Web Application Access Control with Java SE Security

Java Forum Stuttgart 2009

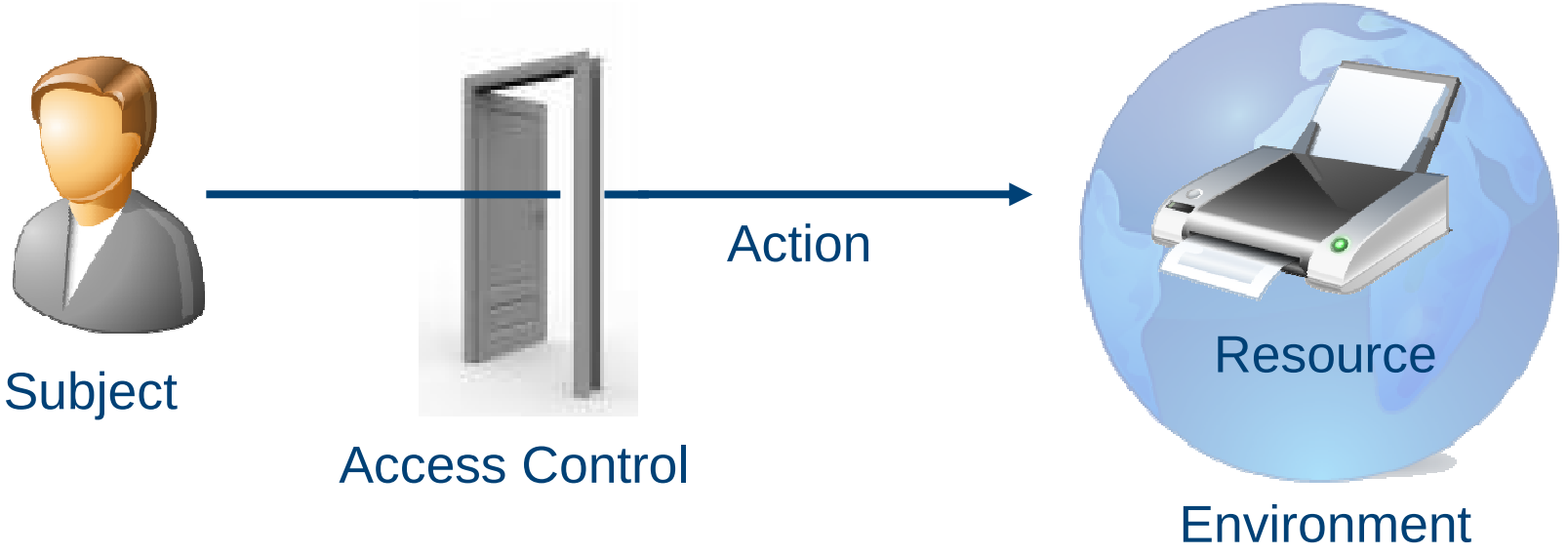
Jürgen Groothues

Stuttgart, 02.07.2009

Agenda

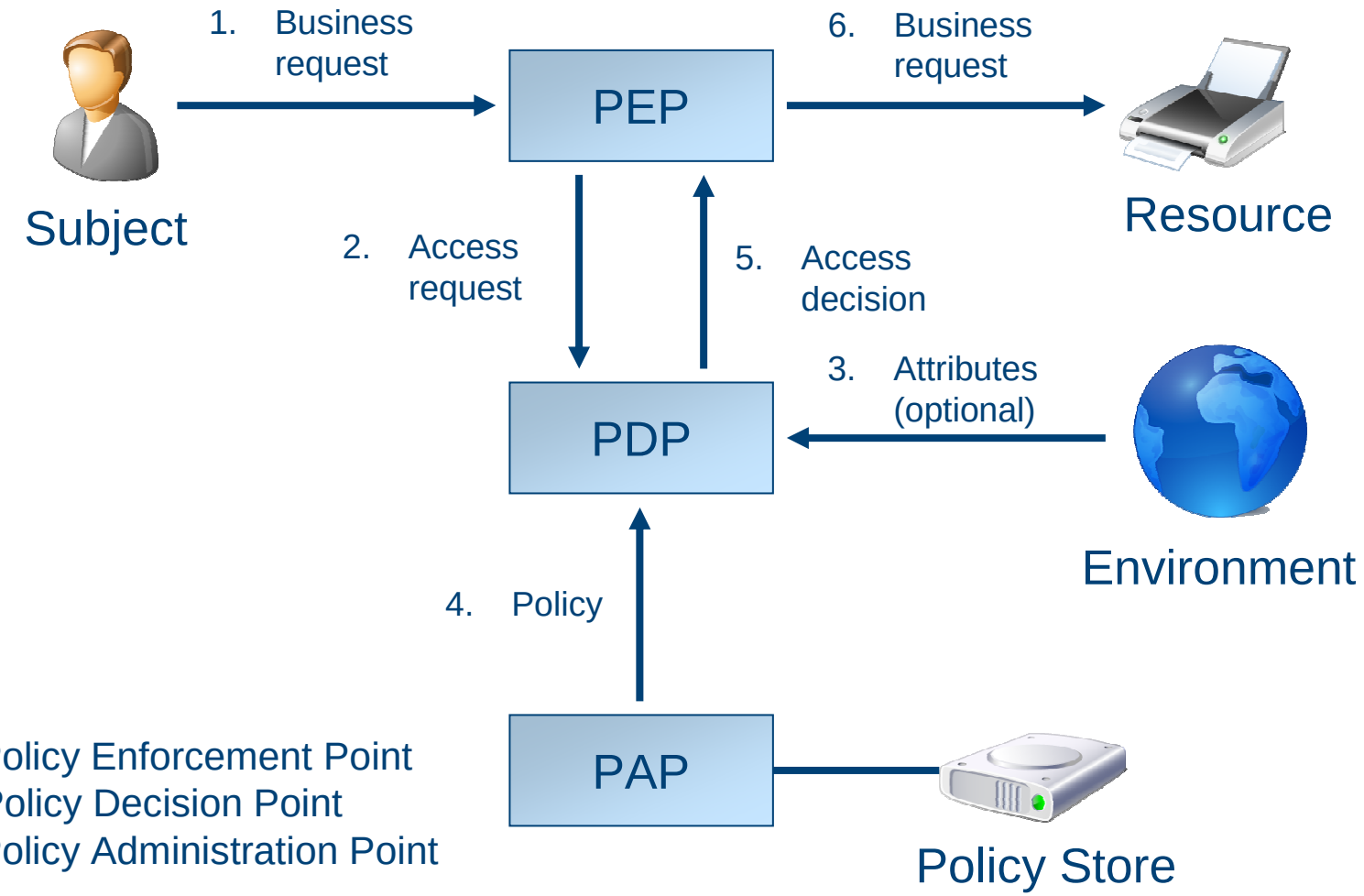
1. Access Control Basics
2. The Java Authentication and Authorization Service (JAAS)
3. Enhancement and Application of JAAS
4. Role-Based Access Control
5. Instance-Based Access Control
6. Sample Application: A Personal Health Record

Access Control Basics



- **Subject**: A user, system, etc.
- **Resource**: A file, printer, domain object, etc.
- **Action**: An operation on a resource (read, print, create, etc.)
- **Environment**: Access control relevant attributes not available from Subject, Action or Resource (time, location, …)
- **Access Control**: Controls performing an Action in accordance with a Policy

Access Control Architecture



Agenda

1. Access Control Basics
2. The Java Authentication and Authorization Service (JAAS)
3. Enhancement and Application of JAAS
4. Role-Based Access Control
5. Instance-Based Access Control
6. Sample Application: A Personal Health Record

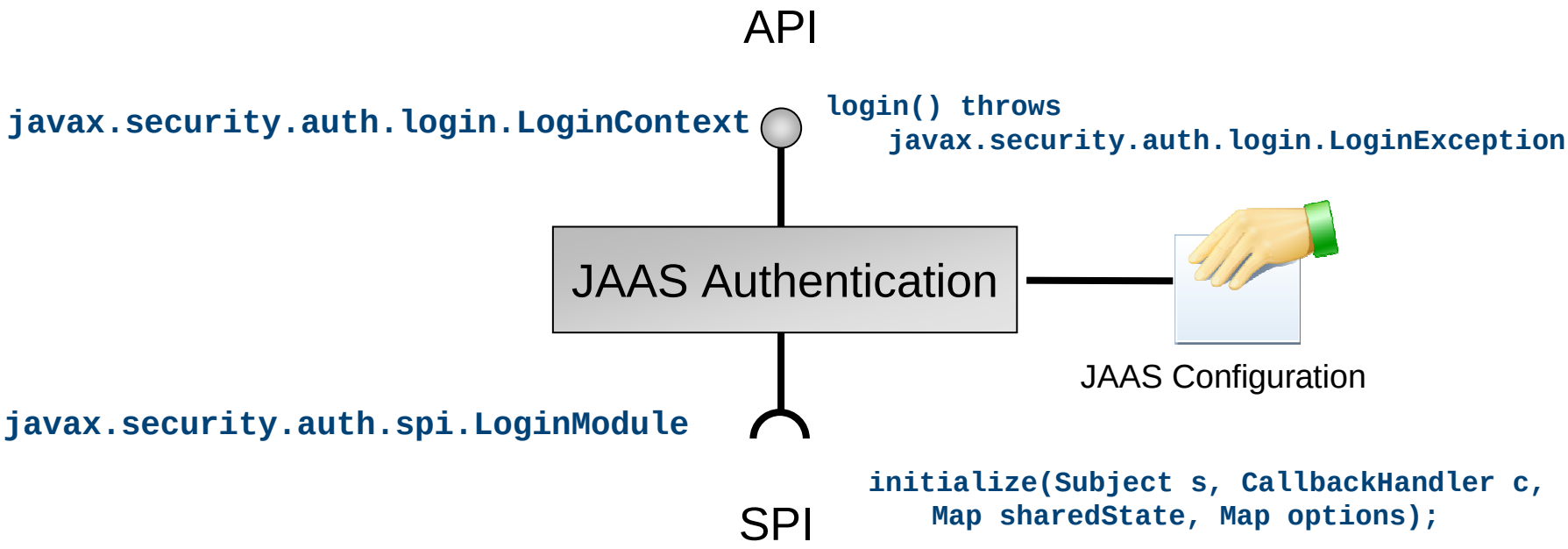
Limitations of JEE Security

- With the declarative JEE security model, it is difficult to change security policies when new requirements arise
- The declarative model limits the expressiveness of security policies
- Only one authentication method per application allowed
- Supports only limited set of authentication methods

Java Authentication and Authorization Service (JAAS)

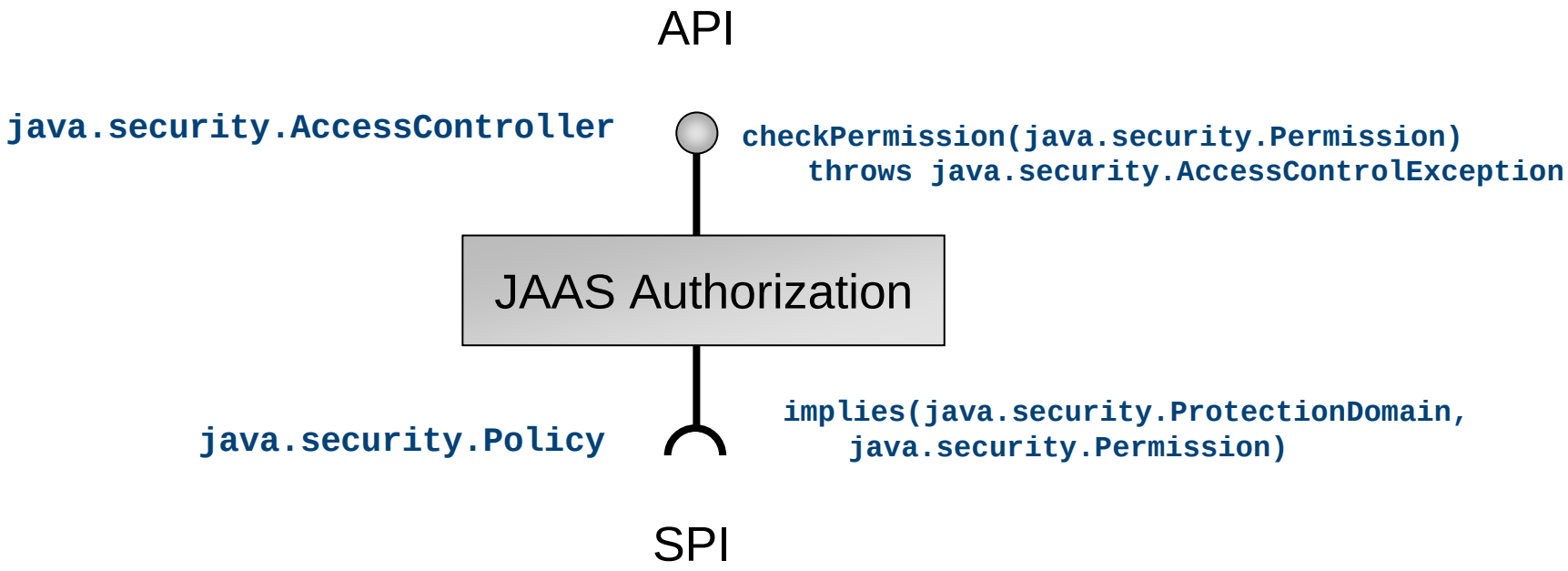
- Originally, Java SE Authorization was based exclusively on the code accessing resources and authentication was based on digital signatures applied to the code
- JAAS was designed to address this shortcoming and is part of Java SE since V. 1.4
- JAAS authentication is an implementation of the Pluggable Authentication Module (PAM) framework and allows applications to authenticate independently from the underlying technology (user/password, certificate,…)
- JAAS authorization allows access control based on who is executing the code

JAAS Authentication




```
initialize(Subject s, CallbackHandler c,
    Map sharedState, Map options);

login() throws LoginException;

commit() throws LoginException;

abort() throws LoginException;

logout() throws LoginException;
```

JAAS Authorization

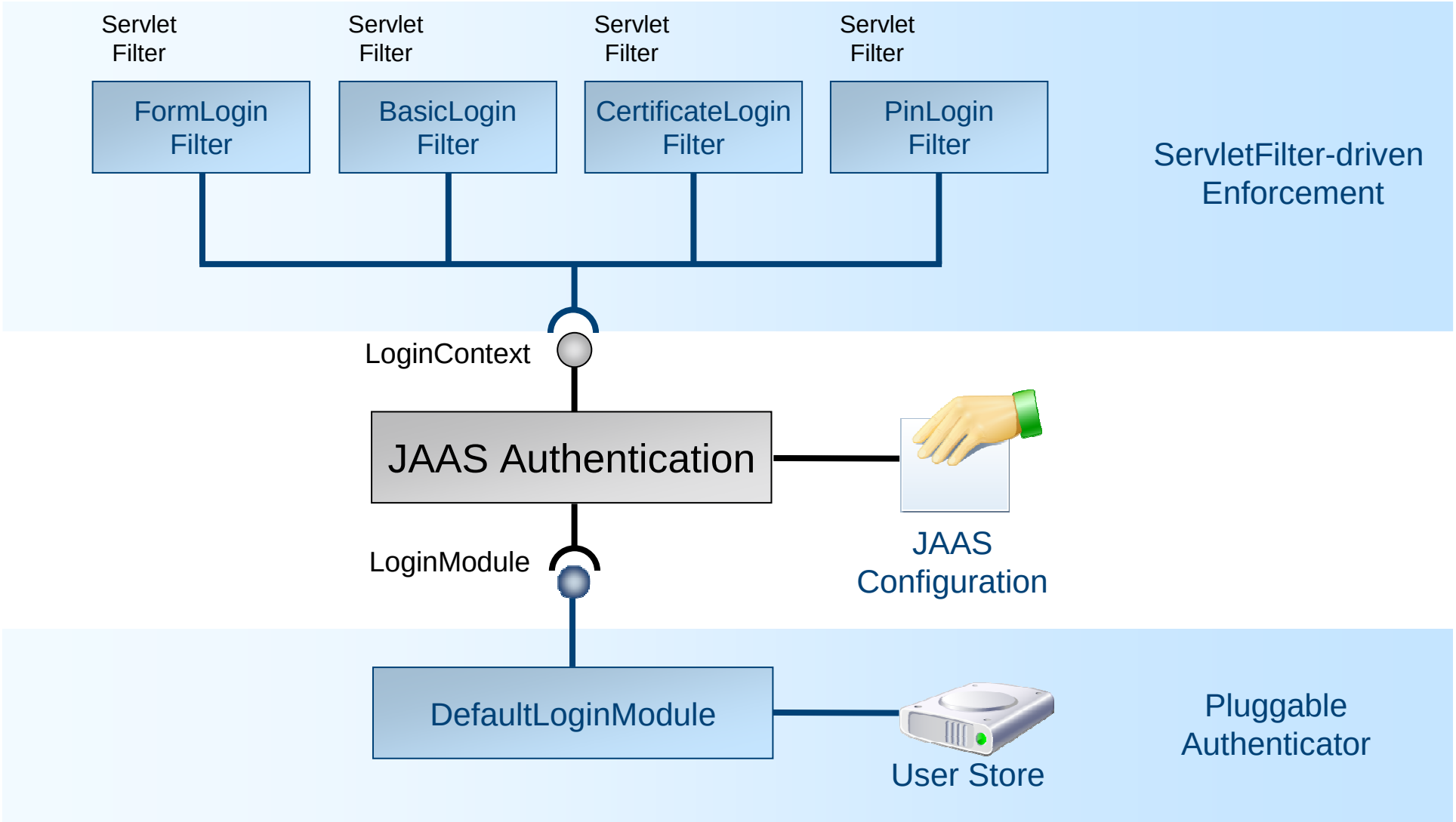


Permission: encapsulates access control relevant attributes of Action and Resource
AccessControlException: thrown if access to Resource is denied
ProtectionDomain: provides Subject

Agenda

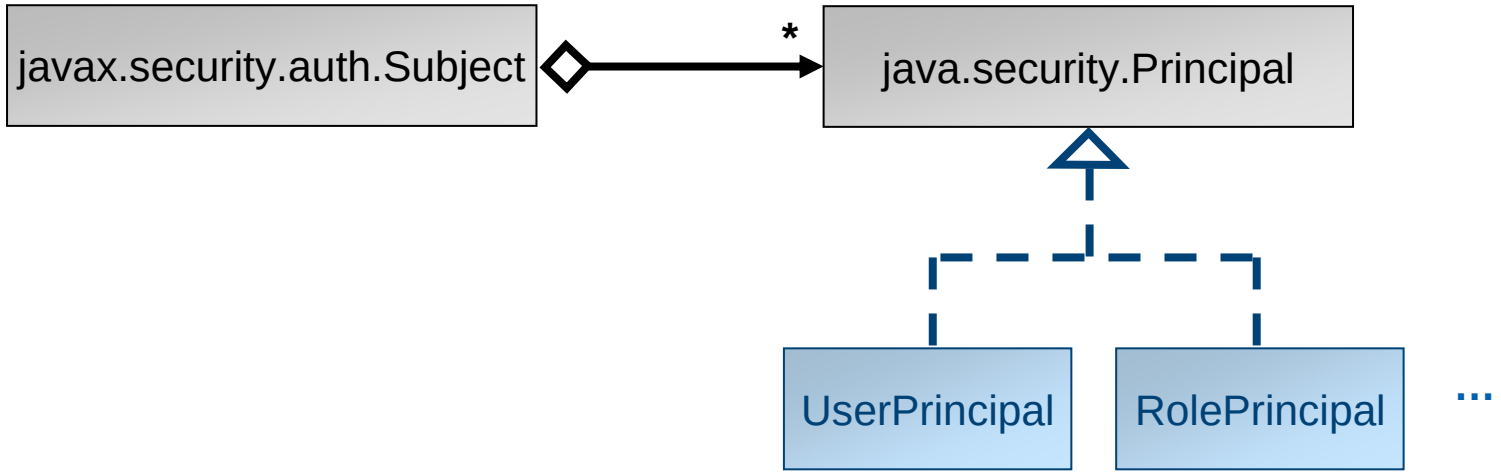
1. Access Control Basics
2. The Java Authentication and Authorization Service (JAAS)
3. Enhancement and Application of JAAS
4. Role-Based Access Control
5. Instance-Based Access Control
6. Sample Application: A Personal Health Record

Application of JAAS Authentication

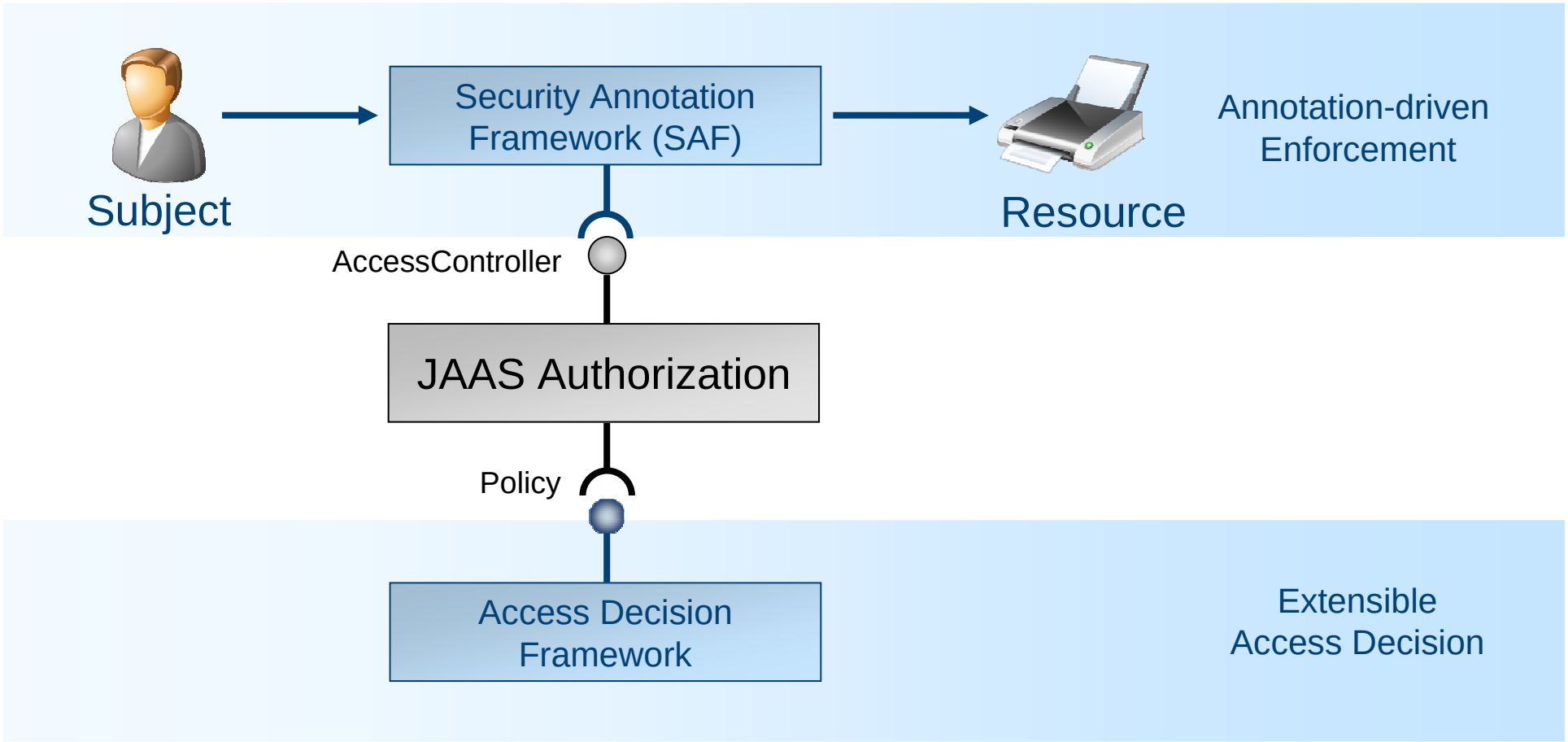


Subject and Principals

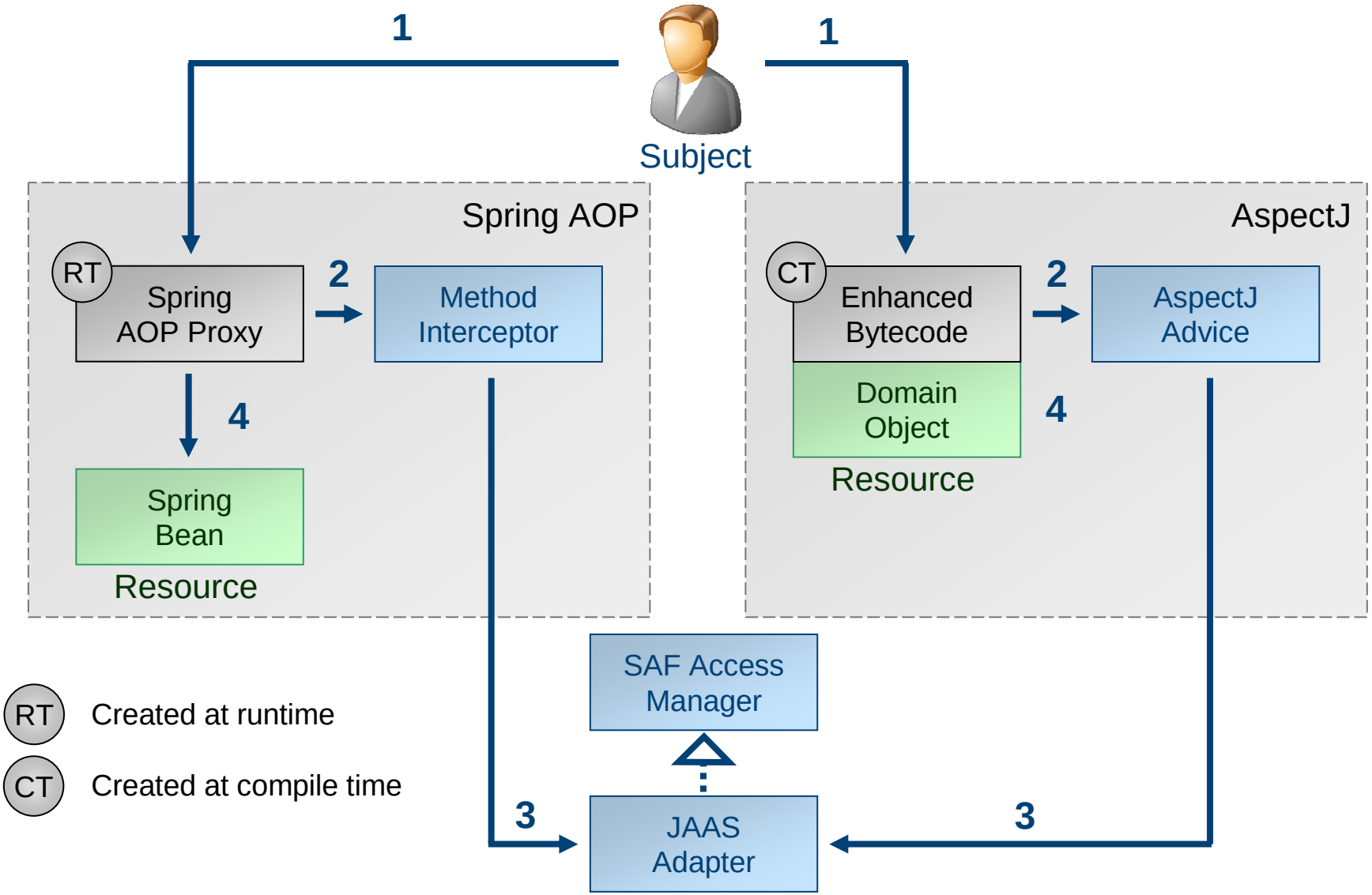
- A successful authenticated subject is represented in Java by a javax.security.auth.Subject instance
- A subject is associated with identities. In Java an identity is represented by the java.security.Principal interface
- An application provides Principal implementations



Application of JAAS Authorization



Security Annotation Framework (SAF)



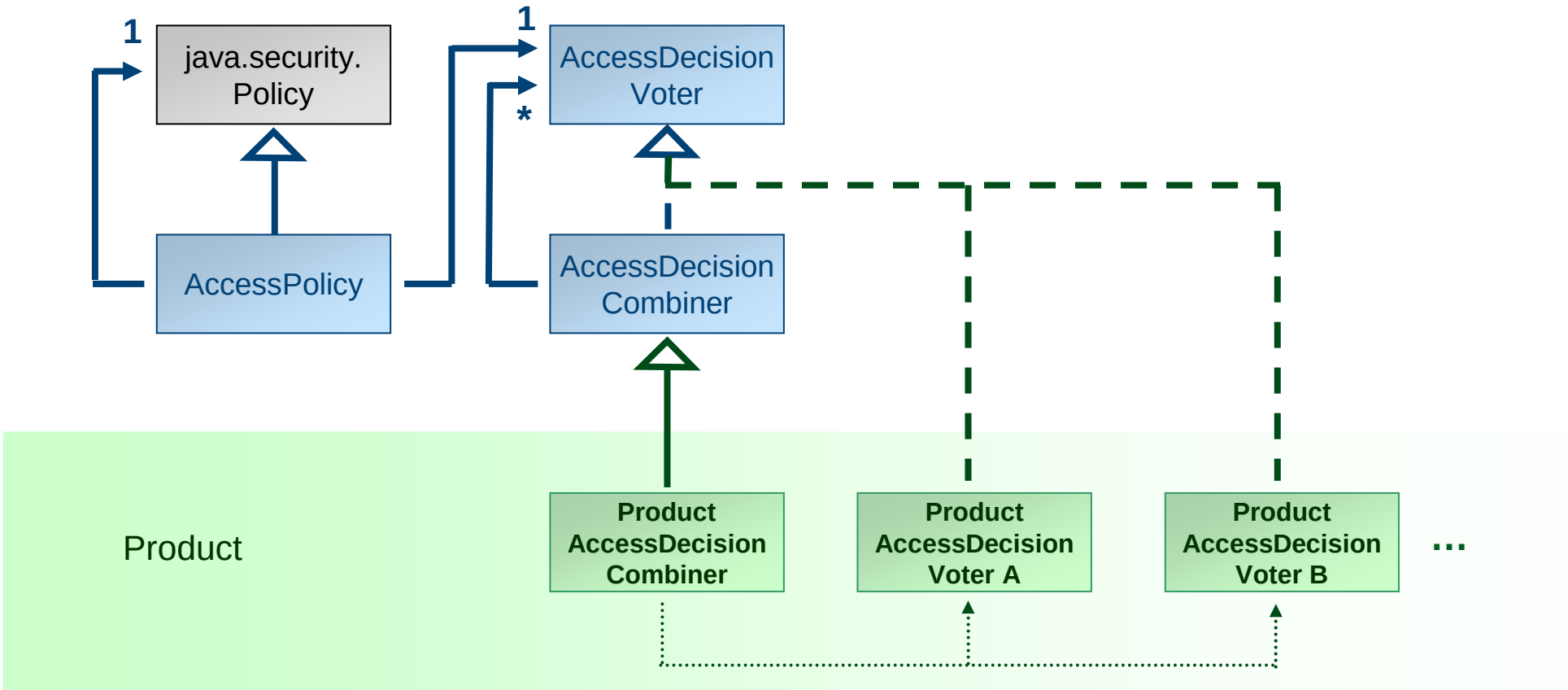
SAF Service Annotations

```
public interface RecordService {
   @Filter
    public Set<Record> findAll();

    public Record create (@Secure (SecureAction.CREATE) Record record);
}
```

SAF Domain Object Annotations

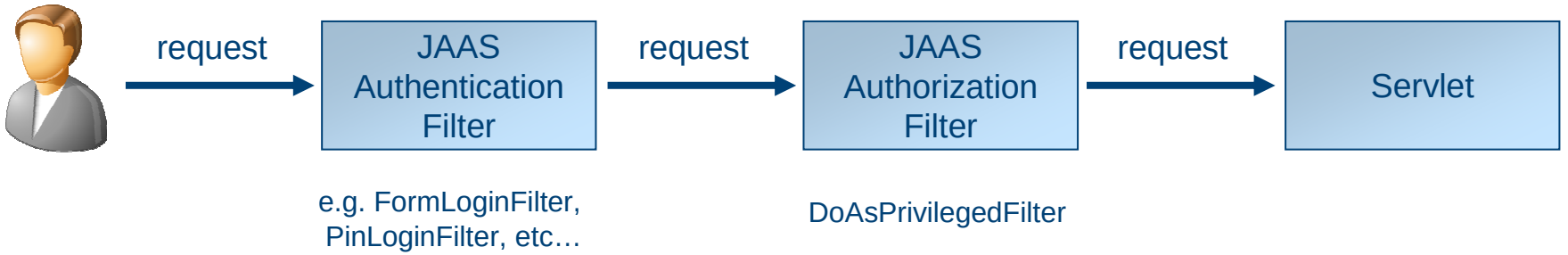
```
@SecureObject
public class Record {
    private Set<Medication> medications;

    @Secure (SecureAction.UPDATE)
    public void addMedication(Medication medication) {
         medications.add(medication);
    }
    ...
}
```

Access Decision Framework



Enabling JAAS in a Web Application

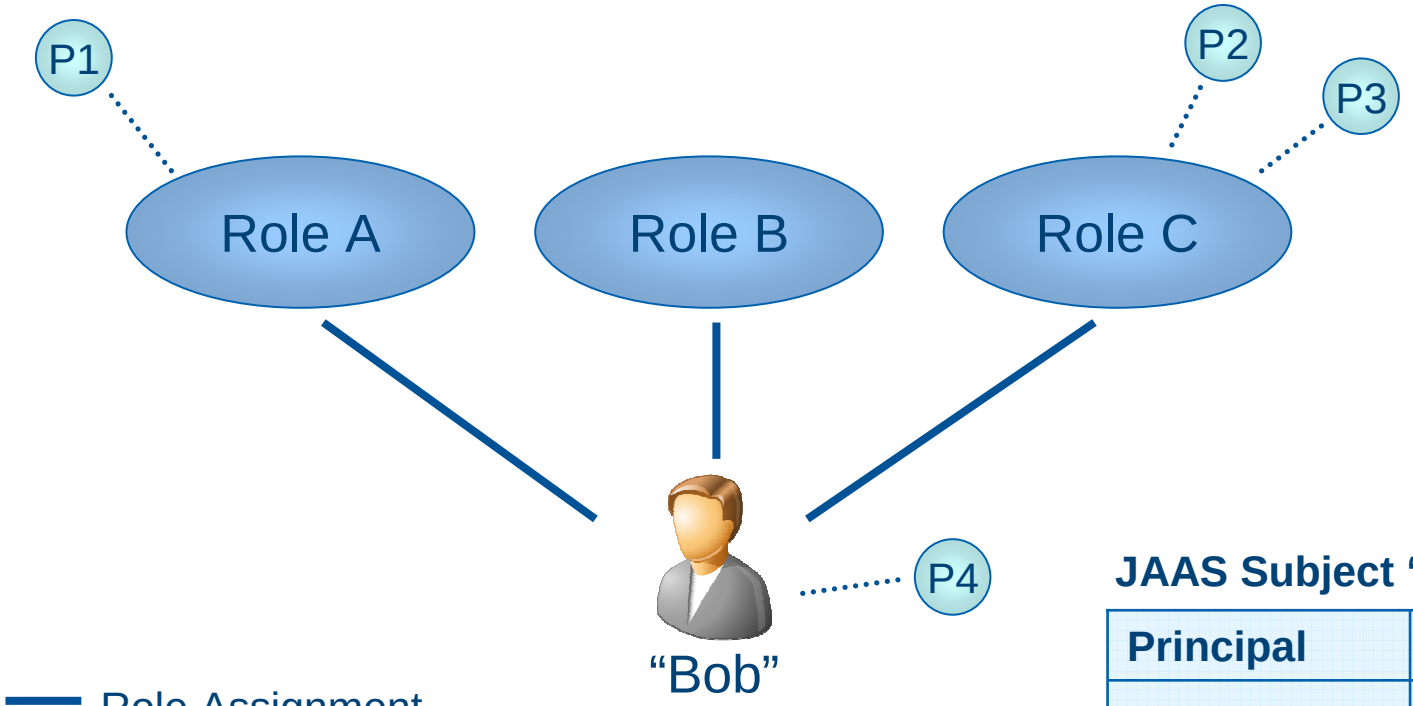
- A JEE-compliant Web Application Server (WAS) is required to support JAAS
- However, the JEE specification does not require a WAS to use JAAS as its own authentication and authorization mechanism
- JAAS has to be enabled by the web application itself:
  - Set the JAAS policy during application startup,
    i.e. call java.security.Policy.*setPolicy(*customPolicy*)* in the ContextListener
  - Use one (or more) JAAS authentication servlet filter/s
  - Use a JAAS authorization servlet filter that adds a Subject to the JAAS access control context (i.e. call javax.security.auth.Subject.doAsPrivileged(…)



Agenda

1. Access Control Basics
2. The Java Authentication and Authorization Service (JAAS)
3. Enhancement and Application of JAAS
4. Role-Based Access Control
5. Instance-Based Access Control
6. Sample Application: A Personal Health Record

Role-Based Access Control (RBAC)



Role Assignment

Pn Permission n

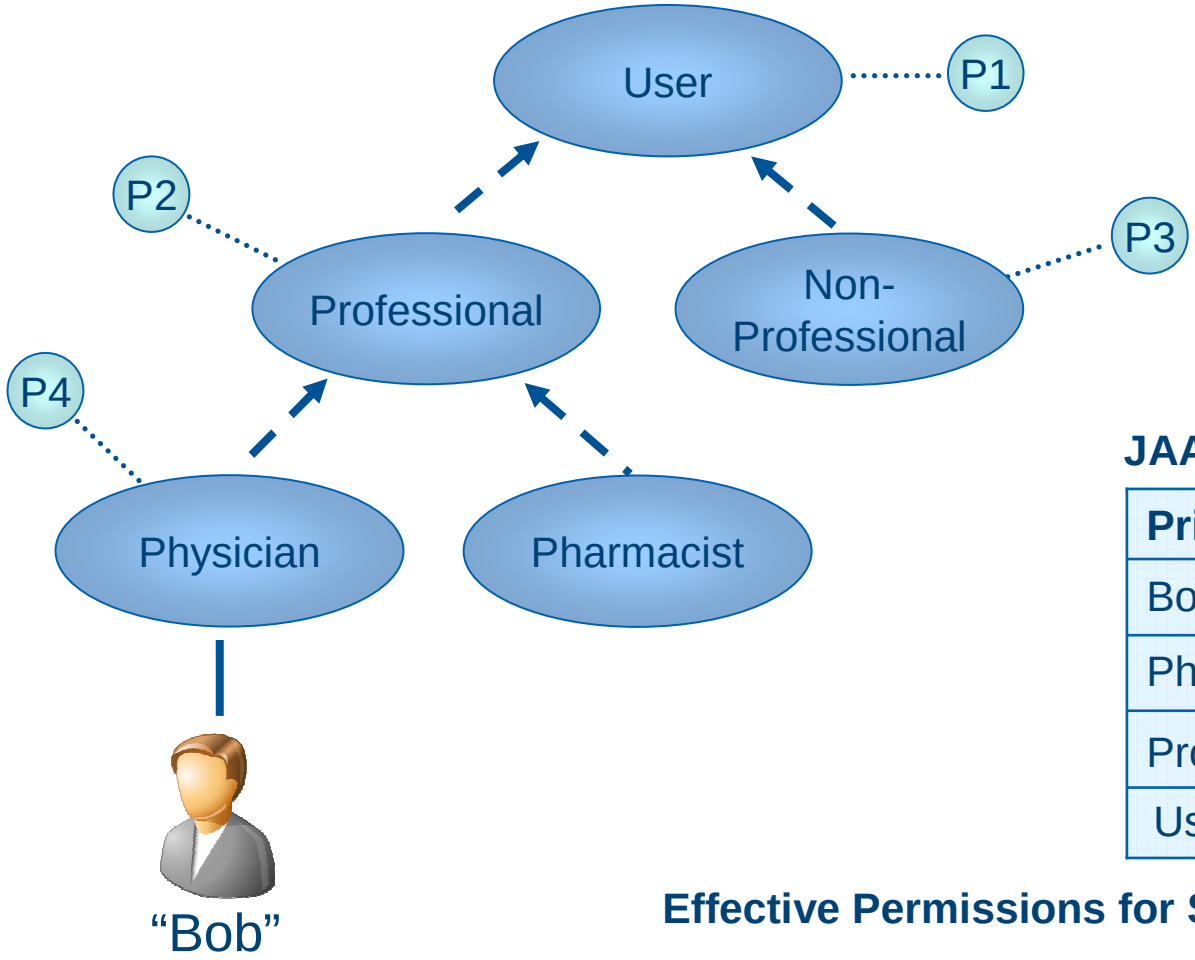
Permission Assignment

JAAS Subject "Bob":

Principal	Permission
Bob	P4
Role A	P1
Role B	-
Role C	P2,P3

Effective Permissions for Subject "Bob": P1,P2,P3,P4

Hierarchical Role-Based Access Control (HRBAC)



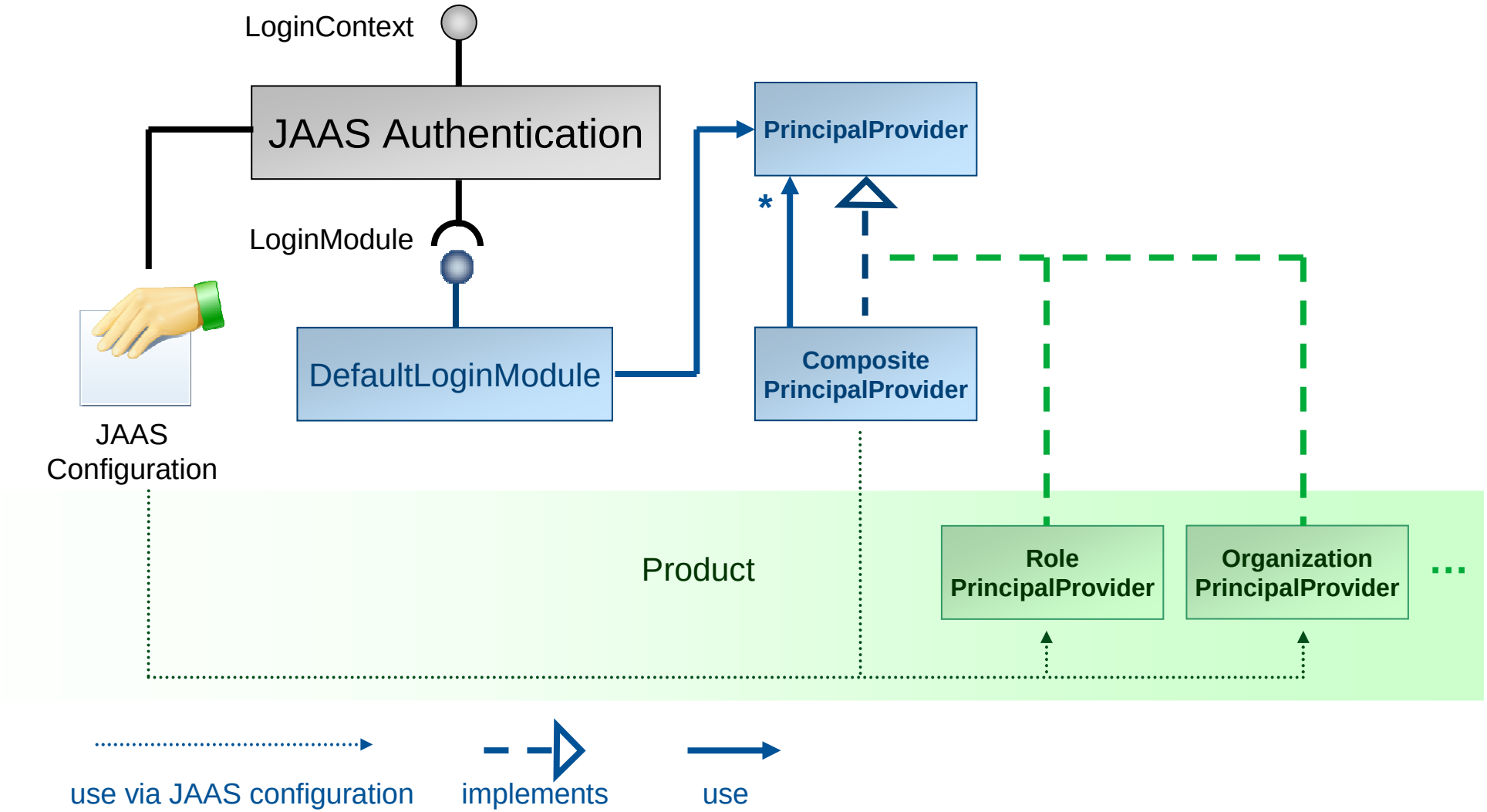
JAAS Subject "Bob":

Principal	Permission
Bob	-
Physician	P4
Professional	P2
User	P1

Effective Permissions for Subject "Bob": P1,P2,P4

Pn Permission n ····· Permission Assignment – → Parent ═══ Role Assignment

Principal Provider Pattern

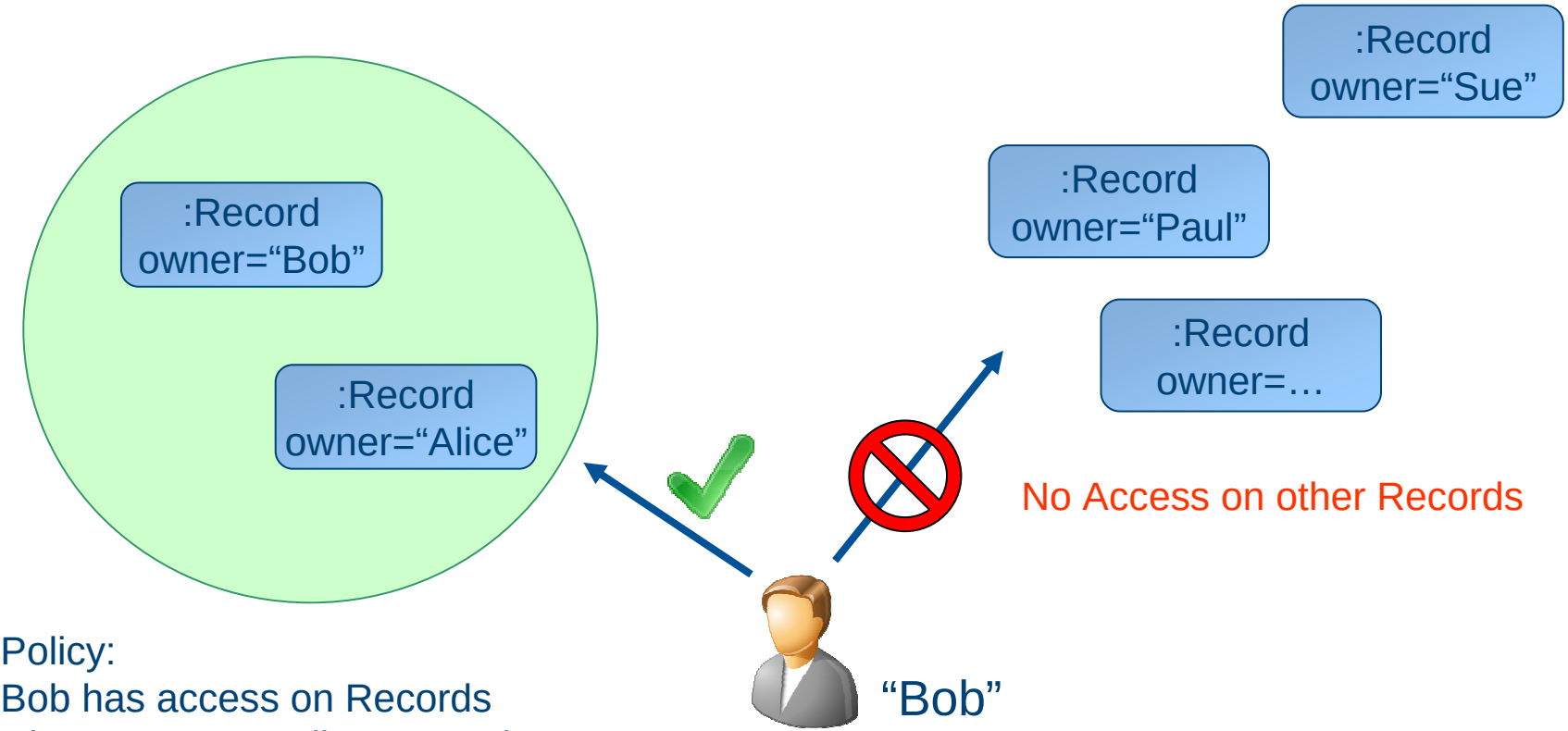


Agenda

1. Access Control Basics
2. The Java Authentication and Authorization Service (JAAS)
3. Enhancement and Application of JAAS
4. Role-Based Access Control
5. Instance-Based Access Control
6. Sample Application: A Personal Health Record

Instance-Based Access Control (IBAC)

- Instances of domain objects are secured resources
- Access decisions are based on the state of the instances



Policy:
Bob has access on Records
where owner = "Alice" or "Bob"

Annotation-Driven IBAC

```
@SecureObject
public class Record {
    ...

    @SecurityRelevant
    private String owner;

    ...
}
```

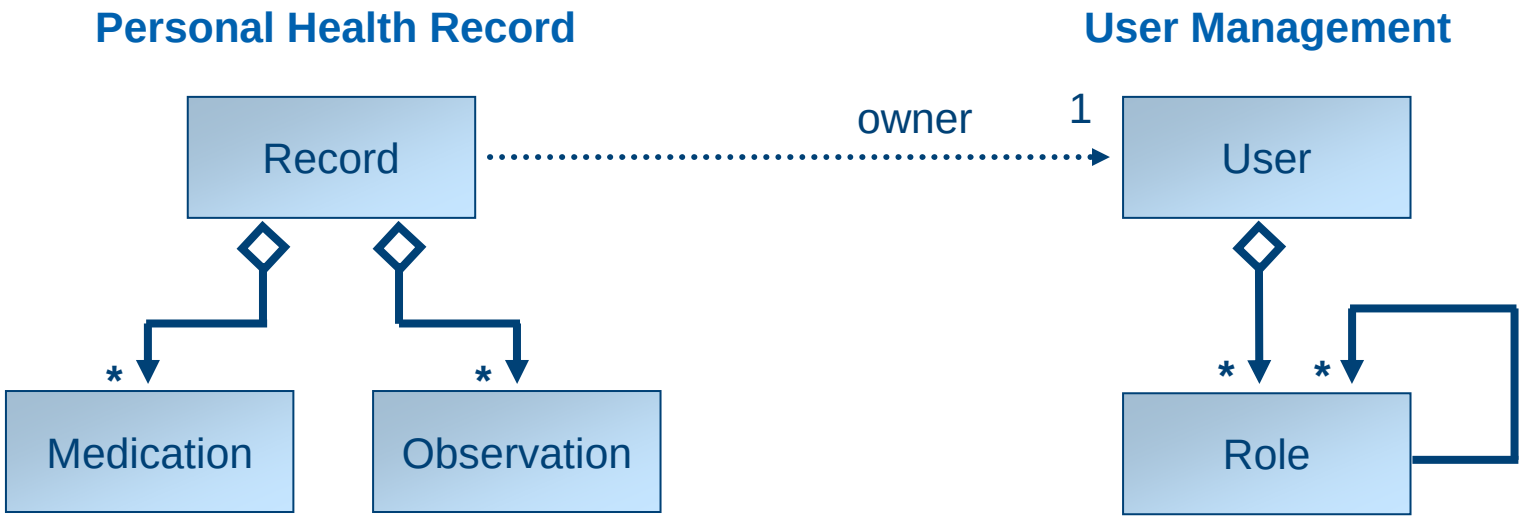
- The PEP extracts all security relevant attributes from the domain object instance and puts them into a java.security.Permission implementation.

Agenda

1. Access Control Basics
2. The Java Authentication and Authorization Service (JAAS)
3. Enhancement and Application of JAAS
4. Role-Based Access Control
5. Instance-Based Access Control
6. Sample Application: A Personal Health Record

Sample Application

- Domain Model



- Use Cases
  - Create a new user with roles and an associated new record
  - Grant access rights to a user
  - Add medications and observations to a record

Sample Application Technology

- Libraries:
  - Java SE 6
  - Spring IOC / AOP / MVC V. 2.5
  - Security Annotation Framework (SAF) V. 0.9
  - Servlets V. 2.5
  - AspectJ V. 1.6
- Testing
  - JUnit V. 4.4
  - EasyMock V.2.4
- Build
  - Maven V. 2.0
- Platform
  - Tomcat V. 6.0

Resources

- Java SE Security
  http://java.sun.com/javase/technologies/security/
- Security Annotation Platform (SAF)
  http://safr.sourceforge.net/
- OASIS eXtensible Access Control Markup Language (XACML)
  http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml
- Enterprise Java Security: Building Secure J2EE Applications
  by Marco Pistoia et al., Addison Wesley, 2004
- Creative Commons Icons
  http://creativecommons.org/licenses/by-nd/3.0/

Contact

InterComponentWare AG

Jürgen Groothues

Industriestraße 41

69190 Walldorf, Germany

E-Mail: juergen.groothues@icw-global.com

www.icw-global.com