Alpine: A User-Level Infrastructure for Network Protocol Development

David Ely, Stefan Savage, and David Wetherall

University of Washington
I. Kernel Development is a Pain
II. Network Protocols are in the Kernel
∴ Network Protocol Development is a Pain
Alpine moves the networking stack into **user-space** for development.

Changes are **easily moved** back to the kernel.

Works for any transport protocol.
Many systems have moved the networking stack out of the kernel for development

- Entrapid, OSKit, x-Kernel

This has always come at a price

- Modifications were required in either
  - the operating system
  - the applications using the stack
  - the networking stack

- Administrative barriers
  - a second IP address or network card
  - root access
Alpine's Goal

- Alpine runs a FreeBSD 3.3 stack in a user-space library
- No modifications to
  - the operating system
  - the applications using the stack
  - the networking stack
- No administrative barriers
  - uses the same IP address as the kernel
  - doesn't require root access
Traditional Stack

Operating System

<table>
<thead>
<tr>
<th>Application</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>System Calls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Socket</td>
</tr>
<tr>
<td>TCP/UDP</td>
</tr>
<tr>
<td>IP</td>
</tr>
<tr>
<td>Ethernet Driver</td>
</tr>
</tbody>
</table>

User Space
Kernel Space

Ethernet
- **Faux-Ethernet Driver**
  - send/receive packets
- **Pseudo-System Calls**
  - interface between application & sockets
- **Support Functions**
  - convince the stack it's in the kernel
Pseudo-System Calls

- Overrides the socket API
  - send, receive, connect, bind, accept
  - read, write, close, select
- Alpine mirrors the kernel's file descriptor table
  - multiplex between different types of "files"
- Applications use Alpine's sockets by either
  - linking with libAlpine.a before libc.a
  - LD_PRELOAD=libAlpine.so
Support Functions

- Calls stack initialization code at startup
- Kernel timer functions
  - timeout and tsleep/wakeup
- Synchronization functions
  - splnet, splx, etc.
- Memory allocation routines
  - kmem_malloc, zalloci, zfreei
  - copyin/copyout
Faux-Ethernet Challenges

- Sending packets from user-level
- Receiving packets from user-level
- Gracefully sharing state with the kernel
Open **raw socket** to bypass protocol stack

Normal: send (Message)  Raw: send (TCP HDR, IP HDR, Message)

IP is not modified because it already sends raw packets
- **Problem:** Alpine can't directly access the interface to receive packets.

- **Solution:** Use packet capture library (libpcap) to get packets.

Copies of all packets are available to applications.
Allocating Ports

- **Problem:** Alpine and the kernel cannot allocate the same ports
- **Solution:** bind a “dummy” socket to each port Alpine allocates
  - success $\Rightarrow$ kernel will not reallocate the port
  - failure $\Rightarrow$ kernel has already allocated port
Filtering the Kernel's Packets

- **Problem:** Alpine must not receive the kernel’s packets

- **Solution:** only capture Alpine's packets
  - dynamically install a filter in libpcap to only capture Alpine's packets

Only buffer Alpine's packets

<table>
<thead>
<tr>
<th>IP</th>
<th>libpcap</th>
<th>Ethernet Driver</th>
</tr>
</thead>
</table>

Ethernet
Filtering Alpine's Packets

- **Problem:** The kernel must not receive Alpine's packets
- **Solution:** Install a software firewall to filter out Alpine's packets

- BPF program only buffers Alpine's packets
- discard Alpine's packets

```
+---------+        +---------+
|  IP     |        | Firewall|
|  BPF     |        |  libpcap|
| Ethernet Driver |
```
Privileged Operations

**Problem:** many operations are privileged

- Kernel's stack
- libpcap
- Firewall

Privileged Operations

- Send raw packets
- Capture packets
- Update firewall

Alpine

libpcap

Firewall

telnet

Alpine

ftp
**Solution:** central proxy running as root
- central proxy performs access control
- individual users don't need root access

Kernel's stack
- Send raw packets
- Capture packets
- Update firewall

Central Alpine Proxy

- Alpine telnet
- Alpine ftp
Application calls pseudo-system call `bind()`
- Validate that the file descriptor is valid
- Create & bind a dummy socket to the port
  - ensures port is not in use
  - prevents kernel from allocating the port
- Contact the central proxy
  - updates the libpcap filter
  - updates the firewall
FreeBSD Implementation

- Alpine runs an unmodified FreeBSD 3.3 stack
- No modifications to kernel, applications, or stack
- 3043 Non-commenting source statements
  - 1188 Support functions + miscellaneous
  - 785 Pseudo-system calls
  - 285 Faux-Ethernet
  - 786 Central proxy
- Experience: makes protocol development easier
  - no reboots, easier debugging
  - running client and server on same machine
Current Limitations

- Alpine only runs on FreeBSD 3.X
  - porting to FreeBSD 4.X and Linux mainly requires sorting out header files
- TCP and UDP
- Maximum sockets used by Alpine ~100
  - limit of 512 instructions in BPF program
- fork() is currently not supported
  - parent and child stacks interfere for shared open connections
Uses of Alpine

- Easier development
- Environment for class projects
- Application specific protocol extensions
- User level overlay networks
Alpine keeps up until 10 Mbit/s
- too many copies
- Latency increases by 2ms

- 300 MHz P-Pro, 100 Mbit/s Ethernet
Demo of Alpine's Usefulness

- We downloaded the same file using three different networking stacks
  - Alpine running a normal TCP receiver
  - Alpine running an accelerated TCP receiver
  - the kernel running an accelerated TCP receiver
Demo of Alpine's Usefulness

Seconds since starting experiment

Sequence Number (bytes)

- Alpine Normal TCP
- Alpine Accelerated TCP
- Kernel Accelerated TCP

0 0.1 0.2 0.3 0.4 0.5 0.6 0.7
0 10000 20000 30000 40000 50000 60000
247 247.1 247.2
Current Work

- Porting FreeBSD 3.3 version to Linux
Future Work

- Porting Alpine to FreeBSD 4.x and Linux
- Support applications that fork
- New release in about a month
Conclusion

- Alpine is a tool that lowers the barrier of protocol development
- Requires no modifications to
  - operating system
  - applications
  - networking stack
- [http://alpine.cs.washington.edu](http://alpine.cs.washington.edu)