CIM-based Resource Information Management for Integrated Access Control Manager

Fumio Machida[1], Kumiko Tadano[1], Masahiro Kawato[1]
Takayuki Ishikawa[2], Yoichiro Morita[3], and Masayuki Nakae[3]

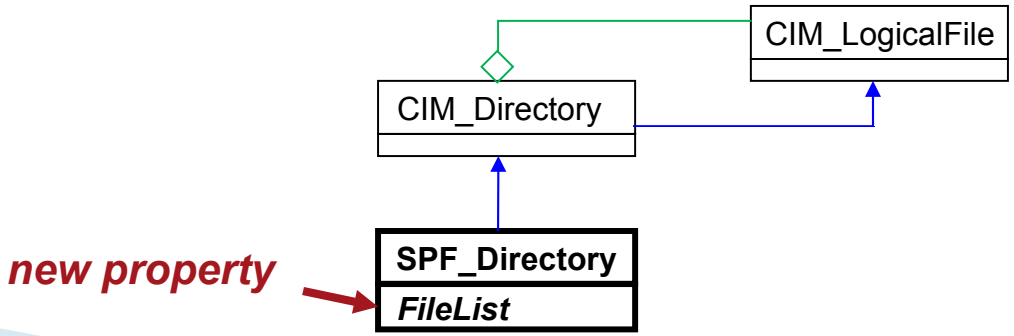
[1] NEC Service Platforms Research Laboratories, [2] NEC Business Inovation Center
[3] NEC Common Platform Software Research Laboratories

This work is a part of the Secure Platform project (SPF) supported by Japanese ministry of Economy, Trade and Industry, and Association for Super-Advanced Electronics Technologies

Contribution

- Model extension for effective directory search
  - We propose an extension of CIM_Directory class to explore directories quickly on the GUI

- Study of an architecture for CIM-based integrated access control management
  - We implemented the CIM-based access control manager by introducing additional CIM models for “reference monitor”



Outline

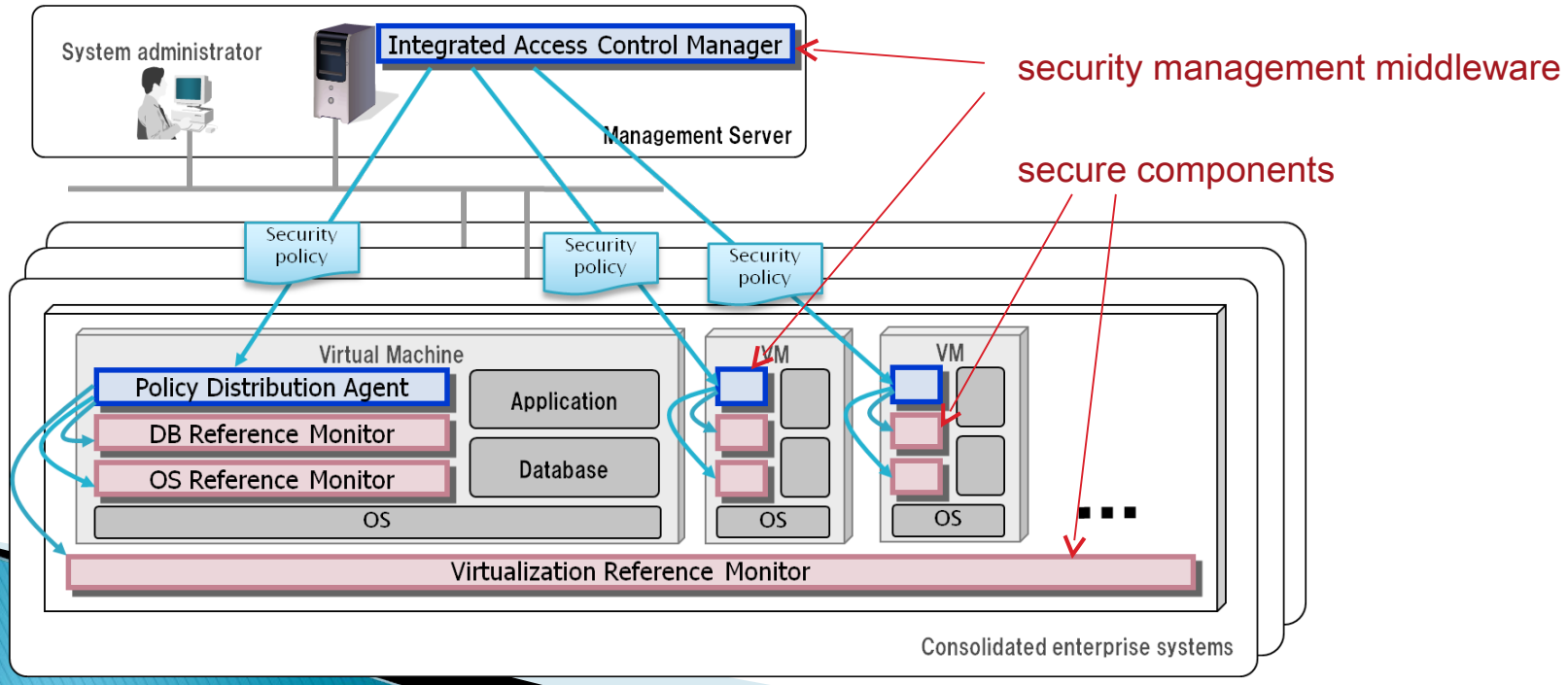
- Introduction
  - The overview of *Secure Platform* project
  - Related work
- Integrated Access control Manager (IAM)
  - Architecture
  - Component interactions
  - Information models
- Implementation
  - Policy Manipulation GUI
  - Query performance evaluation
- Conclusion

Introduction

- Server virtualization is used for server consolidation
- Concerns for security and reliability
  - Vulnerability of virtualization software
  - Risk of spreading of security incidents or performance problems across the systems
- Complexity of the configurations of security management tools
  - Administrators have to configure all security management tools consistently

Secure Platform project (SPF)

- Make consolidated server systems secure and reliable
  - Develop the security management middleware integrating various access control policies
  - Develop the secure components such as secure hypervisor



Integrated Access Control

- Issues on the access control management for consolidated server systems
  - Access control modules are distributed over software layer as well as over servers
  - All access control modules need to be configured consistently

Administrator suffers from the tasks for configuring access control modules

To improve the manageability, integration of access control management is required

Requirements

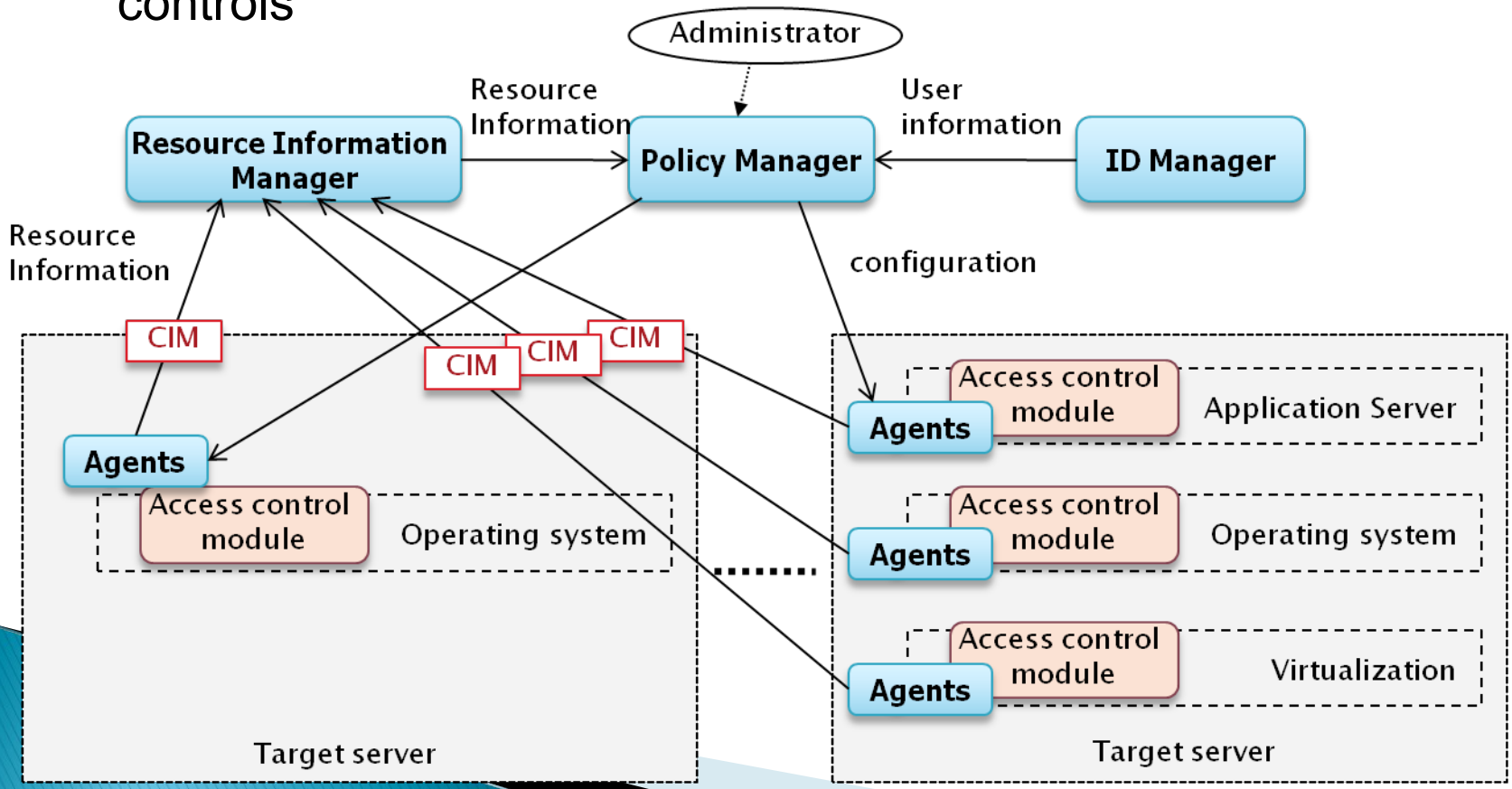
- Management integration
  - Managing various access control modules from an integrated console
- Policy abstraction
  - Introducing abstract policy that can be translated into the specific policies for access control modules
- Operation automation
  - Automating the operations such as lookup of target resource information and configuration of access control modules

Related Work

- Secure components
  - SELinux and AppArmor are known as secure components for Linux OS using LSM framework
  - ACM and Flask are known as secure components for Xen's virtualization using XSM framework
  - Configurations of these components are complex tasks
- Integrated access control systems
  - Integrated access control systems for distributed systems have been studied in several works
  - There is no work addressing the architecture for integrated access control for different resources in consolidated server environments

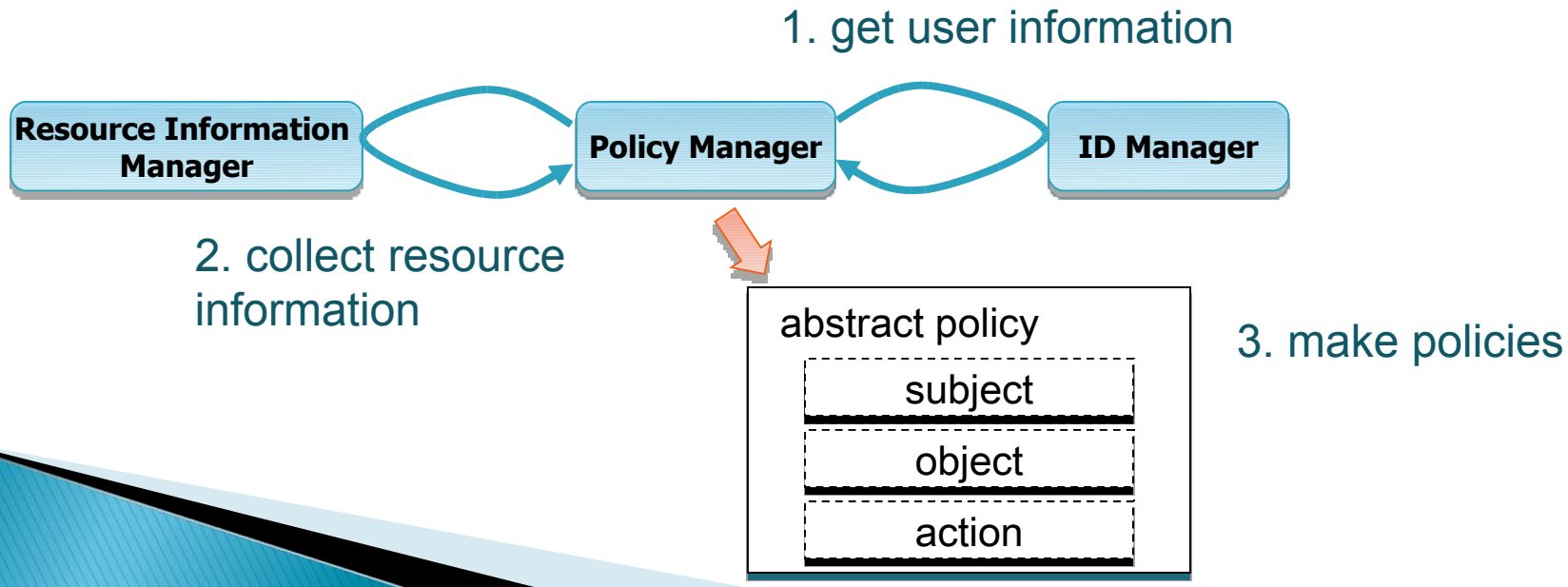
Proposed Architecture

- Integrated Access control Manager (IAM)
  - is organized for satisfying all the requirements
  - adopts CIM standards for integrating various types of access controls



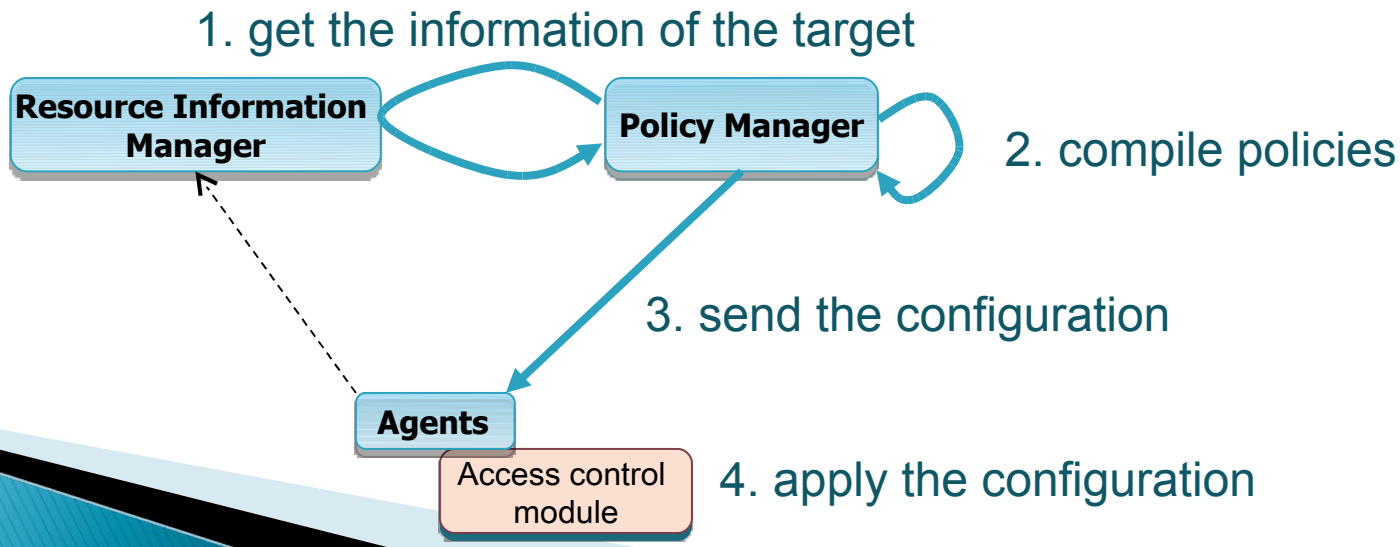
Policy manipulation

1. Policy Manager queries ID Manager to get the user information
2. Policy Manager collects target resource information from Resource Information Manager
3. Administrators make abstract policy



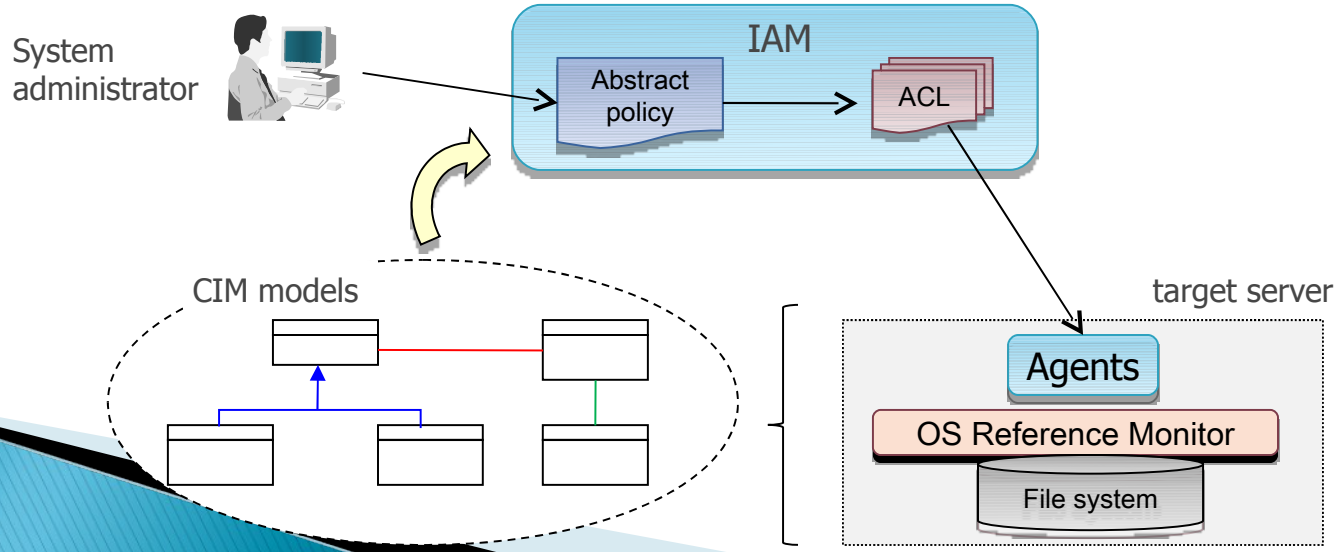
Policy deployment

1. Policy Manager queries Resource Information Manager to get the information of the target access control module
2. Policy Manager compiles the abstract policy
3. Policy Manager sends configurations to the Agents
4. Agent applies the received configurations to the target access control module



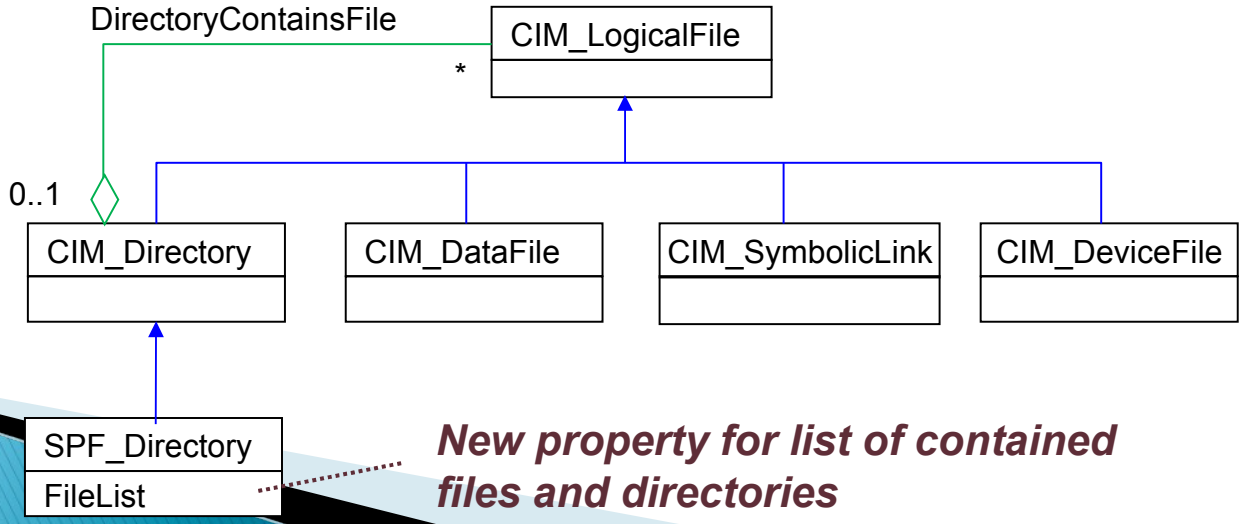
File Access Control Scenario

- CIM models are used in the pilot implementation for file access control
- Integrated file access control
  - OS reference monitor controls the file accesses on an OS by access control list (ACL)
  - IAM manages access controls for distributed multiple OS reference monitors with abstract policy



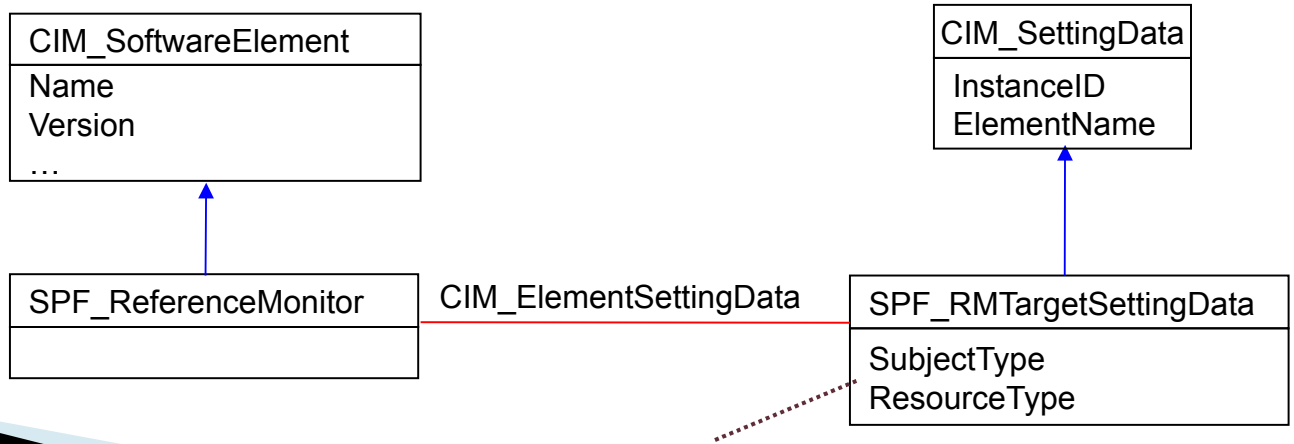
File and Directory

- Files and Directories are the target resoruces of the OS reference monitor
- CIM_Directory inherits CIM_LogicalFile and logically represents a group of files contained in it
- SPF_Directory has a new additional property “FileList”
  - “FileList” allows us to lookup the list of files and directories contained in the directory without retrieving all related CIM_LogicalFile instances



Reference Monitor

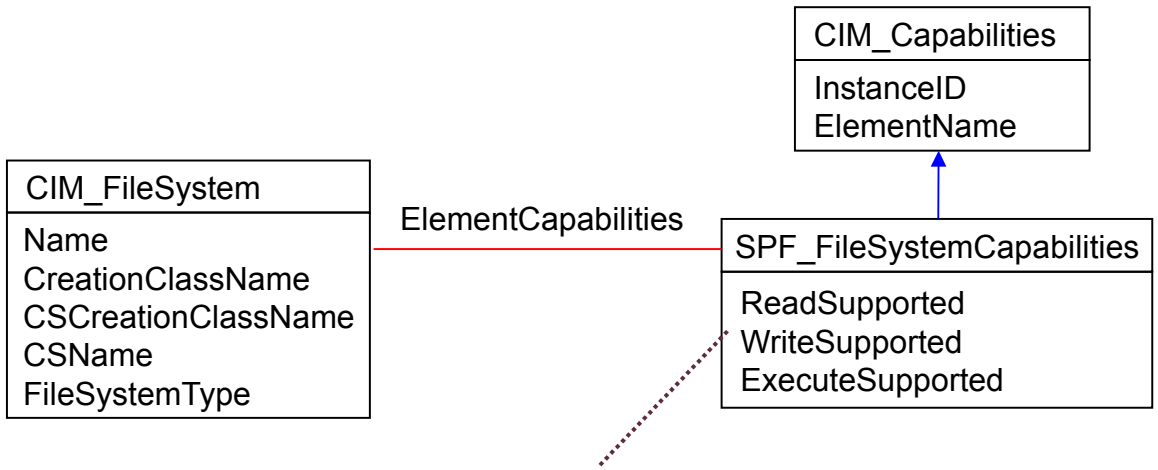
- The property information of the OS reference monitor is required at policy translation
- The model of OS reference monitor is defined by extending CIM_SoftwareElement
- Types of “subject” and “object” supported by the OS reference monitor are expressed within the SPF_RMTagetSettingData



Properties for identifying the types of subject and object

File Access Capabilities

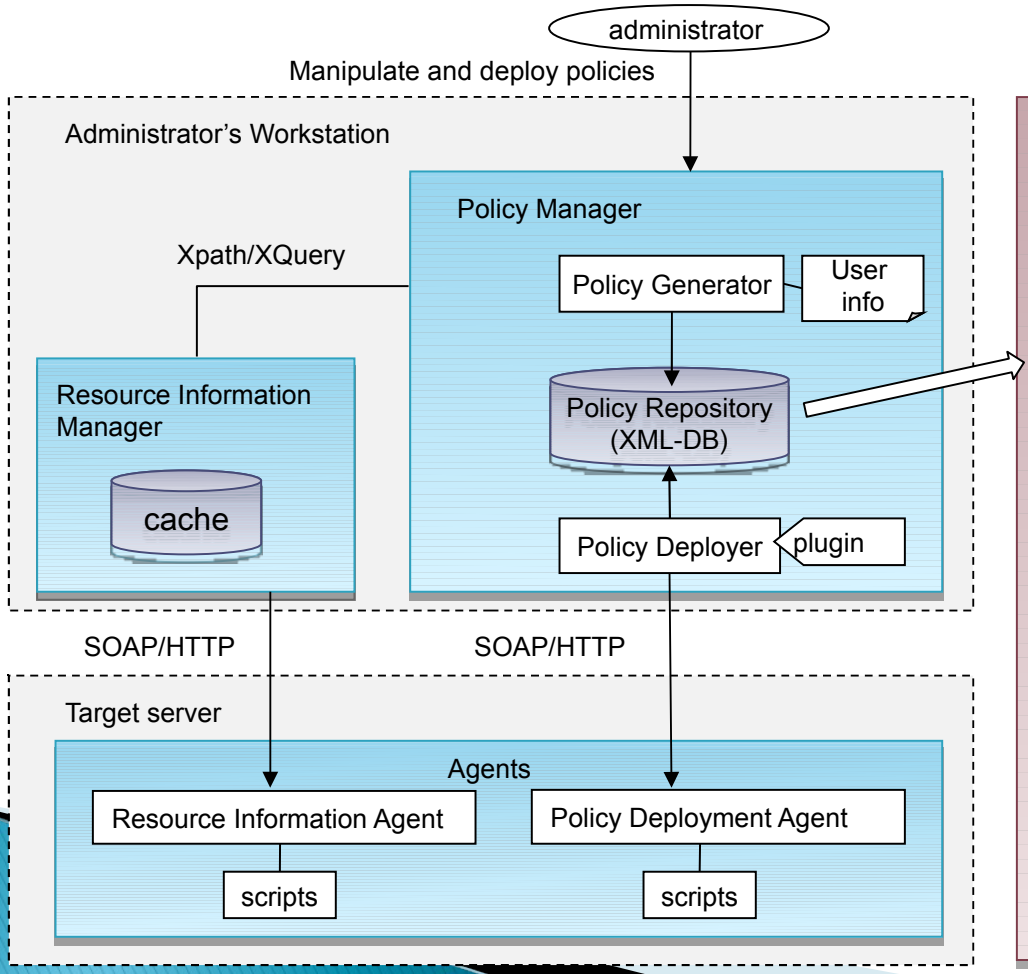
- The actions need to be controlled are "read", "write", and "execute“
- The action types are modeled by extending the CIM_Capabilities



Properties for identifying the set of actions supported by the file system

Implementation

- We implemented the IAM using Java, XMLDB, XACML, CIM-XML, Xpath/Xquery, SOAP/HTTP

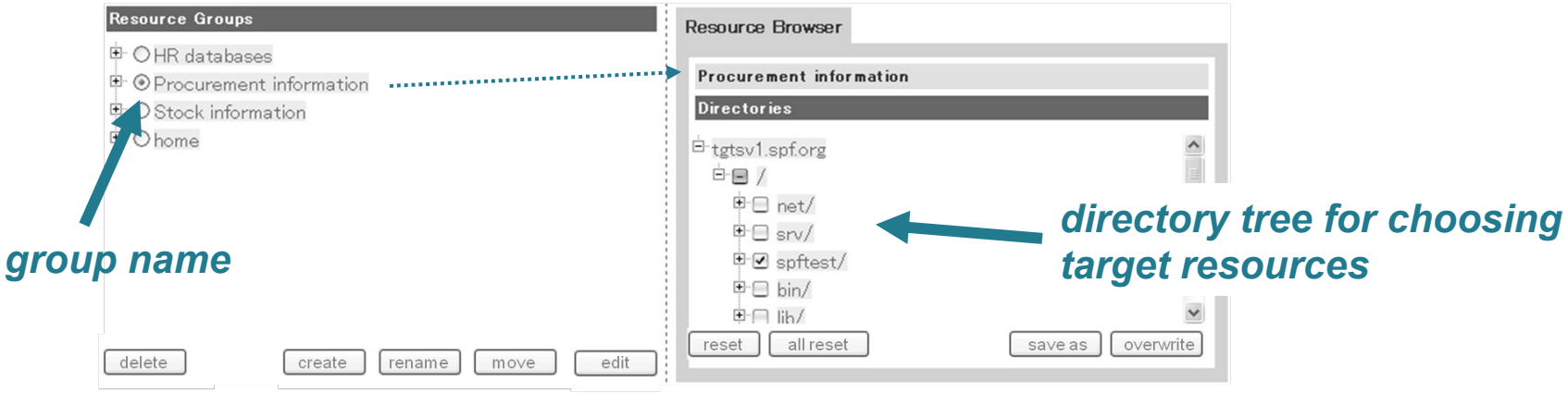


XACML policy

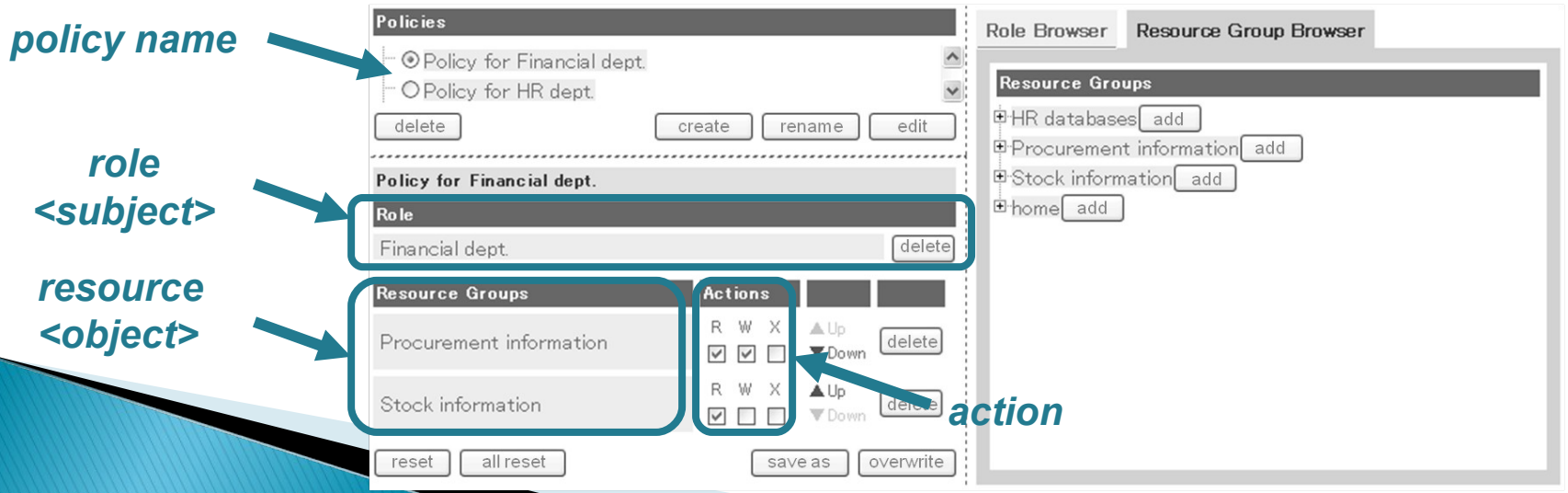
```
<Policy PolicyId="uuid-837423801-4837290">
  <Target>
    <Subjects>
      <Subject>
        <SubjectMatch MatchId="string-match">
          <SubjectAttributeDesignator … />
          <AttributeValue>
      AGlobalRoleId</AttributeValue>
        </SubjectMatch>
      </Subject>
    </Subjects>
    <Resources><AnyResources/></Resources>
    <Actions><AnyActions/></Actions>
  </Target>
  <Rule RuleId="rule-1" effect="deny">
   <Target>
     <Subjects><AnySubjects/></Subjects>
     <Resources>
       <Resource>
         <ResourceMatch MatchId="string-match">
           <ResourceAttributeDesignator  .. />
           <AttributeValue>
             AGlobalResourceId</AttributeValue>
         </ResourceMatch>
       </Resource>
     </Resouces>
     <Actions>
       <Action>read</Action>
       <Action>write</Action>
     </Actions>
   </Target>
  </Rule>
</Policy>
```

User Interface

(1) Making resource groups on the ***Resource Group Editor***

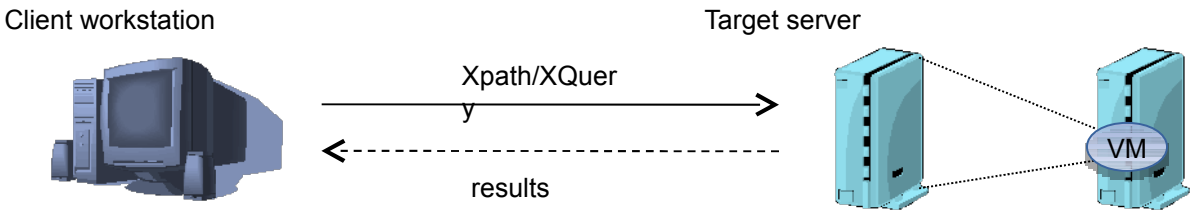


(2) Generating abstract policies on ***Abstract Policy Editor***



Query Performance

- Query response time is an important factor in the usability of the IAM
- We measured the query response time to Resource Information Manager



	Client
CPU	Pentium4 3.2GHz
Memory	1 GB
OS	Fedora Core 6

	Virtual Machine	Host Machine
CPU	Intel Xeon 3GHz	Intel Xeon 3GHz
Memory	512 MB	2 GB
OS	Cent OS 5	Windows Server 2003 VMware Player

Evaluation Results

- Most of queries take 2.5 seconds to get results
- Query for getting all CIM_LogicalFile instances below the root directory takes 5.7 seconds
  - We can avoid this inefficient query by using proposed SPF_Directory model

Query target	XQuery	Response time (s)
Instances of computer systems	for $instance in //INSTANCE[@CLASSNAME="CIM_ComputerSystem"] return {$instance}	2.493
An instance of root directory	for $instance in //INSTANCE[@CLASSNAME="SPF_Directory"] …	2.435
All instances just below root directory	for $instance in //INSTANCE[@CLASSNAME="SPF_Directory"] …	5.770
An instance of file access capabilities	for $instanceFSC in //INSTANCE[@CLASSNAME="SPF_FileSystemCapabilities"] …	2.523

Conclusion

- We proposed the architecture of the integrated access control manager (*IAM*) for the consolidated server systems
- IAM employs *CIM standards* for managing various types of access control modules
- In the pilot implementation, we apply CIM to model the file and directory information, reference monitor, and capabilities of file system
- We propose an *extension of the CIM_Directory* to improve the efficiency of directory browsing

Thank you !