

# Design and Implementation of MAC Framework

Julian Qian  
julian@ercist.iscas.ac.cn

Institute of Software, CAS

September 2, 2005

# References:

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba  
Details  
Summary

- R. Watson, et al. "Design and Implementation of the TrustedBSD MAC Framework"
- C. Vance, R. Watson, "Security Enhanced BSD"
- "FreeBSD Architecture Handbook: Chapter 6 The TrustedBSD MAC Framework"

# Outline

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba  
Details  
Summary

## 1 MAC Framework Design

- A Layered Approach of Framework Design
- MAC Interfaces in the Layered Design

## 2 MAC Framework Implementation

- Components of the MAC Framework
- Framework Startup
- Policy Registration, Synchronization and Concurrency
- Entry Point Invocation, Composition
- Label Management

## 3 Policy Module Design & Implementation

- Kernel Policy Module Design
- Kernel Policy Module Implementations
- Biba Integrity Policy
- Details about Biba Integrity Policy
- Summary of Biba Integrity Policy

# Kernel Framework Aim of Design

## 1 MAC Framework Design

- A Layered Approach of Framework Design
- MAC Interfaces in the Layered Design

## 2 MAC Framework Implementation

- Components of the MAC Framework
- Framework Startup
- Policy Registration, Synchronization and Concurrency
- Entry Point Invocation, Composition
- Label Management

## 3 Policy Module Design & Implementation

- Kernel Policy Module Design
- Kernel Policy Module Implementations
- Biba Integrity Policy
- Details about Biba Integrity Policy
- Summary of Biba Integrity Policy

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba  
Details  
Summary

# A Layered Approach of Framework Design

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba  
Details  
Summary

**Aim** Extensions of both the kernel and user applications ought to support new security models

## Kernel

- A modular framework providing common dependencies for policies (labeling)
- Ability to augment "important" security decisions
- Composition of multi-modules in the future

## User

- Permit policy-independent security-aware applications to interact with the system
- Export info from framework to permit policy monitoring, etc.

# MAC Interfaces in the Layered Design

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba  
Details  
Summary

- **To Kernel Services**  
Provide a set of entry points to selected kernel services.
- **To Security Policies**  
Provide a set of interface to security policy implementations.
- **To User Processes**  
Provide a set of APIs to users, which reflect operations exported by policies.

# High Level View of Framework

MAC Framework

Julian

Outline

Framework Design

Requirement Interfaces

MAC Implementation

Components

Startup

Policies

Entry Point

Labels

Policy Module

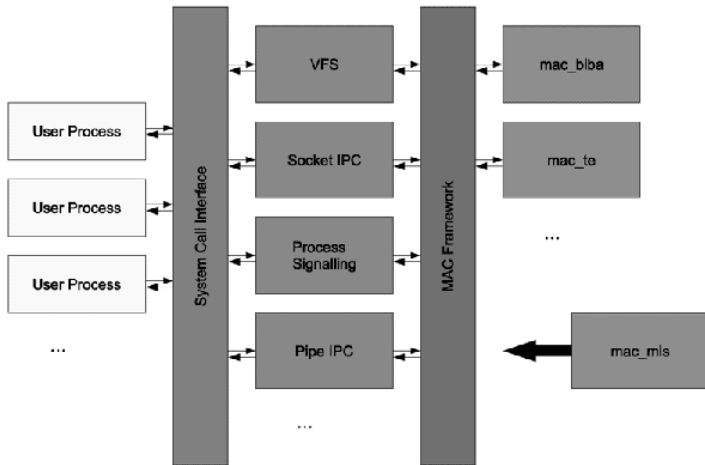
Design

Implementations

Biba

Details

Summary



High Level View of Framework

# MAC Framework Implementation

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba  
Details  
Summary

- 1 MAC Framework Design
  - A Layered Approach of Framework Design
  - MAC Interfaces in the Layered Design
- 2 **MAC Framework Implementation**
  - Components of the MAC Framework
  - Framework Startup
  - Policy Registration, Synchronization and Concurrency
  - Entry Point Invocation, Composition
  - Label Management
- 3 Policy Module Design & Implementation
  - Kernel Policy Module Design
  - Kernel Policy Module Implementations
  - Biba Integrity Policy
  - Details about Biba Integrity Policy
  - Summary of Biba Integrity Policy

# Components of the MAC Framework

## MAC Framework

Julian

Outline

Framework Design

Requirement Interfaces

MAC Implementation

Components

Startup

Policies

Entry Point

Labels

Policy Module

Design

Implementations

Biba

Details

Summary

- **MAC Framework Interfaces for Kernel Services**
  - sys/mac.h APIs for entry points
  - sys/\_label.h struct `label1`, store policy-agnostic label data
- **Framework Kernel Service Entry Points**

Affect object initialization, association/creation, and destruction.

Kernel services interact with the MAC Framework in two ways:

  - Invoke a series of APIs to notify the framework of relevant events
  - Provide a policy-agnostic label structure pointer in security-relevant objects.
- **Framework Implementation**

Entry point implementations, label primitives, policy registration and user/kernel APIs, defined in kern\_mac.c

# Components of the MAC Framework

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components

Startup

Policies

Entry Point

Labels

Policy Module

Design

Implementations

Biba

Details

Summary

## ■ Framework Interface for Policies

Entry point, registration interface and common access methods, defined in sys\_mac\_policy.h

## ■ Policy Implementations

Typically, one policy one C file, except complex ones.

## ■ Interfaces to User Processes

Defined in sys/mac.h, implemented in `libc`

# Framework Startup

## ■ Initialize

Early in the boot process, shortly after the kernel memory allocator, console and locking primitives, but before high level device probing and any kernel or user processes have started.

## ■ Early Registration, ubiquitously

- Apply any policy linked into the kernel itself
- Load policy modules by the boot loader prior boot

## ■ After early registration, a point

After early policy registration, a variable is set to indicate that any policies registered later will not be able to ubiquitously label kernel objects.



### **Policy Life Cycle**

# Policy Registration

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components

Startup

**Policies**

Entry Point

Labels

Policy Module

Design

Implementations

Biba

Details

Summary

- Registration process: receive events, reserve label space and access MAC Framework services
- Registered once the corresponding modules loaded
- `load-time` flag, indicate if the policy can be (un)loaded after boot
- Policy list synchronization and concurrency

# Policy List Synchronization and Concurrency

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup

**Policies**  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba  
Details  
Summary

- Policy list maintained by a busy count  
While the busy count is elevated, policy list changes are not permitted.
- The busy count protected by a mutex  
A condition variable is used to wake up sleepers waiting on policy list modifications.
- The busy count only used to dynamic entries.

# Policy Synchronization and Concurrency

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components

Startup

**Policies**

Entry Point

Labels

Policy Module

Design

Implementations

Biba

Details

Summary

- Use existing FreeBSD synchronization primitives  
Including mutexes, sleep locks, condition variables, and counting semaphores.
- Respect existing kernel lock orders  
Some entry points are not permitted to sleep
- When calling other kernel subsystems, release any in-policy locks

# Entry Point Invocation, Composition

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

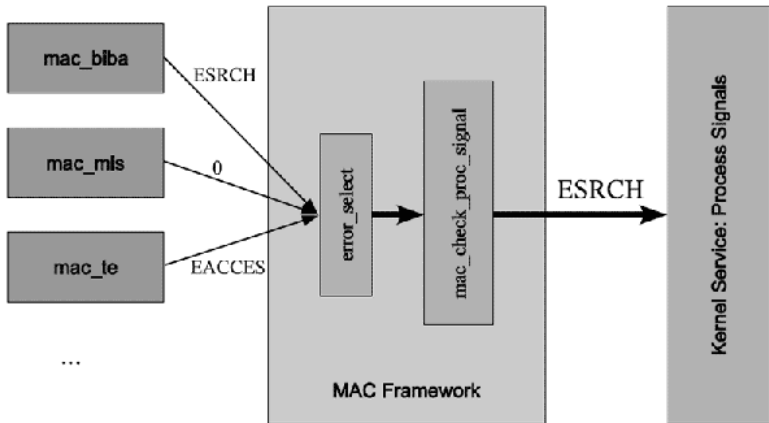
Policy Module

Design  
Implementations  
Biba  
Details  
Summary

## Three kinds of entry point invocation:

- **MAC\_PERFORM** No return values, used to post an event to interested policies to change it or manage labels, etc.
- **MAC\_CHECK** An `errno` return, used to access control entry points. The return values or the composition will be encoded an ordering by the function `error_select`, and provide to kernel services as processes signals.
- **MAC\_BOOLEAN** An arbitrary boolean return, used to some special case scenarios.

# Figure of Policy Composition



**Policy composition**

MAC Framework

Julian

Outline

Framework Design

Requirement Interfaces

MAC Implementation

Components Startup Policies Entry Point Labels

Policy Module

Design Implementations Biba Details Summary

# Structure of Policy and Label

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components

Startup

Policies

Entry Point

Labels

Policy Module

Design

Implementations

Biba

Details

Summary

- Some access control policies rely on security labels.  
Such as Biba, MLS. store sensitivity and integrity info.
- MAC Framework support labeling for several classes of security-relevant objects  
Such as several FS, processes, and network stack kernel elements.
- Each policy structure contains one or more instances of a policy-agnostic label structures  
each policy reserving label state is allocated a slot in the label structure, and each slot holds either a void pointer and an integer.  
In-memory labels{an array of slot unions{void \*, long}}

# Labels initialization, association and destruction

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components

Startup

Policies

Entry Point

Labels

Policy Module

Design

Implementations

Biba

Details

Summary

Labels live with the labeling objects.

- Initialization

When data structure for the kernel object initialized, policies will allocate memory of labels for the object.

- Association and Creation

When an initialized kernel structure is associated with an actual kernel object ( such as a file or process ).

- Destruction

When the kernel object is released.

# Policy Module Design & Implementation

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba  
Details  
Summary

- 1 MAC Framework Design
  - A Layered Approach of Framework Design
  - MAC Interfaces in the Layered Design
- 2 MAC Framework Implementation
  - Components of the MAC Framework
  - Framework Startup
  - Policy Registration, Synchronization and Concurrency
  - Entry Point Invocation, Composition
  - Label Management
- 3 Policy Module Design & Implementation
  - Kernel Policy Module Design
  - Kernel Policy Module Implementations
  - Biba Integrity Policy
  - Details about Biba Integrity Policy
  - Summary of Biba Integrity Policy

# Kernel Policy Module Design

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module  
Design

Implementations  
Biba  
Details  
Summary

- MAC policies are encapsulated in loadable kernel modules. Loaded at build-time, boot-time and run-time, a set of properties determines it.
- Policy modules typically follow a common structure: A description structure specifying the load-time flags, name and label information, and reference to other policy structures.
- Some administrative toggles often defined: Configure debugging, or policy rules.
- Isolate label management from the implementation of policies. Relying on a set of common access control primitives to implement most entry points.

# Table of sample policies

| Policy           | Description   |
|------------------|---|
| mac_bibe         | Hierarchal fixed-label integrity                                  |
| mac_bsdextended  | " File system firewall" using existing credentials/permissions    |
| mac_ifoff        | Interface silencing   |
| mac_lomac        | Hierarchal floating-label integrity                               |
| mac_mls          | Multi-Level Security with compartments                            |
| mac_none         | Prototype stub policy   |
| mac_partition    | Inter-process visibility policy based on process partition labels |
| mac_seeotheruids | Inter-process visibility policy based on existing credentials     |
| mac_test         | Mac Framework invariant tests                                     |
| sebsd            | Port of the SELinux/FLASK/TE                                      |

MAC

Framework

Julian

Outline

Framework

Design

Requirement

Interfaces

MAC Imple-

mentation

Components

Startup

Policies

Entry Point

Labels

Policy Module

Design

Implementations

Biba

Details

Summary

# Kernel Policy Module Implementations

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components

Startup

Policies

Entry Point

Labels

Policy Module

Design

Implementations

Biba

Details

Summary

- Includes some modules that provide a diverse set of kernel access control extension and extensions are also available from third parties.
- Manage policy modules in MAC framework
  - sysctl tunable
  - loader.conf
  - system calls

# TrushBSD Biba Integrity Policy

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

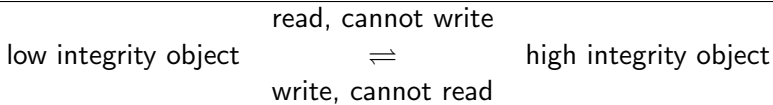
Policy Module

Design  
Implementations

**Biba**

Details  
Summary

Biba is a fixed label policy in that all subject and object label changes are explicit, not like LOMAC.



The dominance operator and access rules are reversed to MLS.

# Description of Biba Integrity Policy

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations

Biba

Details

Summary

- A complete label consists of both hierarchal and non-hierarchal elements.
  - hierarchal grade filed (0~65535)
  - non-hierarchal compartment (0~255)

- Tree special labels often used:

## Label

biba/low

biba/equal

biba/high

## Comparison

lower than all other labels

equal to all other labels

higher than all other labels

# Label Format of Biba Integrity Policy

## ■ Object Label Format

### ■ Form

biba/grade:compartments

### ■ Examples

biba/10:2+3+6

biba/low

## ■ Subject Label Format

### ■ Form

biba/effectivegrade:effectivecompartments( lograde:  
locompartments-higrade:hicompartments)

### ■ Examples

biba/10:2+3+6(5:2+3-20:2+3+4+5+6)

biba/high(low-high)

### ■ Valid ranged labels

rangehigh >= effective >= rangelow

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba

Details  
Summary

# Run-time Configuration of Biba Integrity Policy

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba

Details  
Summary

The following `sysctl(8)` MIBs are available for fine-tuning the enforcement of this MAC policy.

```
security.mac.biba.enabled
```

Enables enforcement of the Biba integrity policy. (Default: 1).

```
security.mac.biba.ptys_equal
```

Label `pty(4)`s as "biba/equal" upon creation. (Default: 0).

```
security.mac.biba.revocation_enabled
```

Revoke access to objects if the label is changed to dominate the subject. (Default: 0).

# Features of Biba Integrity Policy

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components

Startup

Policies

Entry Point

Labels

Policy Module

Design

Implementations

Biba

Details

Summary

- Label values inherited from the parent subject.
- All object labels can be managed using user tools.
- Central access control logic
  - Dominate function compares two Biba elements with respect to their types, grades, and compartments.
- Registration Flags
  - Biba policy cannot be removed, once it attached.

# Advantages of Biba Integrity Policy

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components

Startup

Policies

Entry Point

Labels

Policy Module

Design

Implementations

Biba

Details

Summary

- Isolate the details of the policy implementation from the kernel services
- Reduce the risks of minor changes in one subsystem requiring gratuitous changes in the other.
- Supported through a generalized label management service of MAC Framework Permit the policy implementor to focus on the details of the policy application.

# The End

MAC  
Framework

Julian

Outline

Framework  
Design

Requirement  
Interfaces

MAC Imple-  
mentation

Components  
Startup  
Policies  
Entry Point  
Labels

Policy Module

Design  
Implementations  
Biba  
Details  
Summary

*Thanks!*