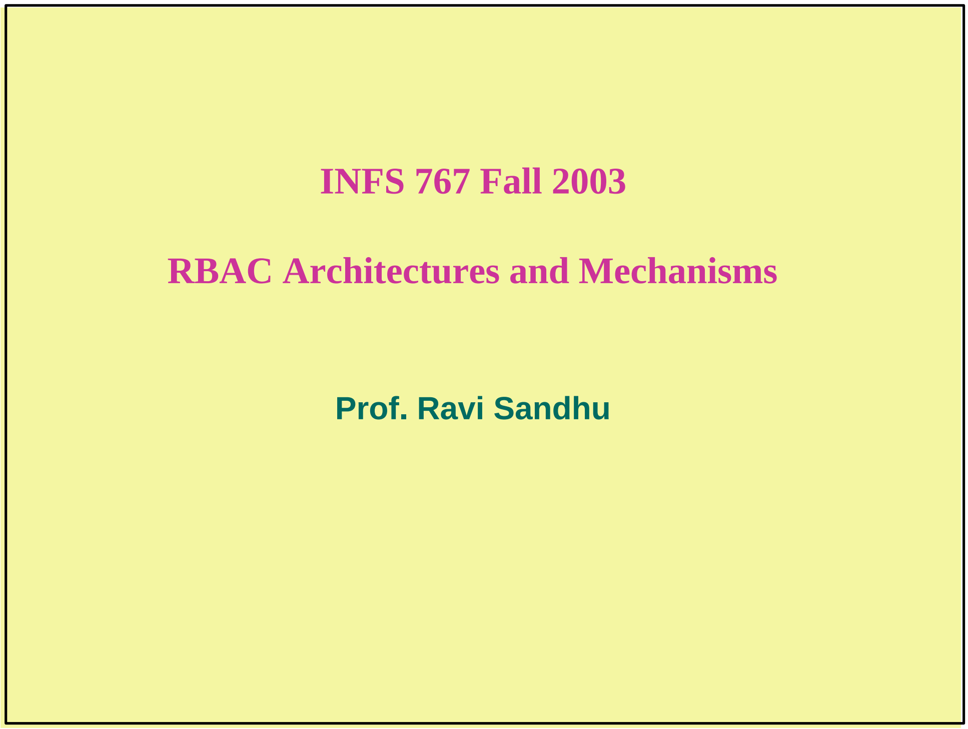
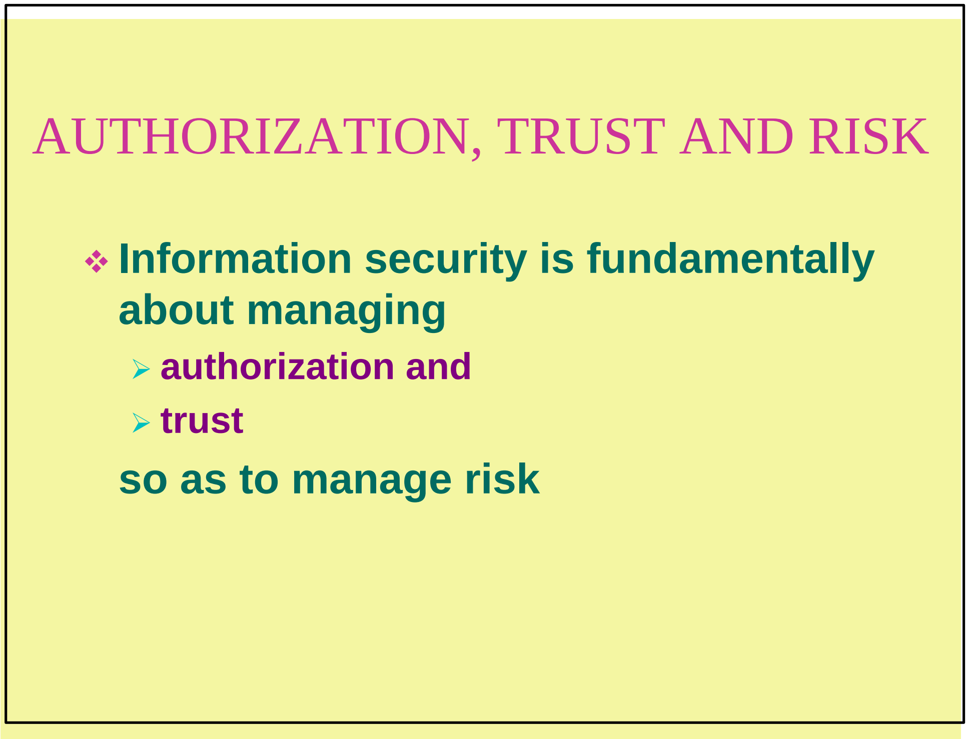


INFS 767 Fall 2003

RBAC Architectures and Mechanisms

Prof. Ravi Sandhu

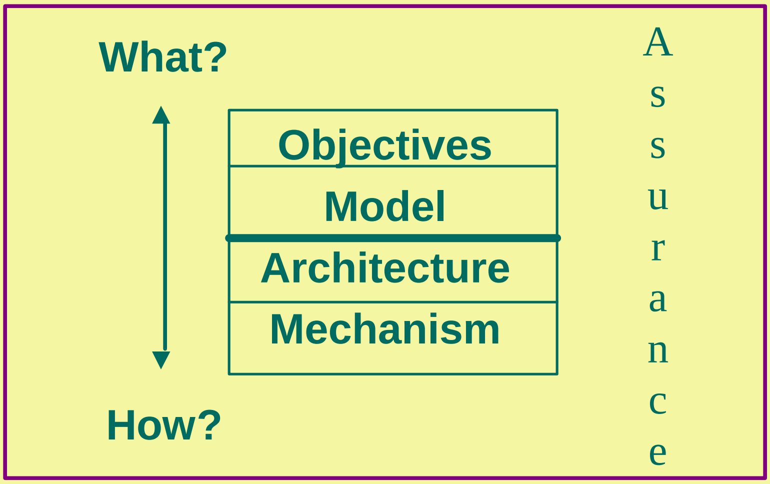


AUTHORIZATION, TRUST AND RISK

- Information security is fundamentally about managing
  - authorization and
  - trust

  so as to manage risk

THE OM-AM WAY



What?

Objectives

Model

Architecture

Mechanism

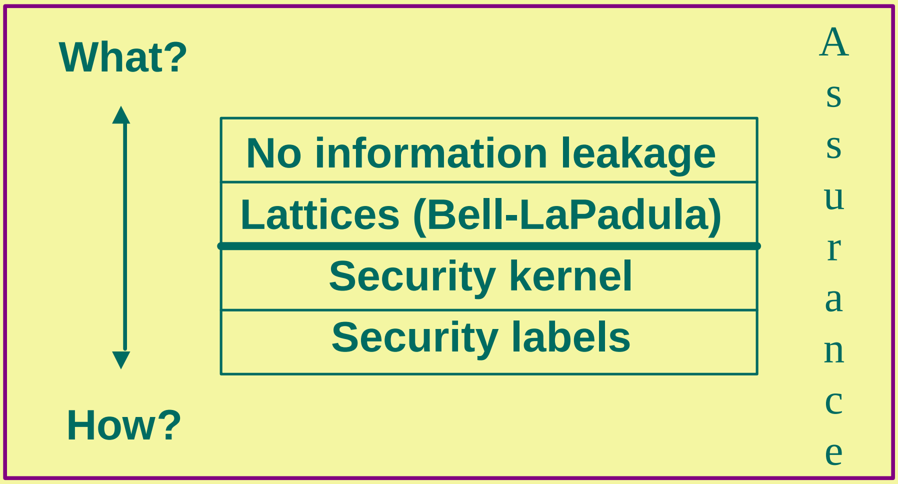
How?

Assurance

LAYERS AND LAYERS

- Multics rings
- Layered abstractions
- Waterfall model
- Network protocol stacks
- Napolean layers
- RoFi layers
- OM-AM
- etcetera

OM-AM AND MANDATORY ACCESS CONTROL (MAC)



What?

No information leakage

Lattices (Bell-LaPadula)

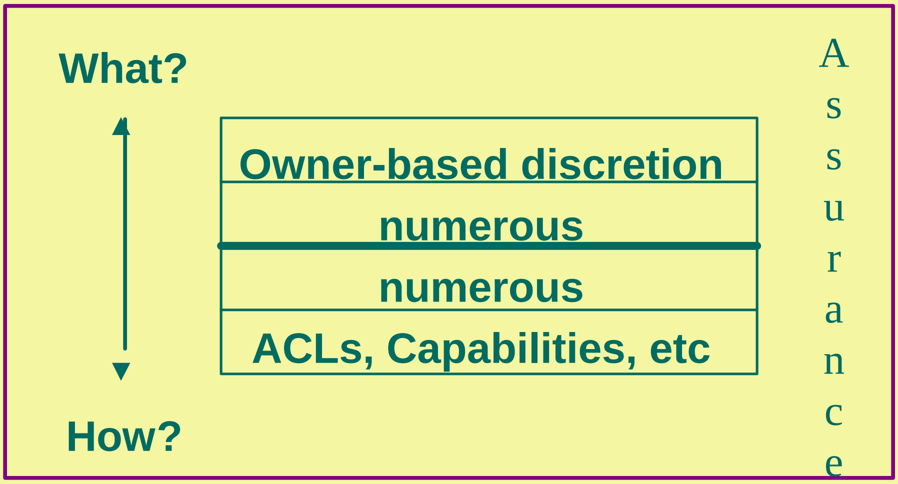
Security kernel

Security labels

How?

Assurance

OM-AM AND DISCRETIONARY ACCESS CONTROL (DAC)



What?

Owner-based discretion

numerous

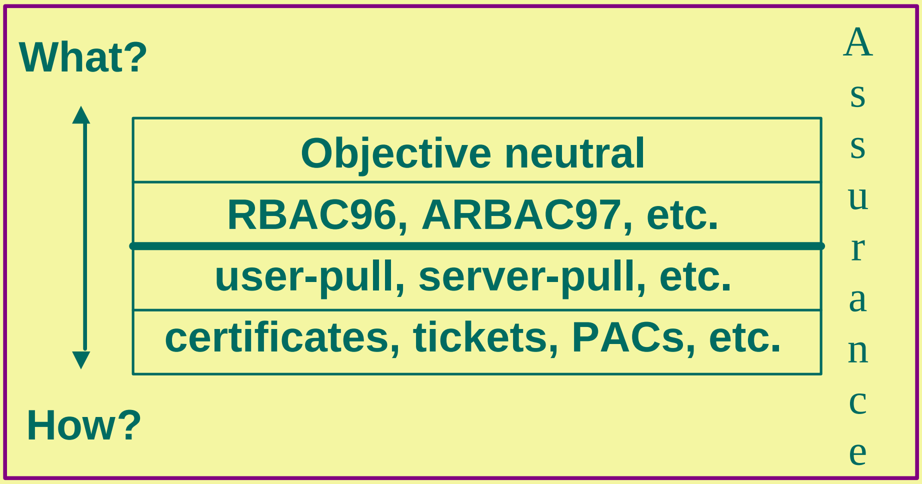
numerous

ACLs, Capabilities, etc

How?

Assurance

OM-AM AND ROLE-BASED ACCESS CONTROL (RBAC)



What?

Objective neutral

RBAC96, ARBAC97, etc.

user-pull, server-pull, etc.

certificates, tickets, PACs, etc.

How?

Assurance

DISTRIBUTED RBAC (DRBAC) CASE STUDY

- Approximately a dozen physical sites
- Approximately 2-3 simulation models/site
- Fewer than 100 roles structured in a very shallow hierarchy
  - A subset of roles is used in any single simulation model
- Fewer than 100 users
- A user uses only one role at a time
  - Convenient but not critical
- Moderate rate of change

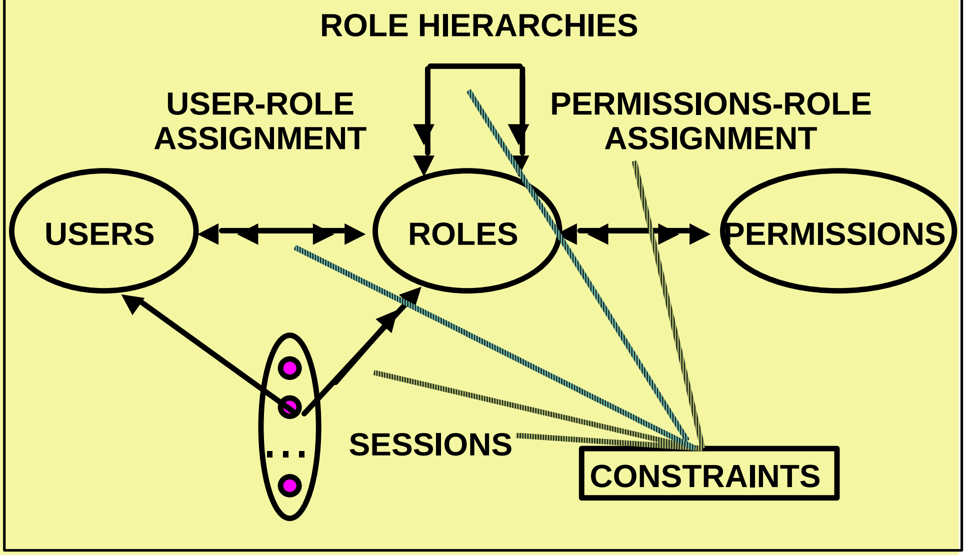
DISTRIBUTED RBAC (DRBAC) CASE STUDY

- Permission-role assignment
  - Locally determined at each simulation model
- User-role assignment
  - A user can be assigned to a role if and only if all simulation models using that role agree
  - A user is revoked from a role if and only if any simulation model using that role revokes the user

DISTRIBUTED RBAC (DRBAC) CASE STUDY

- Each simulation model has a security administrator role authorized to carry out these administrative tasks
- A simulation model can assign permissions to a role X at any time
  - even if X is previously unused in that simulation model
- Consequently any simulation model can revoke any user from any role!

RBAC3



MODEL CUSTOMIZATION

- Each session has a single role
- SM = {sm1, …, smk}, simulation models
- OP = {op1, …, opl}, operations
- P= SM X OP, permissions
- SMA = {sma1, …, smk}, administrative roles
- R Ç SMA = Æ
- Admin: SM « SMA

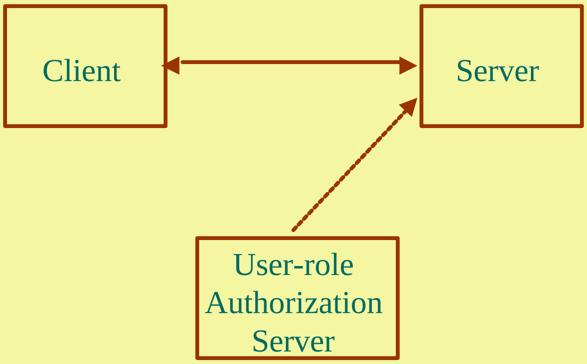
MODEL CUSTOMIZATION

- Can formalize the administrative rules given earlier
- For each simulation model designate a unique user to be the chief security administrator who is authorized to assign and revoke users from the security administrator role for that model

DRBAC ARCHITECTURES

- Permission-role
  - Enforced locally at each simulation model
- Permission-role administration
  - Enforced locally at each simulation model
  - May need to communicate to other simulation models
- User-role
  - See following slides
- User-role administration
  - Centralized or decentralized

SERVER MIRROR

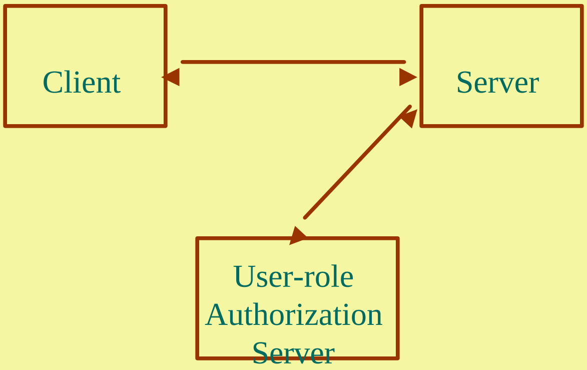


Client

Server

User-role Authorization Server

SERVER-PULL

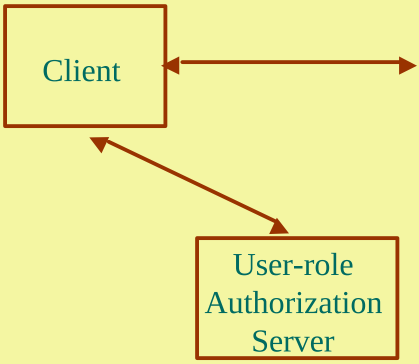


Client

Server

User-role Authorization Server

USER-PULL

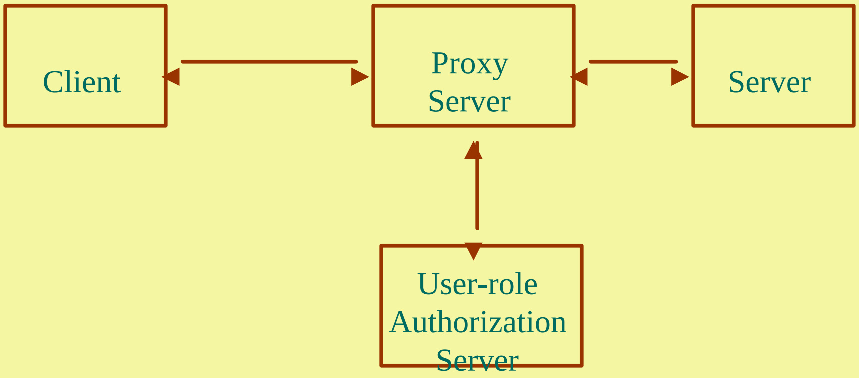


Client

Server

User-role Authorization Server

PROXY-BASED



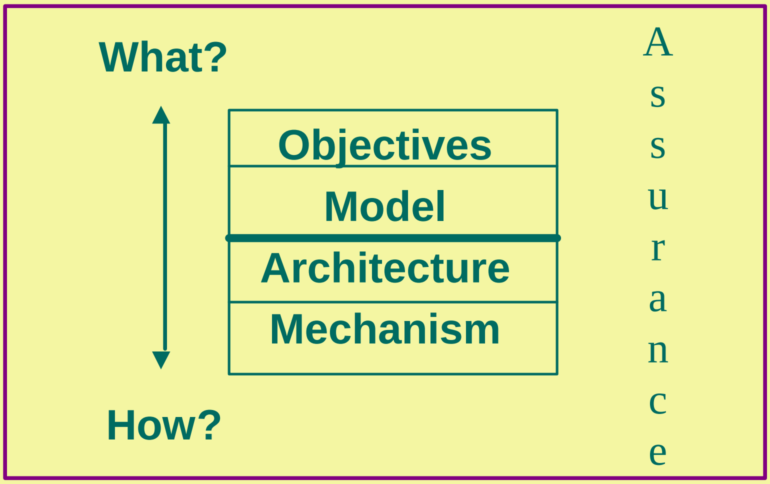
Client

Proxy Server

Server

User-role Authorization Server

THE OM-AM WAY


What?

Objectives

Model

Architecture

Mechanism

How?

Assurance

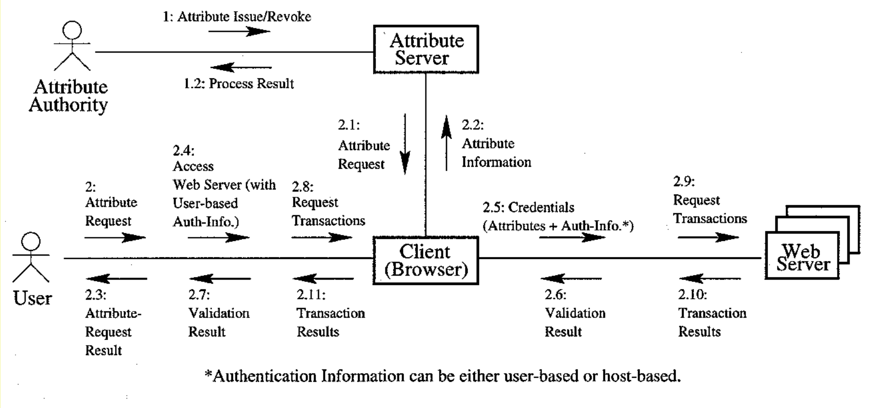
Secure Attribute Services on the Web

- WWW (World Wide Web)
  - widely used for electronic commerce and business
  - supports synthesis of technologies
  - mostly, Web servers use identity-based access control
    - scalability problem

Background

- An attribute
  - a particular property of an entity
    - e.g., role, identity, SSN, clearance, etc.
- If attributes are provided securely,
  - Web servers can use those attributes
    - e.g., authentication, authorization, access control, electronic commerce, etc.
- A successful marriage of the Web and secure attribute services is required

User-Pull Architecture

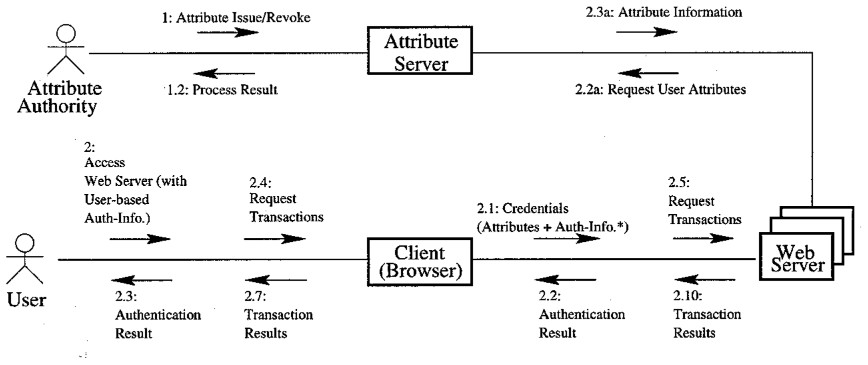

1: Attribute Issue/Revoke
1.2: Process Result
Attribute Authority
Attribute Server
2.1: Attribute Request
2.2: Attribute Information
2.4: Access Web Server (with User-based Auth-Info.)
2: Attribute Request
2.8: Request Transactions
2.5: Credentials (Attributes + Auth-Info.*)
2.9: Request Transactions
User
Client (Browser)
Web Server
2.3: Attribute-Request Result
2.7: Validation Result
2.11: Transaction Results
2.6: Validation Result
2.10: Transaction Results

*Authentication Information can be either user-based or host-based.

User-Pull Architecture

- Each user
  - pulls appropriate attributes from the Attribute Server
  - presents attributes and authentication information to Web servers
- Each Web server
  - requires both identification and attributes from users
- High performance
  - No new connections for attributes

Server-Pull Architecture

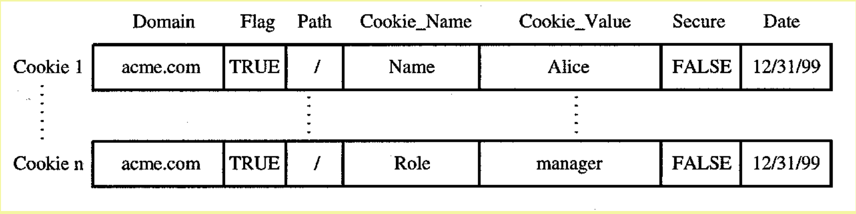

1: Attribute Issue/Revoke
Attribute Authority
1.2: Process Result
Attribute Server
2.3a: Attribute Information
2.2a: Request User Attributes
2: Access Web Server (with User-based Auth-Info.)
2.4: Request Transactions
2.1: Credentials (Attributes + Auth-Info.*)
2.5: Request Transactions
User
Client (Browser)
Web Server
2.3: Authentication Result
2.7: Transaction Results
2.2: Authentication Result
2.10: Transaction Results

*Authentication Information can be either user-based or host-based.

Related Technologies

- Cookies
  - in widespread current use for maintaining state of HTTP
  - becoming standard
  - not secure
- Public-Key Certificates (X.509)
  - support security on the Web based on PKI
  - standard
  - simply, bind users to keys
  - have the ability to be extended

Cookies

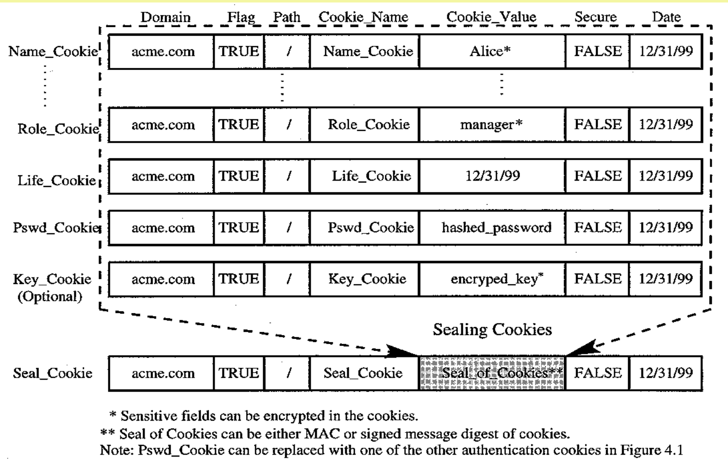


	Domain	Flag	Path	Cookie_Name	Cookie_Value	Secure	Date
Cookie 1	acme.com	TRUE	/	Name	Alice	FALSE	12/31/99
⋮			⋮		⋮		
Cookie n	acme.com	TRUE	/	Role	manager	FALSE	12/31/99

Security Threats to Cookies

- Cookies are not secure
  - No authentication
  - No integrity
  - No confidentiality
- can be easily attacked by
  - Network Security Threats
  - End-System Threats
  - Cookie Harvesting Threats

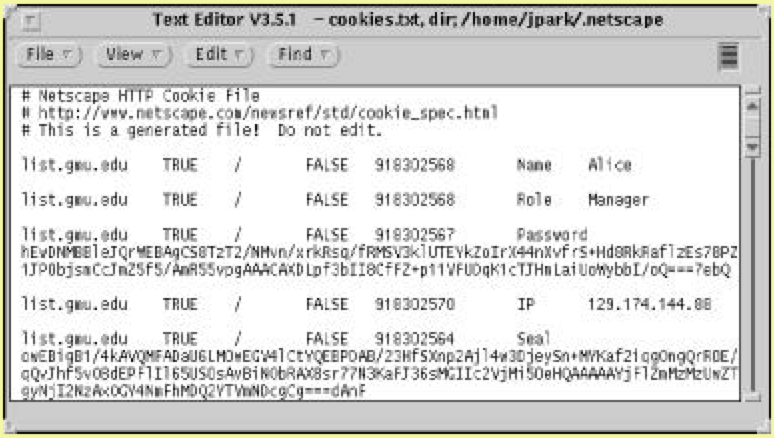
Secure Cookies on the Web



	Domain	Flag	Path	Cookie_Name	Cookie_Value	Secure	Date
Name_Cookie	acme.com	TRUE	/	Name_Cookie	Alice*	FALSE	12/31/99
⋮			⋮		⋮		
Role_Cookie	acme.com	TRUE	/	Role_Cookie	manager*	FALSE	12/31/99
Life_Cookie	acme.com	TRUE	/	Life_Cookie	12/31/99	FALSE	12/31/99
Pswd_Cookie	acme.com	TRUE	/	Pswd_Cookie	hashed_password	FALSE	12/31/99
Key_Cookie (Optional)	acme.com	TRUE	/	Key_Cookie	encryped_key*	FALSE	12/31/99
Seal_Cookie	acme.com	TRUE	/	Seal_Cookie	Seal_of_Cookies**	FALSE	12/31/99

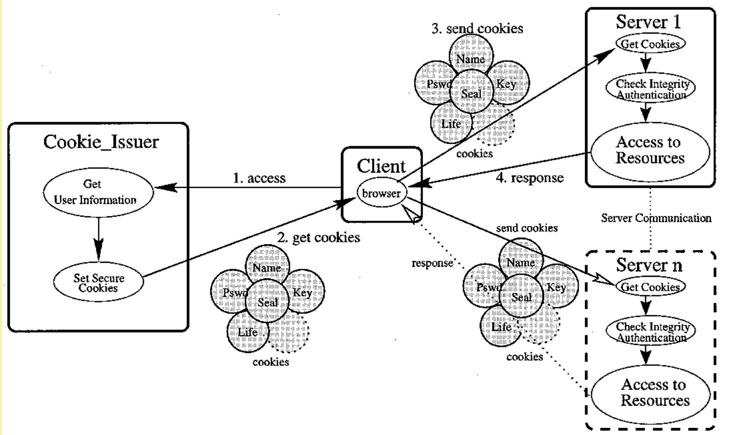
* Sensitive fields can be encrypted in the cookies.
** Seal of Cookies can be either MAC or signed message digest of cookies.
Note: Pswd_Cookie can be replaced with one of the other authentication cookies in Figure 4.1

A Set of Secure Cookies


Text Editor V3.5.1 – cookies.txt, dir; /home/jpark/.netscape
File
View
Edit
Find

```
# Netscape HTTP Cookie File
# http://www.netscape.com/newsref/std/cookie_spec.html
# This is a generated file!  Do not edit.

list.gmu.edu    TRUE    /       FALSE   918302568       Name    Alice

list.gmu.edu    TRUE    /       FALSE   918302568       Role    Manager

list.gmu.edu    TRUE    /       FALSE   918302567       Password
hEvDNMBBleJQrMEBAqCS8TzT2/NHvn/xrkRsq/fRMSV3klUTEYkZoIrX44nXvfrS+Hd8RkRaflzEs78PZ
1JP0bjsmCcJm25f5/AmR55vpgAAACAKDLpf3bII8CfFZ+p11VFUDqK1cTJHmLaiUoWybbI/oQ===7ebQ

list.gmu.edu    TRUE    /       FALSE   918302570       IP      129.174.144.88

list.gmu.edu    TRUE    /       FALSE   918302564       Seal
oWEBigB1/4kAVQMFADaU6LMOwEGV4lCtYQEBPDAB/23HfSXnp2Ajl4w3DjeySn+MYKaf2iqgOngQrROE/
qQvJhf5vO8dEPFlIl65USOsAvBiN0bRAX8sr77N3KaFJ36sMGIIc2VjMi5OeHQAAAAAYjFlZmMzMzUwZT
gyNjI2NzAxOGY4NmFhMDQ2YTVmNDcgCg===dAnF
```

How to Use Secure Cookies

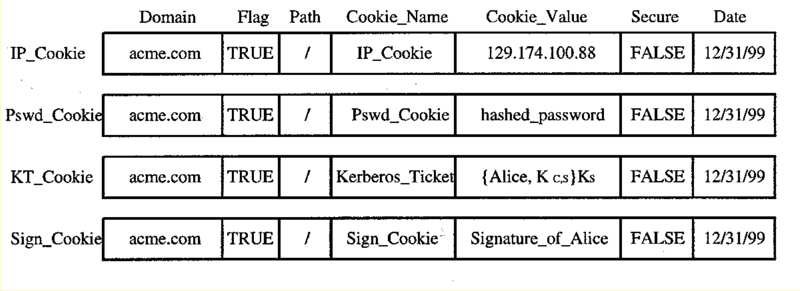


Pswd_Cookie can be replaced with one of the other authentication cookies in Figure 4.1

Applications of Secure Cookies

- User Authentication
- Electronic Transaction
- Eliminating Single-Point Failure
- Pay-per-Access
- Attribute-based Access Control

Authentication Cookies

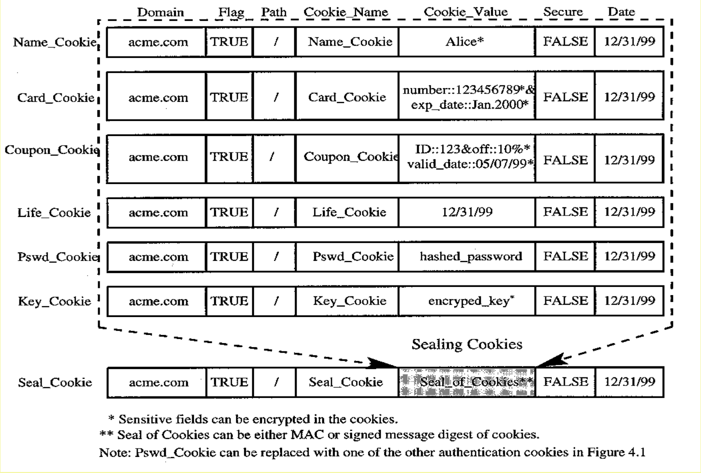


	Domain	Flag	Path	Cookie_Name	Cookie_Value	Secure	Date
IP_Cookie	acme.com	TRUE	/	IP_Cookie	129.174.100.88	FALSE	12/31/99
Pswd_Cookie	acme.com	TRUE	/	Pswd_Cookie	hashed_password	FALSE	12/31/99
KT_Cookie	acme.com	TRUE	/	Kerberos_Ticket	{Alice, $K_{c,s}$}K_s	FALSE	12/31/99
Sign_Cookie	acme.com	TRUE	/	Sign_Cookie	Signature_of_Alice	FALSE	12/31/99

Server-Pull Architecture

- Each user
  - presents only authentication information to Web servers
- Each Web server
  - pulls users’ attributes from the Attribute Server
- Authentication information and attribute do not go together
- More convenient for users
- Less convenient for Web servers

Secure Cookies for Electronic Transactions



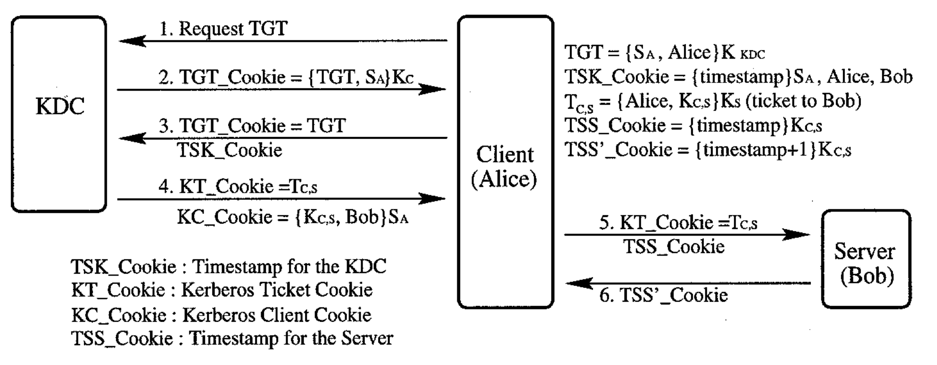
	Domain	Flag	Path	Cookie_Name	Cookie_Value	Secure	Date
Name_Cookie	acme.com	TRUE	/	Name_Cookie	Alice*	FALSE	12/31/99
Card_Cookie	acme.com	TRUE	/	Card_Cookie	number::123456789*& exp_date::Jan.2000*	FALSE	12/31/99
Coupon_Cookie	acme.com	TRUE	/	Coupon_Cookie	ID::123&off::10%* valid_date::05/07/99*	FALSE	12/31/99
Life_Cookie	acme.com	TRUE	/	Life_Cookie	12/31/99	FALSE	12/31/99
Pswd_Cookie	acme.com	TRUE	/	Pswd_Cookie	hashed_password	FALSE	12/31/99
Key_Cookie	acme.com	TRUE	/	Key_Cookie	encryped_key*	FALSE	12/31/99
Seal_Cookie	acme.com	TRUE	/	Seal_Cookie	Seal_of_Cookies**	FALSE	12/31/99

* Sensitive fields can be encrypted in the cookies.

** Seal of Cookies can be either MAC or signed message digest of cookies.

Note: Pswd_Cookie can be replaced with one of the other authentication cookies in Figure 4.1

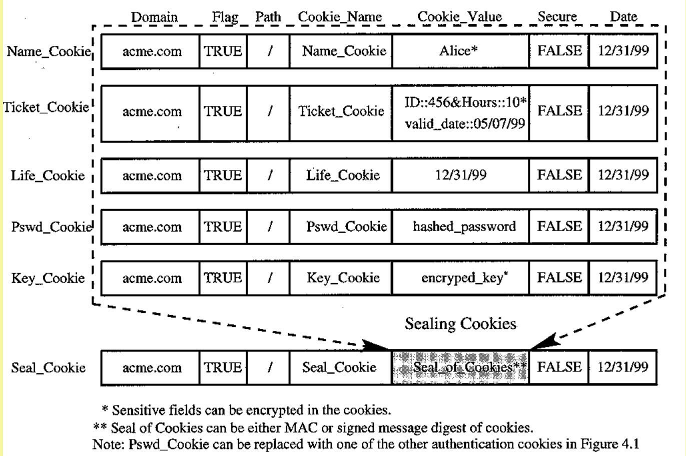
Kerberos-Based Authentication by Secure Cookies



TGT = {S_A, Alice}K_{KDC}
TSK_Cookie = {timestamp}S_A, Alice, Bob
$T_{C,S}$ = {Alice, $K_{C,S}$}K_S (ticket to Bob)
TSS_Cookie = {timestamp}$K_{C,S}$
TSS'_Cookie = {timestamp+1}$K_{C,S}$

TSK_Cookie : Timestamp for the KDC
KT_Cookie : Kerberos Ticket Cookie
KC_Cookie : Kerberos Client Cookie
TSS_Cookie : Timestamp for the Server

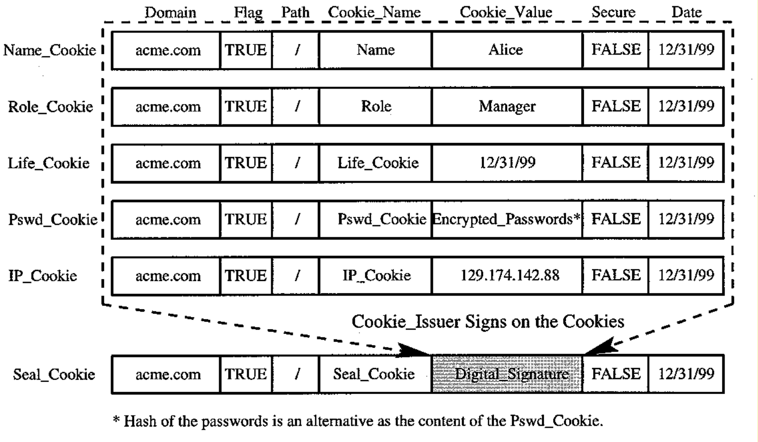
Secure Cookies for Pay-Per-Access



	Domain	Flag	Path	Cookie_Name	Cookie_Value	Secure	Date
Name_Cookie	acme.com	TRUE	/	Name_Cookie	Alice*	FALSE	12/31/99
Ticket_Cookie	acme.com	TRUE	/	Ticket_Cookie	ID::456&Hours::10* valid_date::05/07/99	FALSE	12/31/99
Life_Cookie	acme.com	TRUE	/	Life_Cookie	12/31/99	FALSE	12/31/99
Pswd_Cookie	acme.com	TRUE	/	Pswd_Cookie	hashed_password	FALSE	12/31/99
Key_Cookie	acme.com	TRUE	/	Key_Cookie	encryped_key*	FALSE	12/31/99
Seal_Cookie	acme.com	TRUE	/	Seal_Cookie	Seal_of_Cookies**	FALSE	12/31/99

* Sensitive fields can be encrypted in the cookies.
** Seal of Cookies can be either MAC or signed message digest of cookies.
Note: Pswd_Cookie can be replaced with one of the other authentication cookies in Figure 4.1

Secure Cookies for RBAC

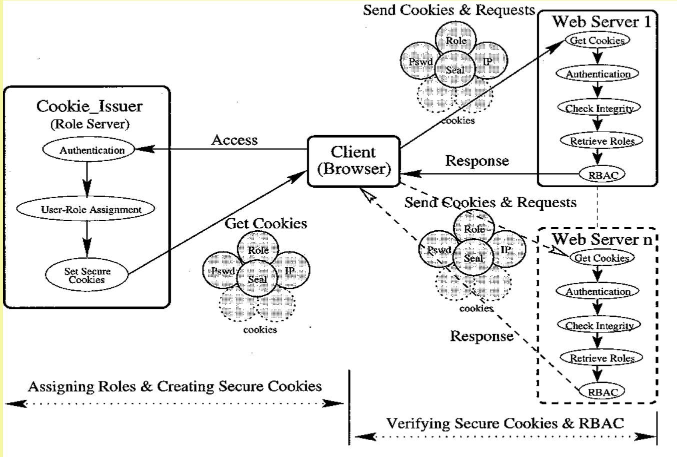


	Domain	Flag	Path	Cookie_Name	Cookie_Value	Secure	Date
Name_Cookie	acme.com	TRUE	/	Name	Alice	FALSE	12/31/99
Role_Cookie	acme.com	TRUE	/	Role	Manager	FALSE	12/31/99
Life_Cookie	acme.com	TRUE	/	Life_Cookie	12/31/99	FALSE	12/31/99
Pswd_Cookie	acme.com	TRUE	/	Pswd_Cookie	Encrypted_Passwords*	FALSE	12/31/99
IP_Cookie	acme.com	TRUE	/	IP_Cookie	129.174.142.88	FALSE	12/31/99
Seal_Cookie	acme.com	TRUE	/	Seal_Cookie	Digital_Signature	FALSE	12/31/99

Cookie_Issuer Signs on the Cookies

* Hash of the passwords is an alternative as the content of the Pswd_Cookie.

RBAC on the Web by Secure Cookies



Cookie_Issuer (Role Server)

Authentication

User-Role Assignment

Set Secure Cookies

Access

Get Cookies

Client (Browser)

Send Cookies & Requests

Response

Web Server 1

Web Server n

Get Cookies

Authentication

Check Integrity

Retrieve Roles

RBAC

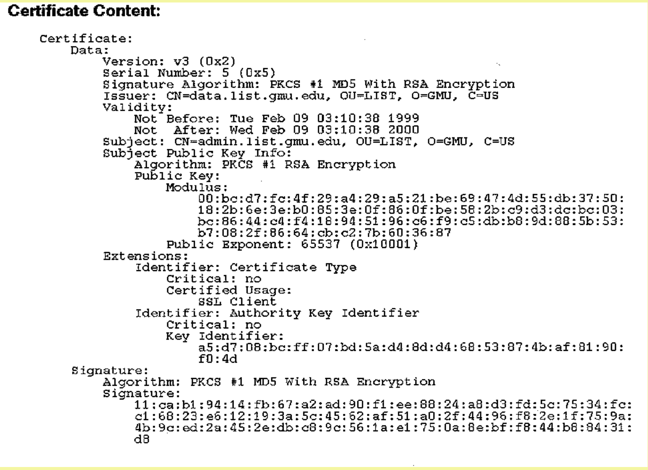
Assigning Roles & Creating Secure Cookies

Verifying Secure Cookies & RBAC

X.509 Certificate

- Digitally signed by a certificate authority
  - to confirm the information in the certificate belongs to the holder of the corresponding private key
- Contents
  - version, serial number, subject, validity period, issuer, optional fields (v2)
  - subject’s public key and algorithm info.
  - extension fields (v3)
  - digital signature of CA
- Binding users to keys
- Certificate Revocation List (CRL)

X.509 Certificate



Certificate Content:

```
    Certificate:
        Data:
            Version: v3 (0x2)
            Serial Number: 5 (0x5)
            Signature Algorithm: PKCS #1 MD5 With RSA Encryption
            Issuer: CN=data.list.gmu.edu, OU=LIST, O=GMU, C=US
            Validity:
                Not Before: Tue Feb 09 03:10:38 1999
                Not  After: Wed Feb 09 03:10:38 2000
            Subject: CN=admin.list.gmu.edu, OU=LIST, O=GMU, C=US
            Subject Public Key Info:
                Algorithm: PKCS #1 RSA Encryption
                Public Key:
                    Modulus:
                        00:bc:d7:fc:4f:29:a4:29:a5:21:be:69:47:4d:55:db:37:50:
                        18:2b:6e:3e:b0:85:3e:0f:86:0f:be:58:2b:c9:d3:dc:bc:03:
                        bc:86:44:c4:f4:18:94:51:96:c6:f9:c5:db:b8:9d:88:5b:53:
                        b7:08:2f:86:64:cb:c2:7b:60:36:87
                    Public Exponent: 65537 (0x10001)
            Extensions:
                Identifier: Certificate Type
                    Critical: no
                    Certified Usage:
                        SSL Client
                Identifier: Authority Key Identifier
                    Critical: no
                    Key Identifier:
                        a5:d7:08:bc:ff:07:bd:5a:d4:8d:d4:68:53:87:4b:af:81:90:
                        f0:4d
        Signature:
            Algorithm: PKCS #1 MD5 With RSA Encryption
            Signature:
                11:ca:b1:94:14:fb:67:a2:ad:90:f1:ee:88:24:a8:d3:fd:5c:75:34:fc:
                c1:68:23:e6:12:19:3a:5c:45:62:af:51:a0:2f:44:96:f8:2e:1f:75:9a:
                4b:9c:ed:2a:45:2e:db:c8:9c:56:1a:e1:75:0a:8e:bf:f8:44:b8:84:31:
                d8
```

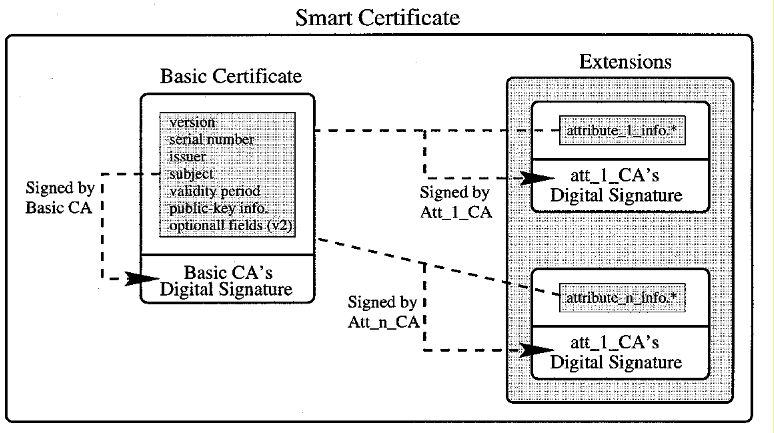
Smart Certificates

- **Short-Lived Lifetime**
  - **More secure**
    - **typical validity period for X.509 is months (years)**
    - **users may leave copies of the corresponding keys behind**
    - **the longer-lived certificates have a higher probability of being attacked**
  - **No Certificate Revocation List (CRL)**
    - **simple and less expensive PKI**

Smart Certificates

- **Containing Attributes Securely**
  - **Web servers can use secure attributes for their purposes**
  - **Each authority has independent control on the corresponding information**
    - **basic certificate (containing identity information)**
    - **each attribute can be added, changed, revoked, or re-issued by the appropriate authority**
      - e.g., role, credit card number, clearance, etc.
  - **Short-lived certificate can remove CRLs**

Separate CAs in a Certificate

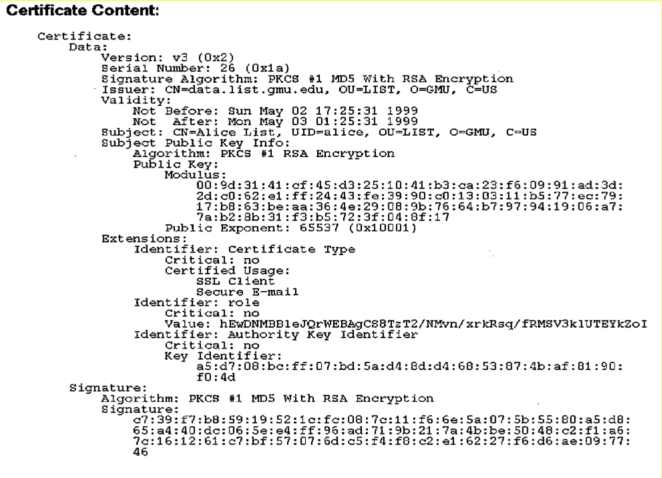


* attribute info.: attributes, attribute issuer, validity period of attributes, etc.

Smart Certificates

- **Postdated Certificates**
  - **The certificate becomes valid at some time in the future**
  - **possible to make a smart certificate valid for a set of duration**
  - **supports convenience**
- **Confidentiality**
  - **Sensitive information can be**
    - **encrypted in smart certificates**
      - e.g. passwords, credit card numbers, etc.

A Smart Certificate



Certificate Content:

```
Certificate:
    Data:
        Version: v3 (0x2)
        Serial Number: 26 (0x1a)
        Signature Algorithm: PKCS #1 MD5 With RSA Encryption
        Issuer: CN=data.list.gmu.edu, OU=LIST, O=GMU, C=US
        Validity:
            Not Before: Sun May 02 17:25:31 1999
            Not  After: Mon May 03 01:25:31 1999
        Subject: CN=Alice List, UID=alice, OU=LIST, O=GMU, C=US
        Subject Public Key Info:
            Algorithm: PKCS #1 RSA Encryption
            Public Key:
                Modulus:
                    00:9d:31:41:cf:45:d3:25:10:41:b3:ca:23:f6:09:91:ad:3d:
                    2d:c0:62:e1:ff:24:43:fe:39:90:c0:13:03:11:b5:77:ec:79:
                    17:b8:63:be:aa:36:4e:29:08:9b:76:64:b7:97:94:19:06:a7:
                    7a:b2:8b:31:f3:b5:72:3f:04:8f:17
                Public Exponent: 65537 (0x10001)
        Extensions:
            Identifier: Certificate Type
                Critical: no
                Certified Usage:
                    SSL Client
                    Secure E-mail
            Identifier: role
                Critical: no
                Value: hEwDNMBBleJQrWEBAgCS8TzT2/NMvn/xrkRsq/fRMSV3klUTEYkZoI
            Identifier: Authority Key Identifier
                Critical: no
                Key Identifier:
                    a5:d7:08:bc:ff:07:bd:5a:d4:8d:d4:68:53:87:4b:af:81:90:
                    f0:4d
    Signature:
        Algorithm: PKCS #1 MD5 With RSA Encryption
        Signature:
            c7:39:f7:b8:59:19:52:1c:fc:08:7c:11:f6:6e:5a:07:5b:55:80:a5:d8:
            65:a4:40:dc:06:5e:e4:ff:96:ad:71:9b:21:7a:4b:be:50:48:c2:f1:a6:
            7c:16:12:61:c7:bf:57:07:6d:c5:f4:f8:c2:e1:62:27:f6:d6:ae:09:77:
            46
```

Applications of Smart Certificates

- On-Duty Control
- Compatible with X.509
- User Authentication
- Electronic Transaction
- Eliminating Single-Point Failure
- Pay-per-Access
- Attribute-based Access Control

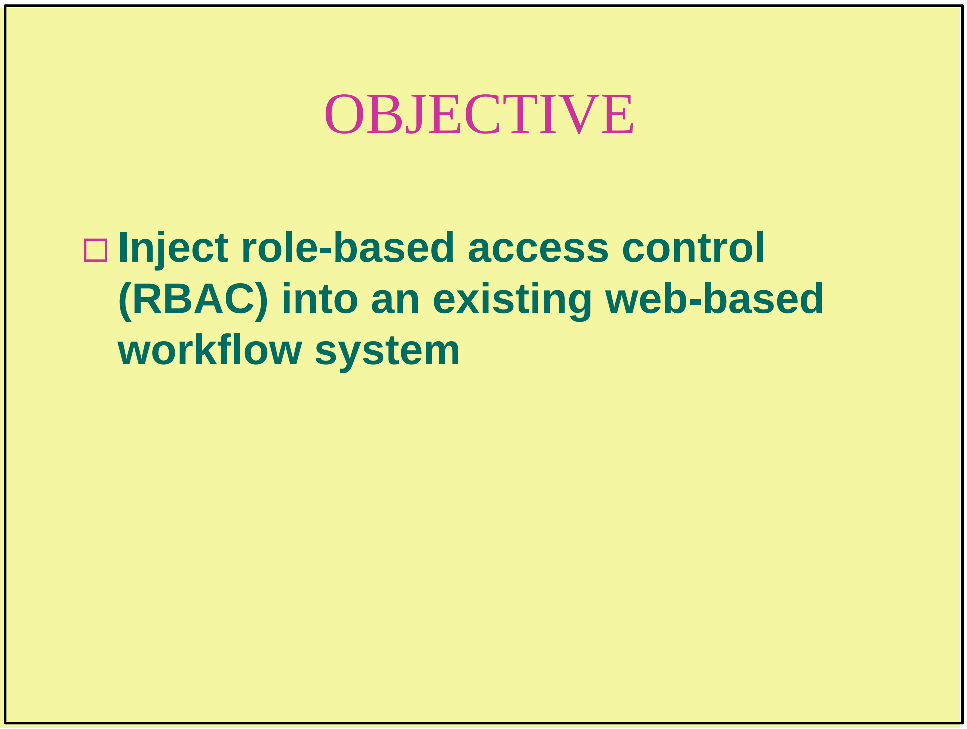
Injecting RBAC to Secure a Web-based Workflow System

Gail-Joon Ahn and Ravi Sandhu
George Mason University

Myong Kang and Joon Park
Naval Research Laboratory

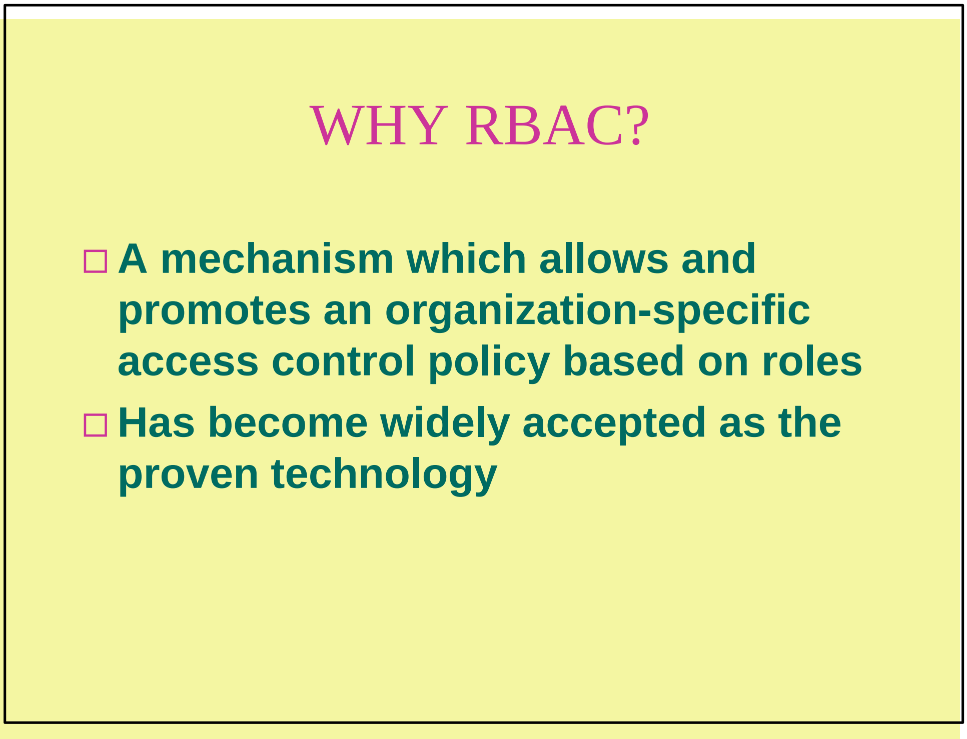
WORKFLOW MANAGEMENT SYSTEMS

- □ Control and coordinate processes that may be processed by different processing entities
- □ Received much attention
- □ Marriage with Web technology
- □ Minimal security services



OBJECTIVE

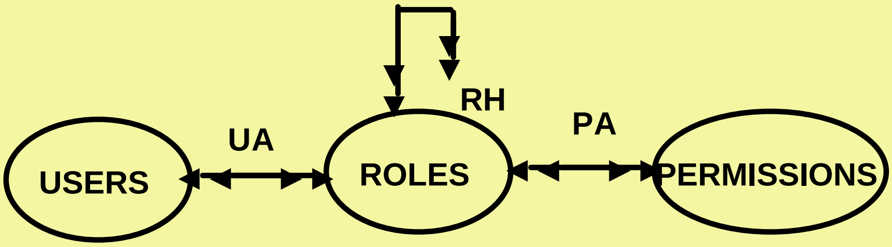
- □ Inject role-based access control (RBAC) into an existing web-based workflow system



WHY RBAC?

- □ A mechanism which allows and promotes an organization-specific access control policy based on roles
- □ Has become widely accepted as the proven technology

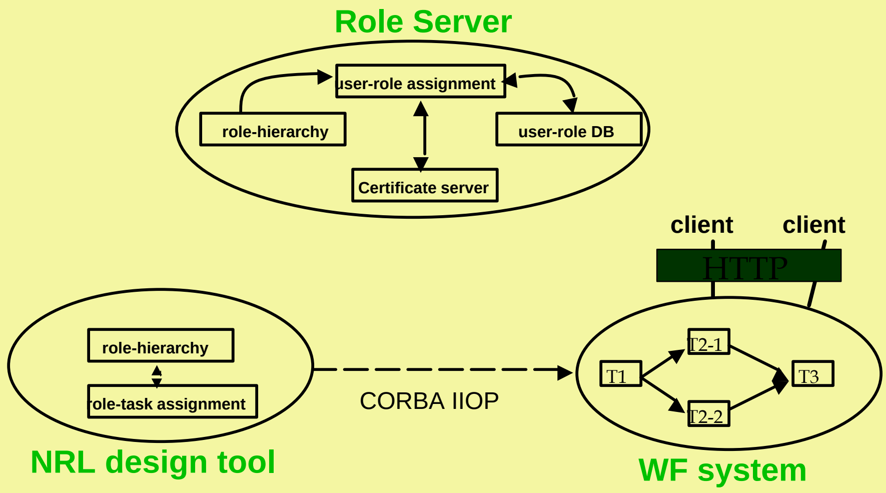
SIMPLIFIED RBAC MODEL



ROLE-BASED SECURE WORKFLOW SYSTEM

- □ Workflow Design Tool
- □ Workflow (WF) System
- □ Role Server

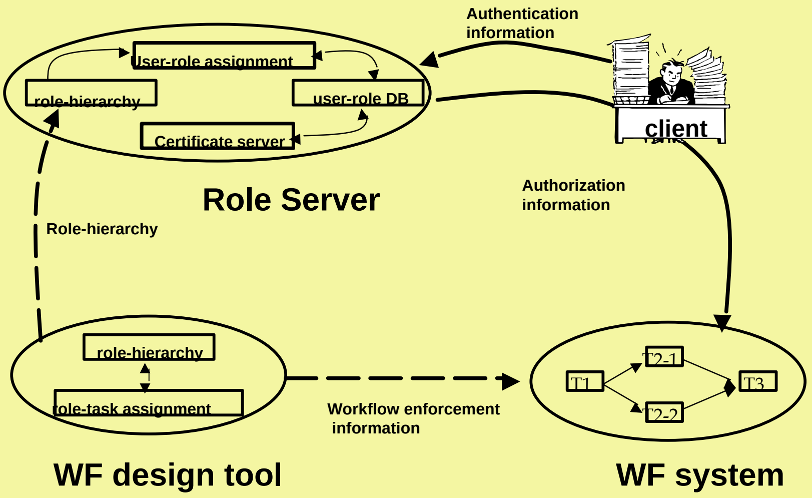
BASIC COMPONENTS



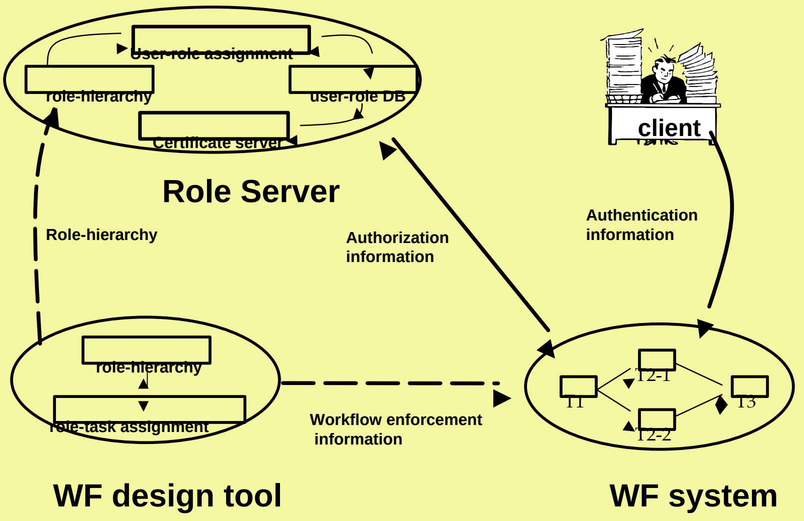
ARCHITECTURES

- □ USER-PULL STYLE
- □ SERVER-PULL STYLE

USER-PULL STYLE



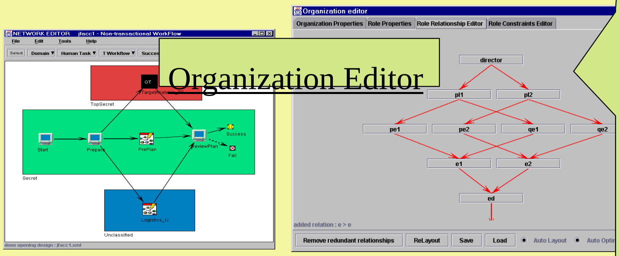
SERVER-PULL STYLE



NRL (Naval Research Lab.) DESIGN TOOL

- □ design workflow model
- □ create role and role hierarchies
- □ assign role to task
- □ exporting role hierarchies to role server

NRL DESIGN TOOL (Cont'd)



Organization Editor

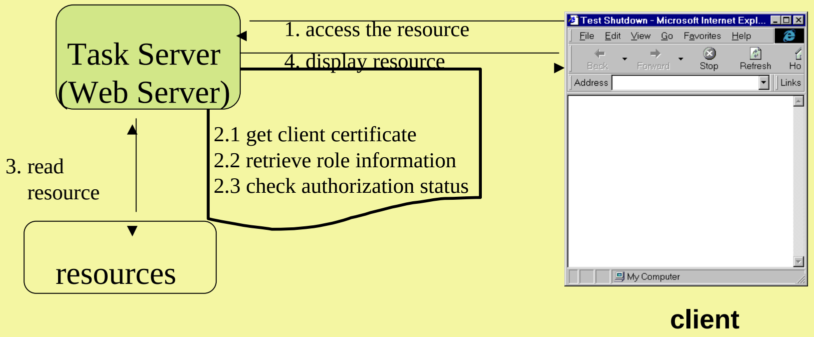
Platform: Windows NT, JDK1.2

```
<?xml version="1.0"?>
<!--URA     Revision: 1
   Mon Dec 07 15:59:28 EDT 1999-->
<!DOCTYPE Organization SYSTEM
   "../dtd/Organization.dtd">
<Organization id="Organization_URA">
<Name>URA</Name>
<Description></Description>
<Role id="director">
<Name>director</Name>
<Description></Description>
<Privileges></Privileges>
<LowRoleList>
 <RoleReference>
                        <Link idref = "pl1"/>
 </RoleReference>
 <RoleReference>
                        <Link idref = "pl2"/>
 </RoleReference>
</LowRoleList>
<HighRoleList>
</HighRoleList>
</Role>
<RoleReference>
```

WORKFLOW SYSTEM

- □ each task server is web server
- □ user should present client authentication certificate
- □ user′s privilege is authorized by content of certificate (specially client′s role information)

ROLE AUTHORIZATION ON WORKFLOW SYSTEM



Task Server (Web Server)

1. access the resource

4. display resource

2.1 get client certificate
2.2 retrieve role information
2.3 check authorization status

3. read resource

resources

client

Test Shutdown - Microsoft Internet Expl...
File Edit View Go Favorites Help
Back
Forward
Stop
Refresh
Ho
Address
Links
My Computer

ROLE SERVER

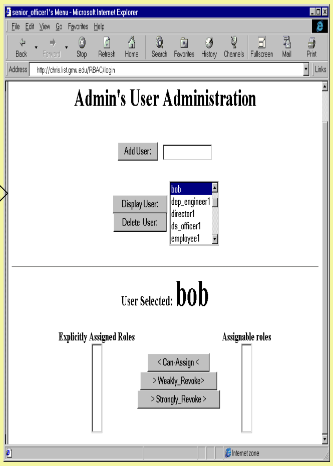
- □ User Role Assignment
- □ Certificate Server

USER ROLE ASSIGNMENT

- □ maintain role hierarchies and user database
- □ assign users to roles
- □ generate user-role database

USER ROLE ASSIGNMENT (Cont’d)

Converted Role Hierarchy File



senior_officer1's Menu - Microsoft Internet Explorer

Address http://chris.list.gmu.edu/RBAC/login

Admin's User Administration

User Selected: bob

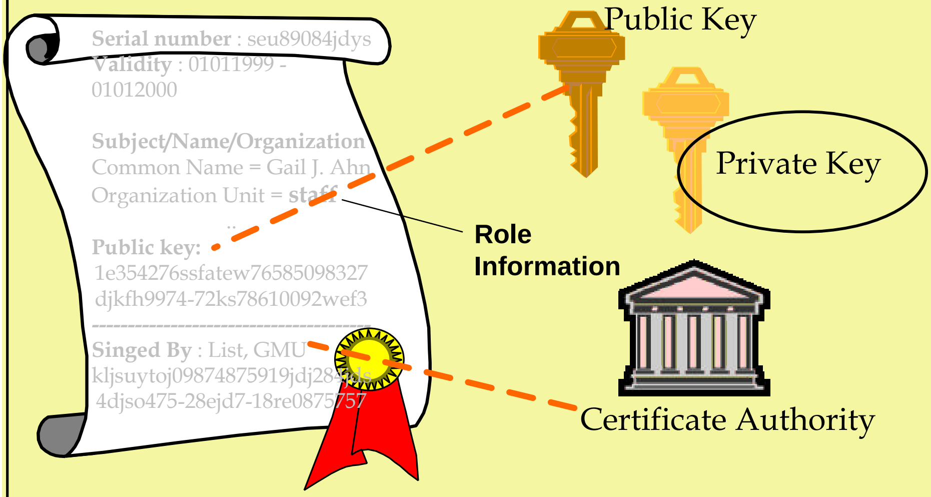
User-Role DB

Alice : director
Bob : engineer
Chris : pro-leader

CERTIFICATE SERVER

- □ authenticate client
- □ retrieve client’s role information from user-role database
- □ issue certificate with client’s role information

X.509 CERTIFICATE



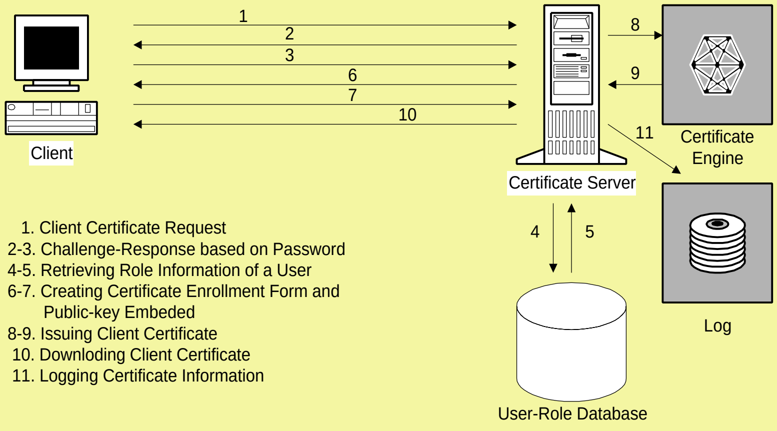
Serial number : seu89084jdys
Validity : 01011999 - 01012000

Subject/Name/Organization
Common Name = Gail J. Ahn
Organization Unit = **staff**
..

Public key:
1e354276ssfatew76585098327
djkfh9974-72ks78610092wef3

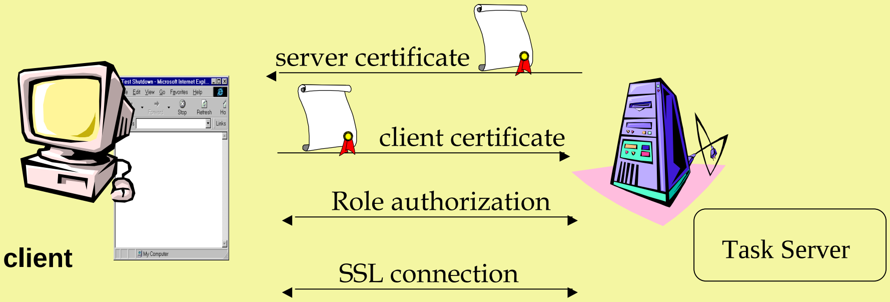
Singed By : List, GMU
kljsuytoj09874875919jdj284jds
4djso475-28ejd7-18re0875757

CERTIFICATE ISSUE



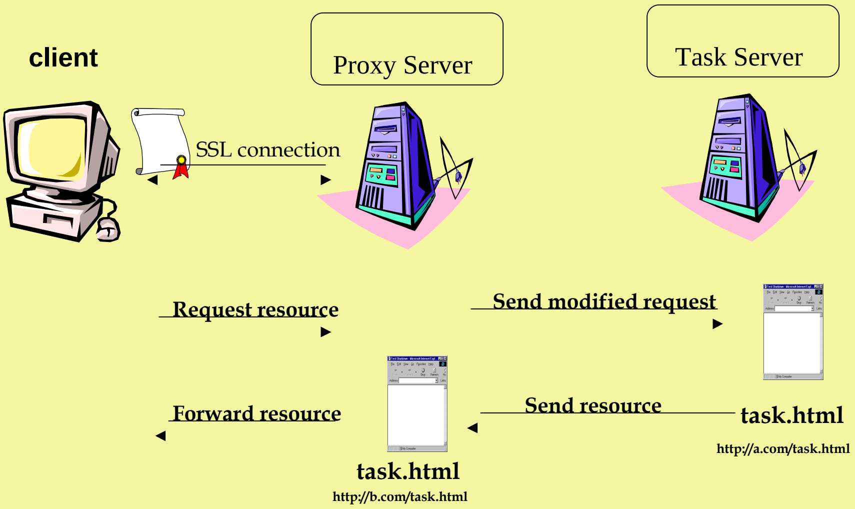
1. Client Certificate Request
2-3. Challenge-Response based on Password
4-5. Retrieving Role Information of a User
6-7. Creating Certificate Enrollment Form and
Public-key Embeded
8-9. Issuing Client Certificate
10. Downloding Client Certificate
11. Logging Certificate Information

CERTIFICATE AUTHORIZATION OVER SSL

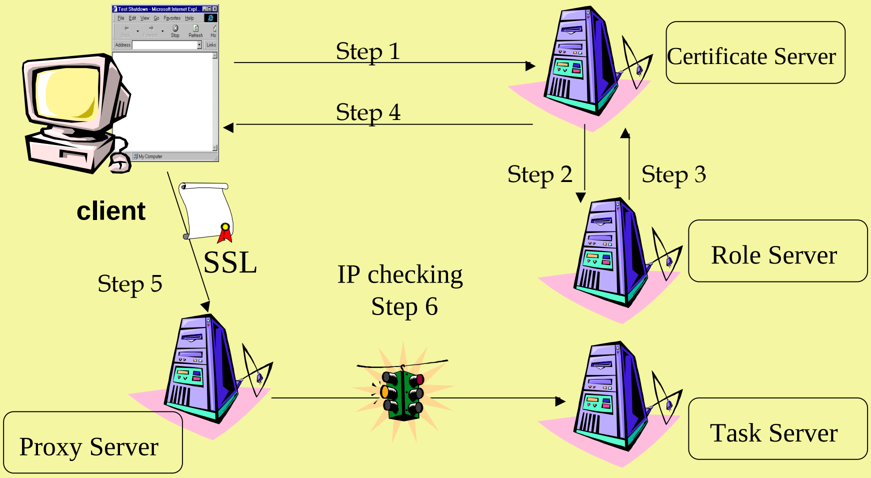


REVERSE PROXYING

(MINIMAL CHANGES IN SERVER SIDE)



FINAL SCENARIO



client

Step 1

Step 4

Certificate Server

Step 2

Step 3

Role Server

SSL

Step 5

IP checking
Step 6

Proxy Server

Task Server