Security Models: Past, Present and Future

Prof. Ravi Sandhu
Executive Director and Endowed Chair
Institute for Cyber Security
University of Texas at San Antonio
July 2009

ravi.sandhu@utsa.edu
www.profsandhu.com
THE BIG PICTURE
Security Objectives

INTEGRITY
modification

AVAILABILITY
access

CONFIDENTIALITY
disclosure
Security Objectives

- **INTEGRITY**: modification
- **AVAILABILITY**: access
- **CONFIDENTIALITY**: disclosure
- **USAGE**: purpose
Security Objectives

INTEGRITY
- modification

AVAILABILITY
- access

CONFIDENTIALITY
- disclosure

USAGE
- purpose

INTegrity
- modify

USABILITY
- access
Security Trends and Change Drivers

Stand-alone computers → Internet

Vandals → Criminals, Nation states, Terrorists

Enterprise security → Mutually suspicious yet mutually dependent security

Few standard services → Many and new innovative services

We are at an inflection point
“Now we face a new challenge to security, a world of shared computing and web services. As with radio, this technology is too valuable to go unused. By contrast with radio, which could be protected with cryptography, there may be no technology that can protect shared computation to the degree we would call secure today. In a decade or a generation, there may be no secure computing.”

Need to be realistic in our security expectations
● Computer scientists could never have designed the web because they would have tried to make it work. But the Web does “work.” What does it mean for the Web to “work”? 
● Security geeks could never have designed the ATM network because they would have tried to make it secure. But the ATM network is “secure. What does it mean for the ATM network to be “secure”? 
Foundational Security Assumptions

- Information needs to be protected
  - In motion
  - At rest
  - In use

- Absolute security is impossible and unnecessary
  - Trying to approximate absolute security is a bad strategy
  - “Good enough” security is feasible and meaningful
  - Better than “good enough” is bad

- Security is meaningless without application context
  - Cannot know we have “good enough” without this context

- Models and abstractions are all important
  - Without a conceptual framework it is hard to separate “what needs to be done” from “how we do it”

We are not very good at doing any of this
Our Basic Premise
- There can be no security without application context
- Courtney’s Law (1970s, 1980s ??):
  - You cannot say anything interesting (i.e. significant) about the security of a system except in the context of a particular application and environment

Corollary
- There can be no security model without application context

Reality
- Existing security models are application neutral
  - Assumption is they can be readily “configured” or “policy-fied” to suit application context

There is also a notion of technology context for security models but out of scope for this lecture
**Application Context**

<table>
<thead>
<tr>
<th>Software-Architect</th>
<th>Project</th>
<th>% Time</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>Vista</td>
<td>25%</td>
<td>U</td>
</tr>
<tr>
<td>Alice</td>
<td>SecureVista</td>
<td>75%</td>
<td>S</td>
</tr>
<tr>
<td>Bob</td>
<td>XP</td>
<td>100%</td>
<td>U</td>
</tr>
</tbody>
</table>

- **What precisely is Secret?**
  - There exists a SecureVista project
  - Alice works on SecureVista
  - Alice’s effort on SecureVista is 75%
  - All or some of the above

- **How do we maintain integrity of the database?**
  - Depends

**Much work and $$$ by researchers and vendors, late 80’s-early 90’s**
Emerging Application-Centric Era (ACE)

Applications are cyber analogs of previously existing enterprise-centric applications

- on-line banking
- brokerage
- e-retail
- auctions
- search engines

Future applications will be fundamentally different

- ?
- ?
- ?
- ?
This lecture is focused on the policy models layer.

At the policy layer security models are essentially access control models.
THE PAST
Access Control Models

- Discretionary Access Control (DAC)
  - Owner controls access but only to the original, not to copies

- Mandatory Access Control (MAC)
  Same as Lattice-Based Access Control (LBAC)
  - Access based on security labels
  - Labels propagate to copies

- Role-Based Access Control (RBAC)
  - Access based on roles
  - Can be configured to do DAC or MAC
### Objects (and Subjects)

<table>
<thead>
<tr>
<th>Subjects</th>
<th>Objects (and Subjects)</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>r w own</td>
</tr>
<tr>
<td>V</td>
<td>G</td>
</tr>
<tr>
<td></td>
<td>r</td>
</tr>
</tbody>
</table>

**Rights**
each column of the access matrix is stored with the object corresponding to that column
each row of the access matrix is stored with the subject corresponding to that row
### Access Control Triples

<table>
<thead>
<tr>
<th>Subject</th>
<th>Access</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>r</td>
<td>F</td>
</tr>
<tr>
<td>U</td>
<td>w</td>
<td>F</td>
</tr>
<tr>
<td>U</td>
<td>own</td>
<td>F</td>
</tr>
<tr>
<td>U</td>
<td>r</td>
<td>G</td>
</tr>
<tr>
<td>V</td>
<td>r</td>
<td>G</td>
</tr>
<tr>
<td>V</td>
<td>w</td>
<td>G</td>
</tr>
<tr>
<td>V</td>
<td>own</td>
<td>G</td>
</tr>
</tbody>
</table>

Commonly used in relational database management systems.
B cannot read file F
TROJAN HORSE EXAMPLE

Program Goodies

executes

A

read

write

File F

File G

B can read contents of file F copied to file G

ACL

A:r

A:w

B:r

A:w

© Ravi Sandhu
- Traditional DAC does not prevent copies from being made and there is no control over copies
  - Modern approaches to information sharing and trusted computing seek to maintain control over copies
- Traditional DAC is weak with respect to confidentiality but may have value with respect to integrity
LATTICE STRUCTURES

dominance ≥ can-flow

Top Secret
Secret
Confidential
Unclassified

© Ravi Sandhu
SIMPLE-SECURITY
Subject S can read object O only if

  • label(S) dominates label(O)

STAR-PROPERTY (LIBERAL)
Subject S can write object O only if

  • label(O) dominates label(S)

STAR-PROPERTY (STRICT)
Subject S can write object O only if

  • label(O) equals label(S)
LATTICE STRUCTURES

Compartments and Categories

\{\text{ARMY, CRYPTO}\}

\{\text{ARMY}\} \quad \{\text{CRYPTO}\}

\{\}\n
© Ravi Sandhu
Hierarchical Classes with Compartments

The product of 2 lattices is a lattice.
Hierarchical Classes with Compartments

\[ \text{TS, \{A,B\}} \]
\[ \text{TS, \{A\}} \]
\[ \text{TS, \{B\}} \]
\[ \text{S, \{A,B\}} \]
\[ \text{S, \{A\}} \]
\[ \text{S, \{B\}} \]
\[ \text{S, \{\}} \]
EQUIVALENCE OF BLP AND BIBA

HI (High Integrity)  \[\Rightarrow\]  LI (Low Integrity)

LI (Low Integrity)  \[\Rightarrow\]  HI (High Integrity)

BIBA LATTICE  \[\Rightarrow\]  EQUIVALENT BLP LATTICE
EQUIVALENCE OF BLP AND BIBA

BLP LATTICE

EQUIVALENT BIBA LATTICE

HS (High Secrecy)  LS (Low Secrecy)

⇒

LS (Low Secrecy)  HS (High Secrecy)
COMBINATION OF DISTINCT LATTICES

\[
\begin{align*}
\text{HS} & \quad \text{HI} \\
\text{LS} & \quad \text{LI}
\end{align*}
\]

\[
\Rightarrow \quad \begin{align*}
\text{HS, LI} & \quad \text{HS, HI} \\
\text{LS, LI} & \quad \text{LS, HI}
\end{align*}
\]

EQUIVALENT BLP LATTICE

BLP \quad BIBA

GIVEN

© Ravi Sandhu
**LEGEND**

S: Subjects  
O: Objects

**S: System Managers**  
O: Audit Trail

**S: System Control**

**S: Application Programmers**  
O: Development Code and Data

**S: System Programmers**  
O: System Code in Development

O: Repair Code

O: Production Code

O: Tools

O: System Programs

O: Production Data

O: Development Code and Data

S: Application Programmers

S: System Programmers

S: System Managers
CHINESE WALL EXAMPLE

BANKS

A   B

OIL COMPANIES

X   Y
CHINESE WALL LATTICE

SYSHIGH

A, X  A, Y  B, X  B, Y

SYSLOW

A, -  -, X  -, Y  B, -
COVERT CHANNELS

High User

Information is leaked unknown to the high user

High Trojan Horse Infected Subject

COVERT CHANNEL

Low Trojan Horse Infected Subject

Low User
MAC/LBAC Summary

- LBAC fails to control covert channels
- LBAC fails to control inference and aggregation
- It is too rigid for most commercial applications
- It has strong mathematical foundations
RBAC: Role-Based Access Control

- Access is determined by roles
- A user’s roles are assigned by security administrators
- A role’s permissions are assigned by security administrators

First emerged: mid 1970s
First models: mid 1990s

Is RBAC MAC or DAC or neither?
- RBAC can be configured to do MAC
- RBAC can be configured to do DAC
- RBAC is policy neutral

RBAC is neither MAC nor DAC!
RBAC96 Model

USER-ROLE ASSIGNMENT

PERMISSIONS-ROLE ASSIGNMENT

SESSIONS

© Ravi Sandhu
ROLE HIERARCHIES

USER-ROLE ASSIGNMENT

SESSIONS

PERMISSIONS-ROLE ASSIGNMENT

USERS

ROLES

PERMISSIONS

© Ravi Sandhu
RBAC96 Model

ROLE HIERARCHIES

USER-ROLE ASSIGNMENT

PERMISSIONS-ROLE ASSIGNMENT

SESSIONS

CONTRAINTS
Example Role Hierarchy

Director (DIR)

Project Lead 1 (PL1)
- Production 1 (P1)
- Engineer 1 (E1)
- Quality 1 (Q1)

Project Lead 2 (PL2)
- Production 2 (P2)
- Engineer 2 (E2)
- Quality 2 (Q2)

Engineering Department (ED)
- Employee (E)

Inheritance and activation hierarchy
Permission-role review is advanced requirement

Overall formal model is more complete

Limited to separation of duties
The RBAC Story

Proposed Standard

Standard Adopted

RBAC96 paper

Amount of Publications

Year of Publication

Pre-RBAC Early RBAC 1st expansion phase 2nd expansion phase

© Ravi Sandhu
THE PRESENT
Institute for Cyber Security

Access Control Models

- **Discretionary Access Control (DAC)**
  - Owner controls access but only to the original, not to copies

- **Mandatory Access Control (MAC)**
  - Same as Lattice-Based Access Control (LBAC)
  - Access based on security labels
  - Labels propagate to copies

- **Role-Based Access Control (RBAC)**
  - Access based on roles
  - Can be configured to do DAC or MAC

- **Attribute-Based Access Control (ABAC)**
  - Access based on attributes, to possibly include roles, security labels and whatever
Founding Principles of RBAC96

- **Abstraction** of Privileges
  - Credit is different from Debit even though both require read and write

- **Separation** of Administrative Functions
  - Separation of user-role assignment from role-permission assignment

- **Least Privilege**
  - Right-size the roles
  - Don’t activate all roles all the time

- **Separation of Duty**
  - Static separation: purchasing manager versus accounts payable manager
  - Dynamic separation: cash-register clerk versus cash-register manager
ASCRAA Principles for Future Access Control

- **Abstraction** of Privileges
  - Credit vs debit
  - Personalized permissions
- **Separation** of Administrative Functions
- **Containment**
  - Least Privilege
  - Separation of Duties
  - Usage Limits
- **Automation**
  - Revocation
  - Assignment: (i) Self-assignment, (ii) Attribute-based
  - Context and environment adjustment
- **Accountability**
  - Re-authentication/Escalated authentication
  - Click-through obligations
  - Notification and alerts
Usage Control Scope

Security Objectives

Privacy Protection
Intellectual Property Rights Protection
Sensitive Information Protection

Security Architectures

Server-side Reference Monitor (SRM)  Client-side Reference Monitor (CRM)  SRM & CRM

DRM
• unified model integrating
  • authorization
  • obligation
  • conditions
• and incorporating
  • continuity of decisions
  • mutability of attributes

Usage Control Model (UCON)

© Ravi Sandhu 51
Usage Control Model (UCON)

- DAC
- LBAC
- RBAC
- ABAC
- … and many, many others
- UCON
  - ABAC on steroids
  - Simple, familiar, usable and effective use cases demonstrate the need for UCON
    - Automatic Teller Machines
    - CAPTCHA at Public web sites
    - End User Licence Agreements
    - Terms of Usage for WiFi in Hotels, Airports
    - Rate limits on call center workers
THE FUTURE
Our Basic Premise
- There can be no security model without application context

So how does one customize an application-centric security model?
- Meaningfully combine the essential insights of
  - DAC, LBAC, RBAC, ABAC, UCON, etcetera
- Directly address the application-specific trade-offs
  - Within the security objectives of confidentiality, integrity and availability
  - Across security, performance, cost and usability objectives
- Separate the real-world concerns of
  - practical distributed systems and ensuing staleness and approximations (enforcement layer) from
  - policy concerns in an idealized environment (policy layer)
This lecture is focused on the policy models layer
Extensive research in the last two decades
- ORCON, DRM, ERM, XrML, ODRL, etc.
Copy/usage control has received major attention
Manageability problem largely unaddressed

Dissemination Chain with Sticky Policies on Objects
Group-Centric Sharing (g-SIS)

- Brings users & objects together in a group
  - Focuses on manageability using groups
  - Co-exists with dissemination-centric
  - Two metaphors
    - Secure Meeting Room (E.g. Program committee)
    - Subscription Model (E.g. Secure multicast)

- Operational aspects
  - Group characteristics
    - E.g. Are there any core properties?
  - Group operation semantics
    - E.g. What is authorized by join, add, etc.?
  - Read-only Vs Read-Write

- Administrative aspects
  - E.g. Who authorizes join, add, etc.?
  - May be application dependant

- Multiple groups
  - Inter-group relationship
g-SIS Operation Semantics

GROUP
Authz (u,o,r)?

Users
join
leave

Objects
add
remove
g-SIS Operation Semantics

GROUP
Authz (u,o,r)?

- Users: Strict Join, Liberal Join, Strict Add, Liberal Add
- Objects: Strict Leave, Liberal Leave, Strict Remove, Liberal Remove
Family of g-SIS Policy Models

Subject Model

<SJ, SL>
<SJ, LL>
<LJ, LL>
Part (e): (a) X (b)
g-SIS models: (e) X (f)

Object Model

<SA, SR>
<LA, SR>
<LA, LR>
Part (f): (c) X (d)

Traditional Groups: <LJ, SL, LA, SR>
Secure Multicast: <SJ, LL, LA, *>

Most Restrictive g-SIS Specification:

□(Authz ↔ (¬SR ∧ ¬SL) S (SA ∧ (¬SL S SJ))))
g-SIS Enforcement Model

CC: Control Center
GA: Group Administrator

Subject Attributes: {id, Join-TS, Leave-TS, ORL, gKey}
ORL: Object Revocation List
gKey: Group Key

Object Attributes: {id, Add-TS}

Refresh Time (RT): TRM contacts CC to update attributes
From Policy to Enforcement

- Additional Trusted/Semi-Trusted Servers
- Approximate Enforcement

- Finally, the Implementation layer models spell out protocol details and details of TRM algorithms
CONCLUSION
THE PAST
- Discretionary Access Control (DAC)
- Mandatory Access Control (MAC)
  - Equivalently Lattice-Based Access Control (LBAC)
- Role-Based Access Control (RBAC)

THE PRESENT
- Usage Control (UCON)
  - Attribute-Based Access Control (ABAC) on steroids

THE FUTURE
- Application-Centric Access Control Models
- Technology-Centric Access Control Models

Models are all important
A Policy Language is not a substitute for a good model