Moving FLASK to BSD Systems
SELinux Symposium 2006

Chris Vance
Information Systems Security Operation,
SPARTA, Inc.
Overview

- Security Frameworks
- A Brief History
- SELinux Inspired…
  - Security Enhanced BSD
  - Security Enhanced Darwin
Security Frameworks

- **Traditional UNIX security isn’t enough**
  - OS hardening
  - Mandatory protection
  - Flexible, manageable, scalable protection
- **Support required in the operating system for new security services**
  - Costs of locally maintaining security extensions are high
  - Framework offers extensibility so that policies may be enhanced without changing the base operating system
- **There does not appear to be one perfect security model or policy**
  - Sites may have different security/performance trade-offs
  - Sites may have special local requirements
  - Vendors unlikely to adopt a single policy
- **Bottom Line: Frameworks for Linux, FreeBSD, Darwin**
How We Got Here...
Focus and Reuse

• **Don’t “reinvent” security**
  – Learn from the past
  – The research is often decades old
  – A good design is durable, doesn’t require constant change
  – FLASK hasn’t changed much recently, what’s that mean?

• **Leverage existing technology, focus on “new” issues**
  – Focus on improving operating system security
  – Spend the time to get Frameworks correct
  – Work with vendor for acceptance
  – Develop rule sets that work
  – Develop effective tools to manage policy

• **Do innovate**
  – Don’t stop thinking!
  – Don’t squash new ideas
  – Re-factor old ideas
• Use MAC Framework to isolate policy from enforcement
• Build on Darwin’s source code and structural similarities to FreeBSD
• Port MAC Framework from FreeBSD
• Port FLASK components from SELinux
• Expand scope for Darwin-specific functionality (Mach IPC)
• Minimize Vendor diffs (OS & SELinux)
• Leverage existing policy & tools

Strong, useful security without sacrificing features, performance, or utility.
FreeBSD and Mac OS X

• Two related operating systems
  – FreeBSD - Server-class network operating system
    » Appropriate for embedded devices, network appliances, storage appliances, web services, firewalls, etc.
    » Already had FLASK components ported from SELinux
    » Already has MAC Framework
  – Mac OS X - Workstation-class operating system
    » Desktop environment, MS office suite, Adobe tools, etc.
    » Provides application and GUI components necessary for widespread adoption

• Open Source availability (Darwin)
• Shared BSD code heritage (user space and kernel)
• Audit support
• We understood FreeBSD, Mac OS X was new to us!
Mac OS X

- **Mac OS X is Apple's next generation operating system**
  - Builds on elements of Mach, NeXTStep, FreeBSD, and Mac OS 9, as well as other open source elements such as KDE
  - Continues Apple's tradition of user interface innovation
  - Potential for a larger user base (desktop users with application suites, programs people are familiar with)

- **Security**
  - UNIX security model as base
  - Mach Ports from NeXTStep for user space to represent privilege and capability
  - Rich application layer services: SecurityServer, SecurityFramework, SecurityAgent, CDSA crypto framework
  - Blend of Unix and Mach offers challenges

- **Mach IPC used extensively, must be considered**
  - Thousands of Mach messages per second!
Mac OS X System Architecture

Applications

Closed source frameworks and daemons

Open source libraries and daemons

XNU Kernel

Applications

Carbon  Cocoa  Java(JDK)
Application Services
Core Services

Libinfo  system_cmds  DirectoryServices  …  mach_init

Libsystem

Processes  UNIX IPC  Networking  VFS
BSD Kernel

IOKit

Scheduling  Virtual Memory  Mach Kernel
Mach IPC
MAC Framework Big Picture

System Call Interface

VFS

Socket IPC

Process Signaling

Mach IPC

mac_test

mac_mls

SEDarwin

(proprietary)
SEBSD/SEDarwin Policy Module

- Policy module wraps FLASK and TE implementations
  - FLASK provides access control abstractions
  - Access Control Vector Cache (AVC) caches policy decisions
  - TE/RBAC/MLS engine stores and implements file-backed policy
- User space components from SELinux
- Kernel components ported easily
  - “normal” issues with allocators, locking primitives, logging, printf, audit, etc.
LSM v. MAC Framework

- Linux, FreeBSD, Darwin similar in operation, very different implementations
- Parallel in construction
  - modular policies, registration based callbacks
- Linux
  - Widespread use
  - Requirements for vendor (Linus) adoption was critical
  - Lightweight Framework, less invasive
- FreeBSD
  - MAC Framework provides more label manipulation and management calls
  - Module composition from the start
  - Vendor buy-in allowed more significant changes
- When SELinux moved from invasive kernel patches to an LSM module, we benefited
- Future focus is to avoid divergence
Unique to Darwin

- **Rich (GUI) applications, desktop integration**
  - Provides motivation to use the system
  - Provides more challenges due to complexity
  - Inter-application messaging is ubiquitous

- **IOKit (another discussion entirely)**

- **Mach IPC**
  - Critical to secure
  - Performance/efficiency concerns
  - Didn’t have to start from scratch
  - Explore DTOS protections for Mach IPC
Darwin Complexity

• Three separate system boundaries (IOMeter, Mach, BSD) and each one must be adequately secured!
• Mach isn’t implemented as a microkernel, there is a blending of the lines between BSD and Mach services
  – BSD is in the kernel address space, not user
  – Threads and the scheduler are Mach constructs while processes are a BSD construct
  – Even worse, virtual memory is shared amongst all three kernel subsystems
• History showed that the complexity of the Mach microkernel led from DTOS to FLASK
  – Yet here were are trying to secure Mach IPC again…
  – It’s no less complex than it was
FLASK Extensions to Support Mach Messaging

- **Add Support for Mach IPC**
  - send, make_send, copy_send, move_recv, hold_send, hold_recv, etc.
- **Binary policy format is unchanged**
- **Add name based checks / generic access checks**
  - Fine grained control for user space services
  - For security aware applications / servers
  - Mach messages are opaque in kernel
  - User space must do some mediation
Project Status

- SEBSD module for FreeBSD 5.1 and up
- SEDarwin module for Mac OS X “Panther” (10.3.8)
- Simple sample policy for Mac OS X
- BSD coverage is fairly complete for both
- Support for Network protections undeveloped
- Mach subsystem coverage is still experimental
Future Directions

• MAC Framework is supported on Mac OS X “Tiger” (10.4.5) but SEDarwin isn’t yet
• More complete support for Mach IPC
• Update FLASK components to current version
• Implement network access controls (Framework already does)
• Develop new policy/rules
• OpenDarwin integration
SEDarwin: For More Information

• **Email:**
  - <Christopher.Vance at SPARTA.com>
  - TrustedBSD Mailing lists
    http://lists.freebsd.org/mailman/listinfo/trustedbsd-discuss

• **Web:**
  - http://sedarwin.org/
  - http://trustedbsd.org/
End.
Learning From History

• You learn each step of the way
  – Domain and Type Enforcement (DTE) on BSD/OS
  – Wrappers (System call interception)
  – Linux Security Modules
  – Security Enhanced Linux
  – TrustedBSD MAC Framework on FreeBSD
  – Security-Enhanced BSD
  – BSM-based Audit on Mac OS X (CAPP/EAL3)
  – TrustedBSD MAC Framework on Darwin
  – Mac OS X DCID 6/3 system
  – Security Enhanced Darwin

• What’s Next?