Network Emulation in the NS Simulator.

Kishore Ramachandran
Emulation – the why and the how.

- **Motivation (Why?)**
  - Simulations make simplifying assumptions.
  - Results not always basis for real-world deployment.

- **Possible approaches (How?)**
  - **Network emulation** – Interaction of simulated components with real-world components.
  - **Environment emulation** – Build an environment where real-world implementations may be executed directly from the simulator.
NS Emulation (NSE)

Network emulation is approach adopted by NSE.

- First approach adopted because
  - Large existing base of protocol implementations in the simulator.
  - Difficulty in implementing all these in operational systems.
- Seen as a logical extension of the ns-2 simulator.
NSE – what can be done.

- **Test performance of** experimental algorithms and **protocols implemented in the simulator using real-world traffic.**

### Hypothetical scenario I

**Traffic**

**WINLAB student working.**

**Simulation** of a router running experimental AQM scheme.

**Traffic**

**Web server (ACM, IEEE, etc)**
NSE – what can be done.

Hypothetical scenario II

Web client using experimental transport protocol implemented in simulator.

Web server (ACM, IEEE, etc) implemented on top of experimental transport protocol implemented in simulator.
NSE – what can be done.

- Evaluate real-world implementations against repeatable interference generated in a simulator.

Hypothetical scenario III

- **WINLAB student working.**
- **Simulation** of a linear multiple bottleneck topology to see the effect of cross-flows on a real-world TCP flow/s.
- **Web server (ACM, IEEE, etc)**.
NSE – Challenges

- Synchronization of event scheduler to real-time.
- Provision of packet generation and capture facilities.
- Translation between internal packet format of simulator and that of real-world implementations.
Synchronization of event scheduler to real-time.

- Scheduler operates in virtual time.
- Current time is immediately advanced to the dispatch time of next pending event.
- Asynchronous arrival of I/O events can restrict execution time of simulation.
Use wall-clock time for the dispatch of events.

Handle asynchronous arrival of I/O events in addition to events in the simulator.

Drawbacks with method used:

- Scheduling not exact; Under sufficient load, scheduler may fall behind.
- Scheduler implemented as a loop due to the Tcl I/O event notifier function that does not provide maximum blocking time.
Network objects provide access to network packets.

Use UDP/IP sockets API for UDP/IP datagrams and raw IP packets.

Use libpcap for frame-level capture and filtering.
NSE – Challenge III

Translation between internal and external packet formats.

- NS-2 uses its own network layer address format.
- Does not understand standard IP address formats.
- Combines port identifiers with network layer node address.
NSE – Solution to Challenge III

- **Tap agents** are used to *encapsulate/de-encapsulate* live network traffic inside/from simulator packets.
- They are network-independent and can be used at various protocol layers.
- Each tap agent is associated with a network object.
NSE – Complete picture.

Tap Agents convert network packets into simulator compatible packets.

Associated Network Objects provide live network access.
NSE – Modes of operation.

- **Opaque mode.**
  - Network data treated as uninterpreted packets.
  - Real-world protocol fields not directly manipulated.

- **Protocol mode.**
  - Ability to interpret and/or generate network data with standard protocol field assignments.
  - Protocol-specific traffic manipulation scenarios are possible here.

Opaque mode is well-supported.
Very limited support for protocol mode.
How NSE fits into EMULAB.

**Thank you ORBIT EMULAB team.**
Future areas of study.

- More detailed EMULAB-NSE details.
- Symphony protocol development environment as an example of environment emulation.
- Wireless emulator from Lucent, Bell Labs.