Agenda Items:

- Protocol Boosters, relationship with A.N.
- Dissemination in Europe
- Secure Active Network Element (SANE)
- Technical Problem Areas for A.N.
- Program Management Challenges
- What’s coming next from Penn
Protocol Boosters and Active Networks

Design Methodology versus Infrastructure
Protocol Design: Current Methodology

- **Pessimistic Design Style**
  - Assume worst-case
  - Pare away functions to get “fast-path”

- **Optimizations Fragile**
  - Environment Changes (WWW)
  - Common Cases Change (delay, loss, ...)
  - Things can break BADLY! (try at home :-())
Protocol Boosters

- Protocol Elements added ‘‘as-needed’’
- Example of “optimistic” design method
- Useful to maintain common case
Performance Potential:

- Thruput: TCP, TCP/FEC, Hybrid
Examples (and leadin to A.N..)

- Implemented over IP on FreeBSD
  » Encryption + Compression Boosters
- FEC Booster at Bellcore
- Hardware Support: The P4*

Q: What’s the network infrastructure needed to support this idea & others???

*see http://www.cis.upenn.edu/~boosters/boosters.html
European Dissemination

4/3/97-8/31/97
Strategic Goal: Enlist others to Solve Hard Problems

- **Formal Methods:** Talk to Milner’s group
  - Pi-calculus to specify distributed behavior
  - Need for first-class time types
  - Integration with mobile work (e.g., Cardelli)

- **Protocol Boosters and A.N.**
  - HIPPARCCH ‘97 Invited Speech

- **SwitchWare and Network Evolution**
  - U.C.L., Lancs, Sussex, Glasgow, BT Labs
Possible Follow-ons

☐ Lancs and Sussex: EPSRC $$ for A.N.

☐ Cambridge DCAN project
  » Restrict Programmability to Admin. Plane

☐ Cambridge Nemesis project
  » Ideal for SwitchWare approach
  » Investigating collaboration
    – upcoming BAAs???
Secure Active Network Element (SANE)

From Bootstrap to Operation
AEGIS Secure Bootstrap

- Integrity Guarantees for Dynamic Integrity Checking

- Trusted Repository
- SwitchLets
- O.S. (BSD)
- ROMs, Boot Block, ...
- BIOS Sec. 1
- POST
Secure Active Network Element (SANE)

- “Trust, but Verify” (U.S. Nuclear Policy..)
- Node-Node Authentication
- Recovery
- PLAN
- Caml/O.S.
- AEGIS
- Dynamic Integrity Checks (Maybe per-packet/SwitchLet?)
- Static Integrity Checks (Done Once)

See http://www.cis.upenn.edu/~waa
Penn/Bellcore SwitchWare Project: A Language-Oriented Model

- **Switchlet Language for users (SL)**
  - formal semantics restrict programs
  - *(Boosters make *fine* Switchlets :-)*
  - Prog. Language for Active Nets (PLAN)

- **Wire Language for communicating (WL)**
  - formal semantics across boundaries
  - Java or Caml bytecodes

- **Infrastructure Language for Virtual Machine (IL)**
  - formal semantics supported on metal: run-time
Active Bridging Paper went over well

A.N. Debate stimulated arguments

- Not clearly won/lost
- Considerable animosity about $$
- Pointed out need for compelling applications
Current Software

- Active Bridging

See http://oilhead.cis.upenn.edu/~salex
Protection of Resources?

 Dynamic versus Static Restrictions?

Programming Language Environment (PLAN)

O.S. Kernel, e.g., Linux, Scout, Nemesis

Device Driver

Device Driver
Example Problem #1: MUX

Want to assign L3 bandwidth 66%/33%
Example #2: Multicast

- Program copies L3 (in) to L1, L2 (out)

Is this Program “safe”?
Restricting Programs

- Node safe versus network safe

Diagram:
- All Programs
- Node Safe Programs
- Network Safe Programs
Model->Modules->Actions

- Syntax, Semantics, Node vs. Network
- Example: Securing a Network

1. System Model
2. Modules loaded into nodes
3. Resulting in a robust Network