Maxwell *Pro*

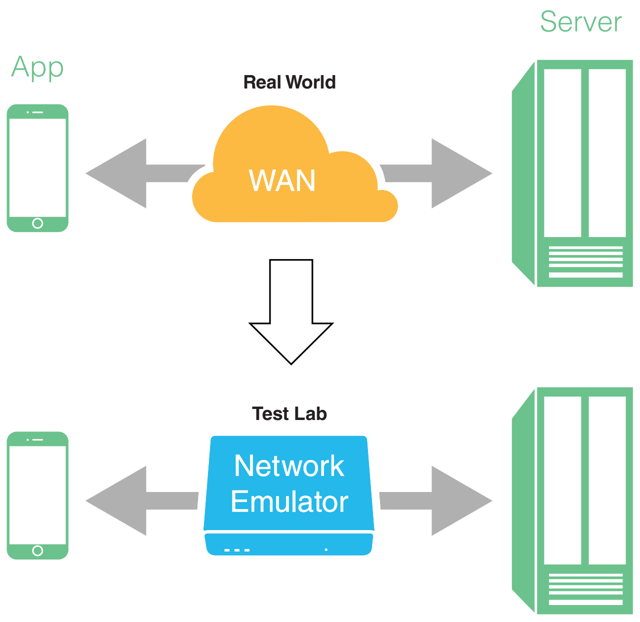
Network Emulator

Maxwell Pro Network Emulator helps network managers, software developers, and testers learn how their products will perform in real-world production networks, including satellites and the Internet. By intercepting and changing network flows, Maxwell can induce conditions that cause network congestion, slow links, time outs, and many other adverse network conditions. Network managers can then see the effects on the device or application in order to find and fix bugs, solve network problems, or learn the limits of device and application performance.

In addition, unlike the real internet, network emulators allow the operator to control these conditions so that products and apps can be subjected to controlled and repeatable tests, by routing selected packets through a series of impairment nodes.

The Result:

- ✓ Deliver robust, high quality products and applications
- ✓ Accelerate time to market
- ✓ Eliminate guesswork and surprises
- ✓ Completely characterize your app’s network performance



Create a Real-World Network in Your Test Lab

Emulate the Real World in Your Lab

The Maxwell Pro network emulator helps network engineers test and measure performance in order to identify and remove defects. Network emulators turn well behaved development and test networks into the kind of slow, congested, and less-than-reliable services encountered on the internet.

Maxwell Pro Test Suites (optionally available)

- SNMP
- TCP
- TLS
- IPv4 and IPv6
- SIP, RTP and RTCP
- UDP
- ICMP
- DHCP
- DOCSIS

InterWorking Labs®

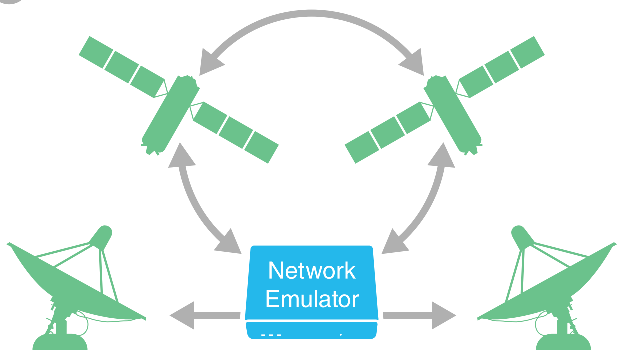
Scenarios

Select a pre-defined, network emulation scenario from our library. Some examples include:

Cross-Atlantic T1 ATM link	Cellular radio handoff
Low earth orbit satellite	VoIP: To remote office via WAN
Geosynchronous satellite	VoIP: multi-way conference over WiFi
Streaming media over satellite	Connection impairments
Periodic network downtime	Failover to backup servers

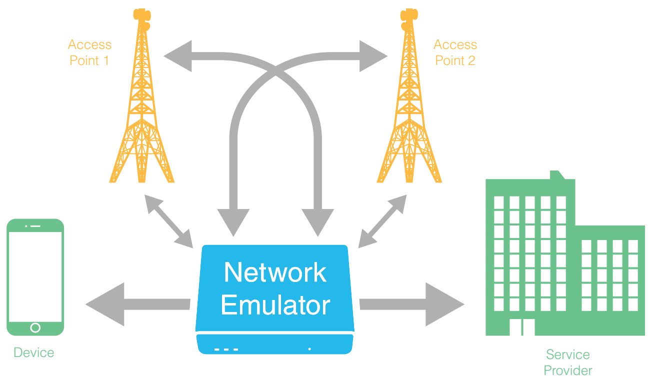
Pre-built library scenarios feature intuitive controls that appear on top of an end-to-end diagram of the whole network, to provide a bird's eye view. You can also add your own custom background diagram. Here are three examples from the library:

1



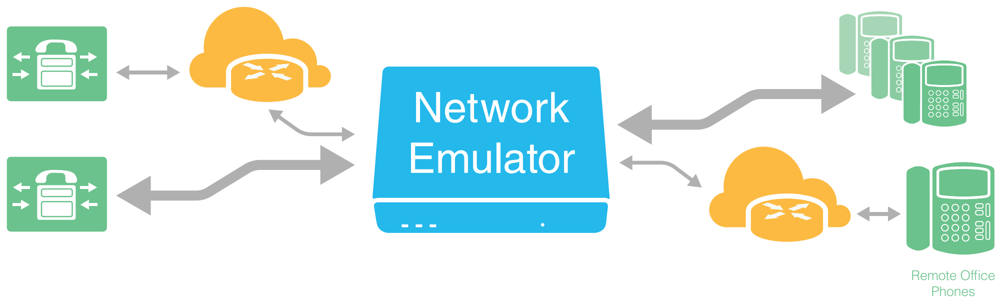
Multi-hop Low Earth Orbit Satellite

2



Cellular Radio Handoff

3



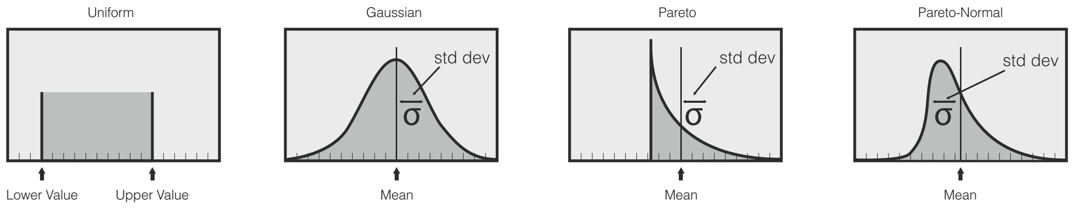
VoiP to Remote Office via WAN

InterWorking Labs®

Create Your Own Scenario

Emulate the real world in your test network by setting basic and advanced parameters for these impairments:

- Packet Drop/Loss: Add "burst mode" for a realistic emulation.
- Packet Delay/Latency: From zero to 10 seconds, with microsecond precision.
- Packet Jitter, with or without reordering. Specify a custom jitter distribution: Uniform, Gaussian, Pareto, or Pareto-Normal:
- Packet Duplication, with rapid back-to-back transmission.
- Packet Corruption: Specify packet-centric or bit-centric probabilities of corruption.
- Rate limiting: Choose From several queue management algorithms.



Model Worst-Case Situations:

- Turn on burst mode for any impairment, to simulate back-to-back real-world stochastic bursts.
- Add time-varying expressions to any parameter

Regardless of the production environment – anything from the routine to the extreme to legacy – Maxwell Pro permits customization to precisely emulate that world.

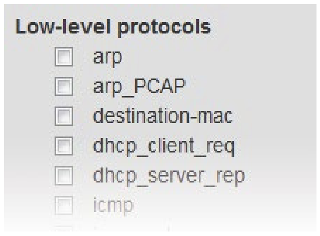
Incorporate Real or Simulated Traffic

Maxwell Pro accommodates all forms of real or simulated network traffic:

- Use real devices to generate packets, then impair only selected packets and protocols under consideration, while filtering out the rest.
- Or, use a packet generator to create simulated network traffic for the device under test, then apply impairments only to those packets, leaving the rest of the network unaffected.

Classify Network Traffic

Create filters for the packet classifier using the Classifier Filter Library, or create your own filter, then apply different impairments to different filter outputs.



Low-level protocols

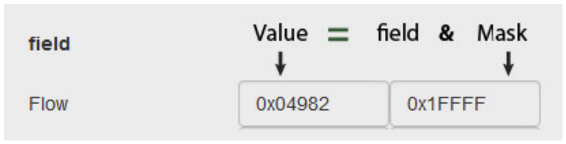
- ☐ arp
- ☐ arp_PCAP
- ☐ destination-mac
- ☐ dhcp_client_req
- ☐ dhcp_server_rep
- ☐ icmp

Examples of filter criteria include:

- IEEE 802 header fields: MAC source/ destination addresses, Ethernet type/ length, MPLS criteria, VLAN tags.
- IPv4 or IPv6 header fields, including chained headers, fragment status, and QoS.
- Higher level protocols: UDP or TCP port numbers.
- Data within the payload, for any layer.

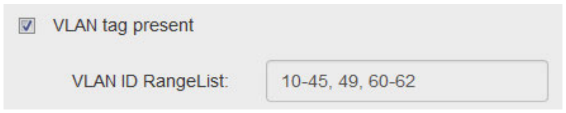
The Maxwell Pro Smart Classifier Filter System puts you in charge to use intelligent criteria for packet classification:

- Specify flexible value/mask comparisons: useful for matching addresses, QoS criterion, and payload bytes:



field	Value	Mask
Flow	0x04982	0x1FFFF

- Specify lists of values and ranges for comparison: useful for port numbers, VLAN IDs, and payload sizes:



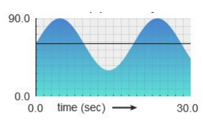
☑ VLAN tag present

VLAN ID RangeList: 10-45, 49, 60-62

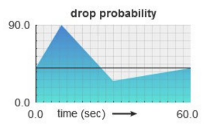
InterWorking Labs®

Add Time-varying Impairments to your Emulation

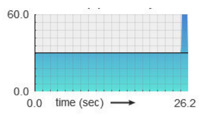
Real world networks change their behavior over time: excessive jitter, dropouts, or reductions in bandwidth can vary smoothly, or jump from one value to another, over the course of minutes, or days. Create a network emulation that varies the impairment parameters over any time period:



Equation



Piecewise Linear



Shaped pulses

Specify a time-varying expression for any metric, using equations, piecewise curves, or shaped pulses:

Examples

- Test your streaming clients: The device under test should handle time-varying impairments, and perform flawlessly, for days.
- Run your protocols continuously, while the time-varying impairments cover all corner cases – perfect for overnight stress testing.

Maxwell Pro Charts and Tables

Monitor your Maxwell Pro network emulation from a Chrome web browser on your tablet or desktop computer, with the **MPtable** and **MPchart** applications.

MPtable provides a bird's eye view of everything that is happening on the Maxwell Pro Network Emulator: Create tables, then expand the advanced tab to see metrics / statistics for all nodes in all flows. If a packet gets impairedanywhere in the network, you'll see a change in the statistics, in real time.

MPchart lets you drill down to see real-time graphs for a few selected statistics. Using linear or logarithmic axes, you can get a better sense of how metrics / impairments change over time. You can also display two metrics on the same graph to directly compare them.

Documentation

Maxwell Pro Documentation:

- Maxwell Pro Installation and Quickstart Guide
- Maxwell Automated TCP/IP Test Guide
- Maxwell Automated TCP/IP Tests
- Maxwell Application Programming Interface Guide
- Maxwell G.1050 Tests
- Maxwell Plugin Programmer's Guide
- Maxwell SIP Tests
- Maxwell Pro System Guide
- Maxwell TCP/IP Plugin Programmer's Guide
- Maxwell TCP Test Output Analyzer

Maxwell Pro Programmability

Customize Maxwell in one of four ways:

1. Use one of our existing scenarios as a base and make small modifications to tailor the impairments, as required.
2. Use the Maxwell graphical interface, select a set of desired impairments, save them, then define addresses and protocol filters.
3. Control Maxwell Pro from a Python, Perl, Tcl, Java, shell script or another application using the Maxwell Control API Guide.
4. Create a plugin in C or C++. Plugins are the mechanism for creating programs in the Maxwell environment.
5. Custom tests can be added to the TCP/IP protocol test suites using the Python language.

Maxwell Pro Network Emulator Specifications

Hardware Enclosure:

- Lunchbox sized: 12.9” x 8.7” x 3.8”
- Rackmount option: 15.7” x 11.8” x 1.7”
- Virtual Machine (VM) HyperVisor: software only

Interfaces:

- 10/100/1000 Mbits/sec
- 10 Gbits/sec for two ports

Operating System: Linux

Impairment Engine: Kernel-driven impairment engine

Transparency / Configuration: Layer 2 transparent bridge

User Interface:

- Command Line Interface
- Script control:
  - TCP API with Python or Java, using JSON
  - RESTful interface, using JSON
  - Load/Save configuration files in easy-to-use YAML format

Standard Impairments:

Packet Delay/Latency:

- Specify packet delay / latency
- Vary packet delay / latency over time *
- Add jitter to the delay, with or without packet reordering
- Resequence groups of packets to simulate queue bursts

Packet Jitter

- Jitter Distribution:
  - Normal / Gaussian / Bell curve
  - Pareto
  - Pareto-Normal
  - Uniform (equal)

Packet Duplication:

- Specify duplication probability
- Vary duplication probability over time *
- Enable bursts:
  - Specify burst probability
  - Specify burst window duration
  - Enable burst skew
- Vary burst probability over time: *

Packet Drop / Loss:

- Specify drop probability
- Vary drop probability over time * :
- Enable bursts:
  - Specify burst probability
  - Specify burst window duration
  - Enable burst skew
- Vary burst probability over time: *

Rate Limitation:

- Specify:
  - target rate
  - queue length
  - overhead bits
  - minimum packet size
  - maximum packet size
- Vary any of the above over time: *
- Support bit clocking rate control
- Support token bucket rate control
- Mark for drop or random early detection
- Specify transition action on queued packets
- Flush queue

Packet Corruption:

- Specify corruption probability
- Limit corruption to Ethernet payload
- Specify burst probability
- Vary burst probability over time *

Emulated Links / Multiple Flows:

- Unidirectional and bidirectional packet flow
- At least 5 flows in each direction; 10 total
- Larger models support > 64 impairment nodes

Classify packets based on:

- IPv4 header fields
- IPv6 header fields
- MAC source or destination addresses
- Ethernet type or length field
- IEEE 802 header fields (includes VLAN tags)
- Data within the packet

Classify packets based on fields in protocols:

- Low-level protocols:
  - arp
  - dhcp
  - icmp
  - mpls
  - spanning-tree
- IP layer:
  - IPv4 packets of varying lengths
  - IPv4 QoS bits
  - IPv6 fragment
  - IPv6 flow range

* This can be done manually, via a simple pulse model, via an equation, or using a list of value/duration pairs

- TCP / UDP layer:
    - Any TCP packet
    - TCP port numbers
    - Any UDP packet
    - UDP port numbers
    - http
    - DNS
    - SIP
- Applications layer:
    - VoIP
    - VoIP with QoS bits in various configs
    - Apply different impairments on each emulated link

Manipulate contents of headers and payloads:

- Randomly corrupt a packet
- Intelligently corrupt a packet
- Create a packet and selectively insert into the flow
- Dynamically create and change distinct flows of packets
- Inspect packet headers to select packets for modification
- Rewrite packet headers
- Rewrite packet data

Alter Impairment:

- Modify a live packet from a real-world network conversation; no need to create a new packet from scratch.
- Supports "man in the middle" operation by sitting between two devices, intercepting the network traffic; it is not limited to creating a packet to send it to an end point.
- Permits multiple, concurrent packets to be altered; operation is not limited to one packet at a time.
- Changes to the Alter code is made via a graphical interface that provides guidance and feedback to the engineer.
- Replicate a specific fuzz test to verify the robustness of a network device.
- Create a new set of tests for a particular feature using the simple Python programming language.
- During the network conversation:
    - Insert or append bytes into a protocol packet's payload or header.
    - Remove bytes from a protocol packet's payload or header.
    - Find and remove specific IPv4, IPv6, or TCP options.
    - Rewrite any header field or payload bytes, such as source or destination IP addresses.
    - Automatically update protocol length fields after byte insertions and deletions.
    - Perform operations on byte strings inside the packet, such as addition, bit-wise "and", "or", and "xor".
- Recompute all checksums upon the engineer's request.
- Perform alterations to a copy of the original packet and send the altered copy before or after the original packet - thus allowing insertion of entirely new packets into the network dialog.
- Supports tunneling
- Any Alter functions defined by an engineer may be saved and reloaded at a future time.
- The Alter functions may also be combined with other standard impairments such as jitter, delay, re-ordering, etc.
- Compared with other commercial and open source tools (such as Scapy), the Alter function:

Maxwell Patent No 7310316

Testing device
US 7310316 B2

ABSTRACT
A test device (21) sits between two or more nodes (20, 22). The nodes (20, 22) communicate in conversations, according to some predetermined protocol. The test device (21), under user control, may introduce jitter, drop packets, create new packets, reroute packets, and reorder packets in the conversations. Particular conversations are detected and tracked by respective virtual state machines (38, 39, 40) within the test device.

Kings Village Center #66190
Scotts Valley, CA 95067
iwl.com
+1.831.460.7010
info@iwl.com