The Architecture of ALIEN

Scott Alexander

Jonathan Smith

Design Goals/ Decisions

- Experimentation
- Prototyping
- Active packets *and* active extensions
- Turing complete
- single address space

Language Choice

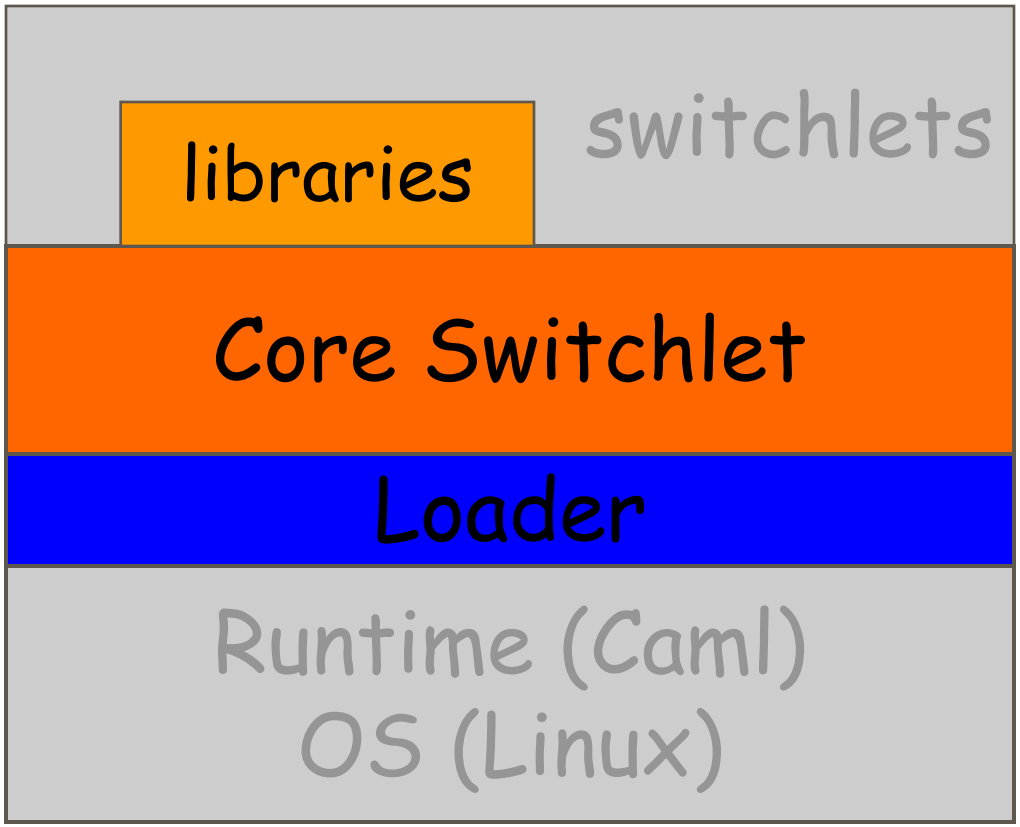
- strong typing
- garbage collection
- module thinning
- dynamic loading
- homogeneous representation of active code
- performance
- Caml

Other Possibilities

- Java
  - SecurityManager to implement module thinning
  - native methods for raw Ethernet access
  - no longer “write-once”
- A new language
  - PLAN
  - Netscript
  - Spanner/Sprocket

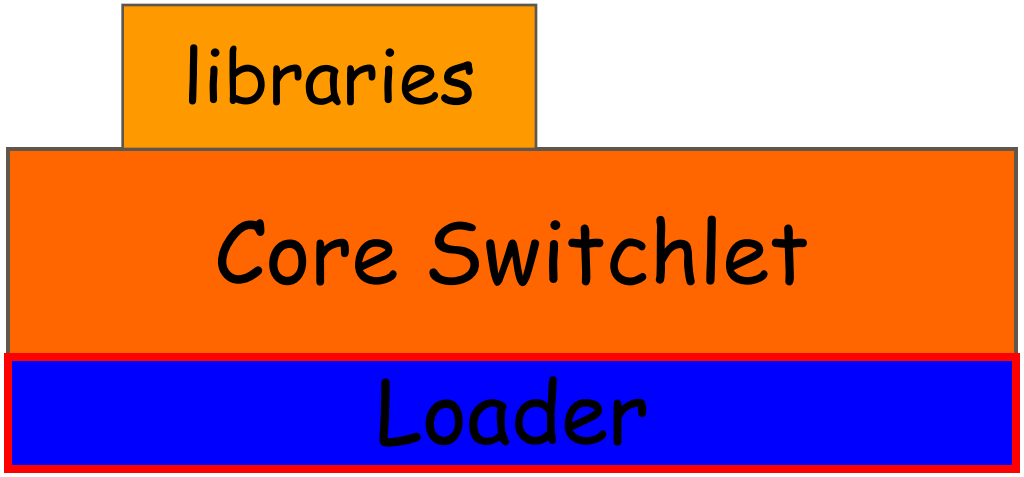
ALIEN Architecture

- Three layer architecture



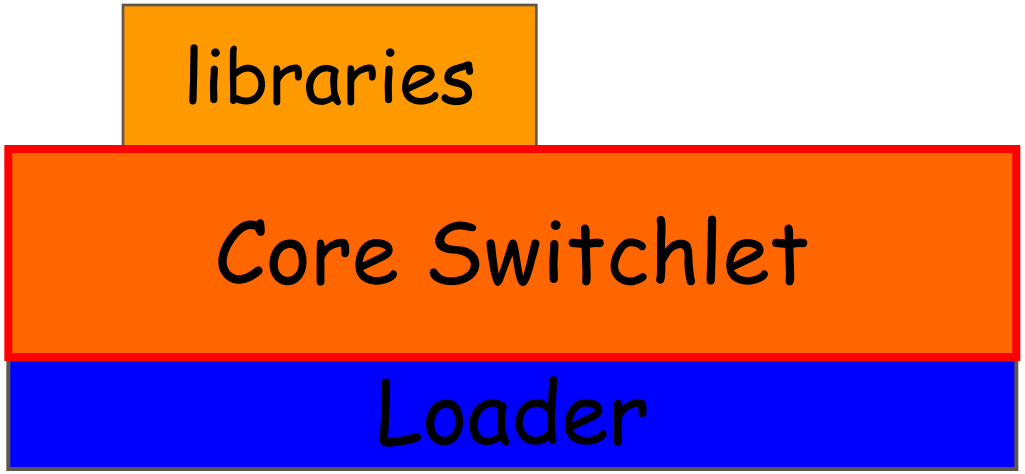
The ALIEN Loader

- startup routines
- active program loading
- system console
- mechanism only



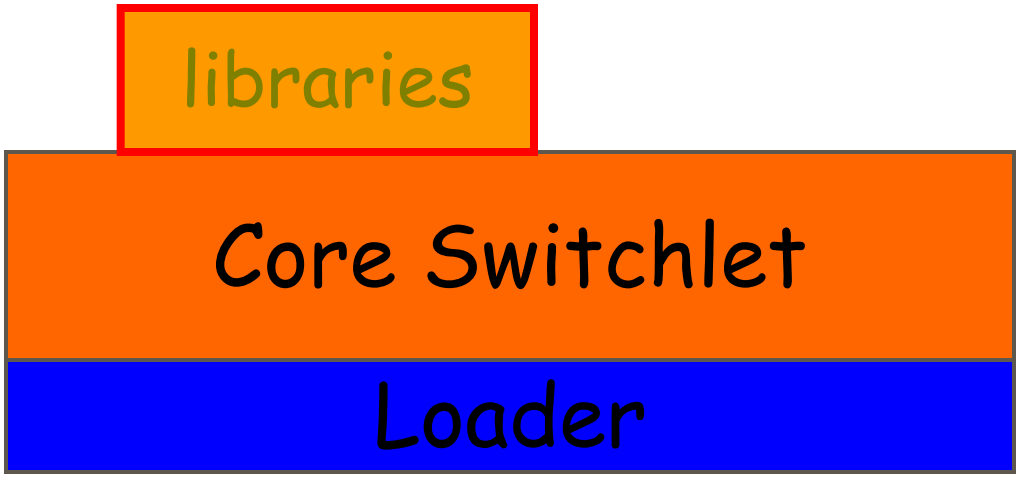
The Core Switchlet

- language primitives
- OS access
- network access
- thread access
- loading support
- message logging
- mechanism and policy



The Library

- "Everything else"
- IP
- UDP
- utility functions



Locating Functionality

- Loader only if needed to boot (implies no policy)
- Core Switchlet if privilege is required
- Otherwise, a library
- Split privileged and non-privileged functions

Implementation Issues

- SHA-1 (and crypto) performance
- extension of runtime for raw Ethernet
- 57 Mbps Active Bridge
- secure active ping

Conclusions

- Organizes 3 interesting cases of privilege versus loadability
- Demonstrates use of modern programming language for AN
- Shows how to build active packets based on active extensions

Future Work

- Runtime performance
- Management of further resources
- Global support for security

For More Information

- salex@research.bell-labs.com
- jms@cis.upenn.edu
- http://www.cis.upenn.edu/~switchware