

-
-
-

ABone Progress & Status Report

Active Nets PI Meeting, Orlando, FL
Dec 4, 2001

Steve Berson, Bob Braden (ISI), Steve Dawson (SRI)
Craig Milo Rogers (ISI), Fred Gilham (SRI)
Hugh Choi, Yu He, Siva Jayaraman, Sarjana Sheth (ISI)



The ABone...

- Is a testbed for active networking
- Is composed of diverse OS platforms distributed across many organizations.
- Executes multiple Execution Environments (EEs).
- Is overlaid on the Internet, but also includes some other testbeds (CAIRN, Emulab)
- Is monitored by ABone Coordination Center (**ABOCC**).



Outline

- Brief introduction
 - (assumes that you stayed awake during the two previous PI meetings)
- Brief Status
- Progress
 - Anetd V2
 - Security
 - AMP Node OS
 - XBone
- Future Directions

ABone Architecture

- Locally-administered nodes (private root pwd) using Linux, FreeBSD, or Solaris.
- **Core** nodes always available; **edge** nodes come/go.
- On core nodes: EEs are remotely managed using **Anetd** and **ABoneShell**.
 - Execute in user mode.
 - Install, restart, kill, configure, debug, monitor EEs.
- “Permanent” EEs: always available for AA developers.
- Variety of network I/O modes

Network I/O Modes

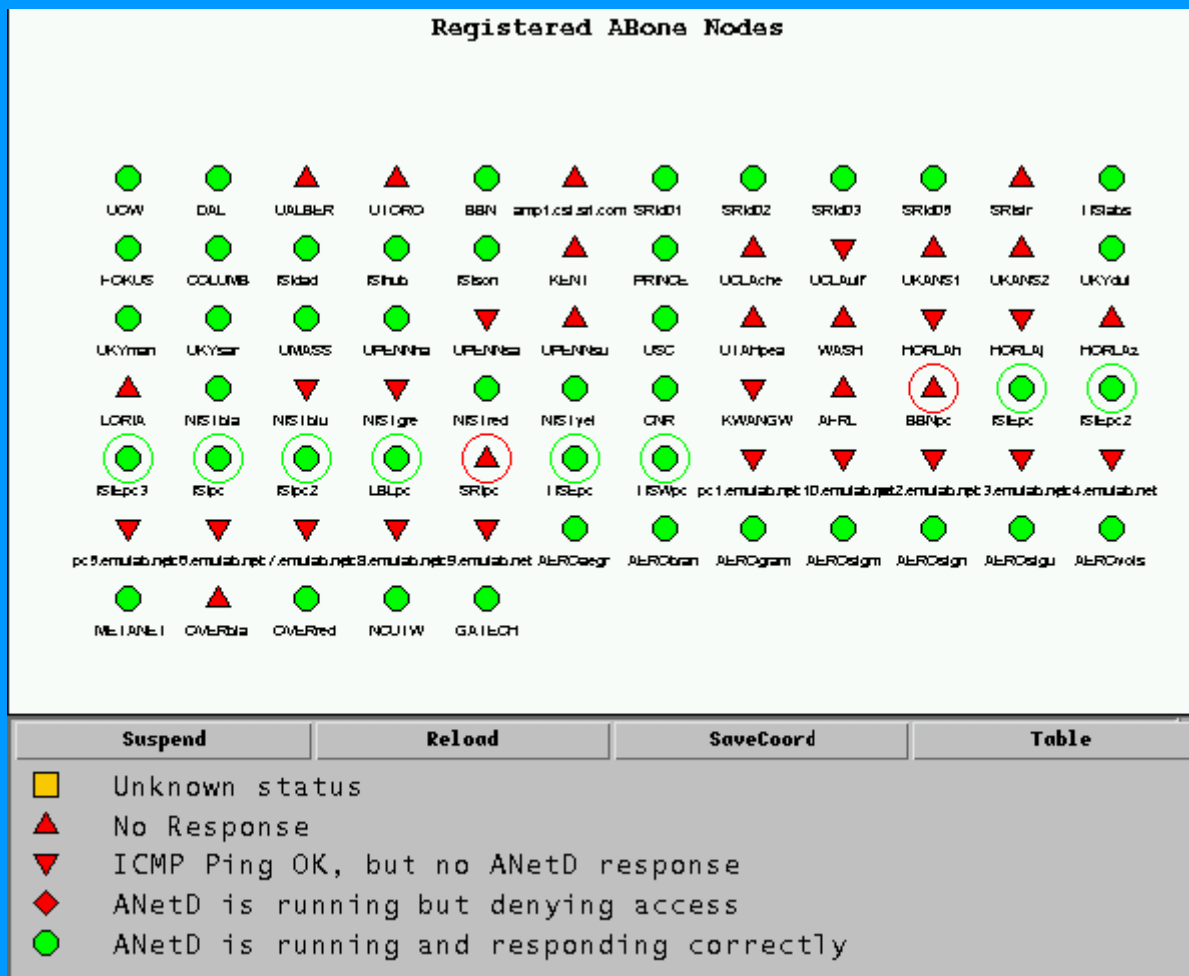
- Virtual connectivity (ANTS & ASP EEs)
 - UDP tunnels, per-EE virtual topology & virtual network addr space.
 - To EE: “if/ipv4/UDP”, demux(ANEP typeid)
- Native IP connectivity (ASP EEs)
 - Running in the Internet ‘porridge’ with real IP addresses.
 - To EE: “if/ipv4”, demux(IP proto ID, etc.)
- Link-Layer connectivity (soon)
 - To EE: “if”
- Virtual native IP connectivity (Xbone)
 - “if/ipv4/ipv4/esh-ipsec/ipv4” -> “if/ipv4”



ABone Overview

- ABOCC: ABone Coordination Center.
- Registration service for nodes and users.
- 7 Unix accounts on each node: abocc, anpub, anee1-5.
- ABone security: PK crypto to control who can load EE from where; ACL and TCL files in each node.
- ABoneShell: front end to Anetd and driver for management EElets.
- Web-based monitoring of nodes, Anetd, EEs.

ABone Node Status



ABone Node Status

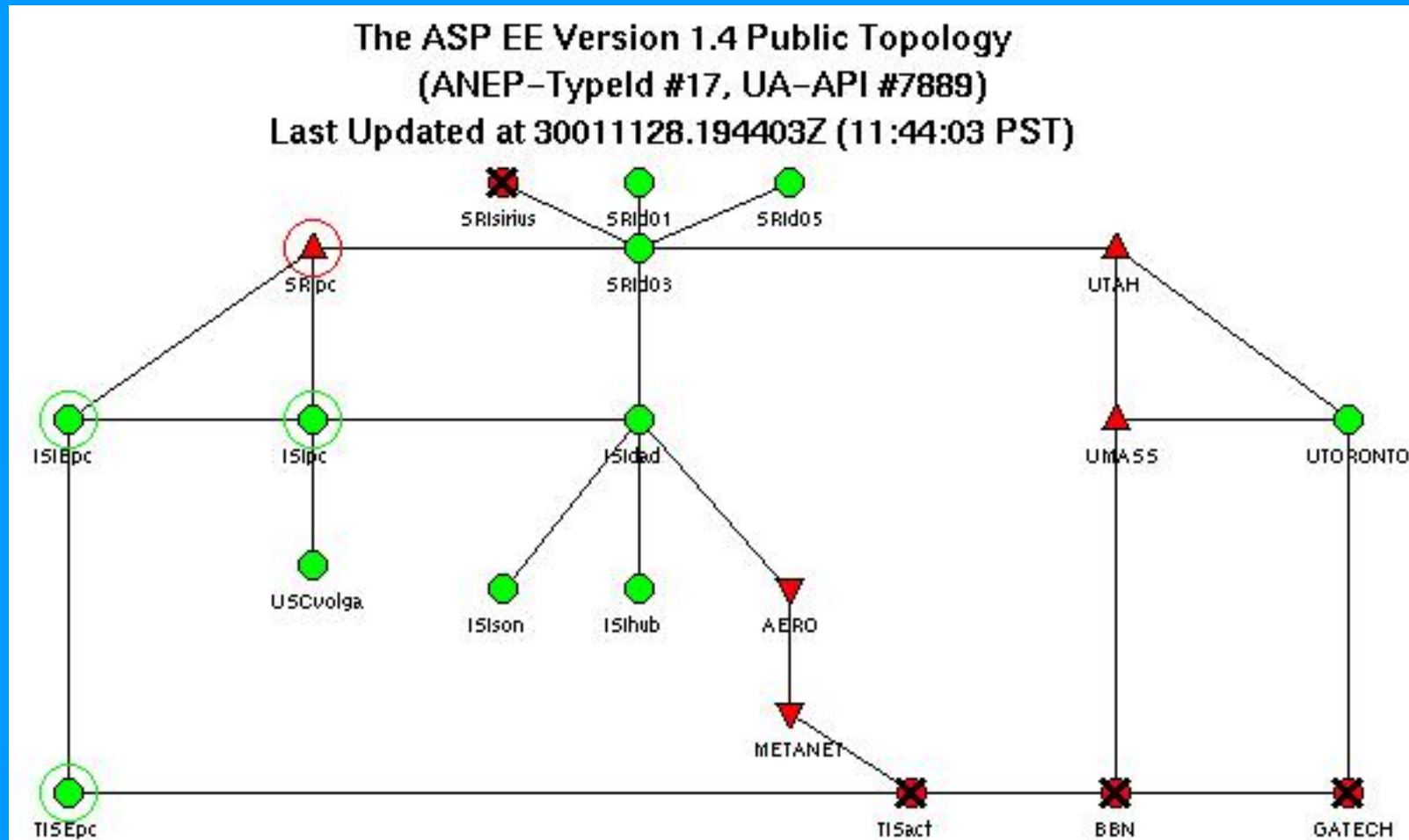
- Reported in June 2001: 77 registered nodes
 - No IP echo: 17
No Anetd echo: 17
Anetd OK: 43
- On November 28, 2001: registered 98 nodes
 - No IP echo: 16
No Anetd echo: 23
Anetd OK: 59
- On October 10, 2001: 96 registered nodes
 - US: 82 (78 DARPA, 14 CAIRN)
France: 4
Canada: 3
Italy: 2
Australia, Finland, Germany, South Korea, Taiwan: 1 each.

Detailed Monitoring example

Status report generated at 23:34:54 GMT on 29-May-2001 (16:34:54 PDT)

Node Name	OS	Version	IP/ICMP Echo (Ping)	anpub	abocc	anee1	anee2	anee3	anee4	anee5
pushkin.abone.uow.edu.au	linux	2.4.0-test11	Up	Up	Up	Up	Up	Up	Up	Up
nemain.cs.dal.ca	linux	2.2.16	Up	Up	Up	Up	Up	Up	Up	Up
newby.cs.ualberta.ca			Down	Down	Down	Down	Down	Down	Down	Down
abone.nal.utoronto.ca			Down	Down	Down	Down	Down	Down	Down	Down
core-abone-bos1.bbn.com	linux	2.4.2-2	Up	Up	Up	Down	Up	Up	Up	Up
amp1.csl.sri.com	exos		Down	Down	Down	Down	Down	Down	Down	Down
d01.csl.sri.com	bsd44	4.2-RELEASE	Down	Down	Up	Up	Up	Up	Up	Up
d02.csl.sri.com	bsd44	3.4-RELEASE	Down	Up	Up	Up	Up	Up	Up	Up
d03.csl.sri.com	linux	2.2.14-5.0	Down	Up	Up	Up	Up	Up	Up	Up
d05.csl.sri.com	linux	2.2.14-5.0	Down	Up	Up	Up	Up	Up	Up	Up
sirius.csl.sri.com	solaris	5.7	Down	Up	Up	Up	Up	Up	Up	Up
active.netsec.tislabs.com	linux		Up	Up	Up	Up	Up	Up	Up	Up
abone.fokus.gmd.de	linux		Up	Up	Up	Up	Up	Up	Up	Up
view.cs.columbia.edu	linux	2.2.18pre20	Up	Up	Up	Up	Up	Up	Up	Up
dad.isi.edu	linux	2.2.16-3	Up	Up	Up	Up	Up	Up	Up	Up
hub.isi.edu	solaris	5.6	Up	Up	Up	Up	Up	Up	Up	Up
son.isi.edu	bsd44	4.1.1-RELEASE	Up	Up	Up	Up	Up	Up	Up	Up
oahu.medianet.kent.edu			Down	Down	Down	Down	Down	Down	Down	Down

Permanent EE Topology



ABone Users

- ABOCC monitoring
 - Comment: Sometimes “Ping” really IS a **killer** application!! (ANTS 2.0).
- BBN Sencomm Project -- Network Management
- Aerospace Corp. -- Active Filter Signaling testbed
- Univ Kentucky (Mary Bond) -- Concast
- AMP nodeOS (soon!)
- Ga Tech: CANES EE (soon!)
- USC grad students (ongoing)

ABone Progress -- Topics

(a) Anetd V2

- ABCd: ABone Control daemon [SRI: Dawson]
- netiod: Network I/O daemon [SRI: Berson]

(b) ABone security design and implementation

(c) AMP nodeOS as ABone node

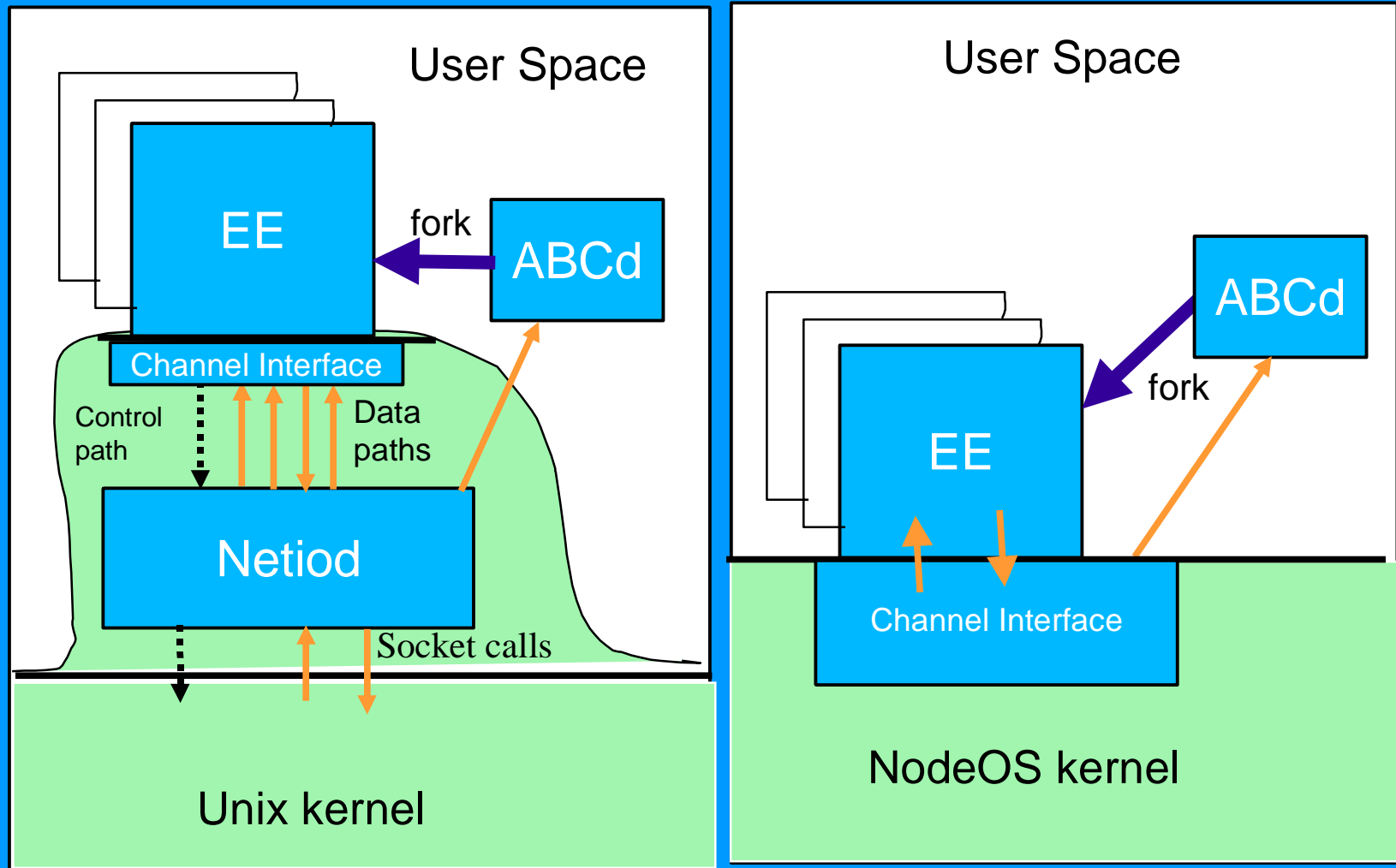
(d) ABone over XBone [ISI]

(e) Applications

Anetd v2

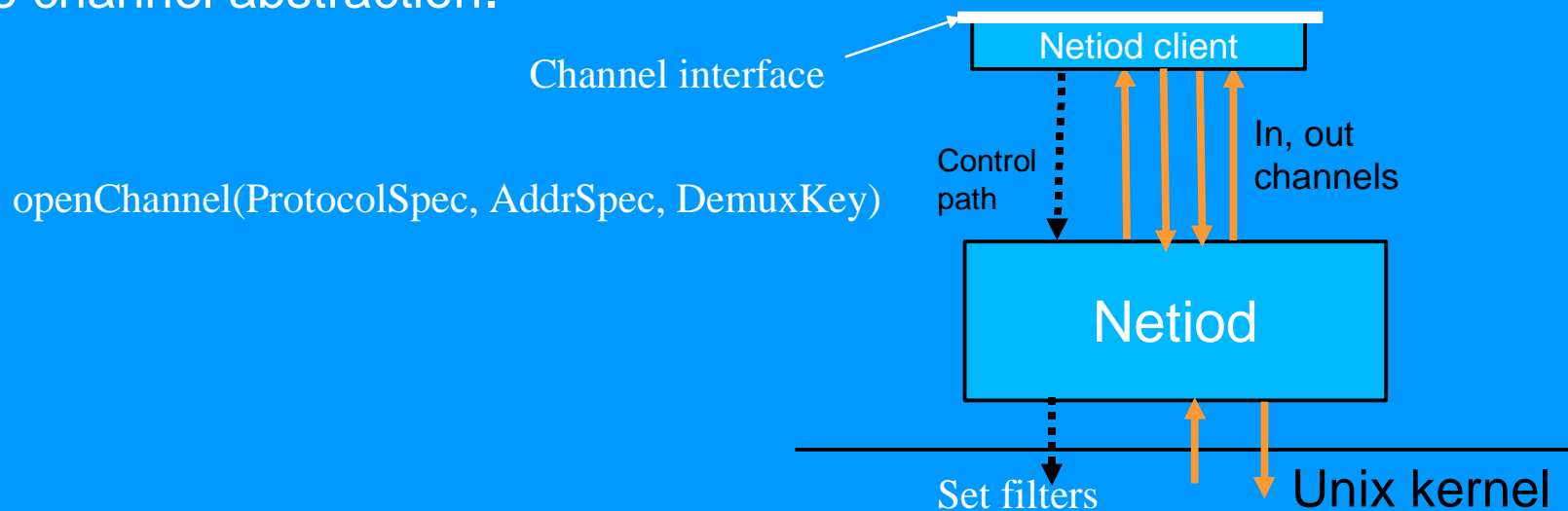
- Anetd 1 -> ABCd + netiod (for Unix)
 - ABCd = ABone Control daemon
 - EE management functions of Anetd
 - Redesigned for easier maintenance and installation
 - Steve Dawson will talk about it
 - Netiod = Network I/O daemon
 - Handles network I/O for Unix platforms
 - Runs as root
 - Presents ~ nodeOS channel interface to EEs
 - Hides from EEs the differences in kernel filtering, etc.

Netiod and ABCd



Network I/O Daemon: Netiod

- Common interface to Unix network I/O and filtering
- Objective: safely run as root (to access kernel filters)
- Intend to support Linux, FreeBSD, and Solaris
- Control path: local TCP connection; trivial protocol based on the channel abstraction.



Channel Issues Encountered

Netiod deviates from nodOS channel spec. Many differences are problems that will be fixed in the nodeOS spec.

- inChannel: Upcall must pass values of wildcard legacy hdr fields
 - **Netiod: Header struct precedes data packet in inChannel.**
(Equiv to passing AddrSpec in upcall)
- outChannel: Want to bind AddrSpec values in send() call -- e.g., specify dest address as in Unix sendto().
 - Netiod: Header struct can precede data in output channel.
- Need additional legacy header fields passed up.
 - **Netiod: defines additional fields in "AddrSpec", e.g., TTL**

More Channel Issues

- “Header” support: “if/ipv4h”, “if/ipv6h”: do IP processing but don’t strip IP header.
 - Netiod: pragmatic, until support “if”.
- *Real* Computer Scientists use LSI/11s and IPv4 for their research. But just in case...
 - Netiod: implements IPv6 as well as IPv4.
 - Netiod: allows “[]” to delimit IP numeric address, to avoid conflict with “:” delimiter of AddrSpecs.
- inChannel: Multiple match semantics ???
 - Netiod: Allows multiple matches and will deliver multiple copies.

More Channel Issues

TCP Channel semantics

- Active/passive open?
 - Netiod: open inChannel => passive, open outChannel => active
- TCP channels half/full duplex?
 - Netiod: Can send on TCP inChannel and receive on TCP outChannel.
- Signal TCP close?
 - Netiod: Closes loop-back local TCP connection.
- Demux keys in TCP channels?
 - Netiod: disallows.

Netiod

- Protocol Specs currently supported:

- Datagrams: if / ipv4 / udp
if / ipv6 / udp*
- Connections: if / ipv4 / tcp
if / ipv6 / tcp
- Network layer (“raw mode”):
if / ipv4[h]
if / ipv6[h]*

*[*Some restrictions may apply, due to Linux problems]*

- User-Mode Option

Convenient to use netiod as universal interface. Don't need to run as root for “if/ipv{4|6}/{UDP|TCP}”, so added user-mode option.

Netiod Implementation

- Tested on Linux and FreeBSD
 - Linux: 2.2 kernels using ipchains.
 - Linux: 2.4 kernels using iptables (or ipchains with reduced function)
 - FreeBSD: 3.x and 4.x kernels using ipfw DIVERT sockets.
- Limited releases July, Sept to NAI, Ga Tech
We appreciate their bug fixes!
- Full release -- real soon...
 - IPv6 support
 - Demux key support (thanks to Andrew Purtell, NAI !)
 - Duplex TCP support
 - Hardening and cleanup

Active Network Security in the ABone

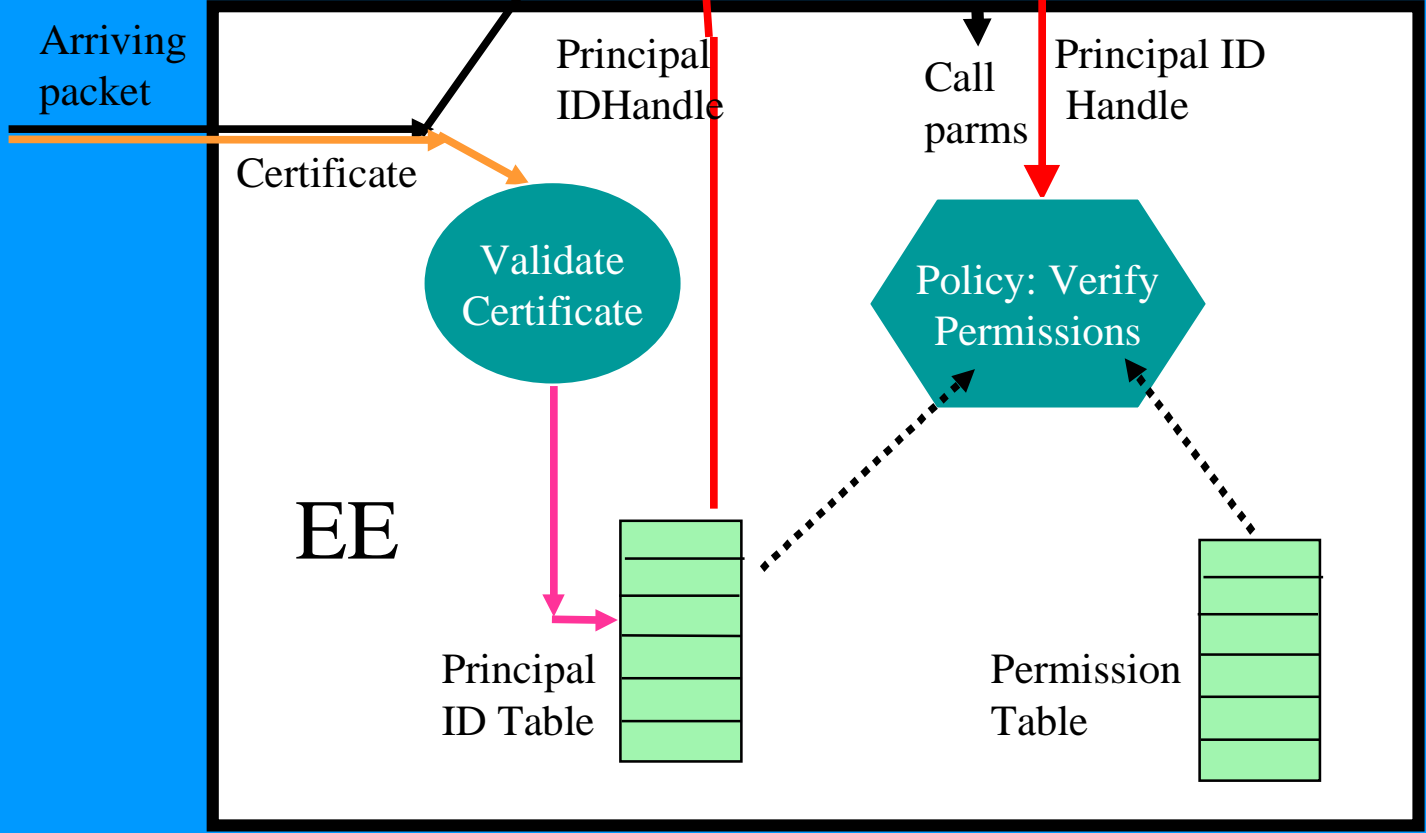
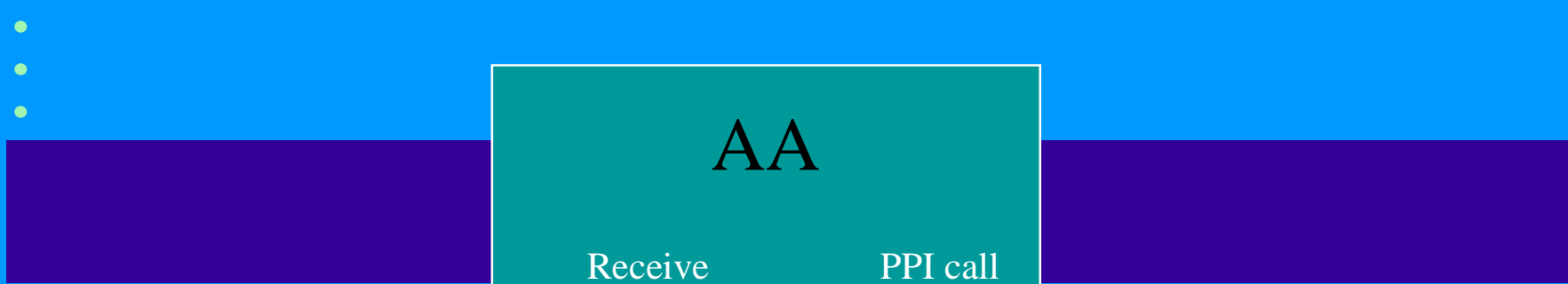
- Want to build realistic mechanism based on ANSA
(*active networks security architecture*).
- ABone: Don't want to modify Unix kernels
 - => cannot do security in nodeOS
 - => Must trust EEs to sandbox AAs, intercept all system calls.
 - Not ultimately secure, but realistic in practice.
- Conclusion: In ABone, do security in EEs.

ABone Security

- Hop/Hop Security Problem
 - Authenticate neighboring AN nodes
 - To protect legitimate AAs from trash packets
 - An *essentially* solved problem: just pick a candidate
 - (e.g., SANTS or AMP code?)
- End-to-End Security Problem
 - Certificate => principal, policy determines what AA can do.
 - Two authorization sub-problems:
 - Resource consumption
 - Privileged EE calls [“PPI”]

End-to-End Security

- Want to develop and deploy E2E security in ANTS and ASP EEs in the ABone
- Use very simple policy mechanism -- e.g., EE-specific files that map principal IDs to privileges.
- Build common library of Java code for this purpose.
 - Had hoped to deconstruct SANTS; now looking at deconstructing code from AMP.
- Use lazy validation of certificates to provide efficiency.
- Punt on resource consumption limits at first.



AMP Nodes in the ABone

- What does this mean?
 - AMP runs Anetd
 - AMP supports both ANTS and ASP EEs
- Aaaaalmost there...
 - Latest Anetd runs on AMP at SRI; firewall issues at NAI.
 - Runs latest ANTS and ASP EEs at NAI, without Anetd.
 - Runs the C version of ABoneShell server, not the Java version.
- Reaaaaalllll soooooon, noooooow...

New ABone EEs and Applications

- IPv4/IPv6 header translation
 - Useful to drive IPv6 support
 - Small part of larger issue: how useful is active networking for experimenting with, and/or deploying, new network-layer protocols?
- AER/NCA
 - Working on porting it from ENHANTS to ASP EE.
 - Currently incomplete
- Concast
- CANES EE

ABone over the XBone

- XBone configures virtual overlay networks.
- Concept: Use XBone to generate virtual native IP ABone configurations, using IP/IP encapsulation.
- Obstacles:
 - Minor Anetd enhancements (DONE)
 - Minor XBone enhancements (DONE)
 - XBone needs to use ABone ACL/TCL security (TO BE DONE)
 - XBone enforces complete virtual isolation -- cannot share anything
=> Need convenient way to configure user's version of common ABOCC repositories: user's EE server, ACL, TCL. (TO BE DONE)
- Status: prototype testing of Anetd in ABone done.
 - Need integration testing, tools, docs.



Future ABoneTasks

- Anetd v2 (ABCd & netiod) completion and transition
- Usability improvements
- Install more EEs
- Install more nodeOS's
- Xbone integration
- Scheduling & auto-configuration of nodes
- Active networks security
- Development of AN debugging tools



ABone Issues

- Regular [mbone/phone] conferences of Abone users?
- Usability
 - Better documentation
 - Debugging facilities
 - Topology generation
- Security: (thou shalt...)
- How much experiment concurrency will actually be possible?
 - Two possible ABone usage models -- need both?
 - Static public topology => some nodes down [*current model*]
 - Dynamically configured, private topologies, scheduling.
- [How] can ABone support loadable kernel modules?



Issue: Integration of Node OSs

- [To what extent] can EEs be portable across Node OSs and Unix/Anetd?
- E.g.: EEs written to Node OS spec; in Unix, run on shim layer that maps Posix into Node OS interface?
 - Probably limitations -- how severe?
 - E.g., implementation of general packet filtering semantics of Node OS may require kernel changes in Linux and/or FreeBSD.
- Accept limitations of Unix systems vs. modify kernels ?



Issue: Debugging AAs and EEs

- Overlaps with network management
- EE debugging
 - Exec EE under jdb/gdb on each node [done]
 - Pipe stdout back to client
- AA debugging
 - Harder, in general: debugging distributed algorithms
 - May need both code breakpoints and “packet breakpoints”
(*Sequester* input packets in EE or in channel, release manually).
 - Another technique: *active probes*

Issue: Active Network Management

- We mean both parsings of the title above.
- Want to use active network techniques for ABone mgt.
 - Active nets => don't need protocol standards;
active NM should => don't need MIB standards.

[“Free us from the tyranny of the MIB and ASN.1!”]
 - Active traps (currently called “probes” at ISI): code that monitors specific state in node, sends spontaneous or periodic msgs.
Needs some design work.
- Need standard instrumentation interfaces