

# Separating Access Control Policy, Enforcement, and Functionality in Extensible Systems

Robert Grimm  
University of Washington

# Extensions

---

- Added to running system
- Interact through low-latency interfaces
- Form tightly integrated system
  
- *Are untrusted*

# Security

---

- Stated in a policy
- Relies on access control
- Expressed through protection domains
  - Structure system into protection domains
  - Enforce domains through access checks
  - Provide auditing of operations

# Problem

---

- Security requires additional structure
- But, want to preserve advantages of extensible systems

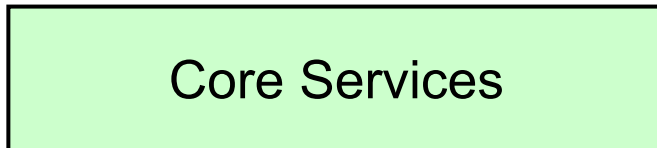
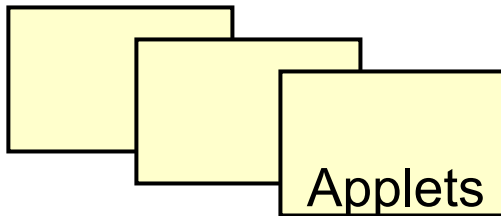
# Outline

---

- Motivation
- Goals and design
- Implementation for SPIN extensible OS
- Implementation for Distributed Virtual Machines (DVMs)
- Discussion and conclusions

# Applet Security

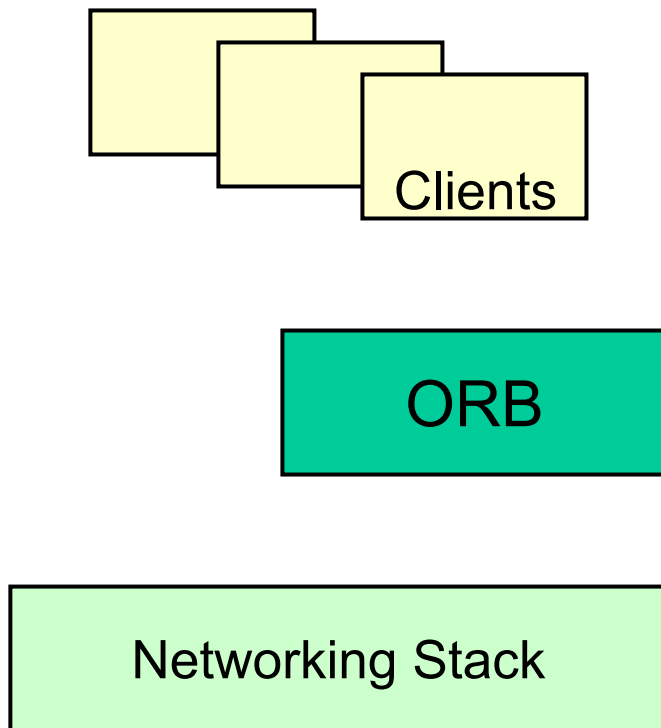
---



- Applets
  - Rely on some core services
  - Interact minimally
- Security for applets
  - Isolate applets from each other
  - Perform access checks on core services

# Real-World Examples

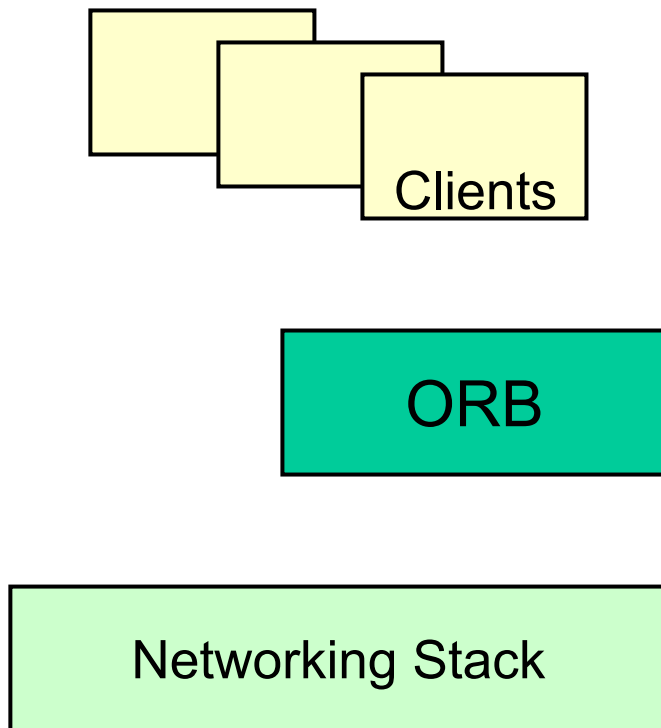
---



- Middle-ware
  - Relies on core services
  - Shared by other extensions
- Complex patterns of interaction

# Real-World Examples

---

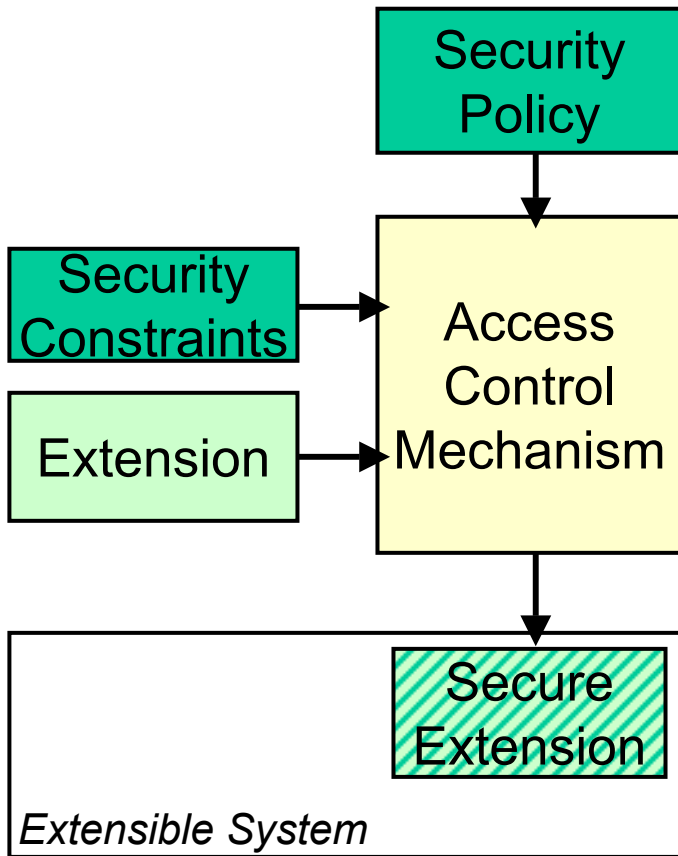


- Middle-ware
  - Relies on core services
  - Shared by other extensions
- Complex patterns of interaction

# Goals

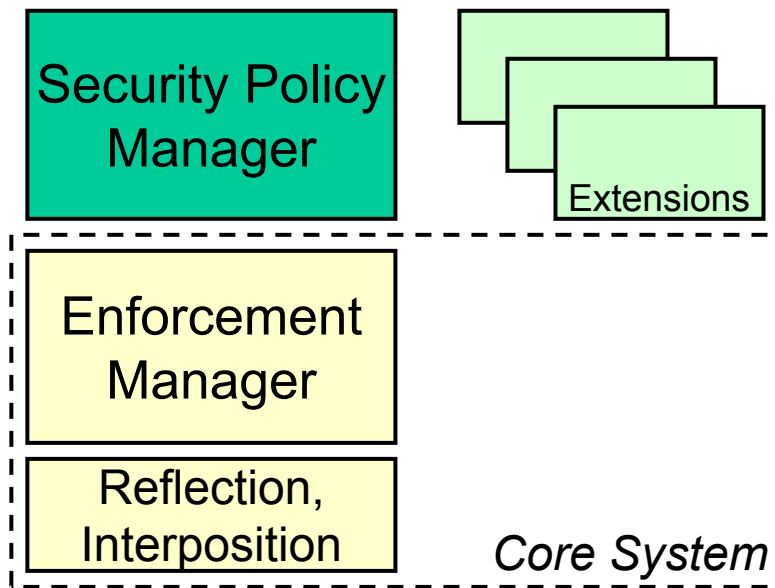
---

- Separate access control and functionality
- Separate policy and enforcement
- Use a simple, yet expressive model
- Enforce transparently in absence of failures



# Design

---



- Enforcement manager
  - Determines types and operations
  - Injects access control operations
- Security policy manager
  - Chooses access control operations
  - Performs mediation

# Basic Abstractions

---

- *Can subject execute operation on object?*
- Security identifiers (SIDs)
  - Associated with subjects and objects
  - Represent privilege
- Access modes
  - Sets of permissions
  - Associated with operations
  - Represent right to perform operation

# Loading an Extension

---

- Authenticate extension
  - SID associated with code
- Choose access control operations
  - Domain transfers, access checks, auditing
- Control interaction with other extensions
  - Can extension execute or extend interface?

# Access Control Operations

---

- Protection domain transfers
  - Establish new domain on procedure entry
  - Restore original domain on procedure exit
- Access checks
  - May call procedure?
  - May pass arguments into / out of procedure?
- Auditing
  - Trace operations

# Protocol

---

- Three mappings
  - $SID_{Thread} \times SID_{Procedure} \rightarrow SID_{Thread}$
  - $SID_{Thread} \times SID_{Object} \rightarrow ACCESSMODE_{Max}$
  - $SID_{Thread} \times TYPE_{Object} \rightarrow SID_{Object}$
- Mediation cache
  - Caches mappings in enforcement manager
  - Controlled by security policy manager

# SPIN Extensible OS

---

- Kernel written in Modula-3
- Static core
  - Hardware support
  - Modula-3 runtime
  - Linker/loader
  - Threads
  - Event dispatcher
- All other services provided by extensions

# Implementation

---

- Part of static SPIN core
- Uses
  - Modula-3 type system for reflection
  - SPIN event dispatcher for interposition
- Provides
  - Binary SIDs, access modes
  - Interface to security policy manager
  - Enforcement manager

# SID Management

---

- Object SID in object header
  - Callback from Modula-3 runtime
- Thread SID in separate SID stack
  - Stack records pre-allocated
  - Pushed and popped in atomic operations

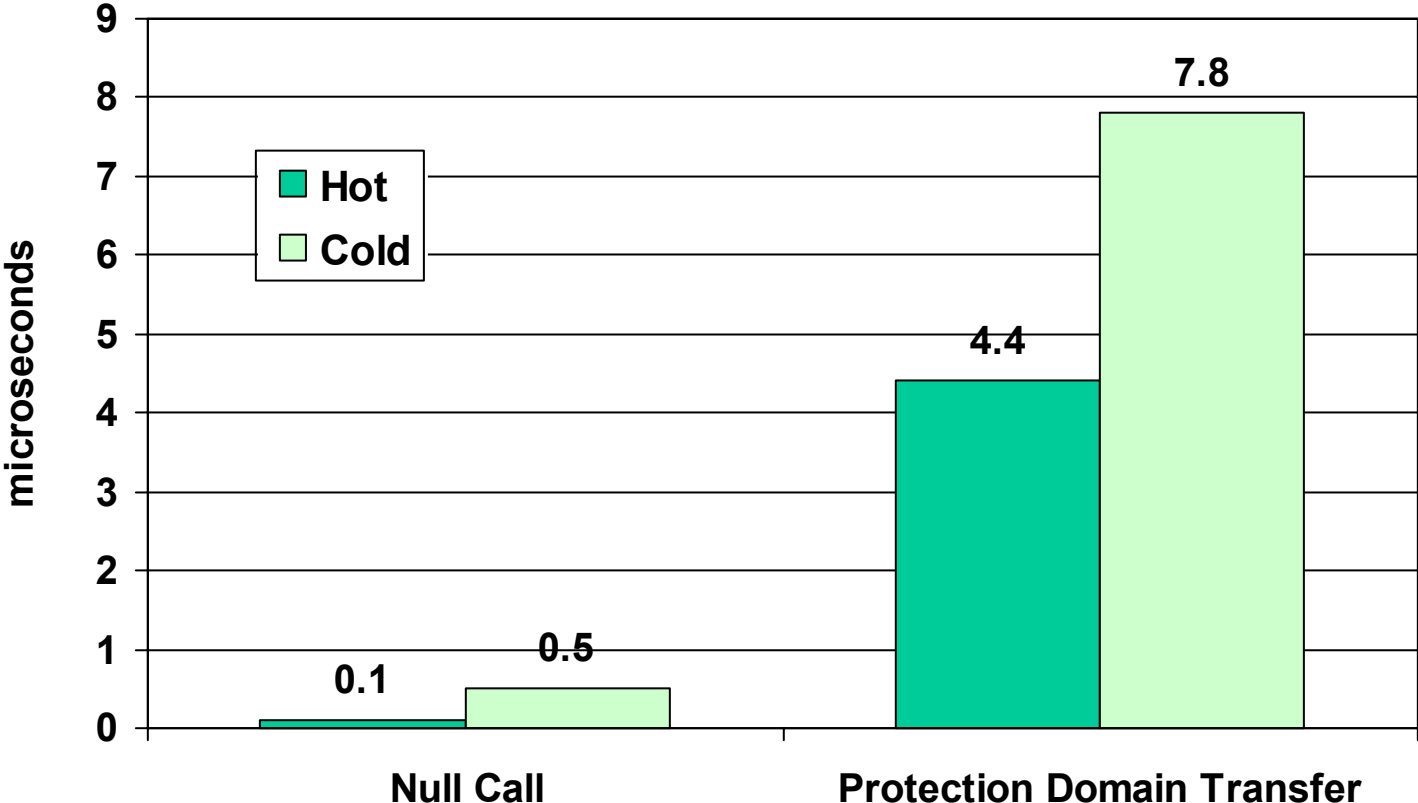
# Performance Evaluation

---

- Micro-benchmarks
  - Protection domain transfer
  - Checks on procedure and arguments
- End-to-end performance
  - Web server benchmark
- Alpha 3000/400 workstation
  - 133 MHz, 64 MB RAM
  - HP 1 GB disk

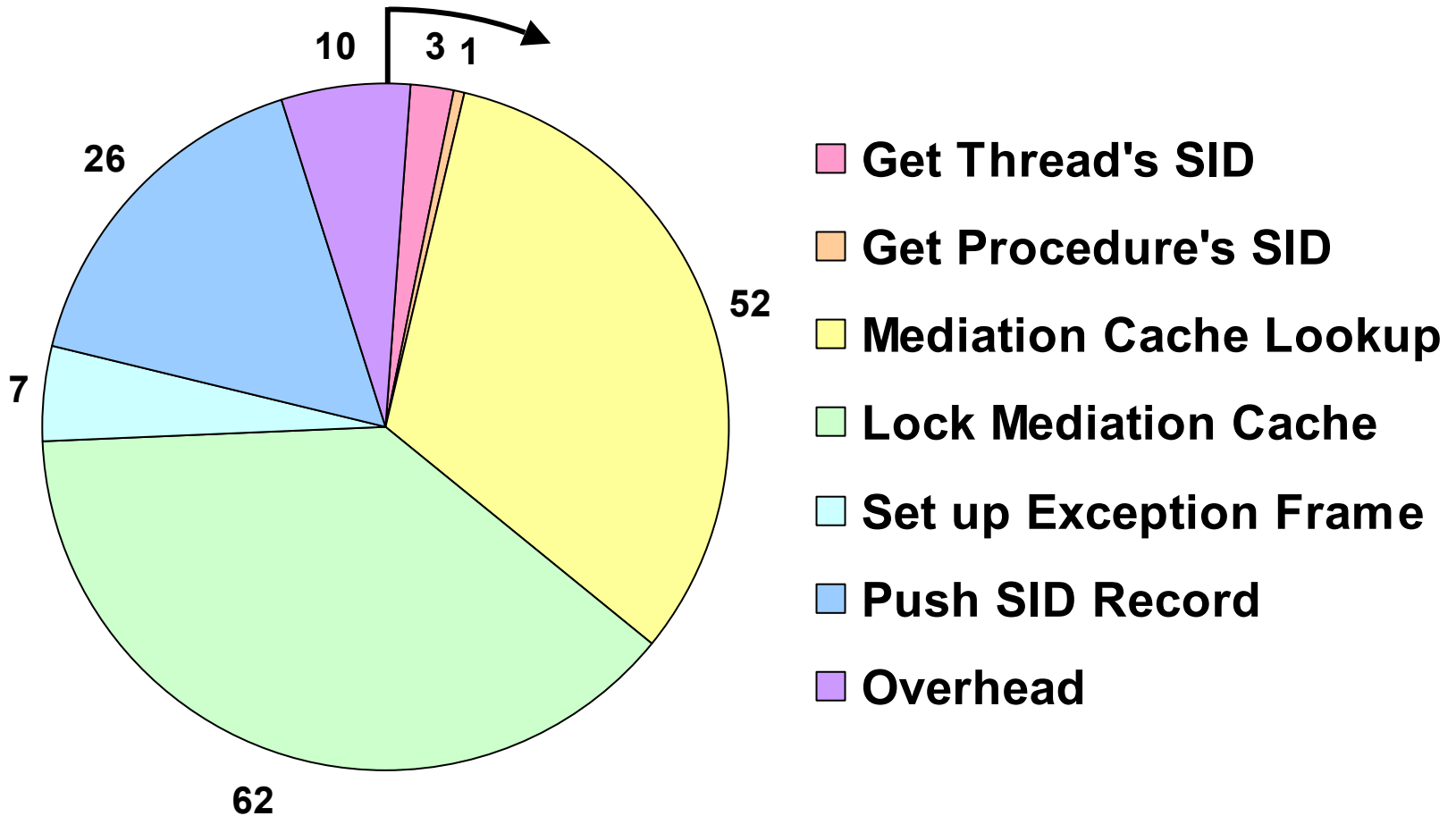
# Micro-Benchmarks 1

---



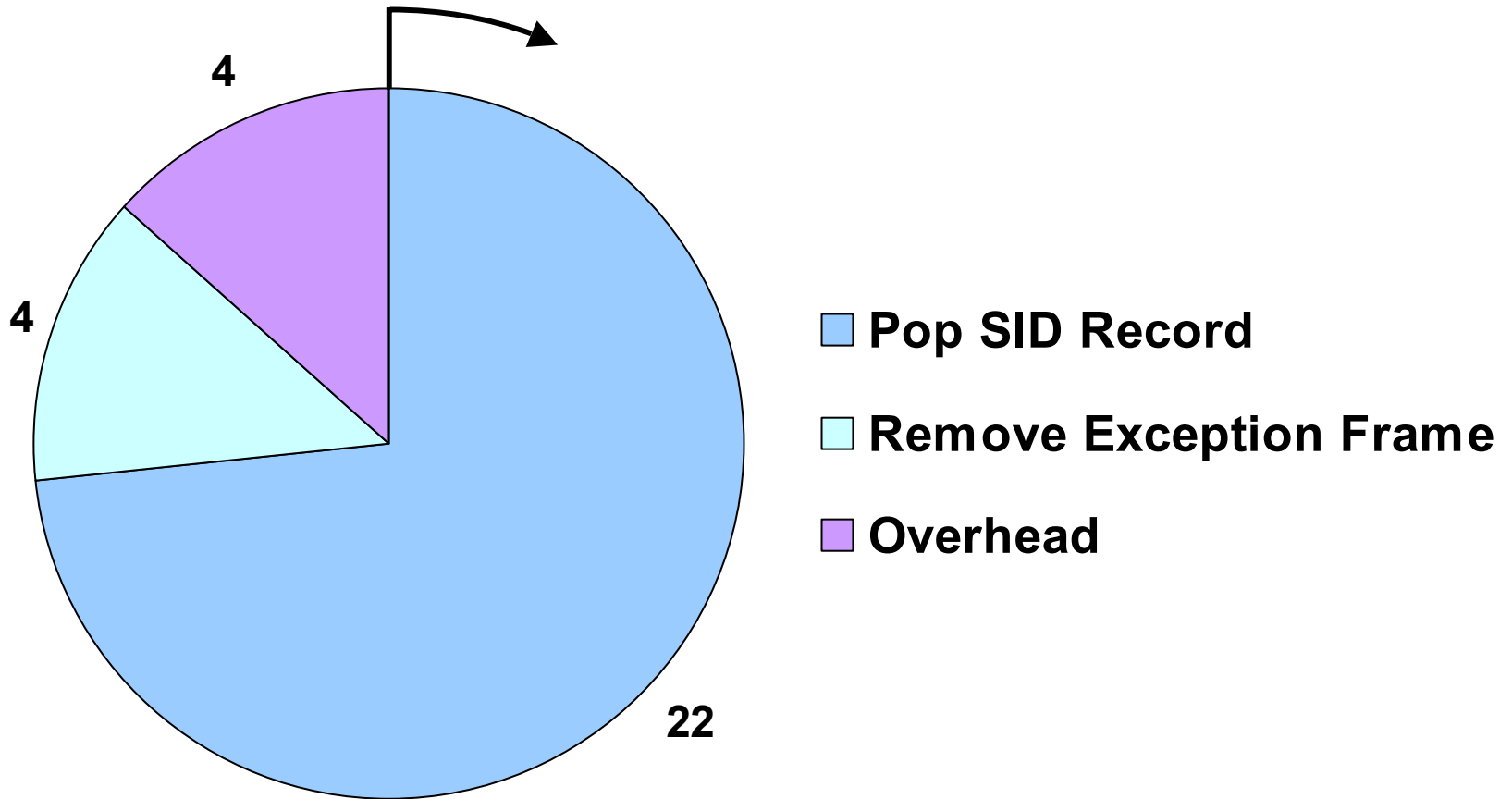
# Enter New Protection Domain

---



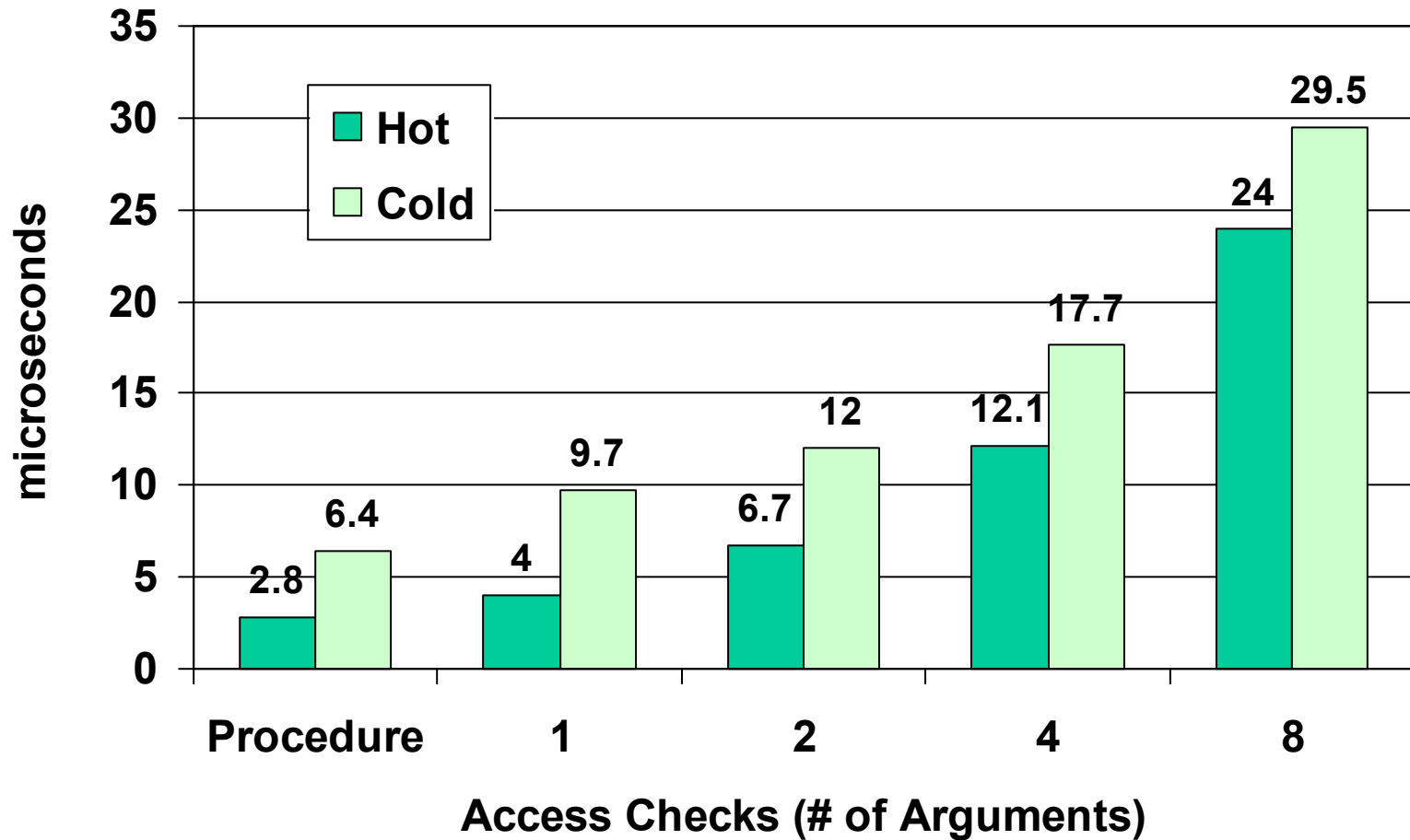
# Restore Old Protection Domain

---



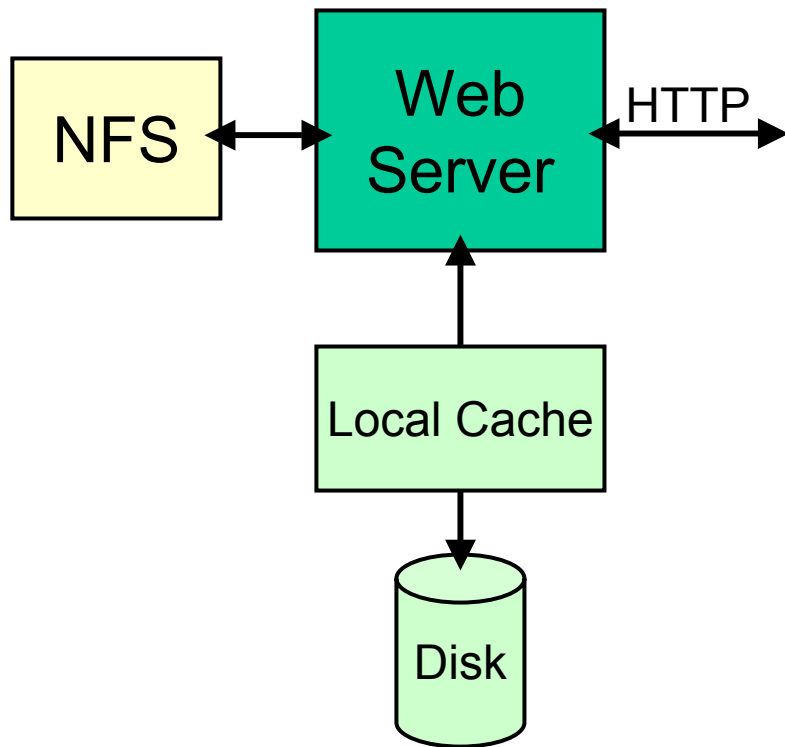
# Micro-Benchmarks 2

---



# Web Server Benchmark

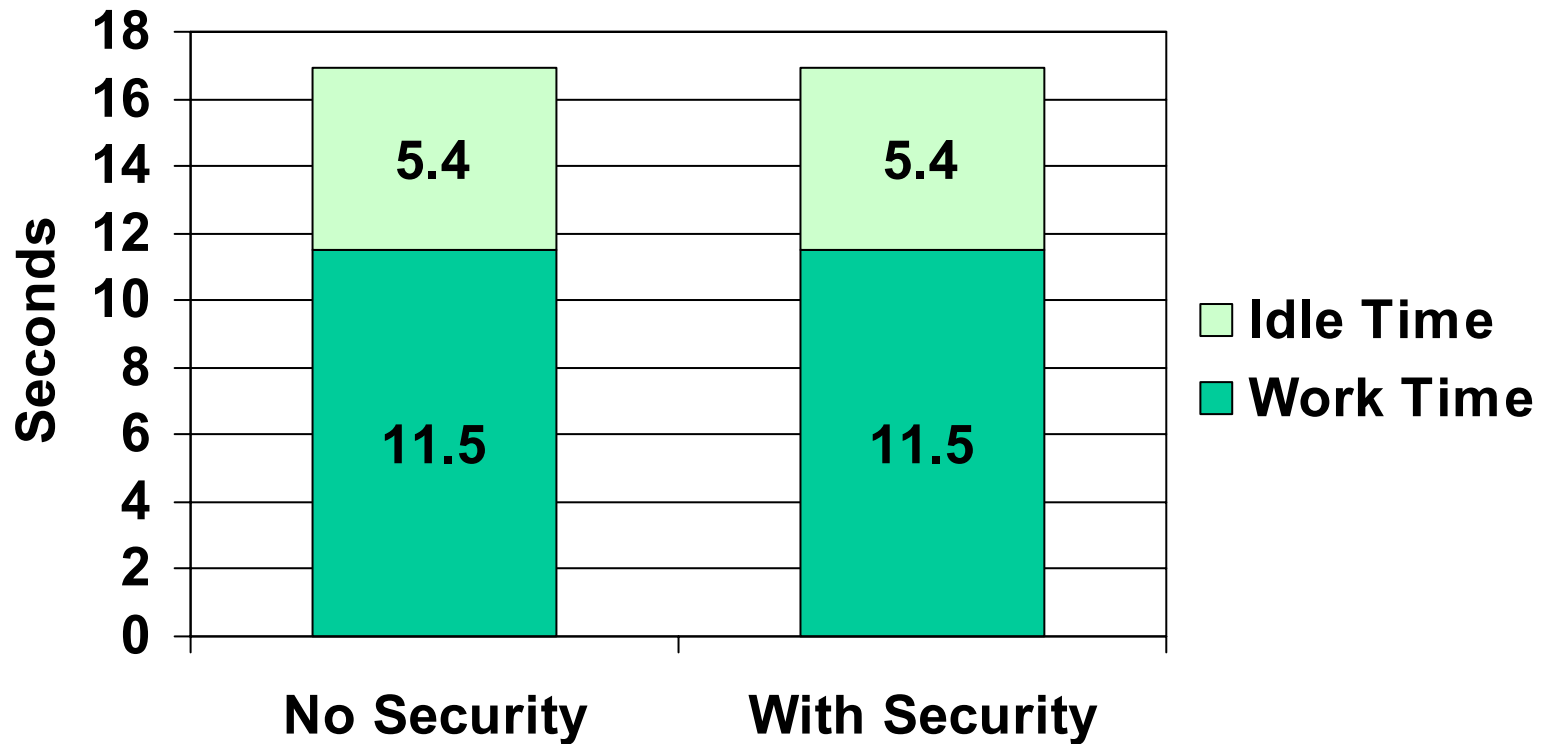
---



- Web server
  - Thread per request
  - NFS to read files
  - Local cache
- Security policy
  - Protection domain for web server
  - Access checks on NFS and local cache

# End-to-End Performance

---



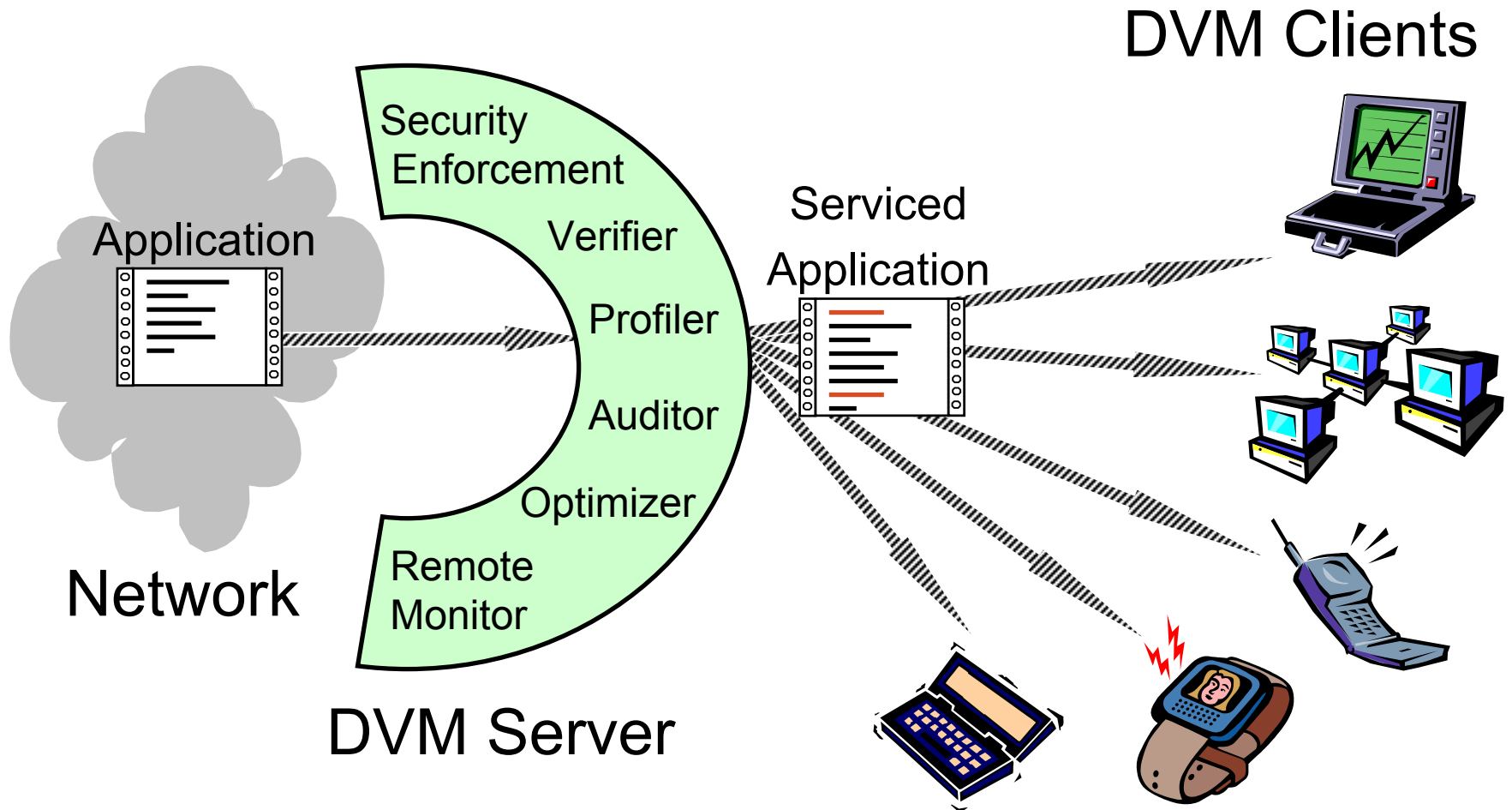
79 files, 5035 KB data, 1573 access checks

# Java Security

---

- State of the JDK
  - Ad hoc protection of system resources
  - Cooperation from programmer
- Separate policy, enforcement, and functionality
- Manage *all* JVMs in an organization
  - Uniform security policy
  - Central point of control

# Distributed Virtual Machines



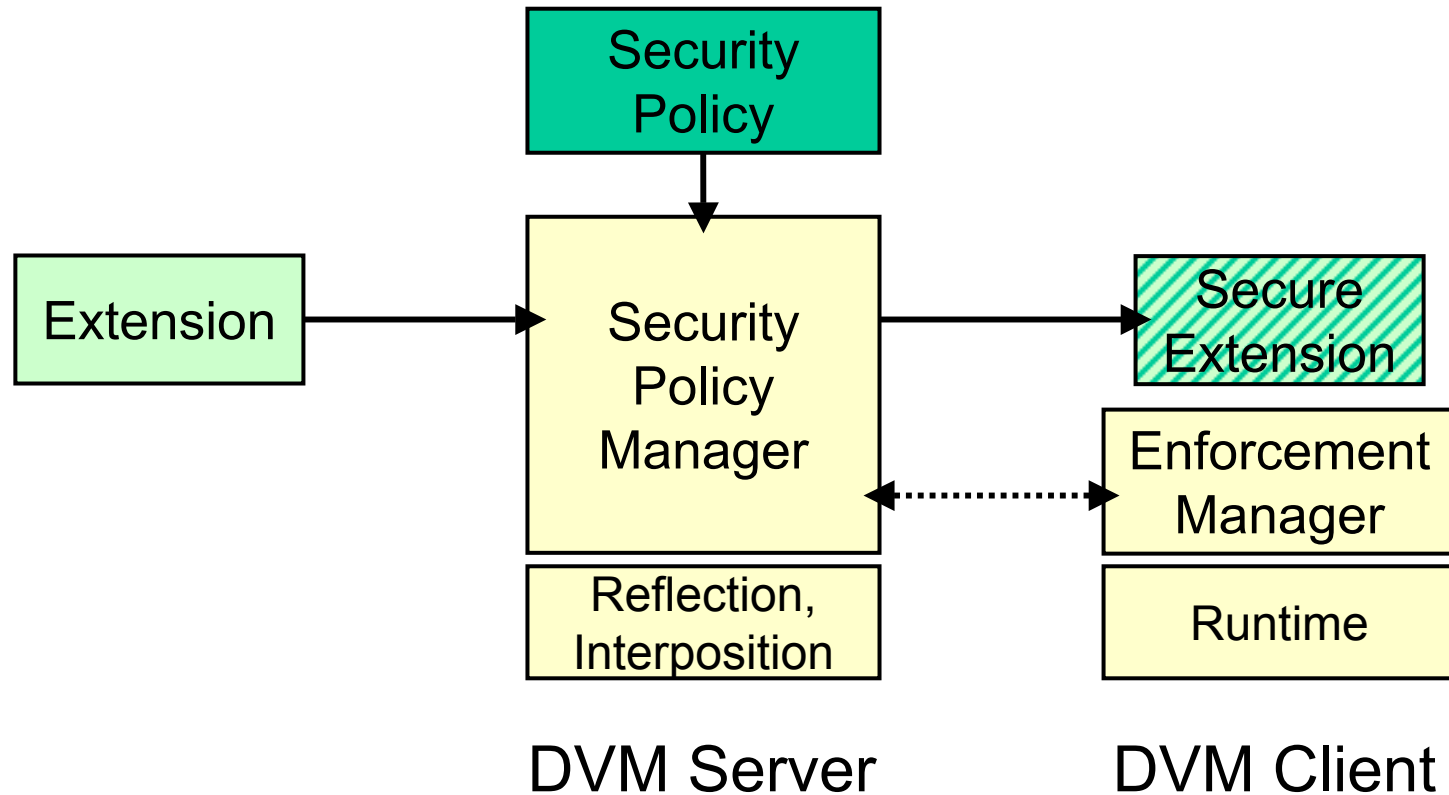
# Design Strategy

---

- Static service component
  - A-priori inspection
  - Fixed, one-time cost
- Dynamic service component
  - Data or context dependent
  - Binary rewriting
  - Centralized control
- Caching

# Security for DVMs

---



# Implementation

---

- Provides
  - Symbolic SIDs, access modes
  - Security policy manager
  - Enforcement manager
- Uses
  - Java class files for reflection
  - Binary rewriting for interposition

# SID Management

---

- Object SID
  - Explicit mapping in enforcement manager
- Thread SID
  - Thread-local SID stack
  - `java.lang.Thread` shadowed by `SecureThread`

# Security Policy Specification

---

- Domain-specific language based on XML
- Access matrix
  - $SID_{Thread} \times SID_{Procedure} \rightarrow SID_{Thread}$
  - $SID_{Thread} \times SID_{Object} \rightarrow ACCESSMODE_{Max}$
- Name spaces
  - $NAME_{Object} \rightarrow SID_{Object}$
- Mapping between code and access control operations

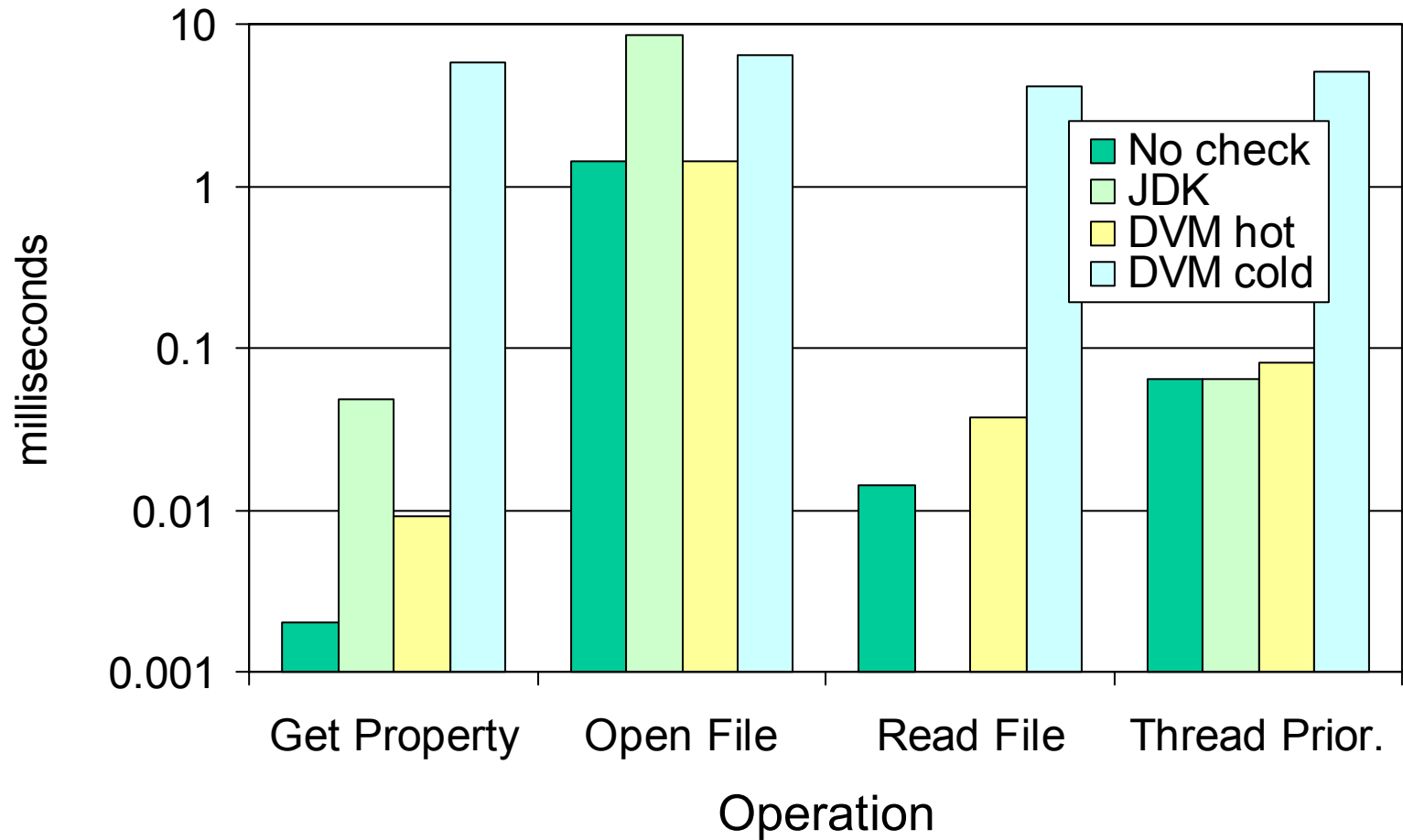
# Performance Evaluation

---

- Micro-benchmarks
  - Checks on operations
- Pentium Pro workstations
  - 200 MHz, 64 MB RAM, 10 Mb ethernet
  - Windows NT 4.0 SP3
  - Sun JDK 1.2
  - DVM server

# Micro-Benchmarks

---



# Discussion

---

- Mechanism relies on extensions' interfaces
- Type-safety
  - Trusted compiler for SPIN
  - Verifier for DVMs
- Expressiveness limited by
  - Abstract data types
  - Granularity of interfaces
  - Calling conventions

# Discussion (continued)

---

- Interposition
  - Event dispatcher for SPIN
  - Binary rewriting for DVMs
    - Re-instrumentation requires application to be restarted

# Systems Building

---

- Original design
  - Based on domain and type enforcement
  - Explicit instrumentation for core services
- Lessons
  - Design, implement, document repeatedly
    - But, beware of second-system effect
  - Separate concerns
  - Use single mechanism

# Conclusions

---

- Access control mechanism
  - Separates policy, enforcement, and functionality
  - Reliably imposes security
    - With small overhead
    - Across network
  - Is portable across extensible systems

# References

---

- Grimm and Bershad. Providing Policy-Neutral and Transparent Access Control in Extensible Systems. In Vitek and Jensen, *Secure Internet Programming*, LNCS 1603, Springer-Verlag, June 1999.
- Sirer, Grimm, Gregory, and Bershad. Design and Implementation of a Distributed Virtual Machine for Networked Computers. In *Proceedings of the 17th Symposium on Operating Systems Principles*, December 1999.
- <http://www.cs.washington.edu/homes/rgrimm/>