



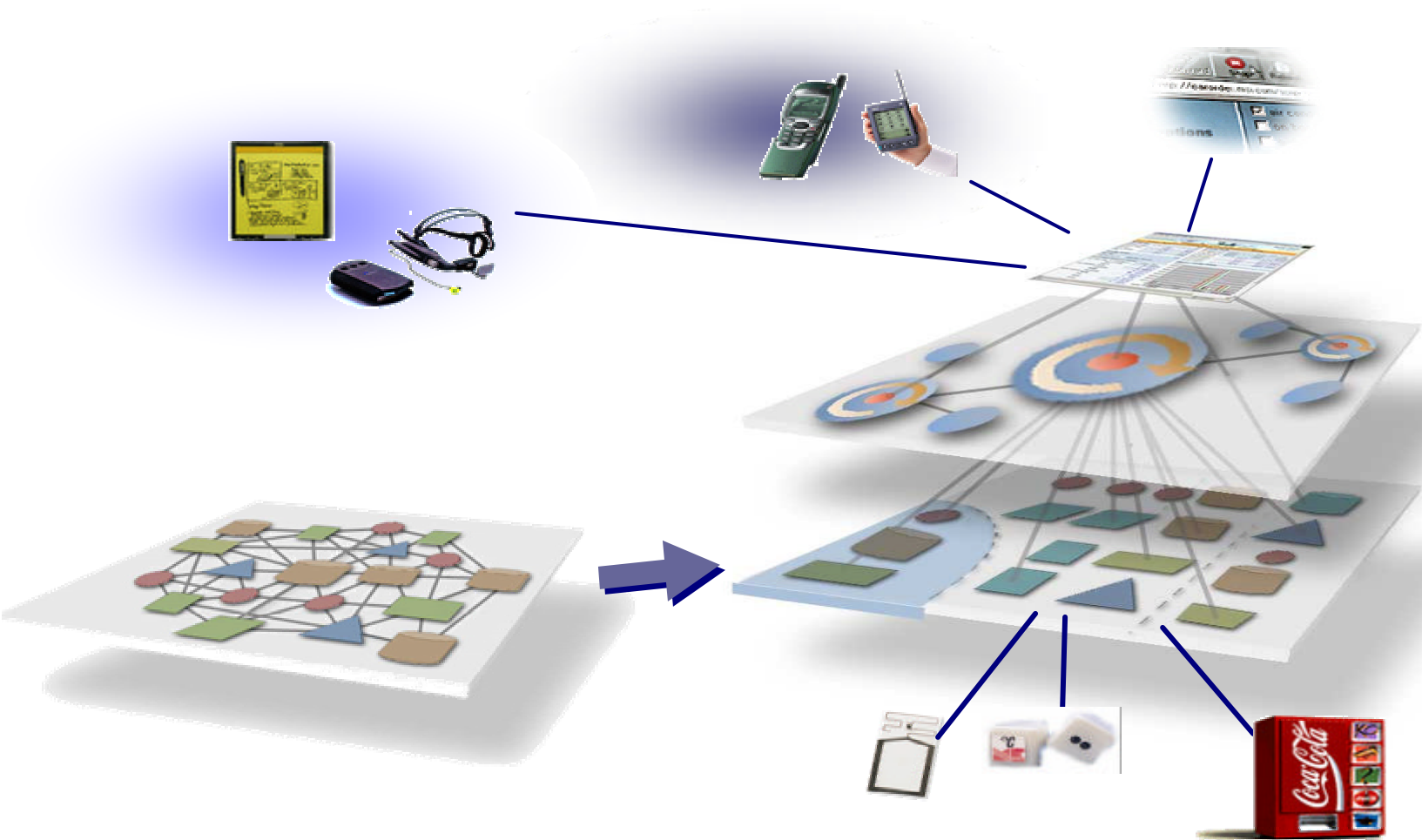
# Separating Business and Security Logic

by Emin Islam Tatli

# Outline

- Problem statement
- Separation of Business and Security Logic
- Policies in WiTness
- Framework and Architecture
- Security Context
- Demo Implementation
- Conclusion

# Mobile Environment

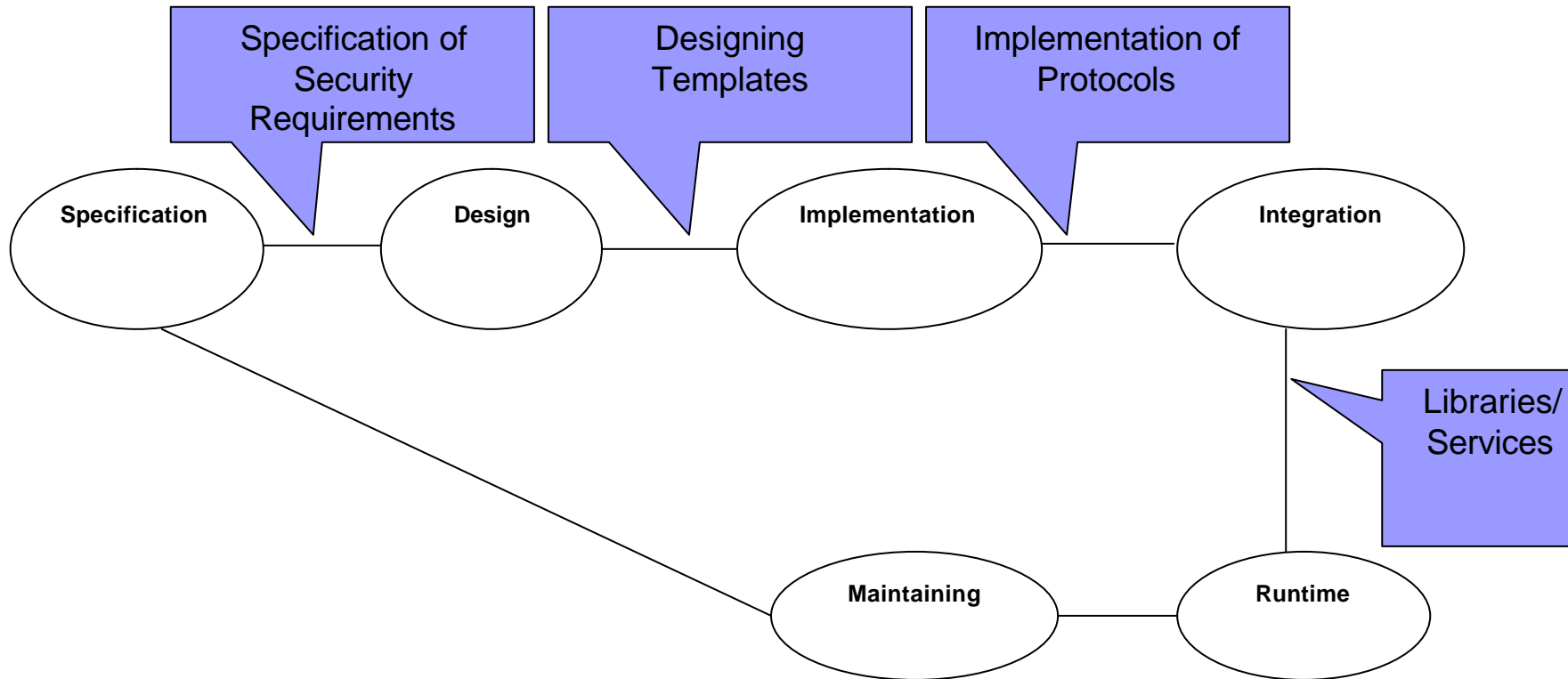


# Problem Statements

- Heterogeneous Devices
- Heterogeneous Environment
- Homogenous Security Requirements
- System Design Challenge

# Application Layer Security

- Enforcement of Security at application layer, beyond the network and transport layer



- Integration of application and security logic

# Separation of Application and Security Logic

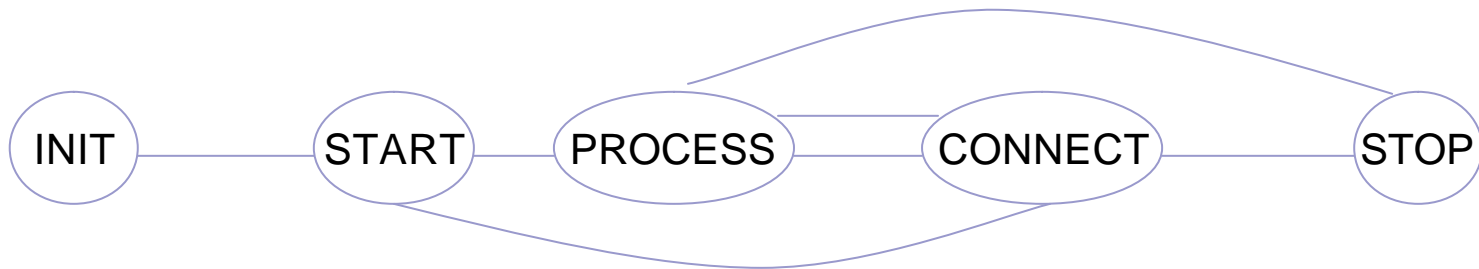
## ■ Why?

- Application programmers are not security experts
- Integration of security within applications more difficult, more complex
- Dynamic security management is desirable
- Capturing Security Expert knowledge required

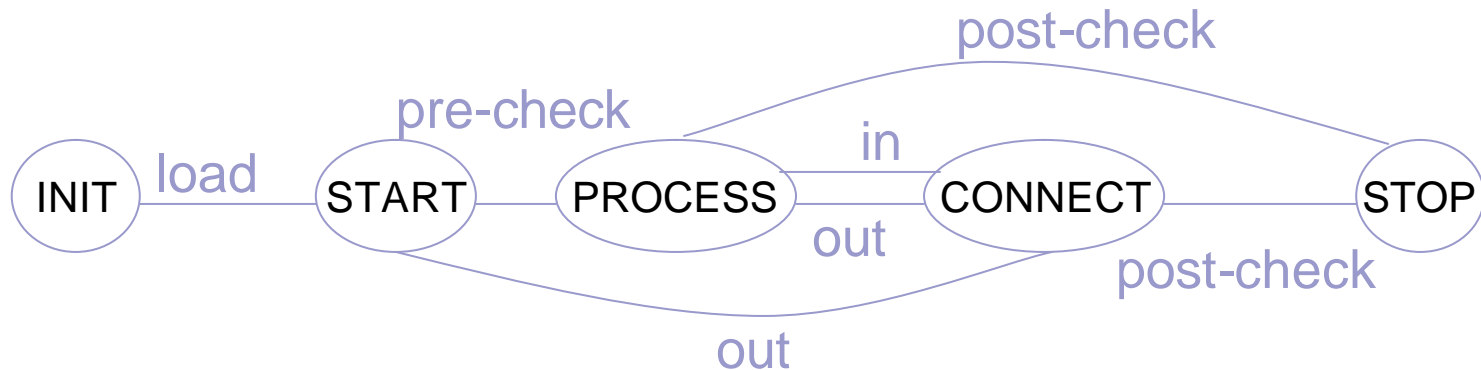
# Separation of Application and Security Logic (cont.)

## ■ How?

**Application Logic** = what the application does at particular states



**Security Logic** = what are the constraints when transitioning



# Policy-Based Management

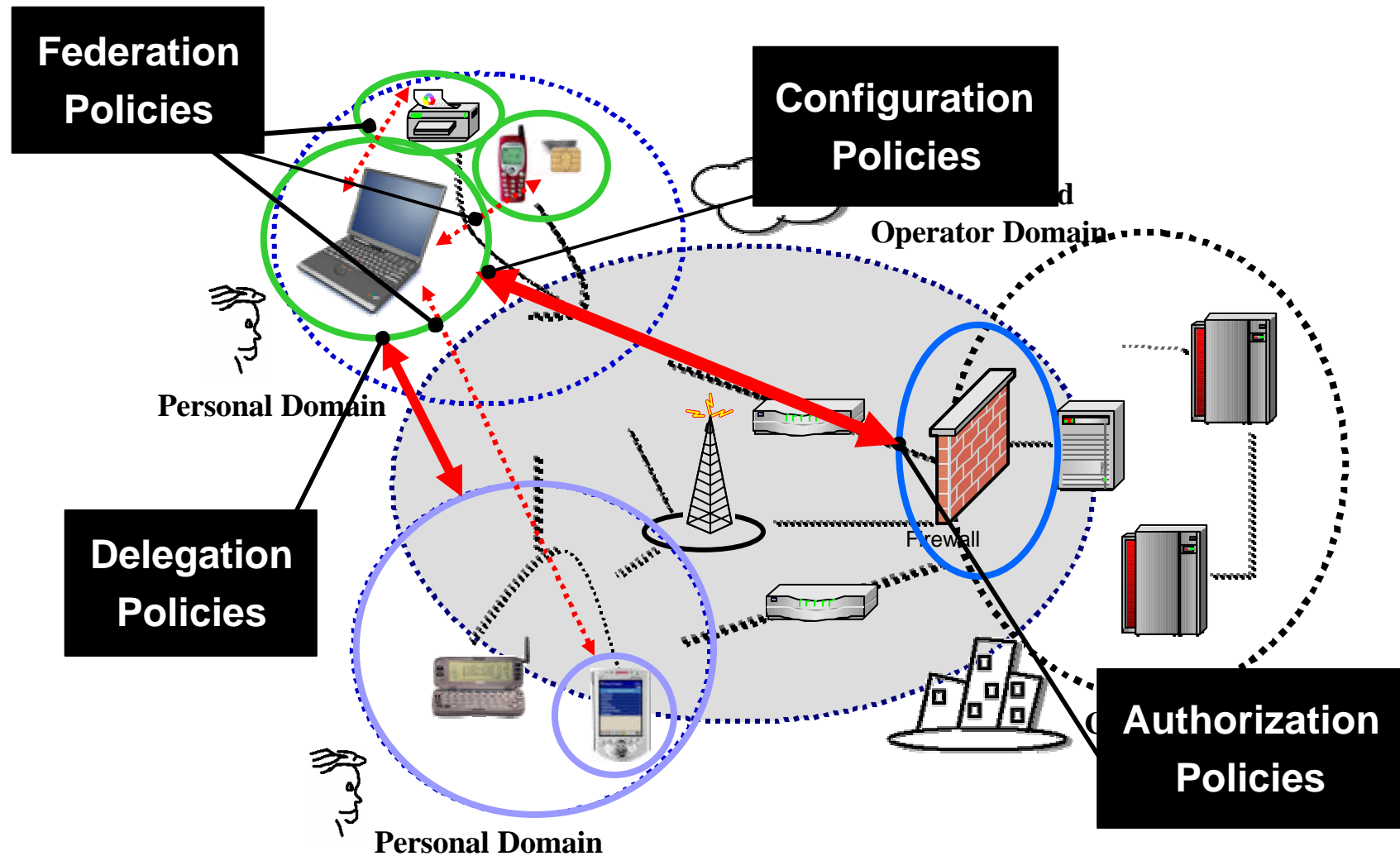
## ■ Policy in WiTness

- *“a means of moving from narrative declaration of security requirements, to one that is technically enforceable”*
- constraints&assertions: rules defining security logic and tiding policies to application transitions

## ■ Advantages of Policy-Based Systems

- Dynamic Management is easy
- High-level Declarative Directives
- Use of Expert Knowledge

# Security Policies in WiTness



# Constraints and Assertions

|                     |         |                    |                     |         |
|---------------------|---------|--------------------|---------------------|---------|
| PROFILE.<br>SERVICE | UTILITY | UTILITY_<br>FILTER | RESOURCE_<br>FILTER | BINDING |
|---------------------|---------|--------------------|---------------------|---------|

- PROFILE.SERVICE: (e.g. SEC.Confidentiality)
- UTILITY: (e.g. Decryption)
- UTILITY\_FILTER : (e.g. instance=des, mode=out)
- RESOURCE\_FILTER: (e.g. KeyMinLength=128)
- BINDING:
  - *Constraint 1 > Constraint 2*
  - *Constraint 1 ; Constraint 2*

# Samples: Constraint/ Assertion

|                            |                         |                                  |                     |   |
|----------------------------|-------------------------|----------------------------------|---------------------|---|
| <b>SEC. Authentication</b> | Authentication Provider | instance=login,<br>mode=precheck | PasswordMinLength=8 | > |
|----------------------------|-------------------------|----------------------------------|---------------------|---|

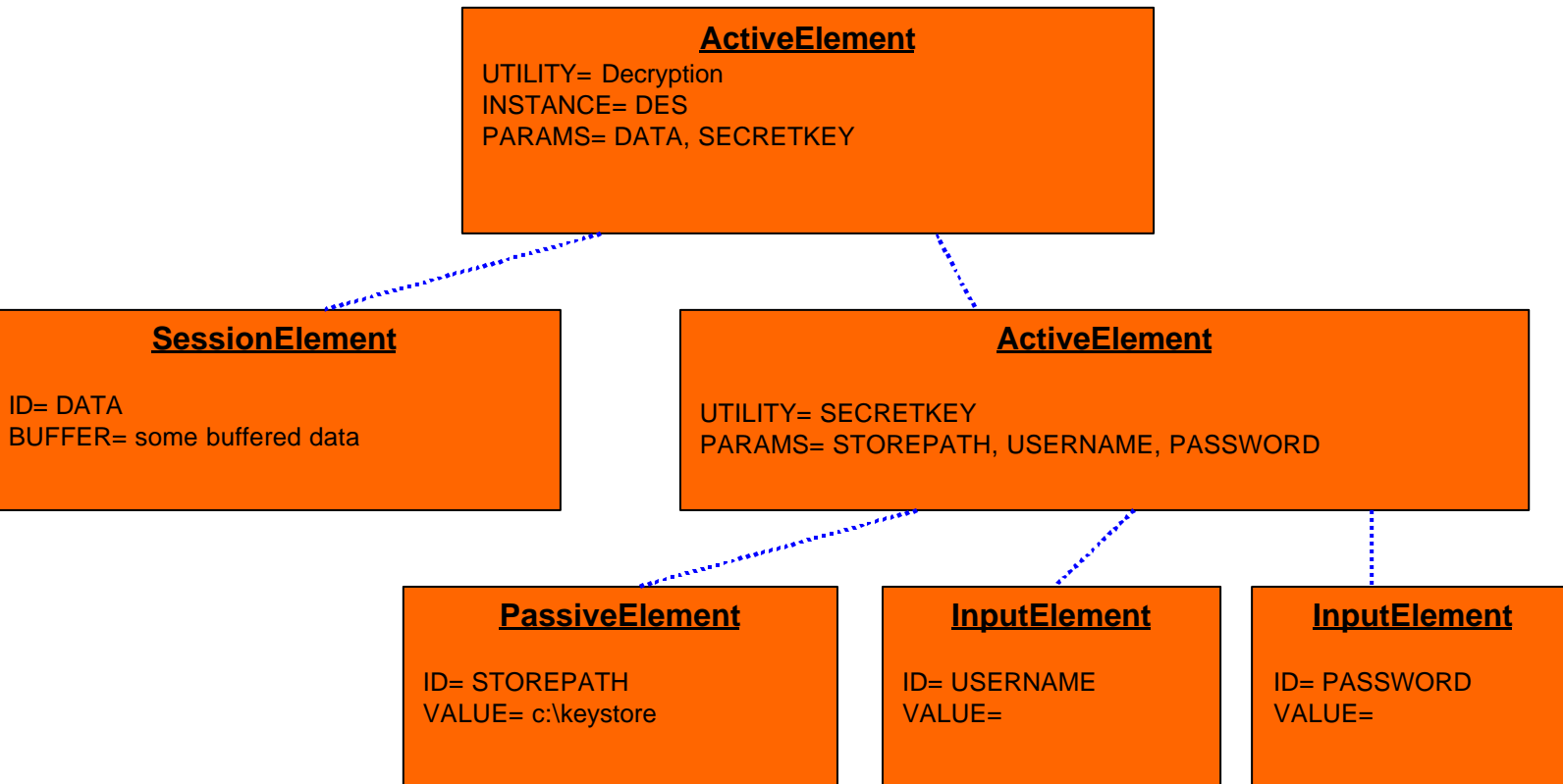
|                      |             |                           |                  |   |
|----------------------|-------------|---------------------------|------------------|---|
| <b>SEC.Integrity</b> | WIMProvider | instance=md5,<br>mode=out | KeyMinLength=128 | ; |
|----------------------|-------------|---------------------------|------------------|---|

|                             |                    |                              |                  |   |
|-----------------------------|--------------------|------------------------------|------------------|---|
| <b>SEC. Confidentiality</b> | ImplCryptoProvider | instance=AES,<br>mode=in-out | KeyMinLength=128 | ; |
|-----------------------------|--------------------|------------------------------|------------------|---|

|                  |                     |                                       |                      |   |
|------------------|---------------------|---------------------------------------|----------------------|---|
| <b>SEC.Trust</b> | CertificateProvider | instance=TrustLevel,<br>mode=precheck | BasicTrustLevel=HIGH | ; |
|------------------|---------------------|---------------------------------------|----------------------|---|

# Mapping Constraint to Security Context

- “SEC.Confidentiality, SEC.Decryption, [mode=in, instance=DES], []”,
- Security Context Tree

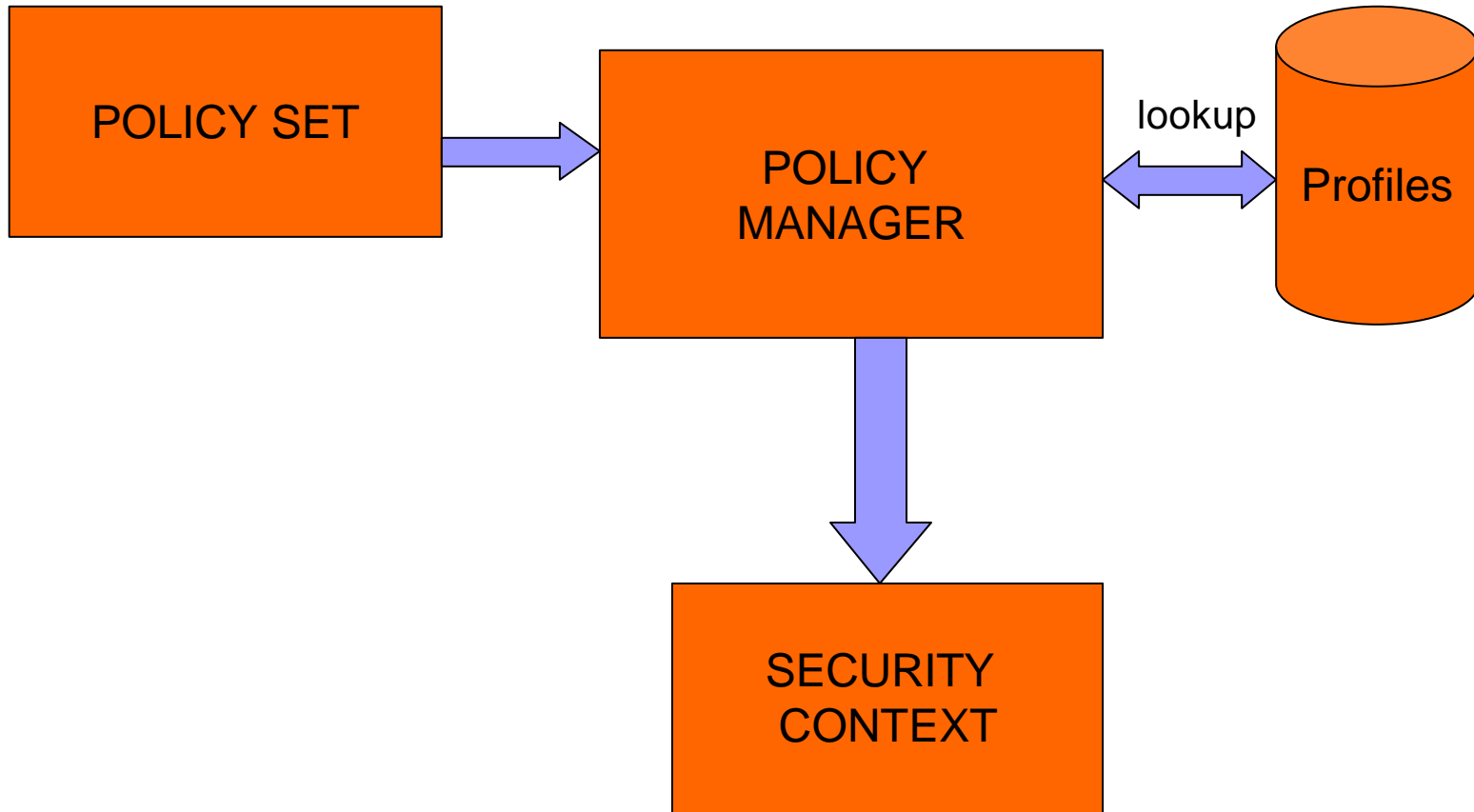


# Security Context

- collection of static information and and active security services representing security logic (*an ordered tree structure*)
  - Keystore Path: static information
  - Encryption: active service
  - Password Prompt: active service
- Security Context can include four types of Security Context Element:
  - PassiveElement
  - InputElement
  - SessionElement
  - ActiveElement

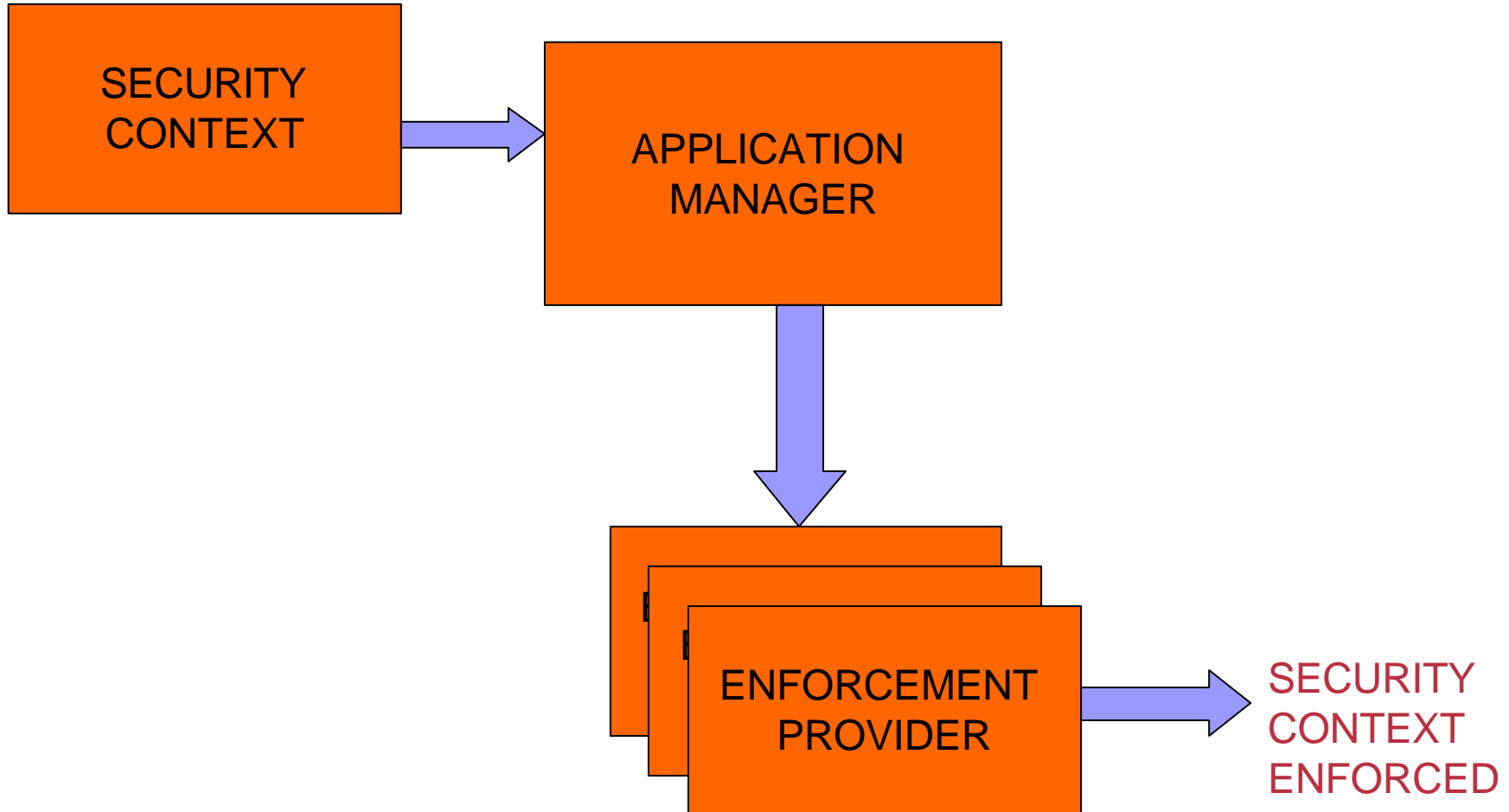
# Framework and Architecture

## ■ CREATING SECURITY CONTEXT



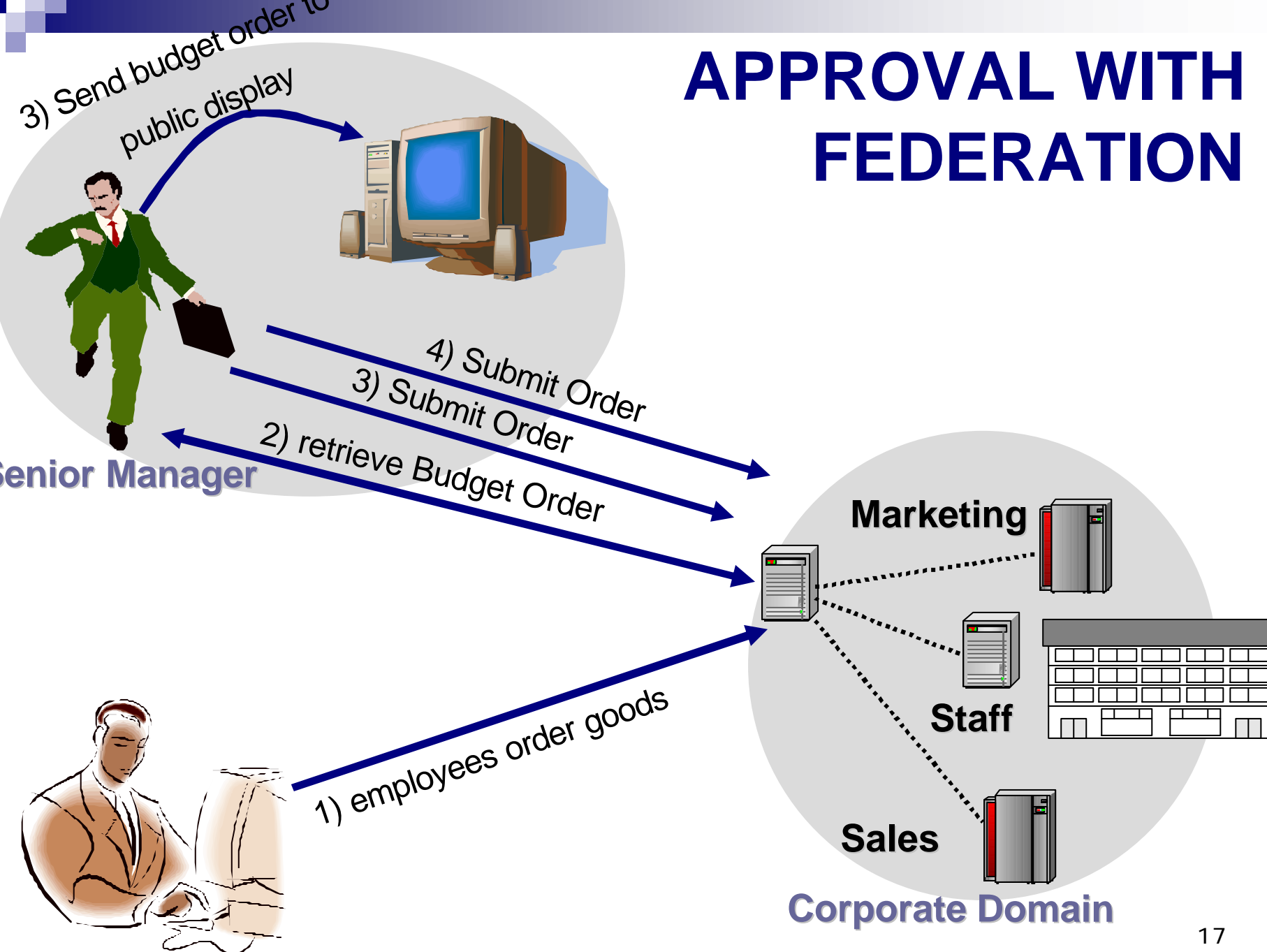
# Framework and Architecture

## ■ ENFORCING SECURITY CONTEXT

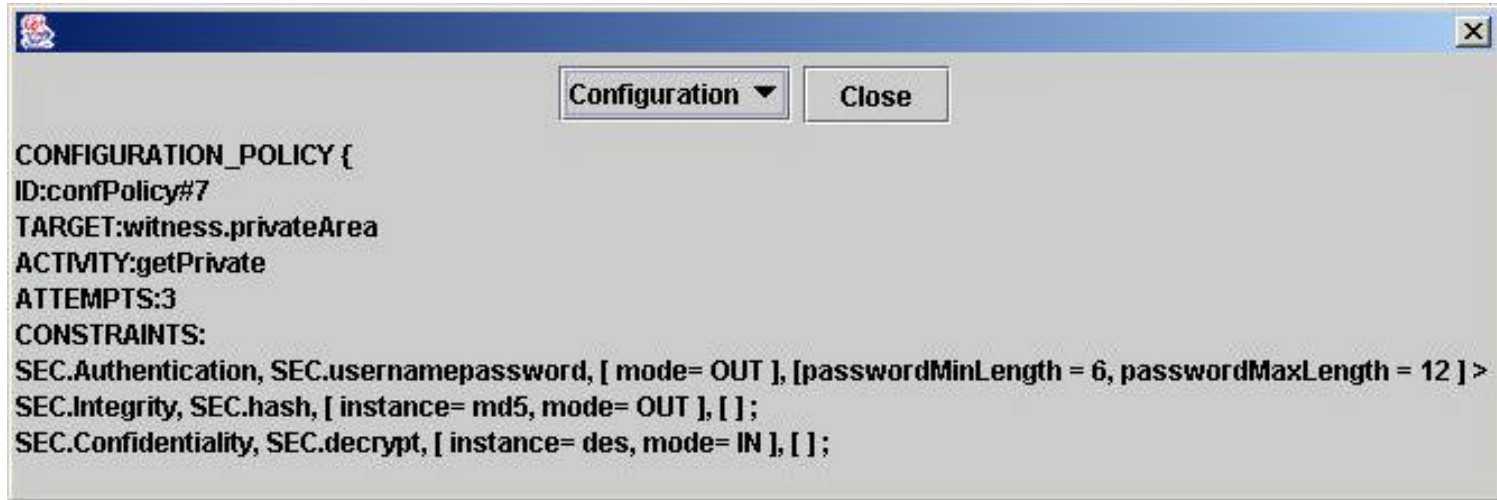


**DEMO**

# APPROVAL WITH FEDERATION

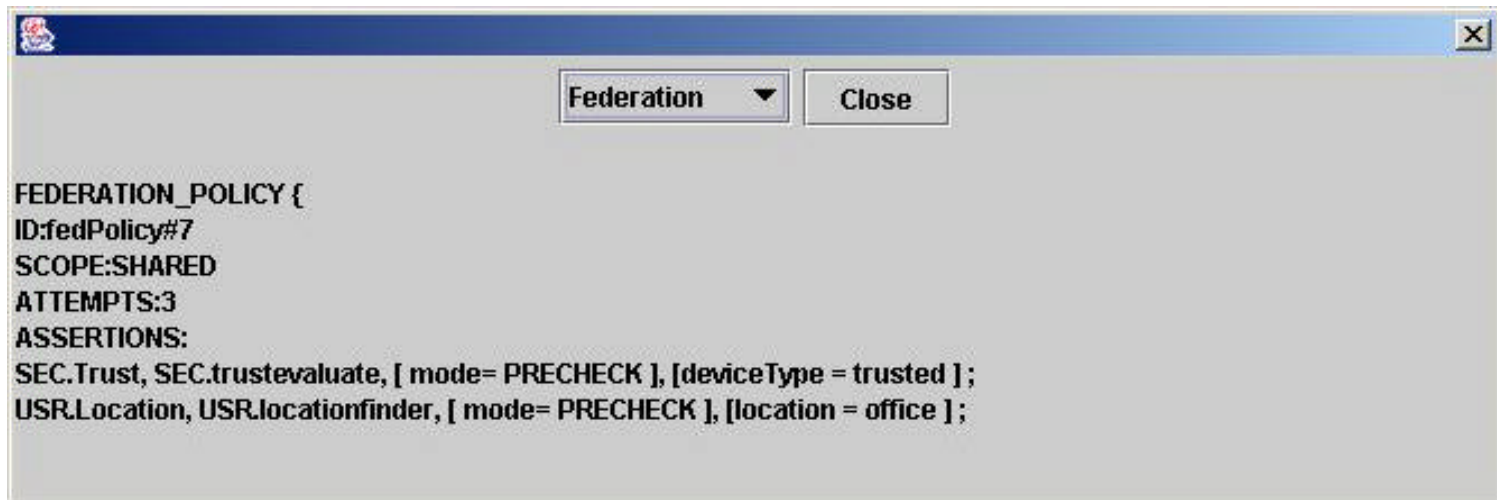


# SAMPLE POLICIES



A screenshot of a software window titled "CONFIGURATION\_POLICY". The window has a blue title bar with a close button (X) in the top right corner. Below the title bar, there are two buttons: "Configuration" with a dropdown arrow and "Close". The main content area displays the following text:

```
CONFIGURATION_POLICY {  
ID:confPolicy#7  
TARGET:witness.privateArea  
ACTIVITY:getPrivate  
ATTEMPTS:3  
CONSTRAINTS:  
SEC.Authentication, SEC.usernamepassword, [ mode= OUT ], [passwordMinLength = 6, passwordMaxLength = 12 ] >  
SEC.Integrity, SEC.hash, [ instance= md5, mode= OUT ], [ ] ;  
SEC.Confidentiality, SEC.decrypt, [ instance= des, mode= IN ], [ ] ;
```



A screenshot of a software window titled "FEDERATION\_POLICY". The window has a blue title bar with a close button (X) in the top right corner. Below the title bar, there are two buttons: "Federation" with a dropdown arrow and "Close". The main content area displays the following text:

```
FEDERATION_POLICY {  
ID:fedPolicy#7  
SCOPE:SHARED  
ATTEMPTS:3  
ASSERTIONS:  
SEC.Trust, SEC.trustevalue, [ mode= PRECHECK ], [deviceType = trusted ] ;  
USR.Location, USR.locationfinder, [ mode= PRECHECK ], [location = office ] ;
```

**DEMO**

# CODE EXAMPLES

## Partlet (Client)

```
String url= new String("http://localhost:8080/servlets-examples/servlet/BudgetServlet");  
Object [] params= {url};  
Class[] paramTypes= {String.class};
```

```
PartletDescription pd= new PartletDescription("targetPartlet.prop");  
Partlet targetPartlet= (RemotePartlet)am.findPartlet(pd).firstElement();
```

```
Budget budget= targetPartlet.makeCall("getBudget",params,paramTypes);
```

## Policy Execution

```
byte[] hashedData= (byte[])  
    securityContext.getConfSecurityContext().execute(WSecurityContext.OUT).get  
  
byte[] decryptedData =(byte[])  
    securityContext.getConfSecurityContext().execute(WSecurityContext.IN).getResult()
```

Authentication  
Enforced

Decryption  
Enforced

# Further Work

- Security Context Negotiation (*interacting with FederationManager*)
- Integration of Policy Framework with WiTness Pilot Mobile Business Application

# Conclusion

- Heterogeneous Mobile Environment
- Application Layer Security
- Separation of Business and Security Logic based on Policies
- Security Context



# Separating Business and Security Logic

by Emin Islam Tatli