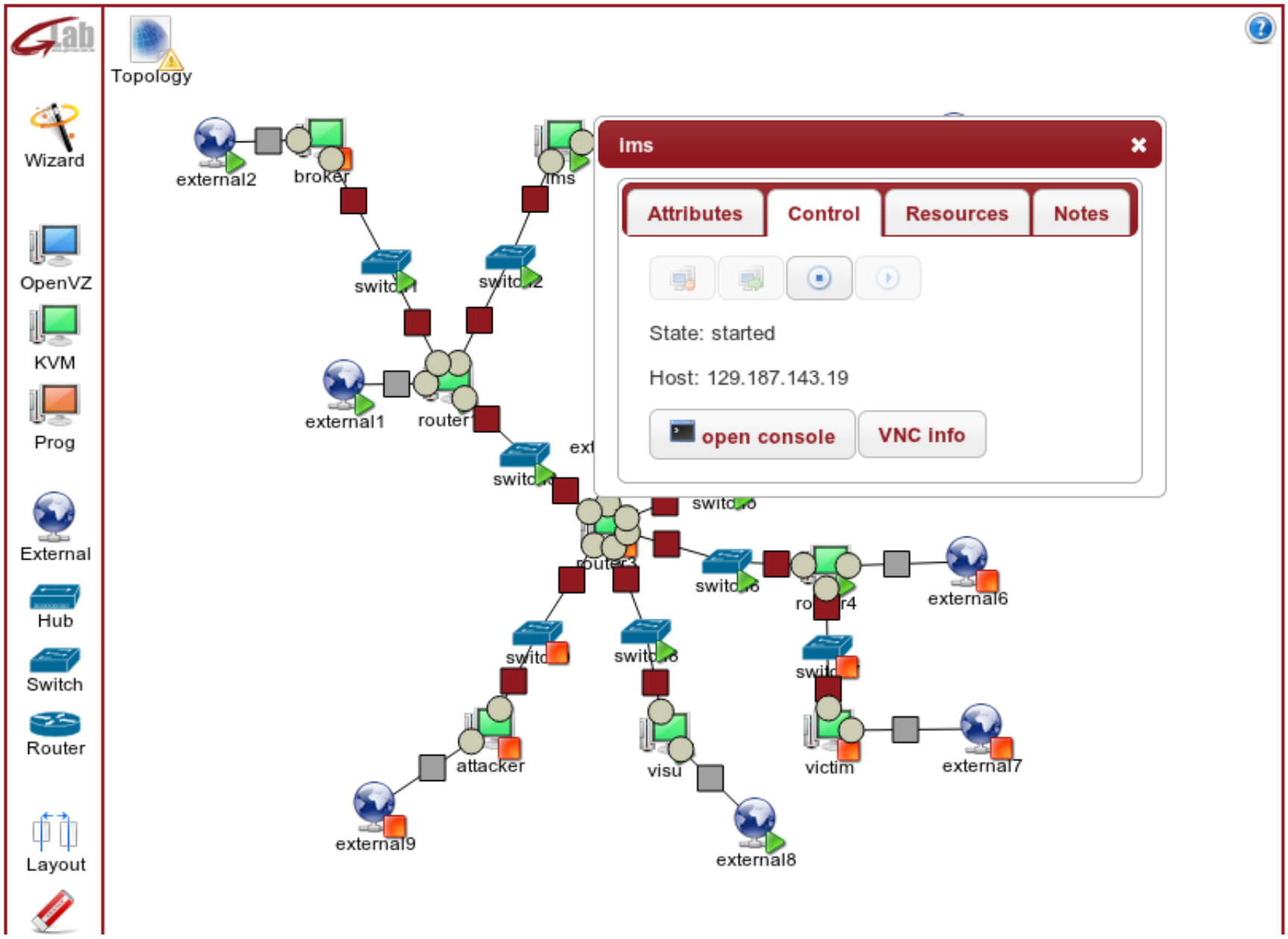
Link emulation

- Add delay to connections, configurable jitter, stochastic distributions
- Add packet loss to connections, optionally correlated
- Add packet dulpication and corruption, optionally correlated
- Bandwidth limits down to a few kb/sec
- All attributes can be configured on both directions individually

Packet capturing

- Capture packets on connections with full tcpdump filter support
- Captured packets can be downloaded in standard pcap format
  - Can be opened in packet analysis tools like Wireshark
  - Can be uploaded and analyzed in Cloudshark (integrated in ToMaTo webfrontend)
- Live-capturing feature sends packets directly to a running Wireshark instance

Topology editor



Remove

Obtaining and contributing

How to get ToMaTo

ToMaTo is Open-Source! It can be simply downloaded from the Github page. There is also a step-by-step tutorial on how to setup ToMaTo in a testbed. ToMaTo includes some nice features that make it pretty easy to install it in an experimental facility:

- All components packaged for debian (updates come automatically)
- Multiple authentication plugins: LDAP, htaccess, SQL-Database, Planet-Lab, ...
- Automatic checks and problem reports for the ToMaTo hosts

How to contribute to ToMaTo

As an Open-Source project ToMaTo is open for hints and contributions.

- Github offers an easy way to fork the project and offer contributions as pull requests
- The wiki is publically editable so everyone can help by adding to the documentation
- The issue tracking system can be used for bug reports and feature requests