Reflections on Active Network Trust

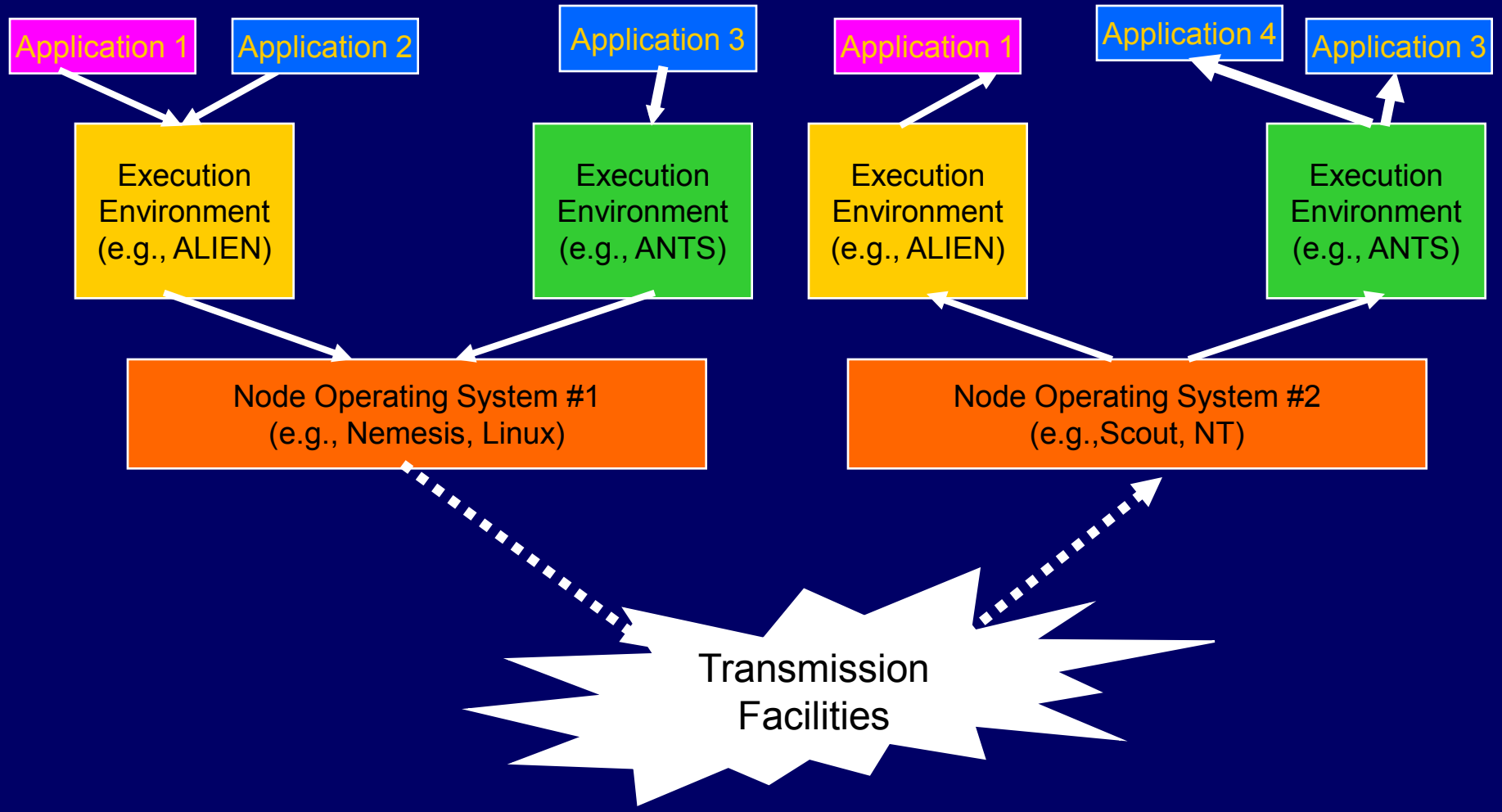
OpenArch Panel, March 26th, 1999

Jonathan Smith
University of Pennsylvania
http://www.cis.upenn.edu/~jms

Inspiration: Ken Thompson Turing Award Paper

- "Reflections on Trusting Trust"
  - Example of self-replicating compiler virus
  - Lesson: You are *trusting* infrastructure!
- A.N. concern so far: trust of code
  - Can the code trust the A.N.?
- Goal in an A.N.:
  - Either operate in untrusted environments
  - Or establish web of trust

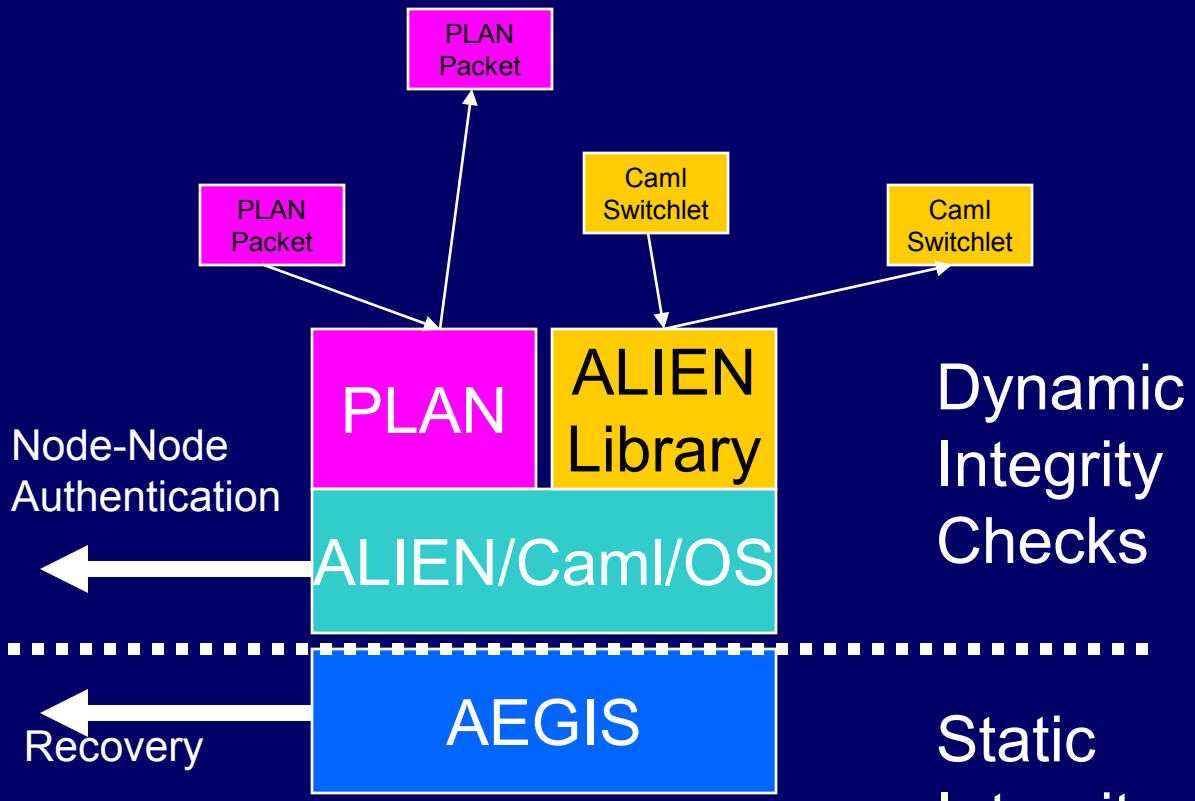
A.N. Internode Interoperation



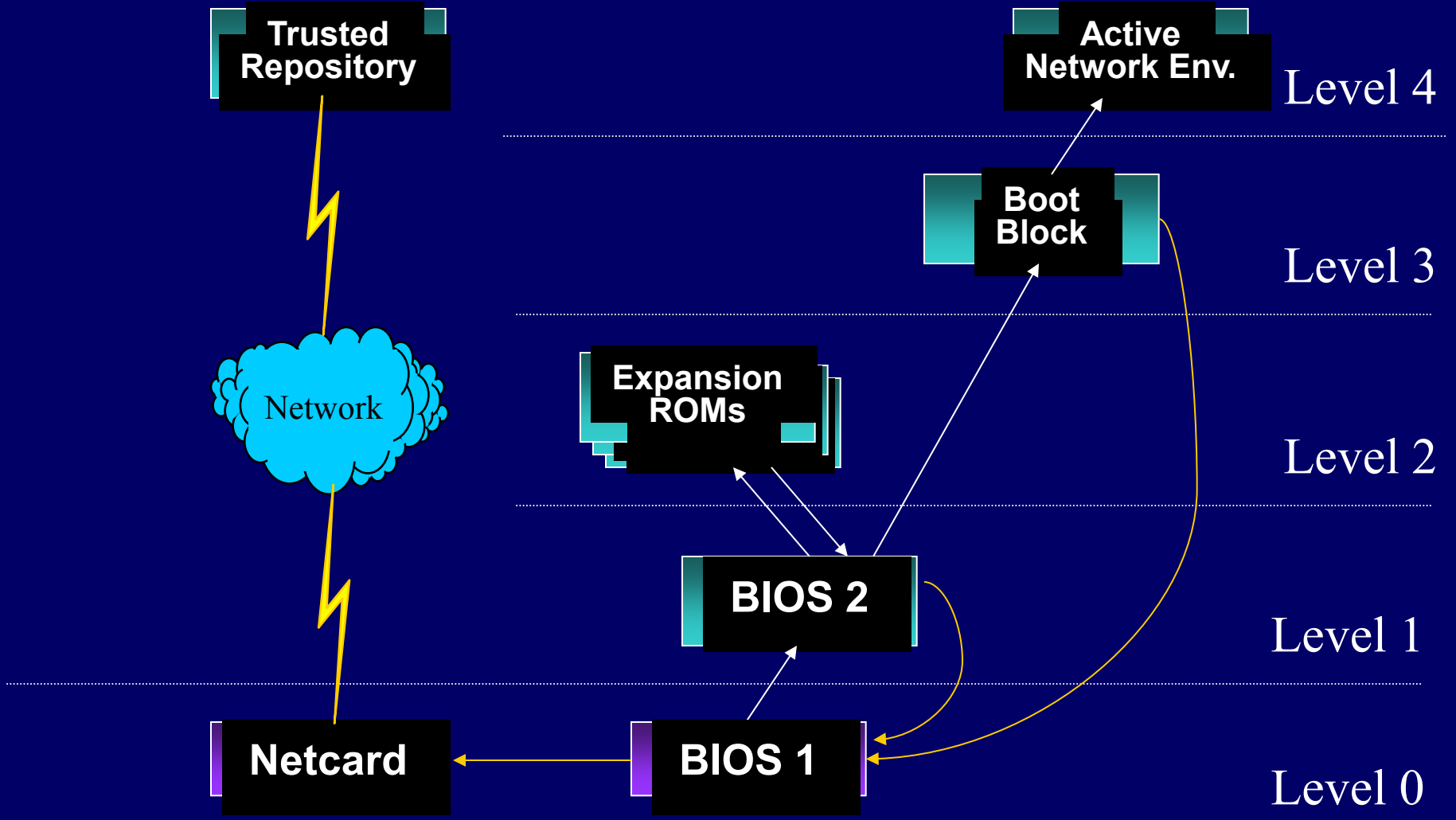
Strategies for paranactive nets

- Carry *all* code with you in a capsule
  - how do you load your code?
- Telescope out trust relationships with cryptography and identities
  - need to think about *ad-hoc* relations
- Pre-establish trust relationships and verify at node

Example: SwitchWare Architecture



Arbaugh's AEGIS Architecture



Result: E.E. in known state, *but…*

- Still trust some hardware
- Also trust repository for recovery
- Need *basis*, like diplomatic pouch containing a one-time pad
- *Applications* aware AEGIS executed?
- Can *applications* know that system integrity has been *preserved?*

Some (maybe crazy) ideas:

- Allow paranactive applications to invoke AEGIS with themselves as target…
  - Awful performance, poor multiplexing :-)
- Paranactive applications "disarm" *gradually* (gradually expose more code and credentials as environment is checked)
- Automated Trust Management (need new acronym - "third rail" of nets!)

Tools and Needs

- AEGIS: http://www.cis.upenn.edu/~waa
- Trust management infrastructure
  - Penn/AT&T work on Keynote
- Scalability is a challenge
- Need paranactive application examples
  - Intrusion detection and response?
  - Mapping and monitoring?