9271

Biometric Authentication for J2EE Applications

Ramesh Nagappan
Staff Engineer
Sun Microsystems

Reid Williams
Member of Technical Staff
Sun Microsystems

Goal

Learn the importance of

Biometric authentication and

How to implement them in J2EE applications.

Agenda

Understanding Biometric Authentication

- Importance of Biometrics
- Biometric Identification Process
- The accuracy of Biometrics
- Enabling technologies
- Logical Architecture

Biometrics in J2EE Applications

- J2EE Tools of the Trade
- Implementing a JAAS BiometricLoginModule
- Implementation Strategies

Biometric Single Sign-On (SSO)

- Biometric SSO to a J2EE based Web Portal
- Sun Java System Access Manager w/ BiObex Demo

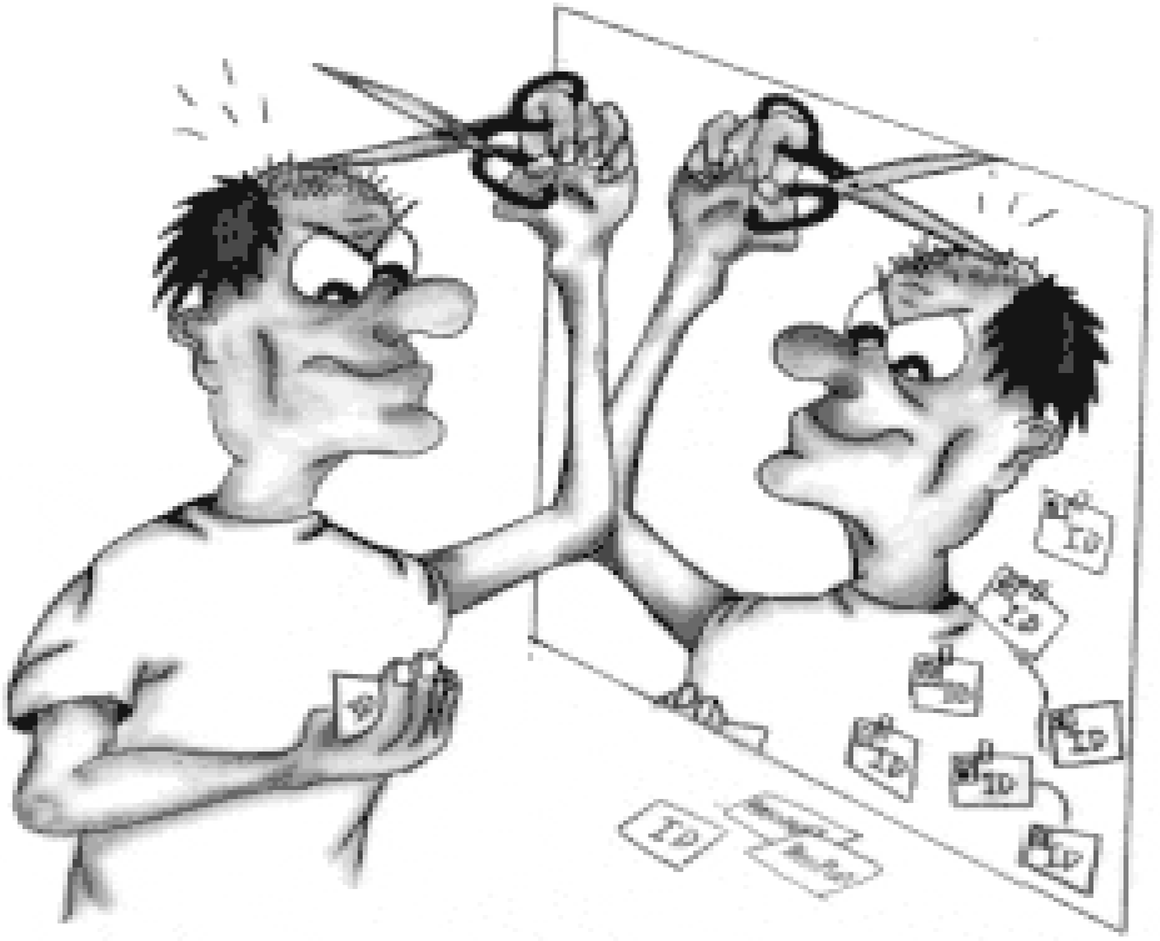
Q & A

On the Internet, Nobody knows you are a dog !

Internet is a faceless Channel...Unless you have a mechanism to physically verify a person....you would not know who is really accessing your application.

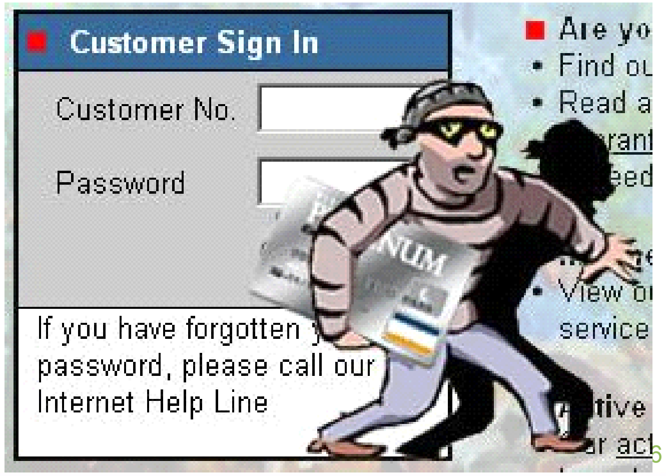
Cartoon by Peter Steiner. The New Yorker, July 5, 1993 issue (Vol.69 (LXIX) no. 20) page 61

How do I know ...it's you ?



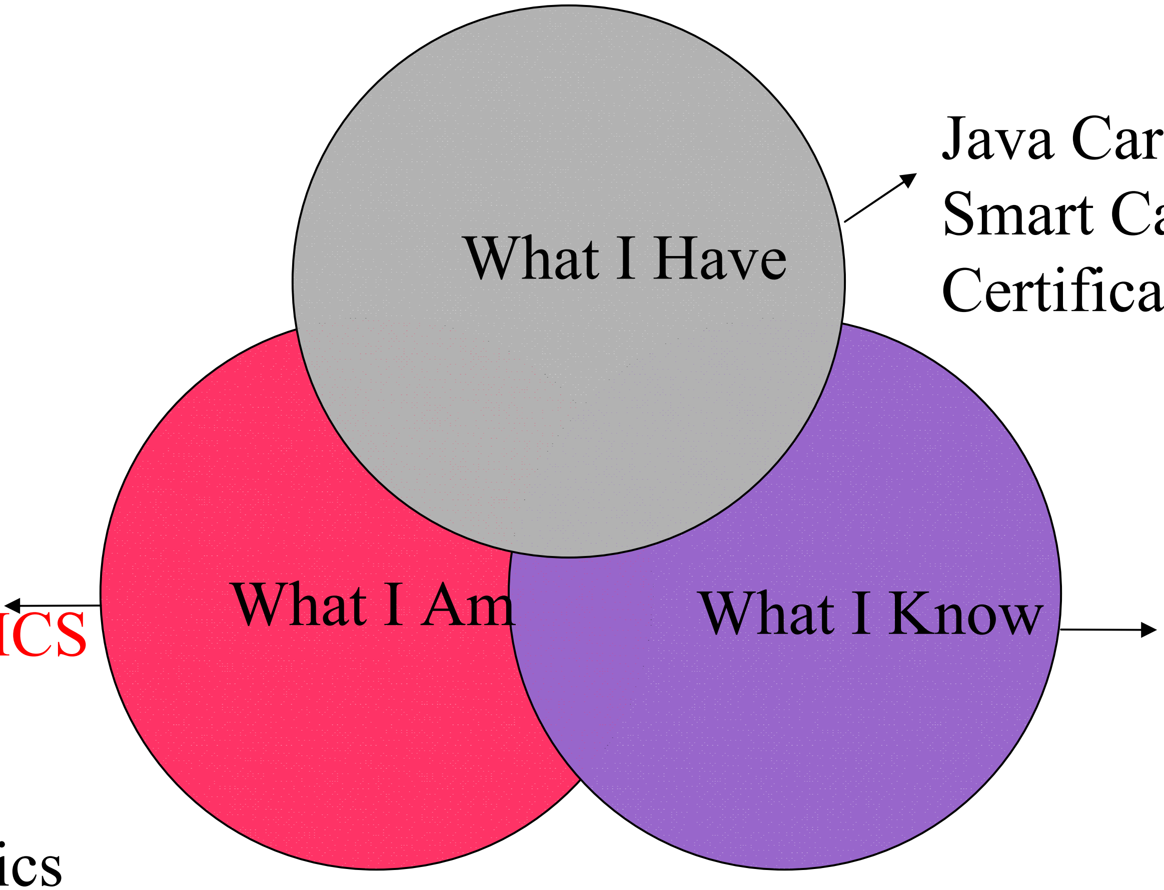
The Identity Crisis

- Impersonation, Identity frauds, Identity theft : Fastest growing crime in the World
  - ***Someone wrongfully obtains or abuses another person's Identity information for economic or personal gain***
  - Password Phishing, Hacked or Stolen authentication credentials (PINs, Passwords and Certificates)
  - Stolen and forged Identity cards
  - Most frauds happens through trusted insiders. (E***mployees, colleagues, friends, and even family members*** .
- Identity theft incurs huge losses
  - Loss of customer confidence
  - Govt. penalties and fines



If you have forgotten y password, please call our Internet Help Line

Three Factors of Authentication



Java Card
Smart Card
Certificates

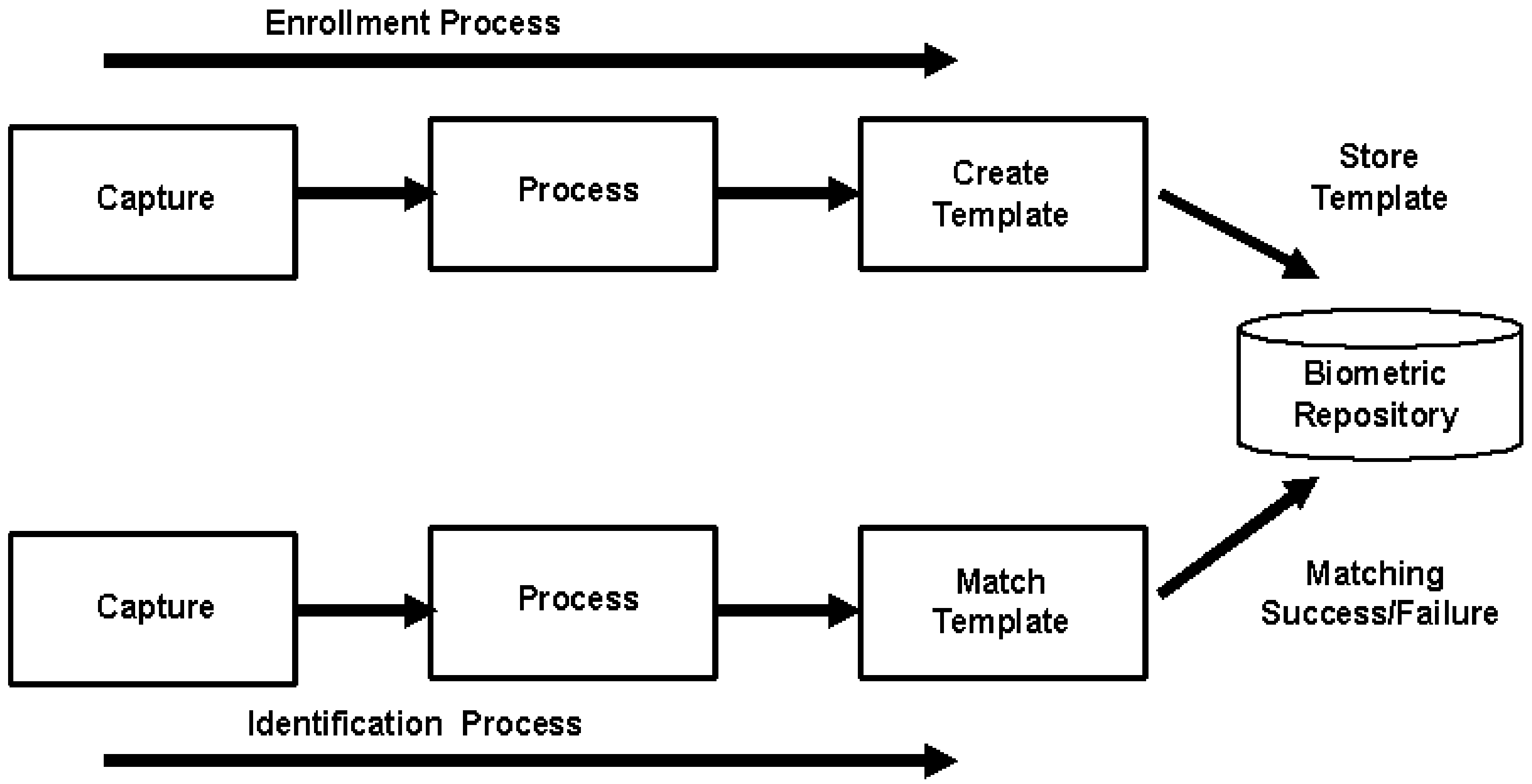
BIOMETRICS
Physical
Behavioral
Characterstics

PIN
Password
Mom's Maide
Name
SS#
DOB
Pet's Name

Biometric Authentication – By Definition

- ***Biometric Authentication refers to the use of physiological or behavioral characteristics of a human being to identify or verify a person.***
- ***A process of verifying a person's identity based on his or her unique physical or behavioral attributes, referred to as biometric samples.***
  - ***Fingerprints, Face geometry, Iris or Retinal patterns, Ear geometry, DNA, Body odor and so forth.***
  - Voice, Hand writing, Key stroke pattern and so forth.
- ***Based on pattern-recognition algorithms that allows determining the authenticity of the biometric sample.***

Biometric Authentication Process

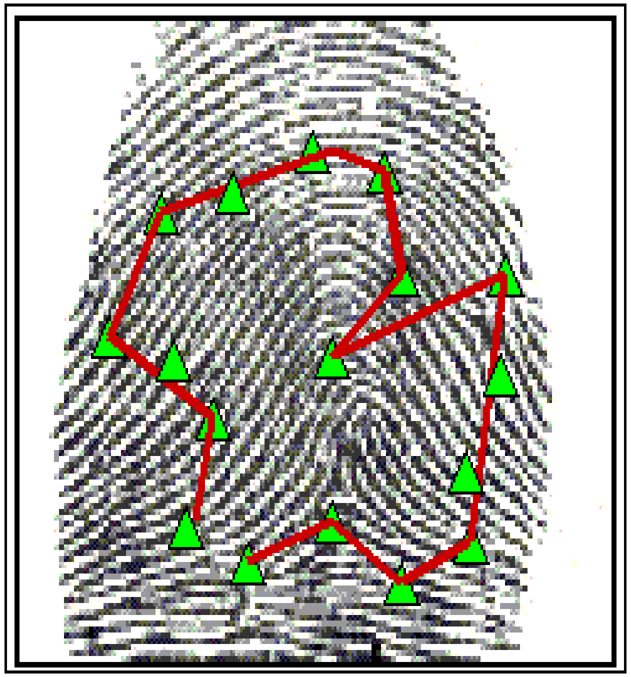


Biometric Template Size

Biometric Sample	Template Size
Fingerprint	256 bytes – 1.2k
Face	84 bytes – 2k
Hand Geometry	9 bytes
Iris	256 bytes – 512 bytes
Retina	96 bytes
Voice	70k – 80k
Signature	500 bytes – 1000 bytes

Fingerprint Matching – How it works ?

- *Fingerprint Matching based Identification and authentication is one of the oldest and most popular methods.*
- *A Fingerprint consists of a series of furrows (shallow trenches) and ridges (crests) on the surface of a finger..*
  - *The uniqueness is determined based on the patterns of ridge-ending, bifurcations, divergences, and enclosures - MINUTIAE points.*
  - *A typical fingerprint template can show from 30 to 40 minutiae points.*
- *Minutiae based approach is commonly adopted by most Fingerprint scanners.*
- *Authentication success is decided by matching score (threshold).*
  - *The provided sample must exceed a predefined threshold limit*



Fingerprint w/ Minutiae p

Accuracy of Biometric Authentication

- *Biometrics authentication is also prone to high err.*
- *Accuracy of a Biometric authentication is often affected by lot of factors.*
  - *Physical condition, weather, injury, position, location, cleanliness.*
- *Accuracy is measured by :*
  - *False Acceptance Rate (FAR)*
  - *False Rejection Rate (FRR)*
  - *Failure to Enroll (FTE)*
  - *Cross-over Error Rate*
  - *Ability to Verify (ATV)*
    - *ATV = (1 – FTE) * (1 – FRR)*
    - *Lower the ATV means the greater the accuracy and reliability of the authentication*

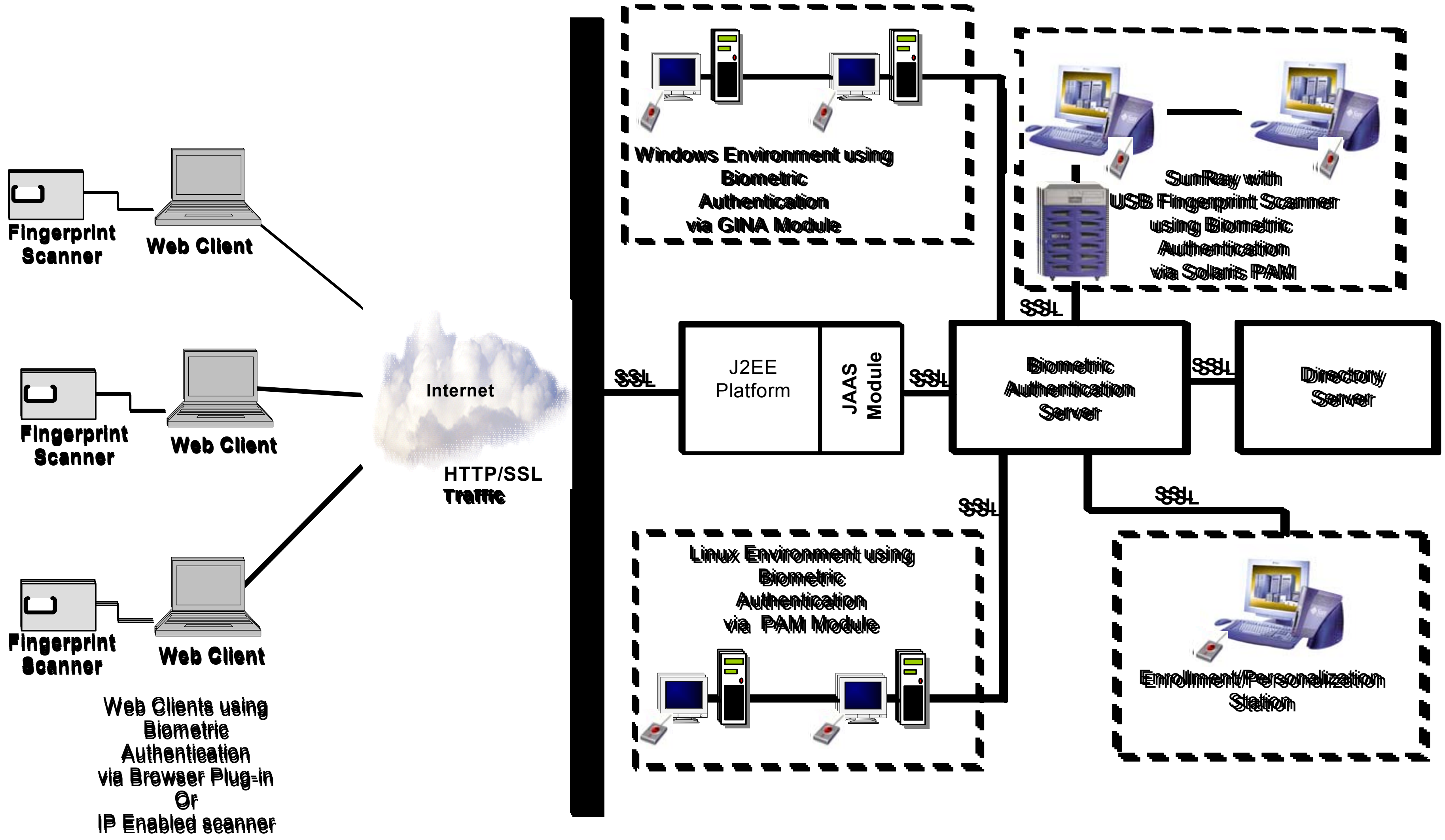
Enabling Technologies

- ***Biometric Authentication Provider***
    - ***The biometrics enrollment and authentication system is provided by a biometric vendor that facilitates enrollment, authentication, management***
- ***Biometric Scanner***
    - ***A Biometric scanner device which allows to capture a biometric sample.***
        - ***For example, a fingerprint scanner device scans the surface of a finger and obtains the patterns from the fingerprint.***
    - ***The scanner device can be integrated using USB or Serial or Ethernet interfaces.***
- ***BioAPI***
    - ***Standard based API for developing personal identification applications that interfaces with biometric verification devices***
    - ***Fingerprint scanners, facial recognition devices, iris and retina scanners, voice recognition systems, and so forth.***
    - ***Most biometric vendors offer Java Implementation for BioAPI.***
    - ***http://www.bioapi.org***

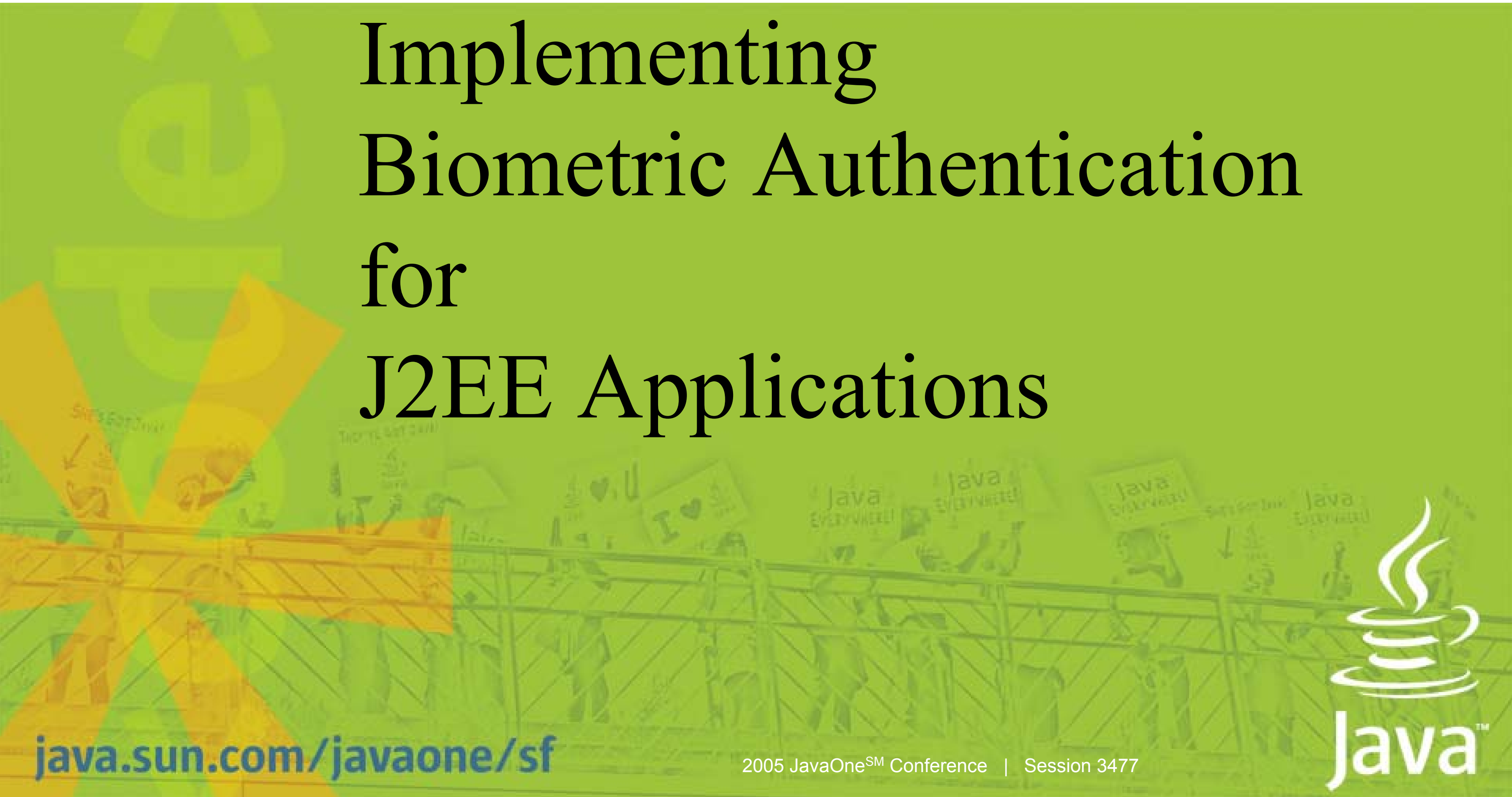
Enabling Technologies ...contd.

- ***JAAS (Java Authentication and Authorization Service)***
  - ***Java API framework that allows implementing authentication and authorization mechanisms in Java applications.***
- ***PAM (Pluggable Authentication Module)***
  - ***PAM allows applications and OSs to be independent of authentication mechanisms in a UNIX environment, particularly Solaris and Linux.***
- ***GINA (Graphical Identification and Authentication)***
  - ***GINA is a Windows dynamically linked library (DLL) in the Microsoft Windows environment that handles the default authentication process of Windows Login.***
- ***Browser Plug-In***
  - ***To support Web browser-based client authentication, browser plug-in that allows interacting with a biometric scanner to acquire biometric samples***

Logical Architecture


Fingerprint Scanner
Web Client
Fingerprint Scanner
Web Client
Fingerprint Scanner
Web Client
Web Clients using Biometric Authentication via Browser Plug-in Or IP Enabled scanner
Internet
HTTP/SSL Traffic
SSL
J2EE Platform
JAAS Module
SSL
Biometric Authentication Server
SSL
Directory Server
Windows Environment using Biometric Authentication via GINA Module
SunRay with USB Fingerprint Scanner using Biometric Authentication via Solaris PAM
SSL
SSL
SSL
Linux Environment using Biometric Authentication via PAM Module
Enrollment/Personalization Station

Implementing Biometric Authentication for J2EE Applications



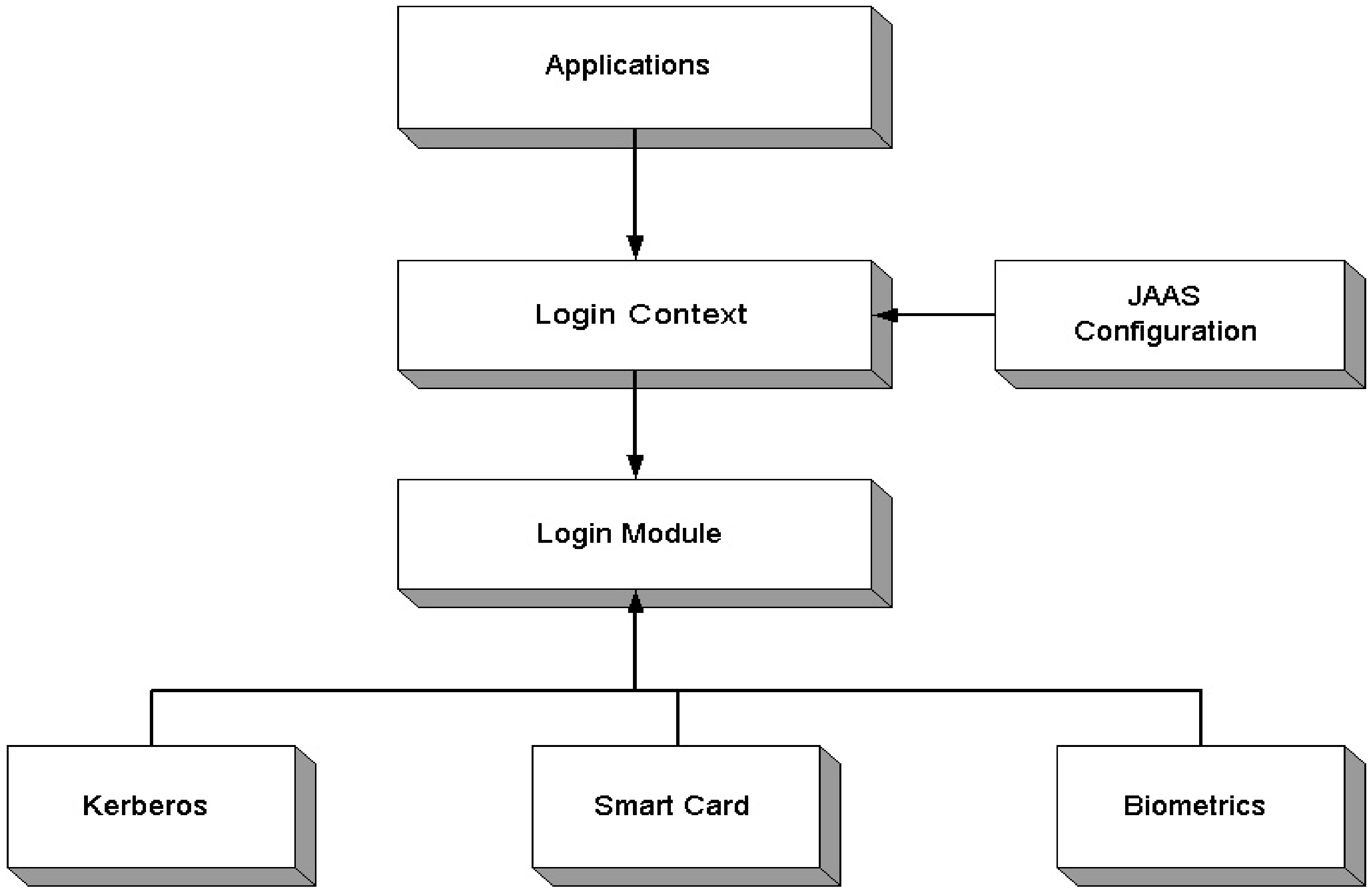
Tools of the trade

- J2EE-Compliant Application Server
- Biometric Authentication Provider
    - *Java API for Biometric Integration (Java BioAPI support)*
- JAAS LoginModule
- Biometric Scanner Device

Using Biometrics in J2EE Applications

- All J2EE compliant containers required to provide support for Java Authentication and Authorization Service (JAAS).
- JAAS allows to enable Biometric authentication in a J2EE environment
  - JAAS facilitates a pluggable authentication solution as JAAS LoginModules.
  - JAAS ensures J2EE environment remain independent of authentication providers.
  - JAAS LoginModules can be configured as J2EE realms.

Understanding JAAS



Implementing a JAAS LoginModule

1. Define a class that represents your `LoginModule`.
2. Implement the `LoginModule` interface methods.
   - `initialize ()` - initializes the authentication scheme and its state information
   - `login ()` - Performs the actual authentication process
     - Also prompts the user for obtaining authentication credentials via a CallbackHandler.
   - `commit ()` - If the `login()` is successful, the `commit()` method adds the `Principal` to authentication state.
   - `abort ()` - If the authentication fails, the abort() method exits the LoginModule and cleans up the authentication state.
   - Logout () - The `logout()` clears the subject and cleans up all `Principal` settings of the `subject` in the `LoginModule`.

Sample JAAS code

```
public class MyBioLoginModule implements LoginModule {
    private Subject subject;
    private CallbackHandler callbackHandler;
    private Map sharedState;
    private Map options;
    private String userName;
    private BioPrincipal userPrincipal;

    /** Implement LoginModule initialize() method */
    public void initialize(Subject subject, CallbackHandler callbackHandler, Map sharedState, Map
      options) {
        this.subject = subject;
        this.callbackHandler = callbackHandler;
        this.sharedState = sharedState;
        this.options = options;
    }
```

Sample JAAS code ... contd.

```
/** Implement the MyBioLoginModule login() method */
 public boolean login() throws LoginException {
     BiometricProvider   myBiometricProvider;
    // connect to the BiometricProvider
     try {
          myBiometricProvider = new BiometricProvider ("myBiometricVendor.properties");
     } catch ( LoginException  lex)  {
     } catch (Exception ex) {
     }
    // Initiate the callbacks to obtain authentication information
     Callback[] callbacks = new Callback[2];
try {
         callbackHandler.handle(callbacks);     . . .
     } catch (java.io.IOException ioe) {
     }
```

Sample JAAS code ... contd.

```
    /** Authenticate the user using the callback information */
       try {
          boolean   result
                = myBiometricProvider.authRequest (NameCallbackObj, DeviceCallbackObj);
       } catch ( LoginException  lex)  {
     }
     return result;
    }
   /** Implement LoginModule commit() method */
     public boolean commit() throws LoginException {
       userPrincipal = new BioPrincipal(userName);
       if (userPrincipal != null && !subject.getPrincipals().contains(userPrincipal)) {
          subject.getPrincipals().add(userPrincipal);
       }
        return true;
    }
```

Sample JAAS configuration

```
MyBioLoginModule {
        com.csp.jaasmodule.BioLoginModule sufficient  debug=true biometricserver=127.0.0.1
               biometricServerPort=9999   keyStoreLocation=/usr/j2se/lib/security/keys
                 keystorePassword=changeit;
 };
```

JAAS Options

- **Required**: Defines that the associated login module must succeed with authentication.
- **Requisite**: Defines that the associated login module must succeed for the overall authentication to be considered as successful
- **Sufficient**: Defines the associated login module's successful authentication sufficient for the overall authentication.
- **Optional**: Defines that the associated login module authentication is not required to succeed.

Implementation Strategies

- Biometric Authentication in J2EE environment
  - Configure JAAS Module as a J2EE realm
    - Realm configuration is often specific to a J2EE vendor.
  - Enables access to J2EE applications with a Biometric Login
- Biometric Single sign-on (SSO)
  - Configure JAAS Module with a SSO Security provider
  - Unified Biometric SSO with heterogenous applications
    - ex. Web Portal
  - All participating application can make use of an Unified Biometric signon process.

Understanding Biometric Single Sign-on

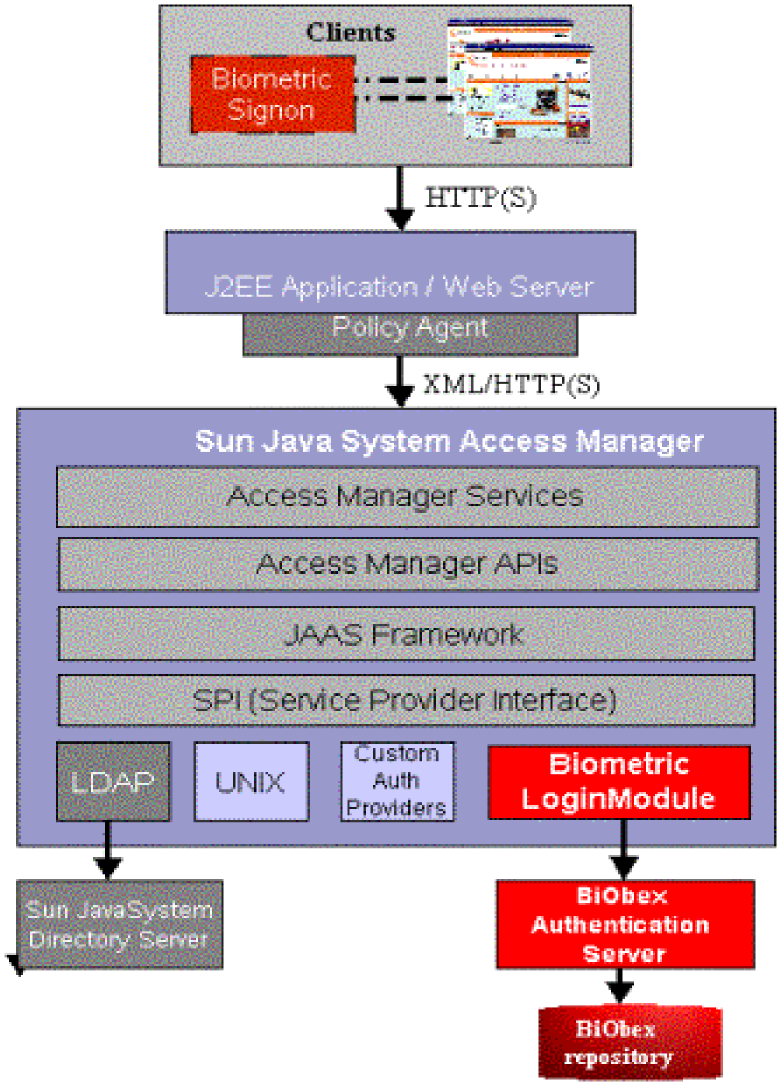
Biometric Single sign-on

- Use Biometric Single sign-on (SSO) to enable unified access to multiple applications.
  - Avoid multiple sign-on scenerios
  - Web portal aggregation
  - Support heterogenous applications
- Once authenticated...
  - Issue an SSO token that represents the user's sign-on and session information.
  - Verifying and validate the user's SSO token for controlling access to resources based on user's policies.

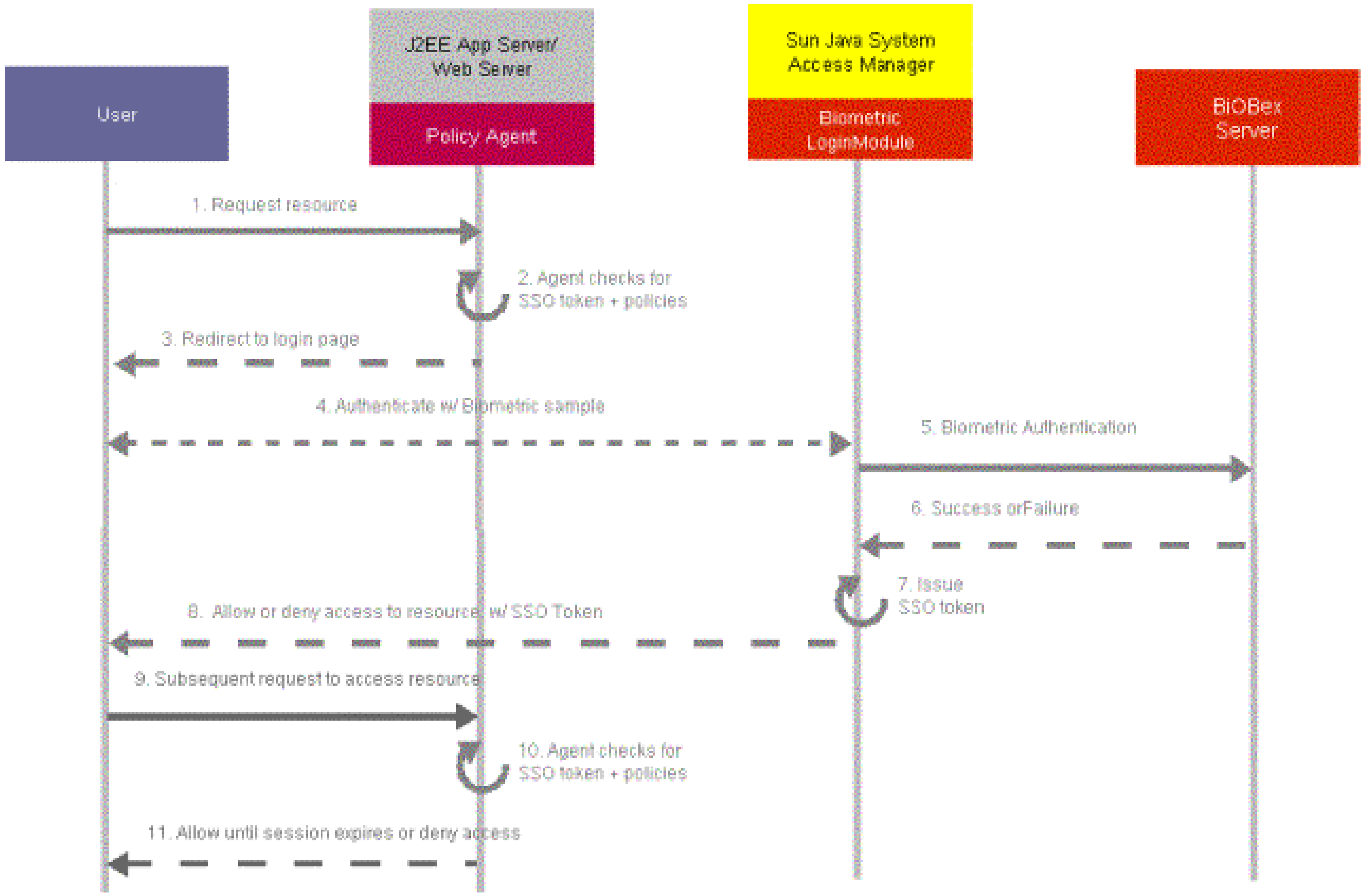
Enabling Technologies

- Sun Java System Access Manager.
  - Runs on a J2EE container
  - JAAS based Authentication Authorization Framework
  - Single Sign-on and Federation Support
  - Supports heterogenous applications – Java, Non-Java, Web-based and Enterprise applications.
- BiOBex (from AC Technology)
  - Java based Biometric authentication provider.
  - JAAS, PAM and GINA modules.
  - Integrates with J2EE application servers, Solaris, Linux and Windows.
  - Military-grade security (Trusted Solaris support).

Architecture



How it works


J2EE App Server/ Web Server
Sun Java System Access Manager
User
Policy Agent
Biometric LoginModule
BiOBex Server
1. Request resource
2. Agent checks for SSO token + policies
3. Redirect to login page
4. Authenticate w/ Biometric sample
5. Biometric Authentication
6. Success orFailure
7. Issue SSO token
8. Allow or deny access to resource w/ SSO Token
9. Subsequent request to access resource
10. Agent checks for SSO token + policies
11. Allow until session expires or deny access

DEMO

Biometric Single Sign-on for a Web Portal

For More Information

- Core Security Patterns

  Chris Steel, Ramesh Nagappan & Ray Lai

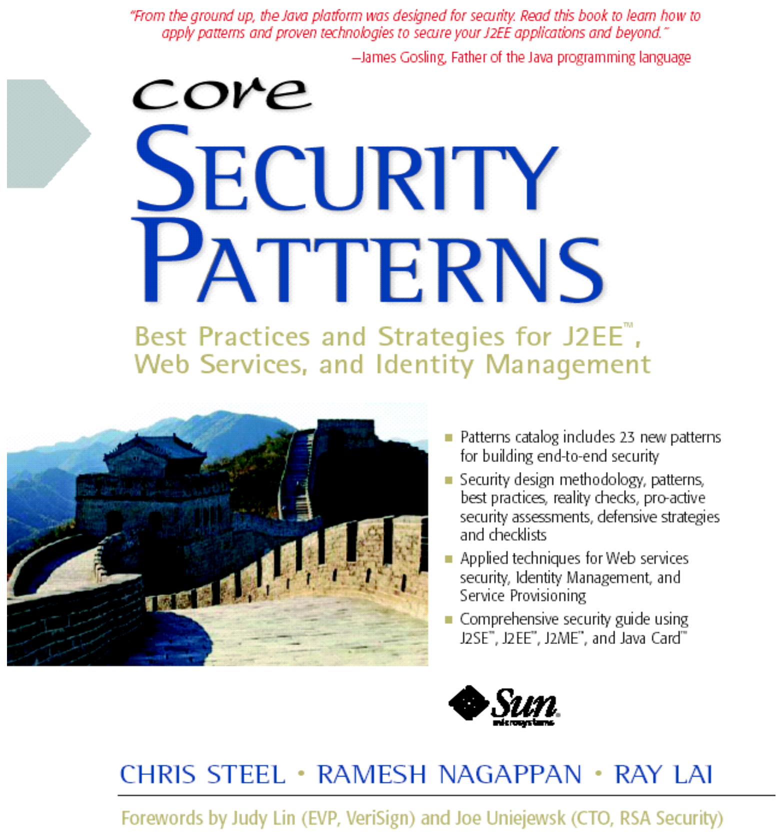
  - Special focus on Architecture and Implementation Strategies for using "Biometrics and Smart cards"
  - Sun Press, September 2005

- Building Biometric Authentication for J2EE, Web and Enterprise applications.

  Ramesh Nagappan and Tuomo Lampinen

  http://developers.sun.com/prodtech/identserver/reference/techart/bioauthentication.html

- More information at "www.coresecuritypatterns.com"


"From the ground up, the Java platform was designed for security. Read this book to learn how to apply patterns and proven technologies to secure your J2EE applications and beyond."
—James Gosling, Father of the Java programming language

core
SECURITY PATTERNS
Best Practices and Strategies for J2EE™, Web Services, and Identity Management

- Patterns catalog includes 23 new patterns for building end-to-end security
- Security design methodology, patterns, best practices, reality checks, pro-active security assessments, defensive strategies and checklists
- Applied techniques for Web services security, Identity Management, and Service Provisioning
- Comprehensive security guide using J2SE™, J2EE™, J2ME™, and Java Card™

Sun microsystems

CHRIS STEEL • RAMESH NAGAPPAN • RAY LAI

Forewords by Judy Lin (EVP, VeriSign) and Joe Uniejewsk (CTO, RSA Security)

MICHEL

Biometric Authentication for J2EE Applications

Ramesh Nagappan
nramesh@post.harvard.edu

Reid Williams
reid.williams@alum.mit.edu