The Anatomy of a Secure Java Web App Using Apache Fortress

March 10, 2019

SCaLE 17X, Pasadena

Objective

Think about how we should be securing web apps.

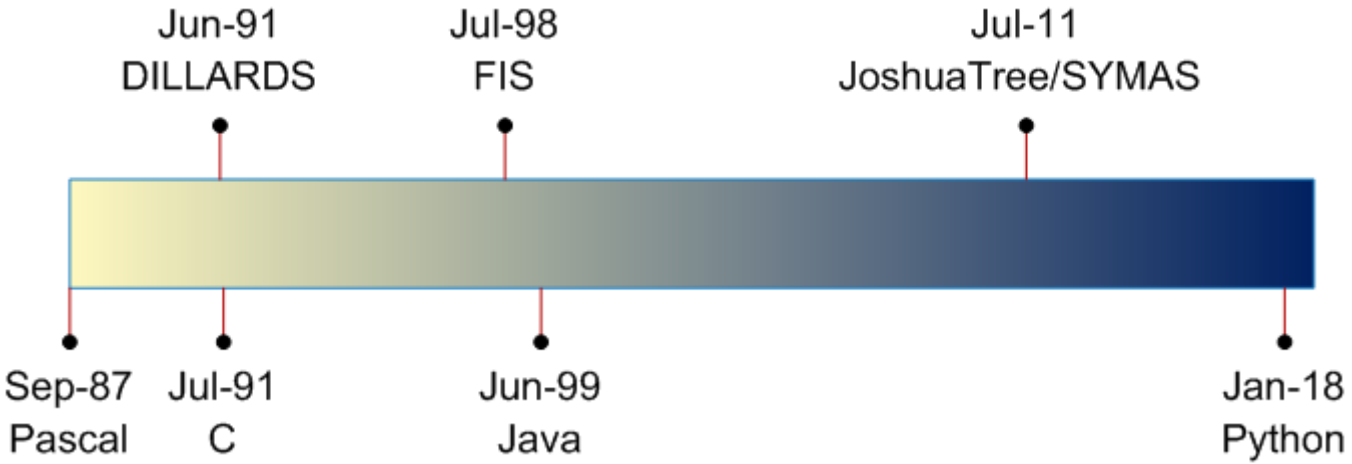
(If we pulled out all stops)

Intro

Shawn McKinney

https://github.com/shawnmckinney

Code Monkey



- symas Software Architect
- Apache Directory PMC Chair
- OpenLDAP™ PROJECT Engineering Team

Agenda

1. Discuss recent security breaches and vulnerability scanning.
2. Talk about Principle of Least Privilege
3. Look at Remote Code Execution Vulnerability
4. Talk about Defense in Depth
5. Look at Apache Fortress Demo and RBAC
6. “ Fortress ABAC Demo
7. Wrap-up

symas

Recommendation

Listen and absorb *conceptually*. Slides are published and have the *details*.

https://iamfortress.files.wordpress.com/2018/09/anatomy-secure-web-app-scale17x-2019-v1.pdf

There are 2 types of companies: those that have been hacked and those who don't know they have been hacked.

-- John T. Chambers

former CEO Cisco Systems

It's not a network as much as an ocean. Not a tier, machine or container but islands separated by miles of hostile waters. All must be attended to and defended, each with unique challenges.

Problem Statement

Make our systems more resilient to cyber attack via the application of a few common security principles.

Disclosed Breaches in 2018

- Aadhaar, 1.1B
- Exactis, 340M
- Under Armour, 150M
- MyHeritage, 92M
- Facebook, 87M

For Example

- Equifax disclosed a breach in Sep ‘17
  - 148 million consumers in the U.S., U.K. and Canada had their personal information compromised.
  - Included full names, addresses, birth dates, SSNs and drivers license numbers.

Who's Equifax

From Wikipedia, the free encyclopedia

Equifax Inc. is a consumer credit reporting agency. Equifax collects and aggregates information on over 800 million individual consumers and more than 88 million businesses worldwide. Founded in 1899 and based in Atlanta, Georgia, it is one of the three largest credit agencies along with Experian and TransUnion (known as the "Big Three").[2] Equifax has US$3.1 billion in annual revenue and 9,000+ employees in 14 countries.[3] It is listed on the NYSE as EFX.

https://cwiki.apache.org/confluence/display/WW/S2-045

What Happened

Summary

Possible Remote Code Execution when performing file upload based on Jakarta Multipart parser.

Who should read this	All Struts 2 developers and users
Impact of vulnerability	Possible RCE when performing file upload based on Jakarta Multipart parser
Maximum security rating	Critical
Recommendation	Upgrade to Struts 2.3.32 or Struts 2.5.10.1
Affected Software	Struts 2.3.5 - Struts 2.3.31, Struts 2.5 - Struts 2.5.10
Reporter	Nike Zheng <nike dot zheng at dbappsecurity dot com dot cn>
CVE Identifier	CVE-2017-5638

The Exploit

"The Jakarta Multipart parser in Apache Struts 2 2.3.x before 2.3.32 and 2.5.x before 2.5.10.1 mishandles file upload, which allows remote attackers to execute arbitrary commands via a #cmd= string in a crafted Content-Type HTTP header, as exploited in the wild in March 2017."

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-5638

The Exploit

CVE-2017-5638: Apache Struts 2 Vulnerability Leads to Remote Code Execution

Posted on: March 9, 2017 at 5:51 am **Posted in:** Exploits, Vulnerabilities
Author: Suraj Sahu (Vulnerability Research Engineer)

Apache Struts is a free and open-source framework used to build Java web applications. We looked into past several Remote Code Execution (RCE) vulnerabilities reported in Apache Struts, and observed that in most of them, attackers have used Object Graph Navigation Language (OGNL) expressions. The use of OGNL makes it easy to execute arbitrary code remotely because Apache Struts uses it for most of its processes.

Using OGNL, a researcher found a new remote code execution vulnerability in Apache Struts 2, designated as CVE-2017-5638. An exploit has been reported to be already in the wild; our own research and monitoring have also seen attacks using the vulnerability.

https://blog.trendmicro.com/trendlabs-security-intelligence/cve-2017-5638-apache-struts-vulnerability-remote-code-execution/

OGNL, again

Apache Struts RCE payloads often come in the form of Object-Graph Navigation Library (OGNL) expressions. OGNL is a language for interacting with the properties and functions of Java classes and Apache Struts supports it in many contexts.

For example, the snippet below uses OGNL to dynamically insert the value "5" into a webpage by calling a function.

```
<s:property value="%{getSum(2,3)}" />
```

OGNL expressions can also be used for more general code execution:

```
${
    #_memberAccess["allowStaticMethodAccess"]=true,
   @java.lang.Runtime@getRuntime().exec('calc')
}
```

https://blog.cloudflare.com/apache-struts-s2-057/

Behind The Equifax Breach

https://www.brighttalk.com/webcast/13983/280311/behind-the-equifax-breach-a-deep-dive-into-apache-struts-cve-2017-5638

SATURDAY SEPTEMBER 09, 2017

Apache Struts Statement on Equifax Security Breach

UPDATE: MEDIA ALERT: The Apache Software Foundation Confirms Equifax Data Breach Due to Failure to Install Patches Provided for Apache® Struts™ Exploit

2. Establish a process to quickly roll out a security fix release of your software product once supporting frameworks or libraries needs to be updated for security reasons. Best is to think in terms of hours or a few days, not weeks or months. Most breaches we become aware of are caused by failure to update software components that are known to be vulnerable for months or even years.

Once followed, these recommendations help to prevent breaches such as unfortunately experienced by Equifax.

For the Apache Struts Project Management Committee,

René Gielen
Vice President, Apache Struts

https://blogs.apache.org/foundation/entry/apache-struts-statement-on-equifax

The Solution

Ensure all appropriate patches have been applied.

OWASP Dependency-Check

https://www.owasp.org/index.php/OWASP_Dependency_Check

OWASP Dependency-Check

Dependency-Check is a utility that identifies project dependencies and checks if there are any known, publicly disclosed, vulnerabilities. Currently, Java and .NET are supported; additional experimental support has been added for Ruby, Node.js, Python, and limited support for C/C++ build systems (autoconf and cmake). The tool can be part of a solution to the OWASP Top 10 2017 A9:2017-Using Components with Known Vulnerabilities [1] previously known as OWASP Top 10 2013 A9 - Using Components with Known Vulnerabilities.

Quick Download

Version 3.3.2

- Command Line
- Ant Task
- Maven Plugin
- Gradle Plugin
- Jenkins Plugin
- Mac Homebrew

OWASP Vulnerability Scanning (Java)

Add to your Maven pom.xml file:

```
<plugin>
        <groupId>org.owasp</groupId>
        <artifactId>dependency-check-maven</artifactId>
        <version>3.3.1</version>
        <configuration>
        <failBuildOnAnyVulnerability>true</failBuildOnAnyVulnerability>

    <suppressionFile>${project.basedir}…/suppression.xml</suppressionFile>
        </configuration>
</plugin>
```

Apache Struts Statement on Equifax Security Breach

UPDATE: MEDIA ALERT: The Apache Software Foundation Confirms Equifax Data Breach Due to Failure to Install Patches Provided for Apache® Struts™ Exploit

3. Any complex software contains flaws. Don't build your security policy on the assumption that supporting software products are flawless, especially in terms of security vulnerabilities.

Once followed, these recommendations help to prevent breaches such as unfortunately experienced by Equifax.

For the Apache Struts Project Management Committee,

René Gielen
Vice President, Apache Struts

https://blogs.apache.org/foundation/entry/apache-struts-statement-on-equifax

How do we make our software free of *unknown* vulnerabilities?

The Solution (Take 2)

- ✓ Ensure all appropriate patches have been applied.
- Practice the principle of least privilege / mandatory access controls.

Principle of least privilege

From Wikipedia, the free encyclopedia

https://en.wikipedia.org/wiki/Principle_of_least_privilege

Not to be confused with Rule of least power.

In information security, computer science, and other fields, the **principle of least privilege** (also known as the **principle of minimal privilege** or the **principle of least authority**) requires that in a particular abstraction layer of a computing environment, every module (such as a process, a user, or a program, depending on the subject) must be able to access only the information and resources that are necessary for its legitimate purpose.[1][2]

Mandatory Access Control

Restrictions on even administrative users can be enforced at a granular level.

Mandatory Access Control Examples

- POSIX Security
  - sudo, file permissions
- SELinux
- Database access control lists
- Java Security Manager

Employ a Runtime Java Security Policy

```
grant codeBase "file:${catalina.home}/webapps/my-web-app-1/-" {
    permission java.net.SocketPermission "localhost", "resolve";
    permission java.net.SocketPermission "127.0.0.1:32768", "connect,resolve";
    permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
    permission java.io.SerializablePermission "enableSubclassImplementation";
    permission java.io.FilePermission "…/resources/", "execute";
    …
    };
```

^ use w/ caution

Example #1

https://github.com/shawnmckinney/remote-code-execution-sample

remote-code-execution-sample

Example shows how to use the Java Security Manager to prevent remote code execution exploits.

Intro to the Problem

- The Problem: Equifax Breach, 143 million Americans' personal info, including names, addresses, dates of birth and SSNs compromised.
- Only a veneer of security was in place.

The Exploit

- The vulnerability *Apache Struts, CVE-2017-5638.*
- http://www.zdnet.com/article/equifax-confirms-apache-struts-flaw-it-failed-to-patch-was-to-blame-for-data-breach/

The Exploit

CVE-2017-5638: Apache Struts 2 Vulnerability Leads to Remote Code Execution

Posted on: March 9, 2017 at 5:51 am **Posted in:** Exploits, Vulnerabilities
Author: Suraj Sahu (Vulnerability Research Engineer)

Apache Struts is a free and open-source framework used to build Java web applications. We looked into past several Remote Code Execution (RCE) vulnerabilities reported in Apache Struts, and observed that in most of them, attackers have used Object Graph Navigation Language (OGNL) expressions. The use of OGNL makes it easy to execute arbitrary code remotely because Apache Struts uses it for most of its processes.

Using OGNL, a researcher found a new remote code execution vulnerability in Apache Struts 2, designated as CVE-2017-5638. An exploit has been reported to be already in the wild; our own research and monitoring have also seen attacks using the vulnerability.

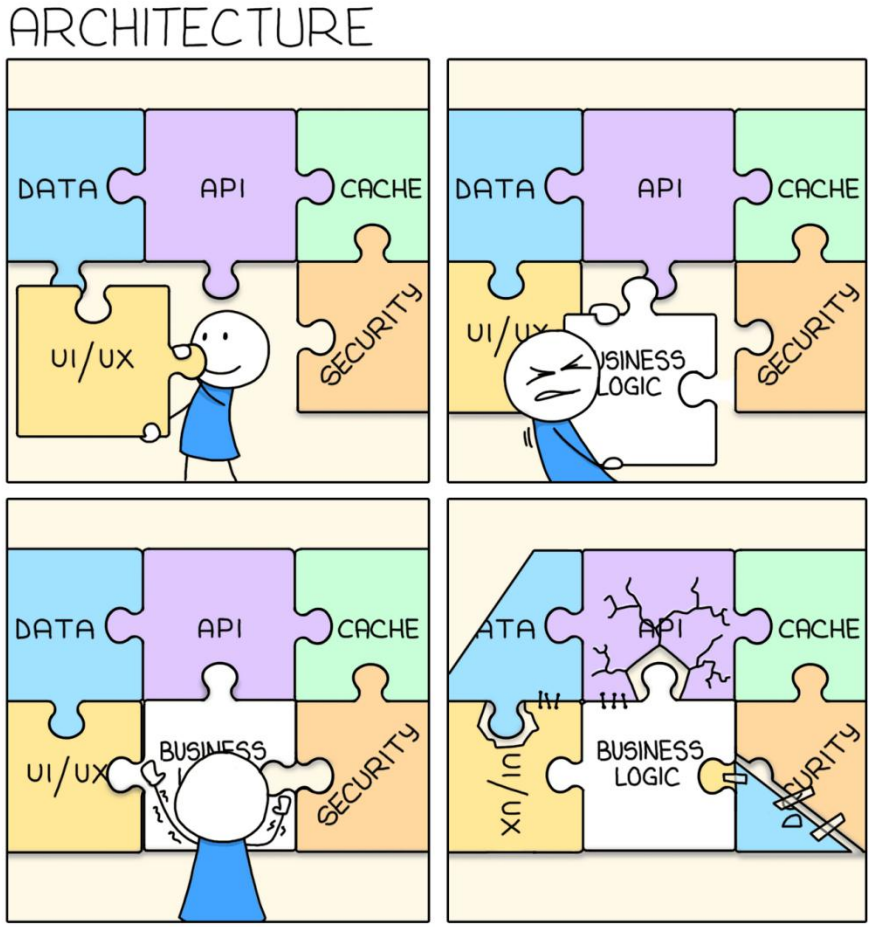
symas

Not a Perfect Solution

```
grant codeBase "file:${catalina.home}/webapps/my-web-app-1/-" {
   permission java.net.SocketPermission "localhost", "resolve";
   permission java.io.FilePermission "…/resources/good-scripts*", "execute";
   permission java.net.SocketPermission "127.0.0.1:32768", "connect,resolve";
   permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
   permission java.io.SerializablePermission "enableSubclassImplementation";
   permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
   };
```

Permission Target Name	What the Permission Allows	Risks of Allowing this Permission
suppressAccessChecks	ability to access fields and invoke methods in a class. Note that this includes not only public, but protected and private fields and methods as well.	This is dangerous in that information (possibly confidential) and methods normally unavailable would be accessible to malicious code.

What now?



https://www.monkeyuser.com/2018/architecture/

SATURDAY SEPTEMBER 09, 2017

Apache Struts Statement on Equifax Security Breach

UPDATE: MEDIA ALERT: The Apache Software Foundation Confirms Equifax Data Breach Due to Failure to Install Patches Provided for Apache® Struts™ Exploit

The Apache Struts Project Management Committee (PMC) would like to comment on the Equifax security breach, its relation to the Apache Struts Web Framework and associated media coverage.

4. Establish security layers. It is good software engineering practice to have individually secured layers behind a public-facing presentation layer such as the Apache Struts framework. A breach into the presentation layer should never empower access to significant or even all back-end information resources.

Once followed, these recommendations help to prevent breaches such as unfortunately experienced by Equifax.

For the Apache Struts Project Management Committee,

René Gielen
Vice President, Apache Struts

https://blogs.apache.org/foundation/entry/apache-struts-statement-on-equifax

Back to the The Exploit

The attackers used the vulnerability to pop a web shell on the server weeks later, and managed to retain access for more than two months, the House panel found, and were able to pivot through the company's various systems by obtaining an unencrypted file of passwords on one server, letting the hackers access more than 48 databases containing unencrypted consumer credit data.During that time, the hackers sent more than 9,000 queries on the databases, downloading data on 265 separate occasions.

https://techcrunch.com/2018/12/10/equifax-breach-preventable-house-oversight-report/

The Solution (Take 3)

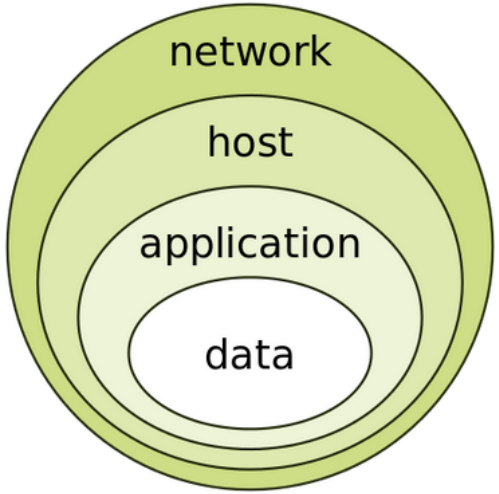
- ✓ Ensure all appropriate patches have been applied.
- ✓ Practice the principle of least privilege.
- Apply defense-in-depth.

Defense in depth [edit]

https://en.wikipedia.org/wiki/Information_security

Main article: Defense in depth (computing)

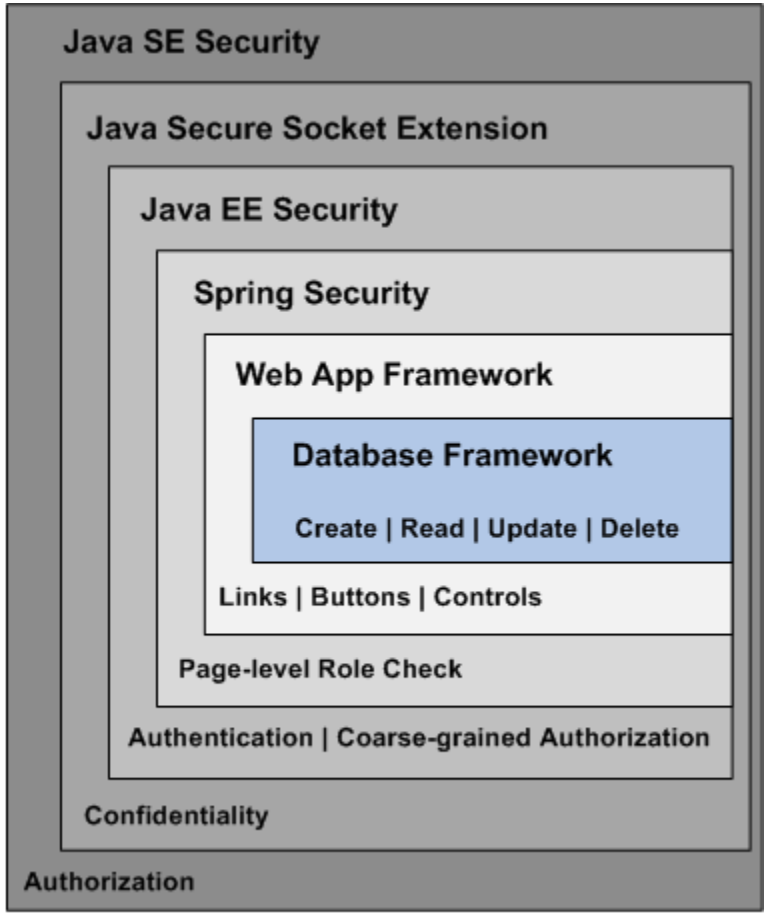
Information security must protect information throughout the life span of the information, from the initial creation of the information on through to the final disposal of the information. The information must be protected while in motion and while at rest. During its lifetime, information may pass through many different information processing systems and through many different parts of information processing systems. There are many different ways the information and information systems can be threatened. To fully protect the information during its lifetime, each component of the information processing system must have its own protection mechanisms. The building up, layering on and overlapping of security measures is called **defense in depth**. In contrast to a metal chain, which is famously only as strong as its weakest link, the defense-in-depth aims at a structure where, should one defensive measure fail, other measures will continue to provide protection.



The onion model of defense in depth

Java Web Security Layers

1. Java SE Security
2. Java Secure Socket Extension (JSSE)
3. Java EE Security
4. Spring Security
5. Web App Framework
6. Database Framework



Each with a specific purpose

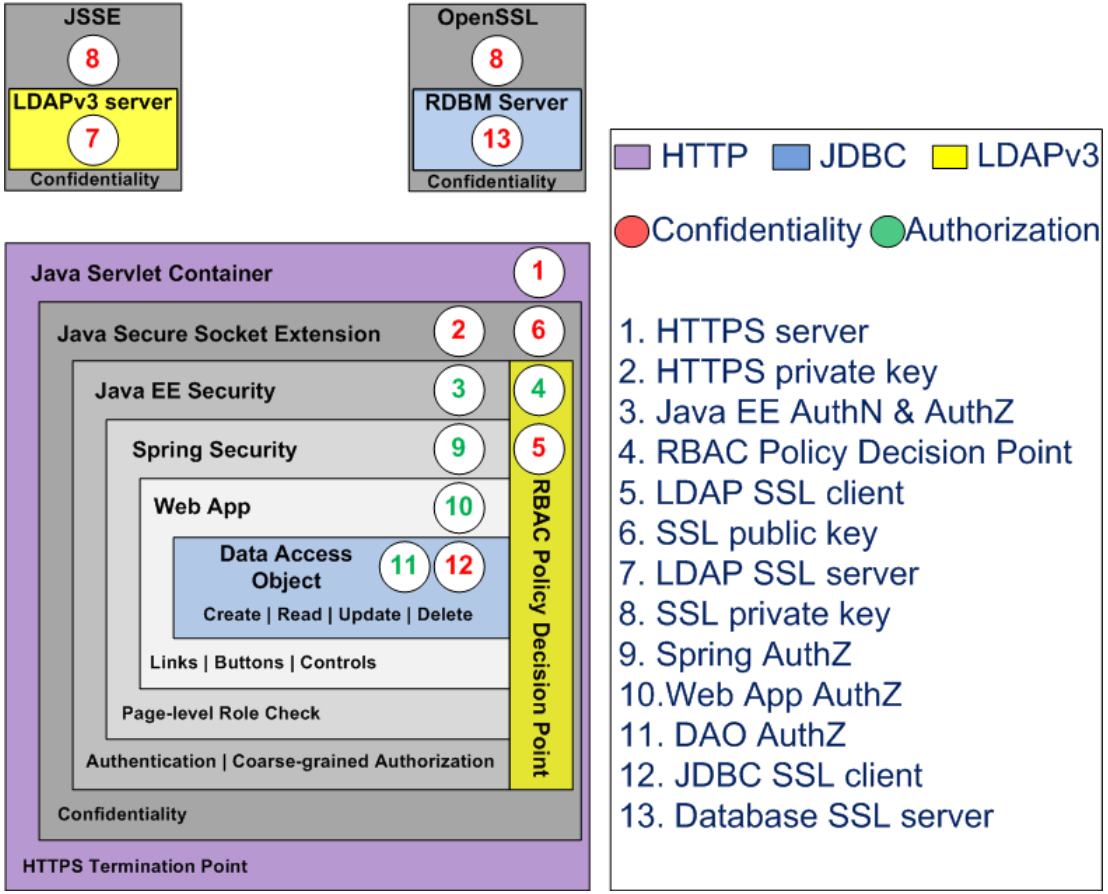
1. Java SE Security ---------- principle of least privilege
2. JSSE ---------- private conversations
3. Java EE Security ---------- deadbolt on front door
4. Spring Security ---------- locks on room doors
5. Web App Framework - locks on equipment in rooms
6. Database Functions ---- content filtering

Example #2

Apache

Fortress

Demo



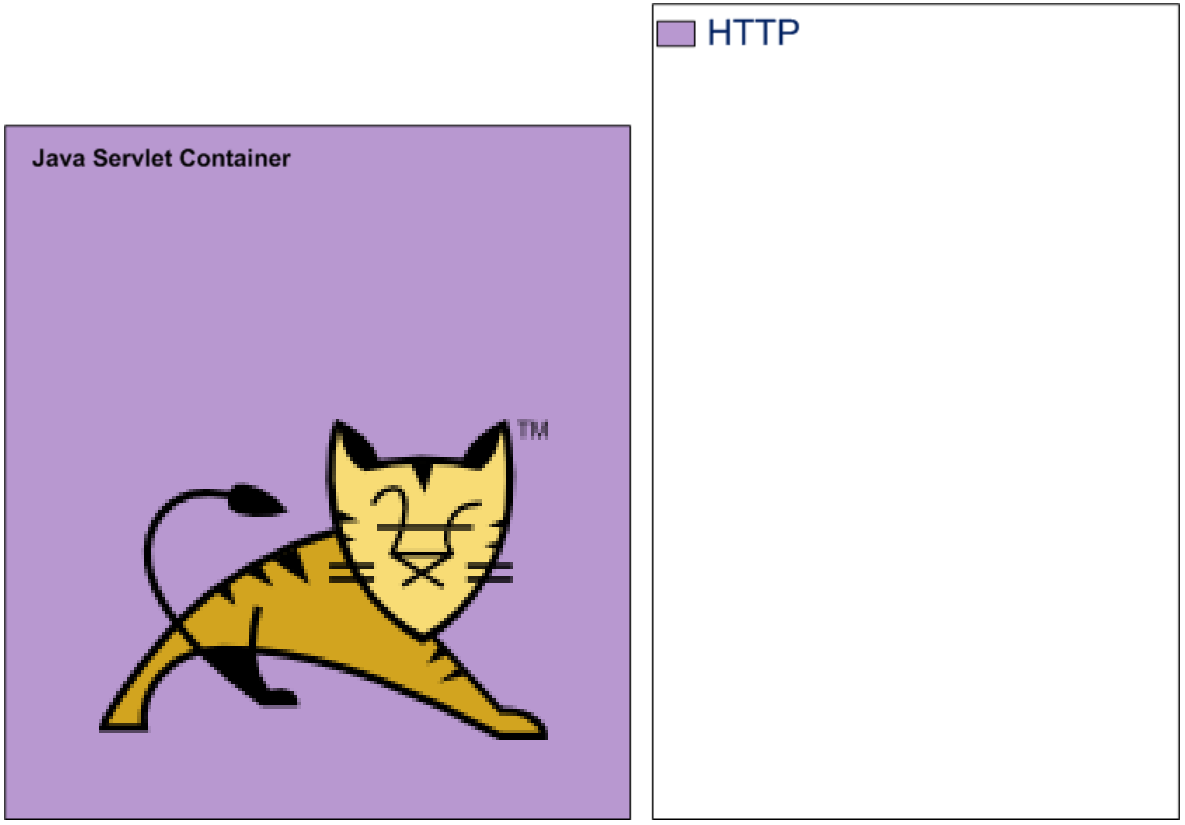
https://github.com/shawnmckinney/apache-fortress-demo

Two Areas of Control

1. JavaSE, JSSE, JavaEE and Spring Declarative controls

2. Programmatic AuthZ controls in the Web and DB layers

Start with Tomcat Servlet Container

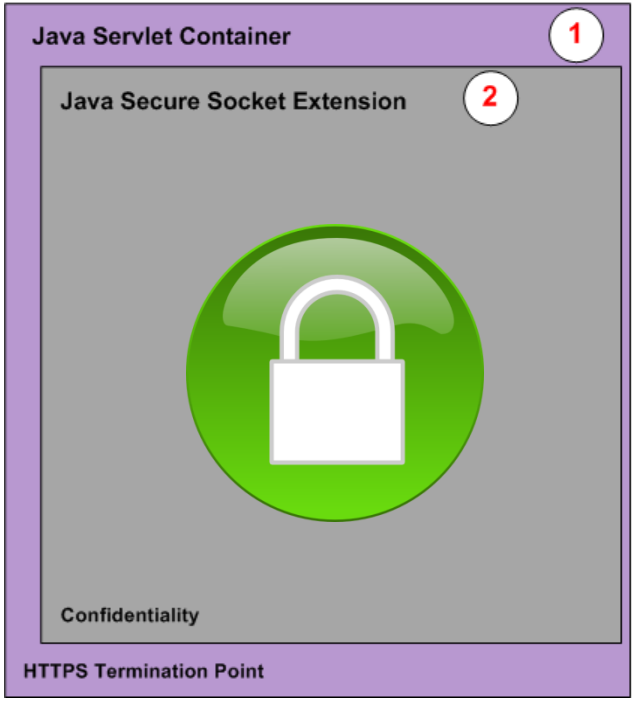


1 & 2. Enable HTTPS

ssssh!!!

1. Update the Server.xml

2. Add private key



HTTP

Confidentiality

1. HTTPS server
2. HTTPS private key

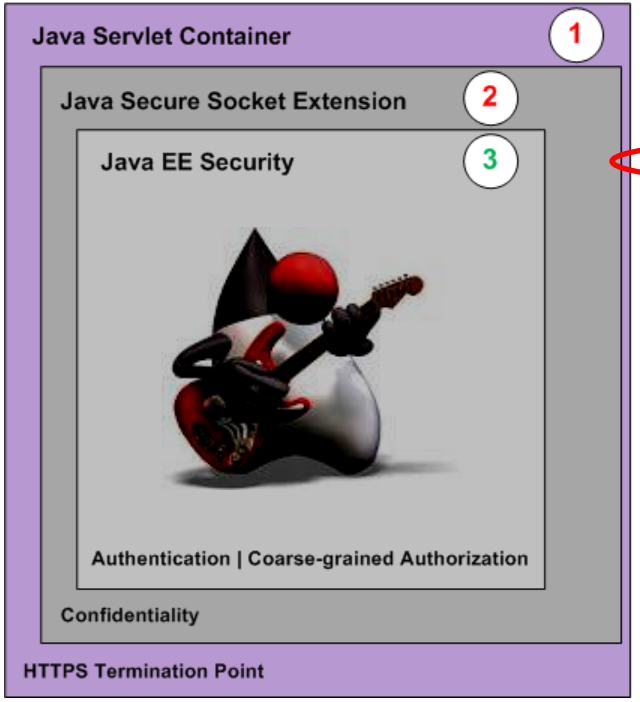
3. Enable Java EE Security the deadbolt

a. Update web.xml

b. Drop the proxy jar

c. Add context.xml

d. Add fortress to pom.xml



Current Specs for Java EE Security

1. JSR-196 – JASPIC - AuthN
2. JSR-115 – JAAC - AuthZ
3. JSR-375 – JavaEE Security API

What is a Realm?

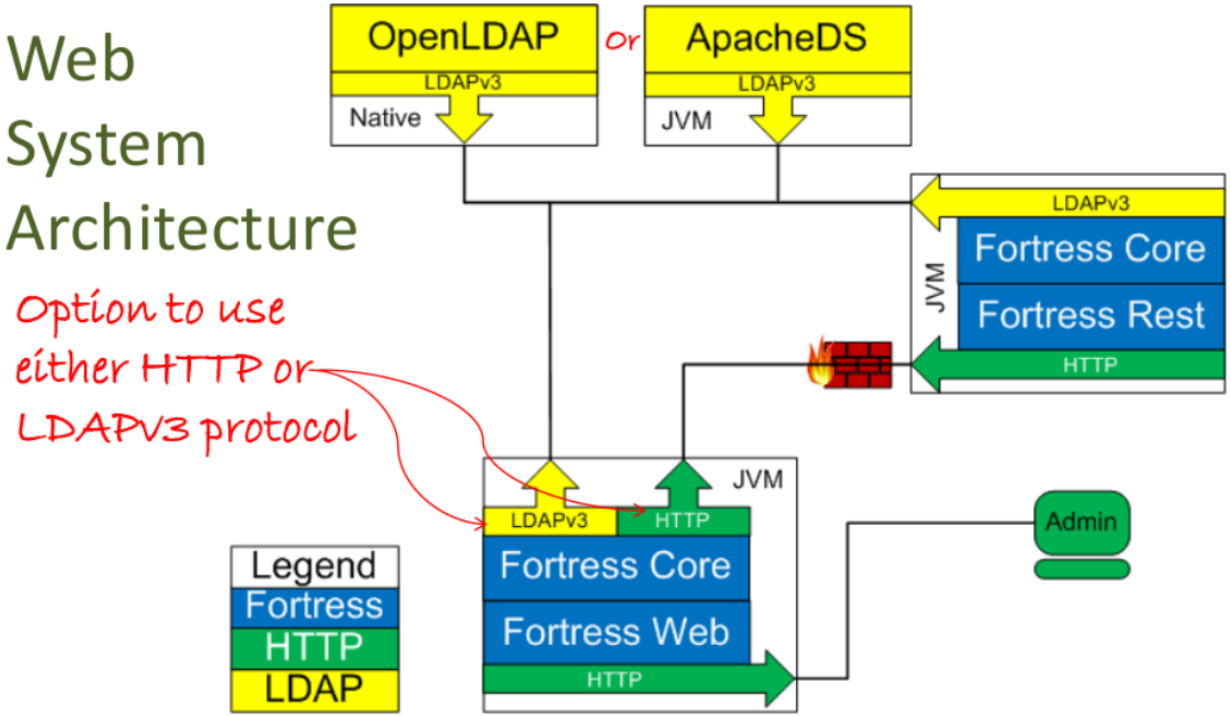
A Realm is a "database" of usernames and passwords that identify valid users of a web application (or set of web applications), plus an enumeration of the list of roles associated with each valid user.

https://tomcat.apache.org/tomcat-9.0-doc/realm-howto.html

Apache Fortress™

Access Management SDK and Web Components

A standards-based access management system, written in Java, supports ANSI INCITS 359 RBAC and more.

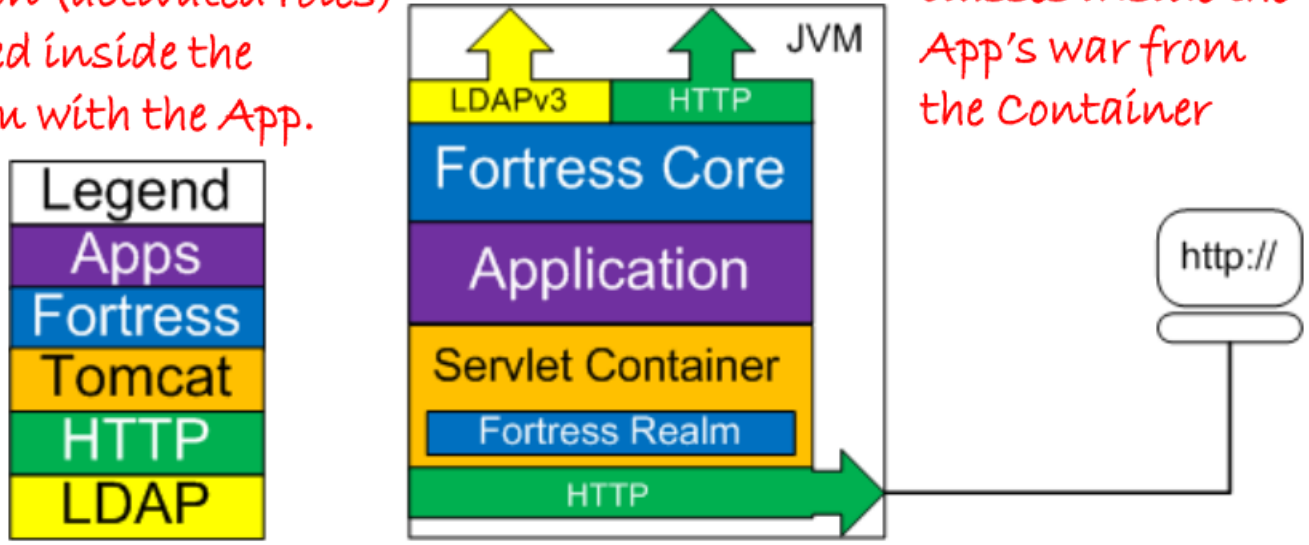


Apache Fortress Context Realm

Realm Context System Architecture

Shares the RBAC session (activated roles) created inside the Realm with the App.

Isolates Fortress classes inside the App's war from the Container



Enable Fortress Tomcat Realm

Add to App's Web.xml

```
<security-constraint>
  <display-name>My Project Security Constraint</display-name>
  <web-resource-collection>
    <web-resource-name>Protected Area</web-resource-name>
    <url-pattern>/wicket/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>DEMO2_USER</role-name>
  </auth-constraint>
</security-constraint>
<login-config>
  <auth-method>FORM</auth-method>
  <realm-name>MySecurityRealm</realm-name>
  <form-login-config>
<form-login-page>/login/login.html</form-login-page>
```

1. Java EE container protects this URL Automatically.

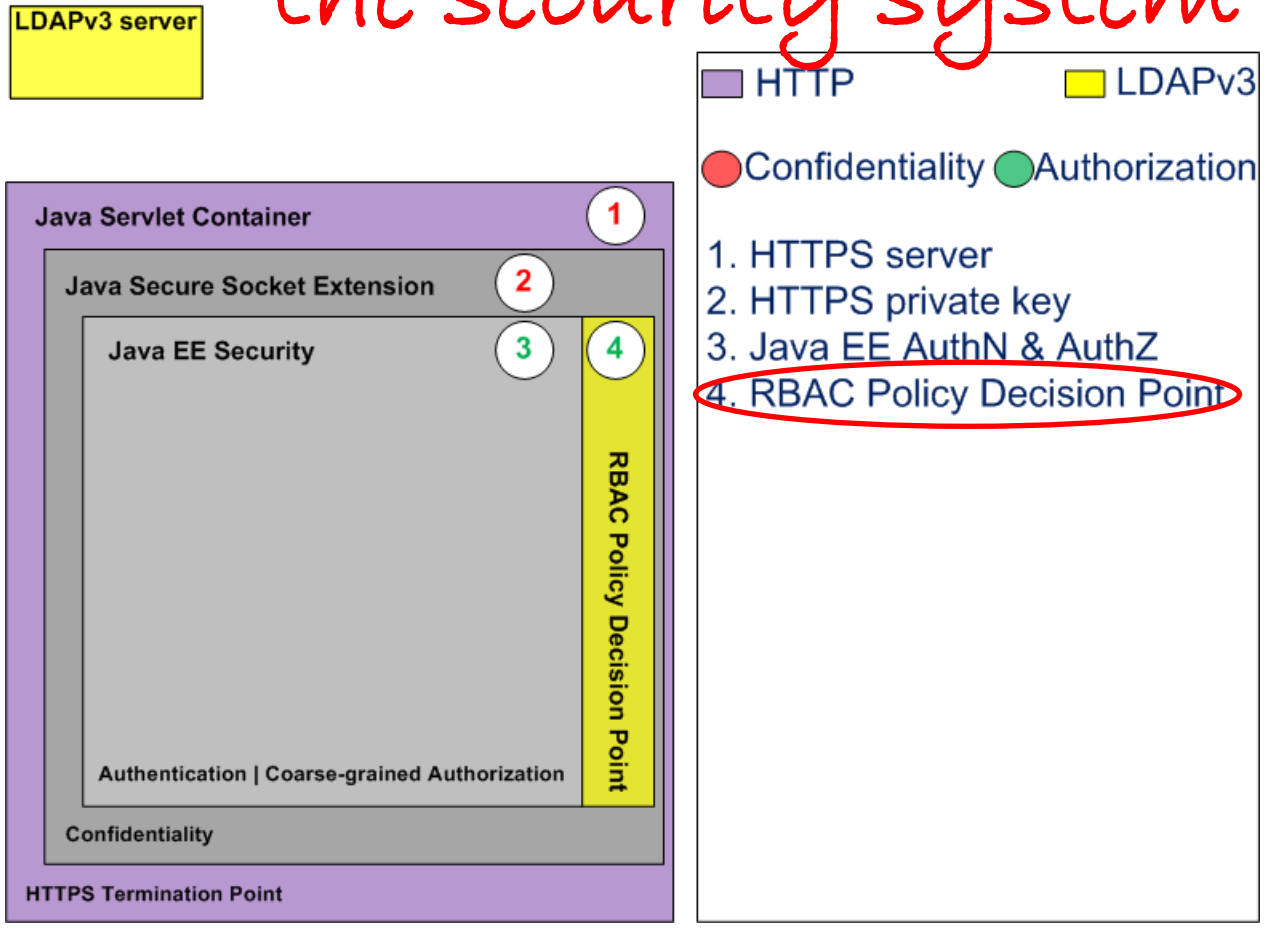
2. All users must have this role to gain entry.

3. Route un-authN requests to my form.

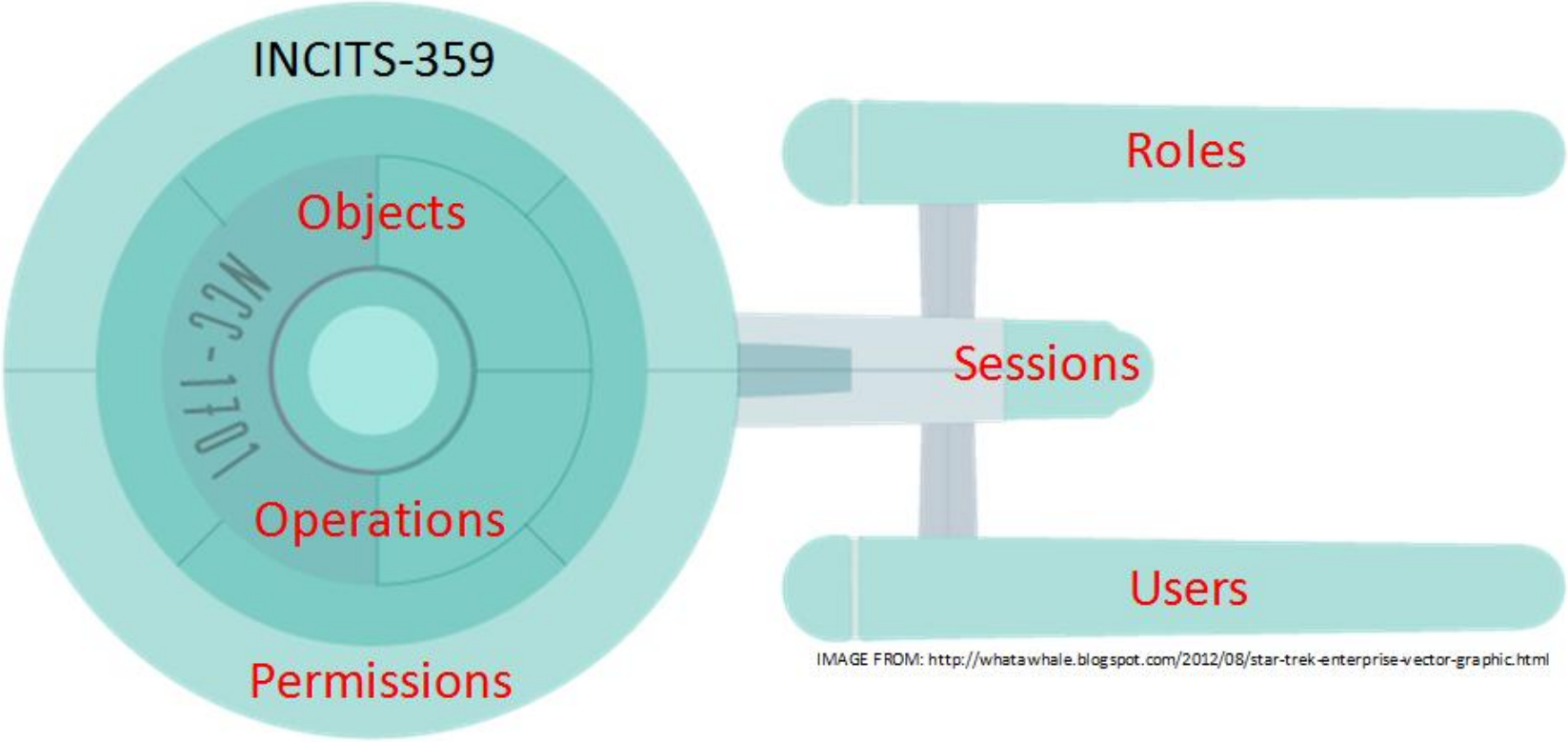
https://github.com/shawnmckinney/apache-fortress-demo/blob/master/src/main/webapp/WEB-INF/web.xml

4. Setup Policy Decision Point

the security system



Intro to the RBAC Standard



Early Years

- The Role-Based Access Control model was formally introduced in 1992 by David Ferraiolo and Richard Kuhn of National Institute of Standards and Technology.

- Their model, already in use for some time, was meant to address critical shortcomings of the Discretionary Access Control. DAC was not meeting the needs of non-DoD organizations.

- In particular integrity was lacking, defined by them, as the requirement for data and process to be modified only in authorized ways by authorized users.

Middle Years

- Eight years later, in 2000, they teamed with Ravi Sandhu and produced another influential paper entitled ‘The NIST Model for a Role-Based Access Control: Towards a Unified Standard’.

- Later the team released the RBAC formal model. One that laid out in discrete terms how these types of systems were to work. The specifications, written in Z-notation, left no ambiguity whatsoever.

- This model formed the basis for the standard that followed:
  - ANSI INCITS 359

Current Years

- INCITS 359-2012 RBAC also known as Core.
- INCITS 494-2012 RBAC Policy Enhanced allows attribute modifiers on permissions specifically to provide support for fine-grained authorization.

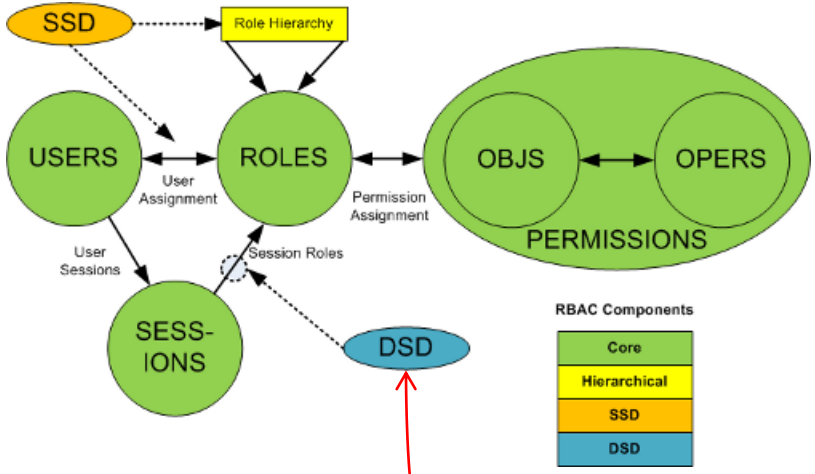
Use ANSI RBAC INCITS 359 Specification

RBAC0:
- Users, Roles, Perms, Sessions

RBAC1:
- Hierarchical Roles

RBAC2:
- Static Separation of Duties

RBAC3:
- Dynamic Separation of Duties



Today we demo this

Use RBAC Object Model

Six basic elements:

1. **User** – human or machine entity
2. **Role** – a job function within an organization
3. **Object** – maps to system resources
4. **Operation** – executable image of program
5. **Permission** – approval to perform an Operation on one or more Objects
6. **Session** – contains set of activated roles for User

Use RBAC Functional Model

APIs form three standard interfaces:

1. Admin – Add, Update, Delete
2. Review – Read, Search
3. System – Access Control

Management and Config processes

Demo runtime processes

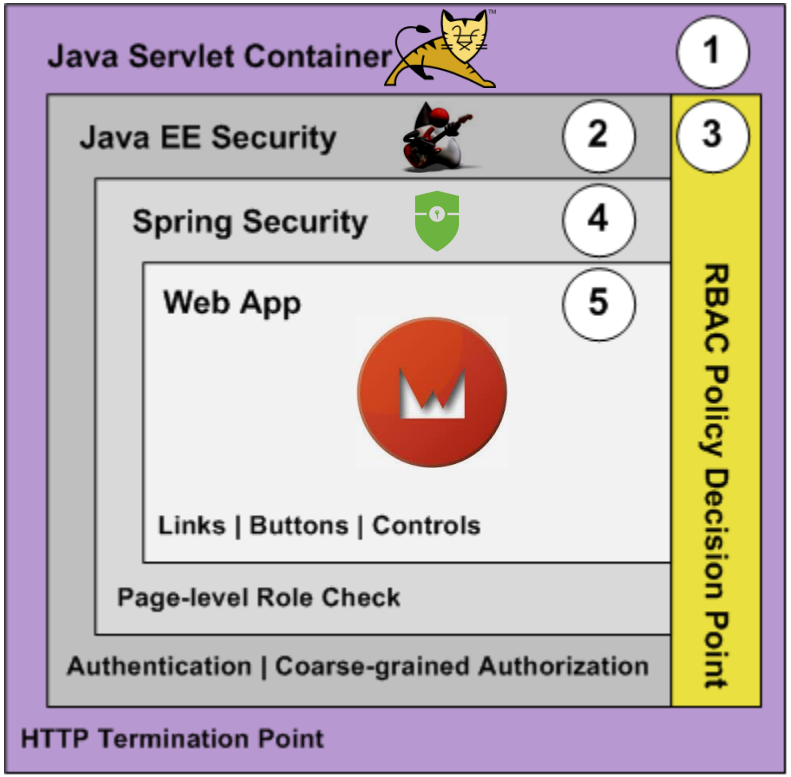
Use RBAC Functional Model

System Manager APIs:

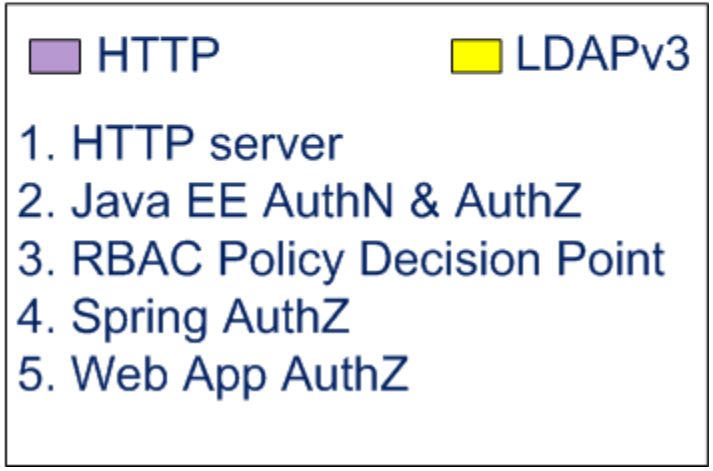
http://directory.apache.org/fortress/gen-docs/latest/apidocs/org/apache/directory/fortress/core/impl/AccessMgrImpl.html

1. createSession – authenticate, activate roles
2. checkAccess – permission check
3. sessionPermissions – all perms active for user
4. sessionRoles – return all roles active
5. addActiveRole – add new role to session
6. dropActiveRole – remove role from session

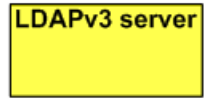
Example #3 : Role Engineering Sample

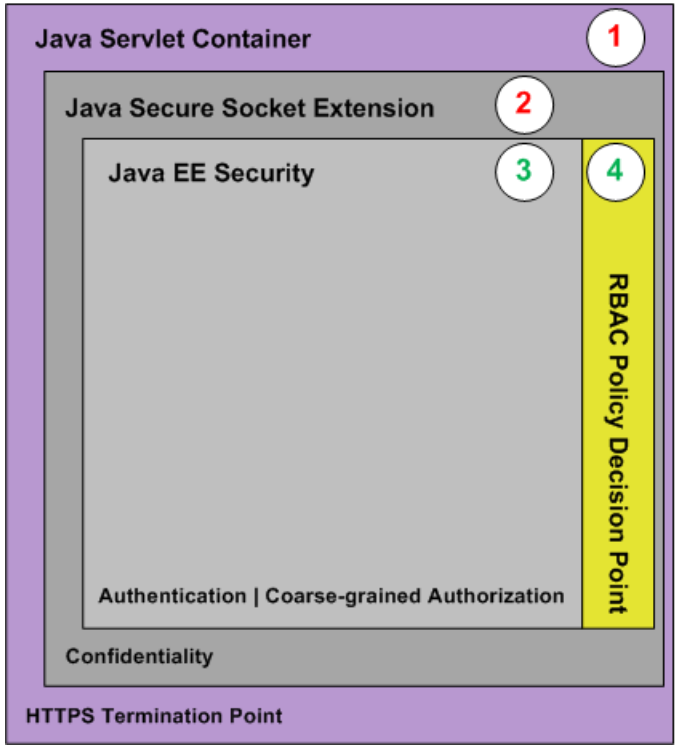


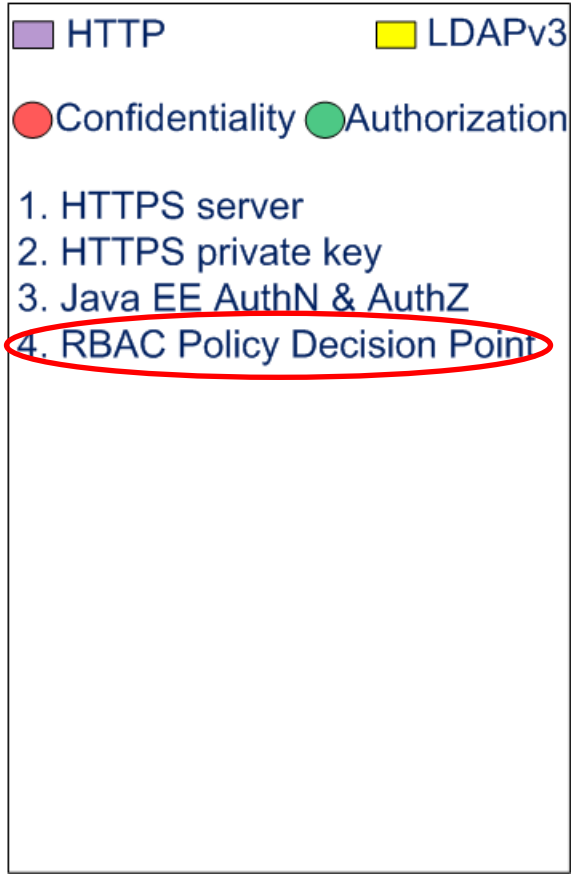
https://github.com/shawnmckinney/role-engineering-sample



4. Back to Installing a policy decision point

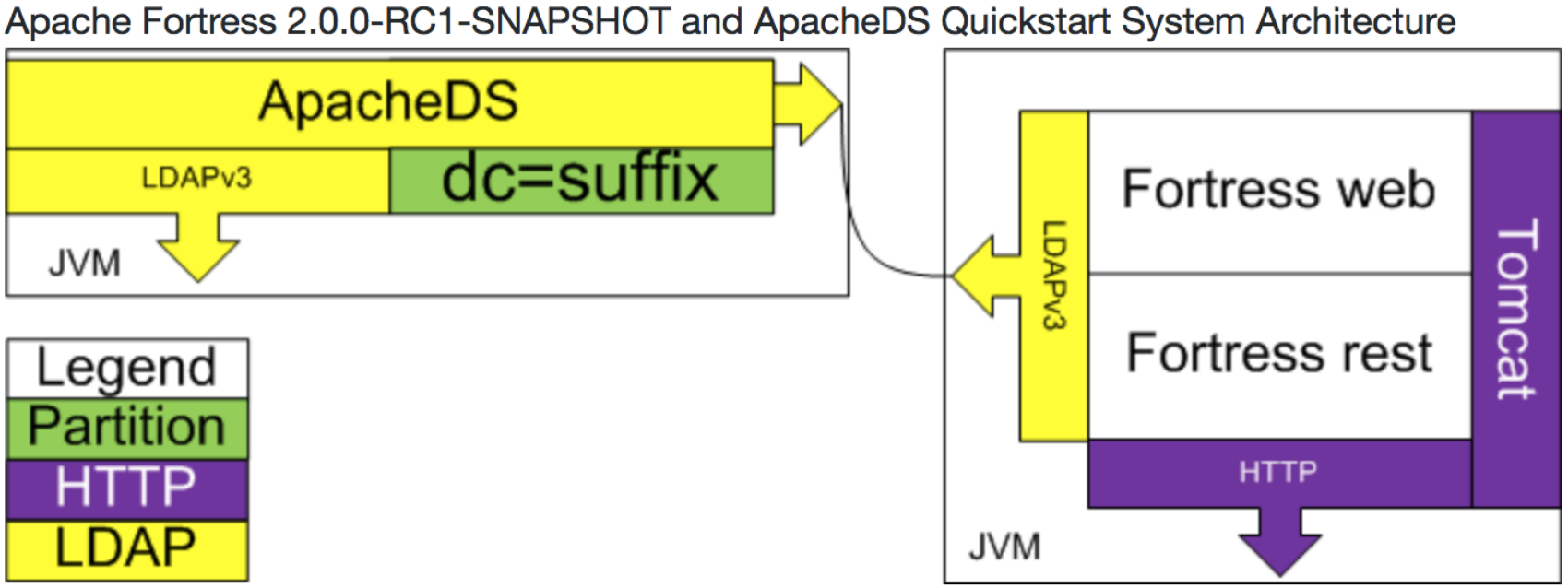






Use

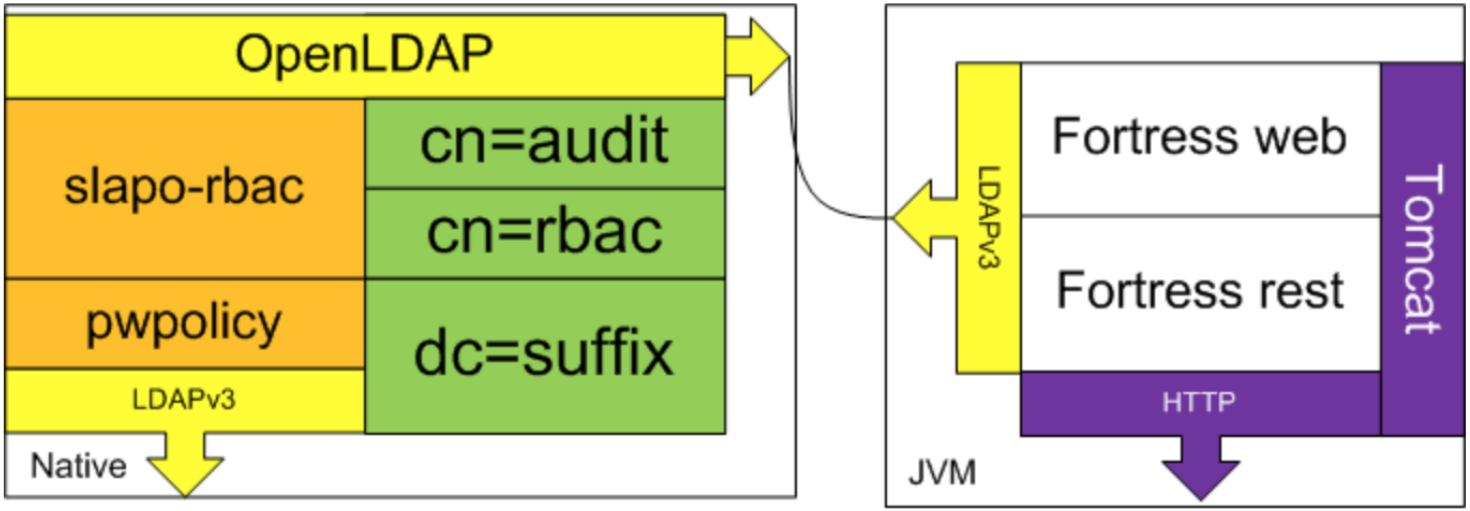
ApacheDS & Fortress QUICKSTART



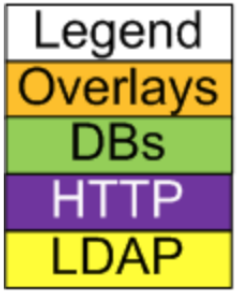
https://github.com/apache/directory-fortress-core/blob/master/README-QUICKSTART-APACHEDS.md

Or OpenLDAP & Fortress QUICKSTART

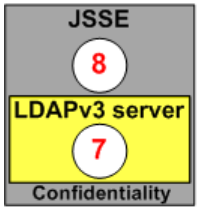
Apache Fortress 2.0.0-RC2 and OpenLDAP Quickstart System Architecture



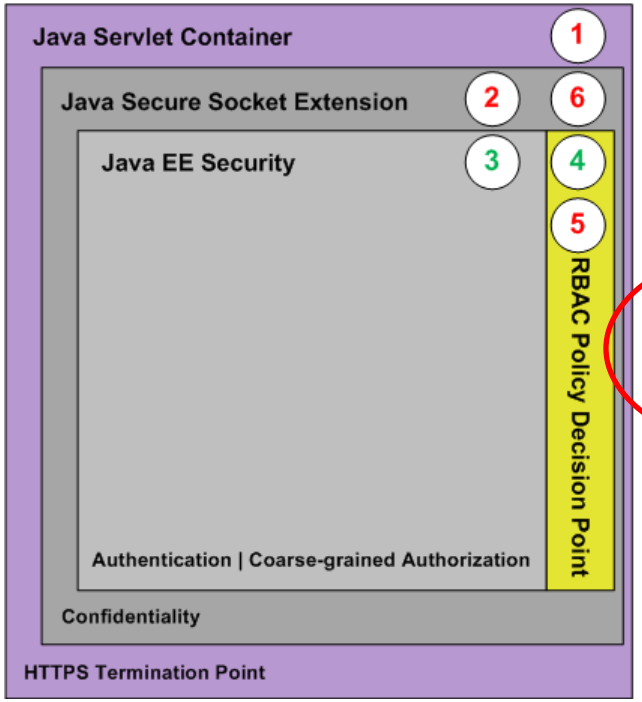
https://github.com/apache/directory-fortress-core/blob/master/README-QUICKSTART-SLAPD.md

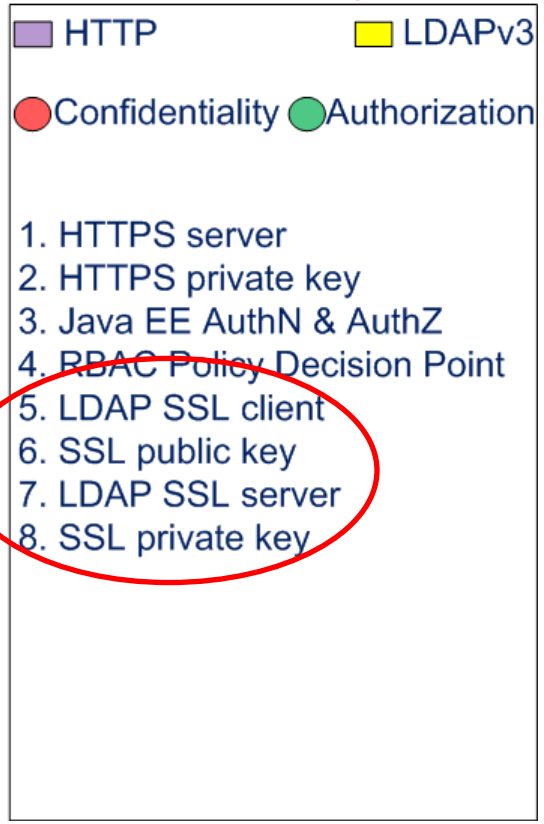


5 – 8 Enable LDAP SSL



confidentiality

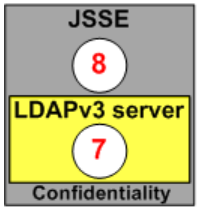




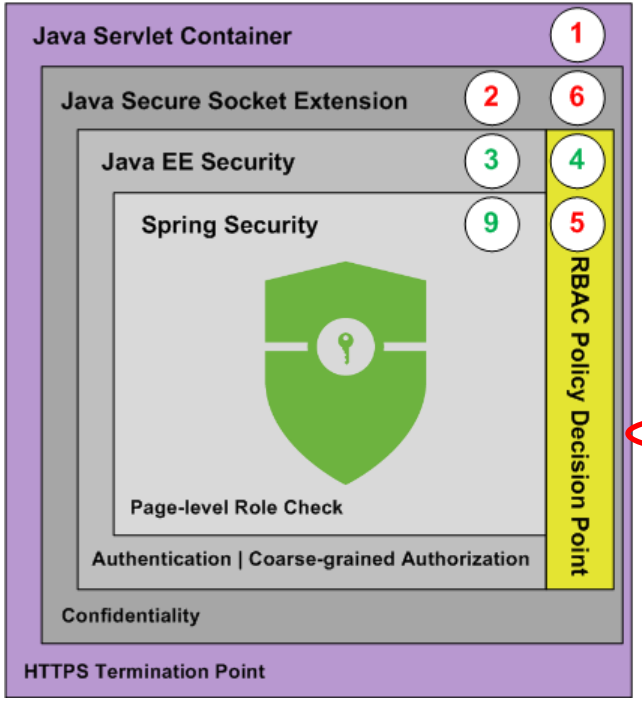
9. Enable Spring Security

a. Authorization

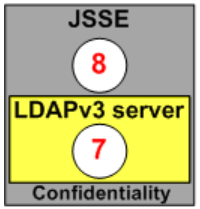
b. Role mapping



locks on the rooms



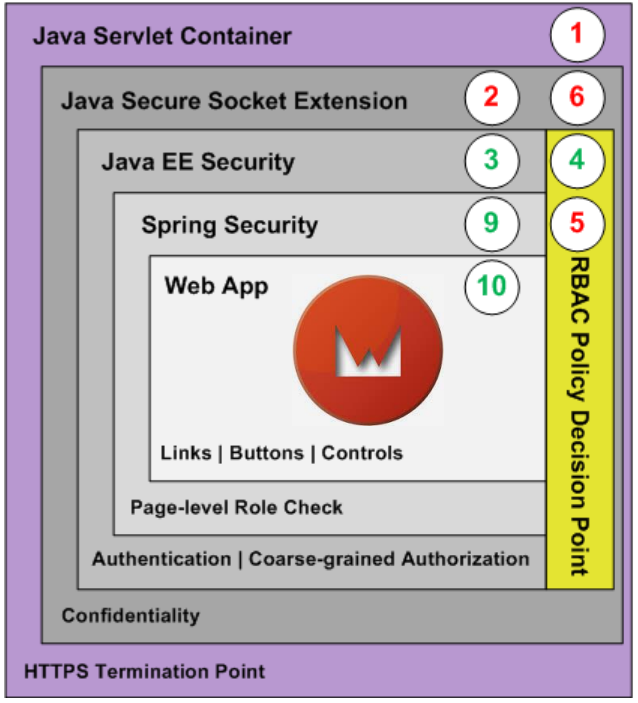
10. Web App Authorization

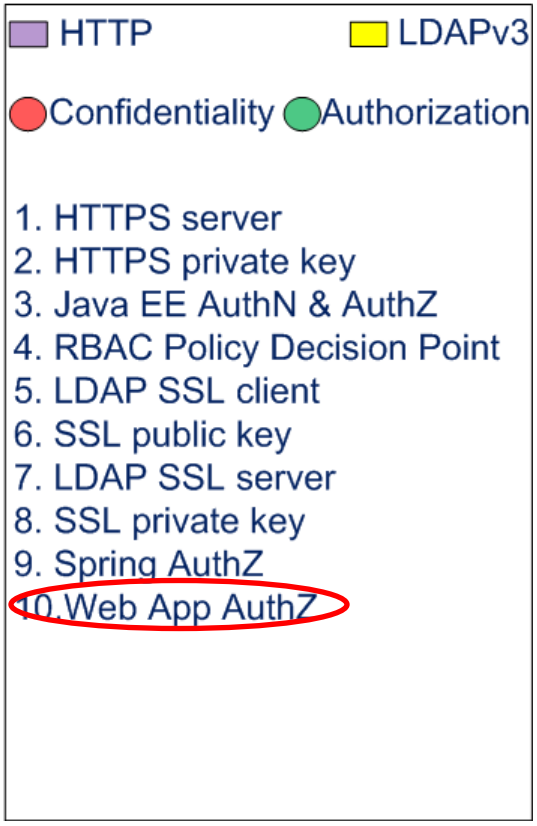


locks on equipment

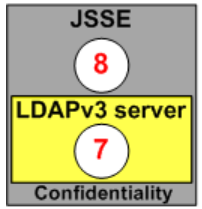
Add fine-grained checks:

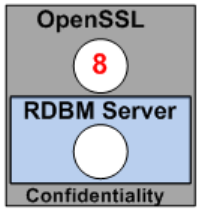
a. Page links
b. Buttons
c. Other controls



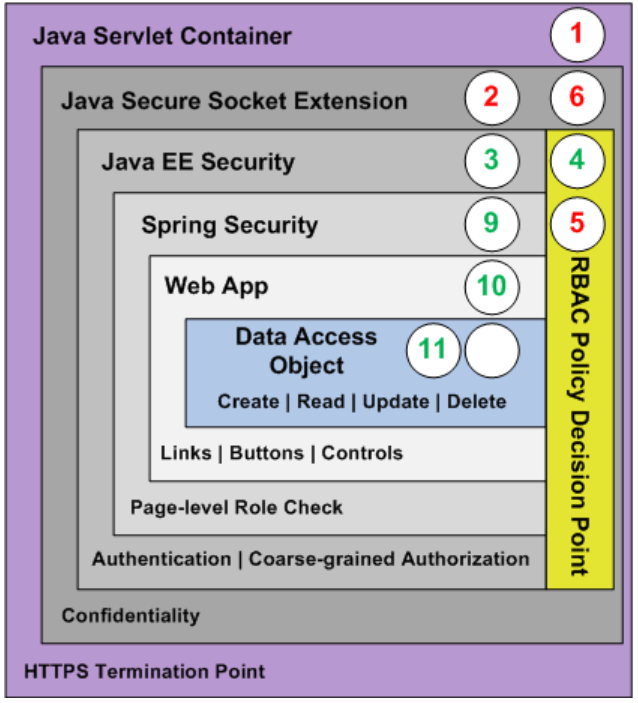


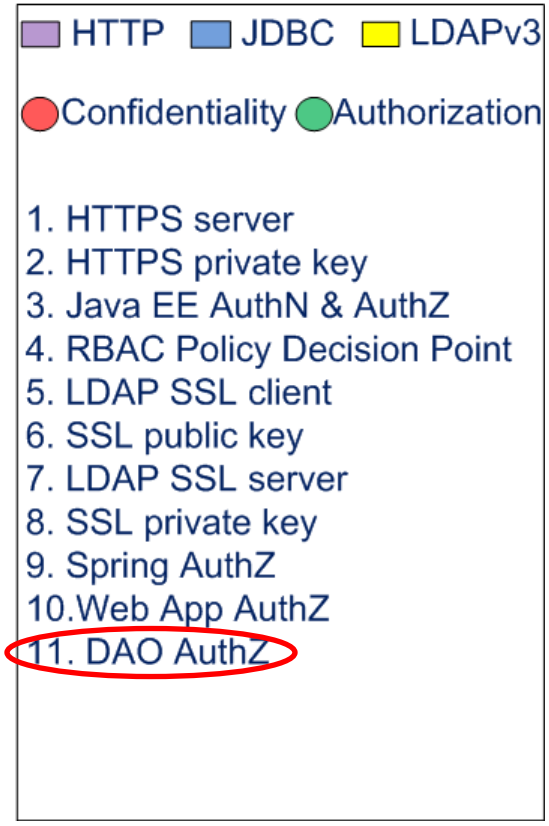
11. DAO Authorization





filtering





Add fine-grained Checks to:

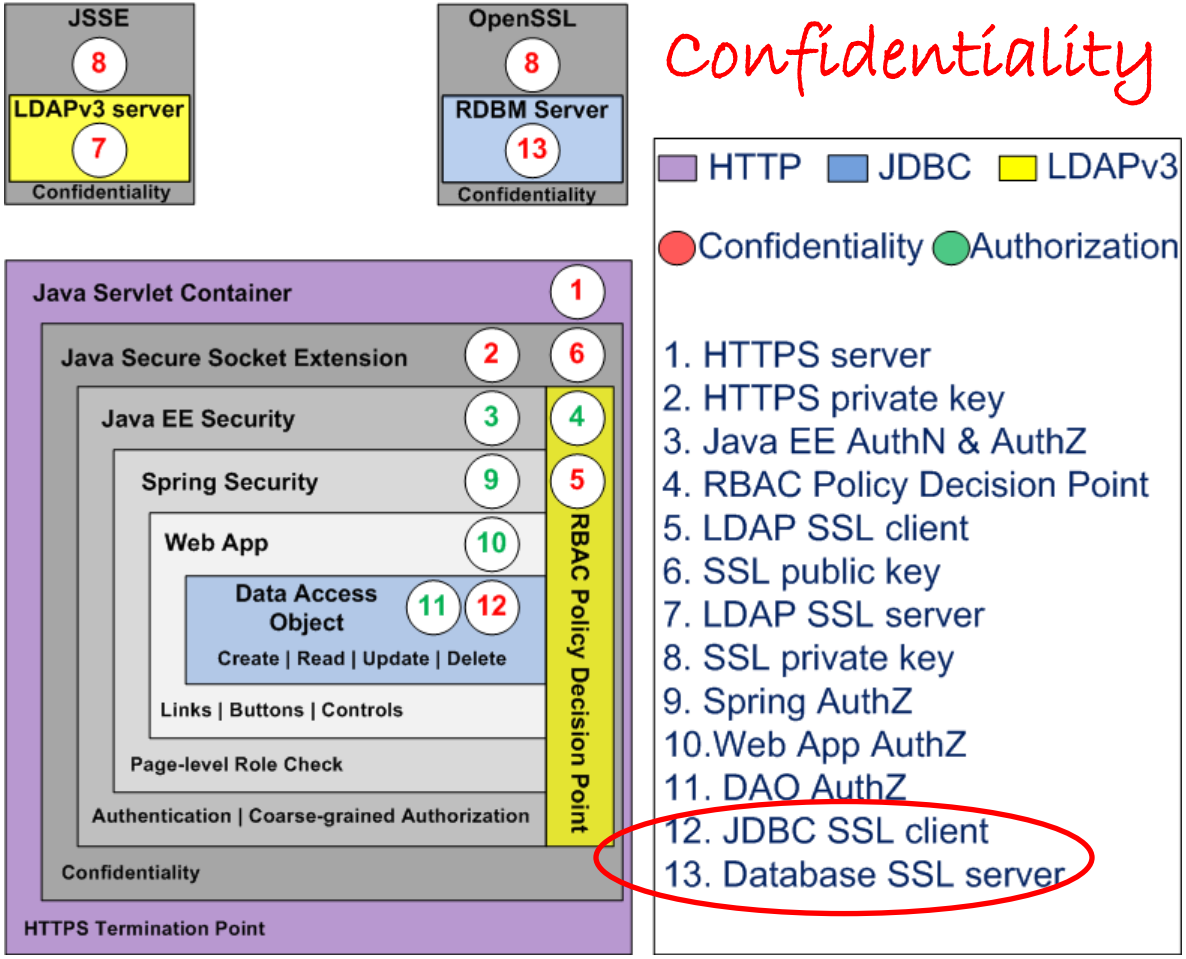
a. Create
b. Read
c. Update
d. Delete

symas

12, 13. Enable DB SSL

12. Client
 a. public key
 b. config

13. Server
 a. private key
 b. config



Apache Fortress Demo

- Three Pages and Three Customers
- One role for every page to customer combo
- Users may be assigned to one or more roles
- One and only one role may be activated

Pages	Customer 123	Customer 456	Customer 789
Page One	PAGE1_123	PAGE1_456	PAGE1_789
Page Two	PAGE2_123	PAGE2_456	PAGE2_789
Page Three	PAGE3_123	PAGE3_456	PAGE3_789

User123	Customer 123	Customer 456	Customer 789
Page1	True	False	False
Page2	True	False	False
Page3	True	False	False

User1	Customer 123	Customer 456	Customer 789
Page1	True	True	True
Page2	False	False	False
Page3	False	False	False

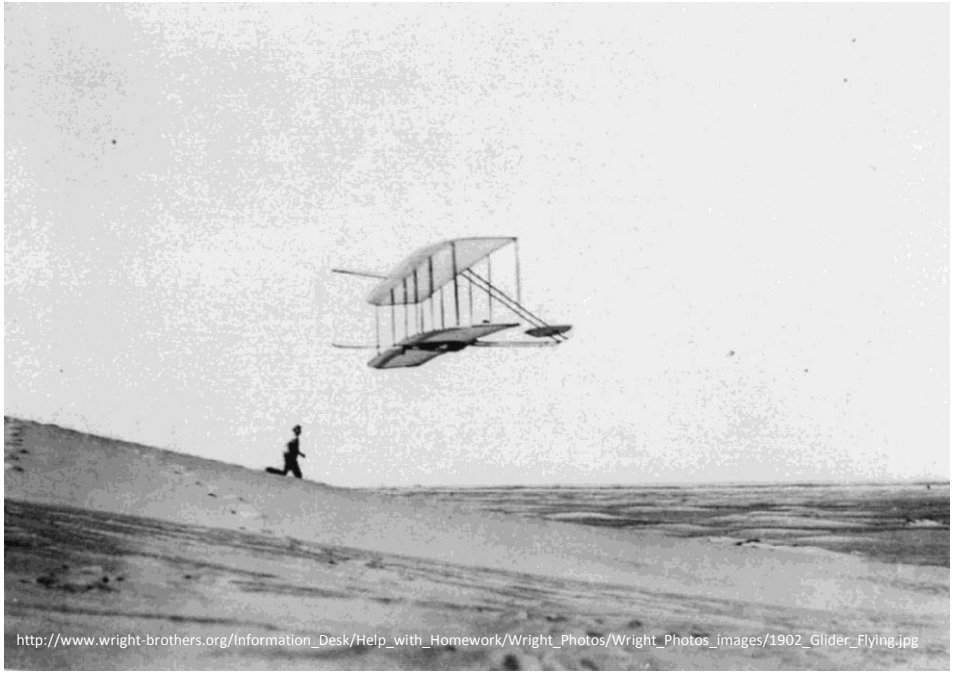
User1_123	Customer 123	Customer 456	Customer 789
Page1	True	False	False
Page2	False	False	False
Page3	False	False	False

User456	Customer 123	Customer 456	Customer 789
Page1	False	True	False
Page2	False	True	False
Page3	False	True	False

User2	Customer 123	Customer 456	Customer 789
Page1	False	False	False
Page2	True	True	True
Page3	False	False	False

User2_123	Customer 123	Customer 456	Customer 789
Page1	False	True	False
Page2	False	False	False
Page3	False	False	False

RBAC Demo


http://www.wright-brothers.org/Information_Desk/Help_with_Homework/Wright_Photos/Wright_Photos_images/1902_Glider_Flying.jpg

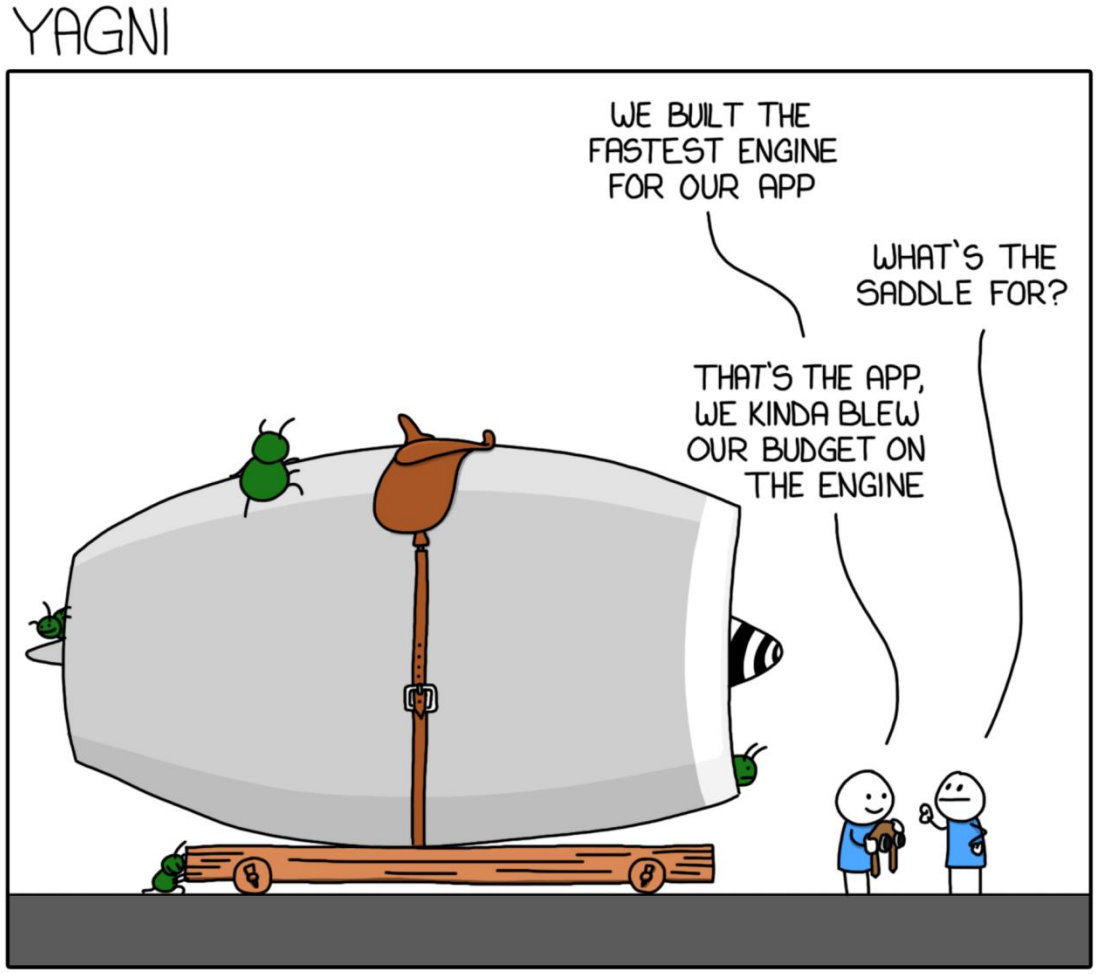
Testing

- Verify security functionality via automation.
- Beware of regressions. Can go unnoticed for weeks, months, years.

https://github.com/shawnmckinney/apache-fortress-demo/.../ApacheFortressDemoSeleniumITCase.java

Performance

- Verify performance of security APIs
- Roundtrips should be < 10ms, better yet < 1ms!
- Link to Apache Fortress JMeter Classes



LDAP Performance

Satisfies the SLAs:

- OpenLDAP
  - Reads/Search/Bind > 75K/second
  - Update/Delete > 10K/second
  - Replication/Highly-Available
  - Audit Trail
  - Runs on most platforms
  - Commercial support options available

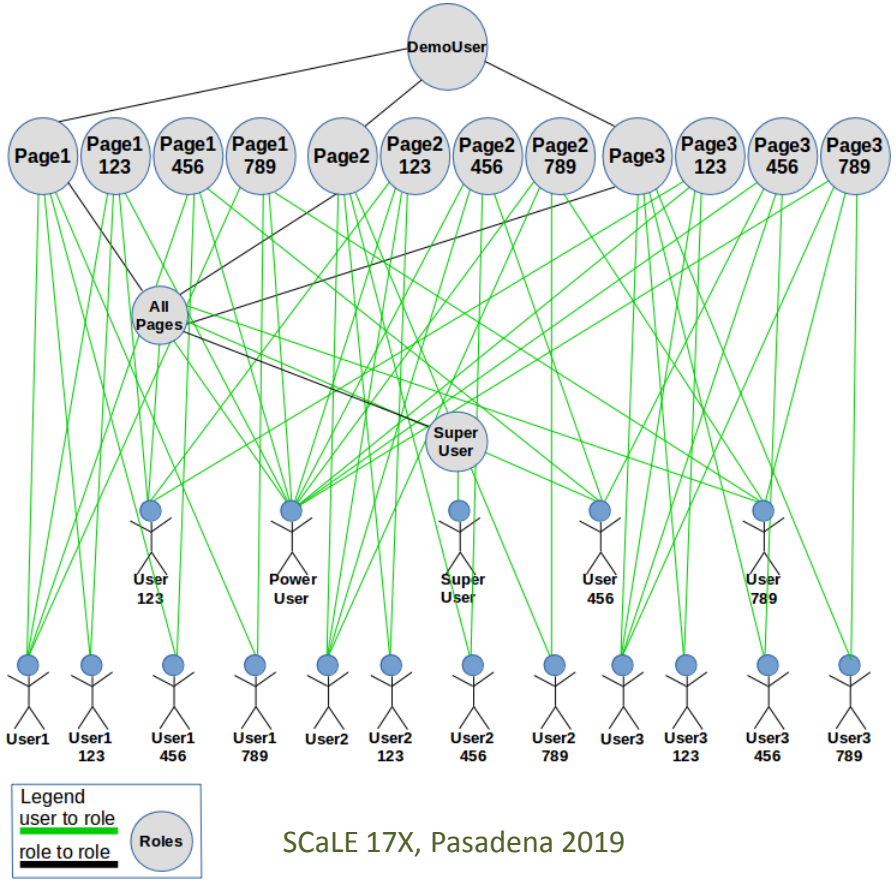
Apache Fortress Demo

- https://github.com/shawnmckinney/apache-fortress-demo

User Foo	Customer 123	Customer 456	Customer 789
Page1	False	True	True
Page2	True	False	False
Page3	True	False	False

We still have a problem…

Roles Have Exploded



Cartesian Product

$A \times B = \{(a,b) \mid a \in A \text{ and } b \in B\}$

- A : role
- B : relationships

Number of Roles = sizeof(A) * sizeof(B)

Roles (A)		Relationships (B)
Role1		Customer 123
Role2	*	Customer 456
Role3		Customer 789

=>

Roles

1. Role1-123
2. Role1-456
3. Role1-789
4. Role2-123
5. Role2-456
6. Role2-789
7. Role3-123
8. Role3-456
9. Role3-789

The Solution

Use attributes to constrain under what conditions roles may be activated.

RBAC only

- ou=Roles (17)
  - cn=PAGE1_123
  - cn=PAGE1_456
  - cn=PAGE1_789
  - cn=PAGE2_123
  - cn=PAGE2_456
  - cn=PAGE2_789
  - cn=PAGE3_123
  - cn=PAGE3_456
  - cn=PAGE3_789

uid	user123
uidNumber	**1237**
description	Apache Fortress Demo User123 Access all Pages for Customer 123
displayName	test2
ftCstr	user123$0$0000$0000$20090101$20990101$none$none$1234567
ftProps	initAttrArrays:
ftRA	PAGE1_123
ftRA	PAGE2_123
ftRA	PAGE3_123
ftRC	PAGE1_123$0$0000$0000$none$none$none$none$all
ftRC	PAGE2_123$0$0000$0000$none$none$none$none$all
ftRC	PAGE3_123$0$0000$0000$none$none$none$none$all

RBAC w/ ABAC

- ou=Roles (23)
  - cn=ABAC_PAGE1
  - cn=ABAC_PAGE2
  - cn=ABAC_PAGE3

uid	user123
uidNumber	**1254**
description	Apache Fortress Demo User123 Access all Pages for Customer 123
displayName	test2
ftCstr	user123$0$$$$$$$
ftProps	initAttrArrays:
ftRA	ABAC_PAGE1
ftRA	ABAC_PAGE2
ftRA	ABAC_PAGE3
ftRC	ABAC_PAGE1$0$$$$$$$
ftRC	abac_page1$type$USER$customer$123$
ftRC	ABAC_PAGE2$0$$$$$$$
ftRC	abac_page2$type$USER$customer$123$
ftRC	ABAC_PAGE3$0$$$$$$$
ftRC	abac_page3$type$USER$customer$123$

symas

Under the Hood

https://appdevcloudworkshop.github.io/images/introduction/image16.png

Role Constraints

```
<roleconstraint role="PAGE1"
  key="customer" … />
<roleconstraint role="PAGE2"
  key="customer" … />
<roleconstraint role="PAGE3"
  key="customer" … />
```

https://github.com/shawnmckinney/fortress-abac-demo/blob/master/src/main/resources/fortress-abac-demo-load-policy.xml

User-Role Constraints

```
<roleconstraint userId="User123" role="PAGE1"
  key="customer" value="123" … />

<roleconstraint userId="User123" role="PAGE2"
  key="customer" value="123" … />

<roleconstraint userId="User123" role="PAGE3"
  key="customer" value="123" … />
```

https://github.com/shawnmckinney/fortress-abac-demo/blob/master/src/main/resources/fortress-abac-demo-load-policy.xml

Code Sample

```
// Nothing new here:
User user = new User("curly");

// This is new:
RoleConstraint constraint = new RoleConstraint( );

// In practice we're not gonna pass hard-coded key-values in here:
constraint.setKey( "customer" );
constraint.setValue( "123" );

// This is just boilerplate goop:
List<RoleConstraint> constraints = new ArrayList();
constraints.add( constraint );

try
{
   // Create the RBAC session with ABAC constraint -- customer=123, asserted:
   Session session = accessMgr.createSession( user, constraints );
   ...
}
```

https://github.com/shawnmckinney/fortress-abac-demo/blob/master/src/main/java/com/mycompany/MyBasePage.java

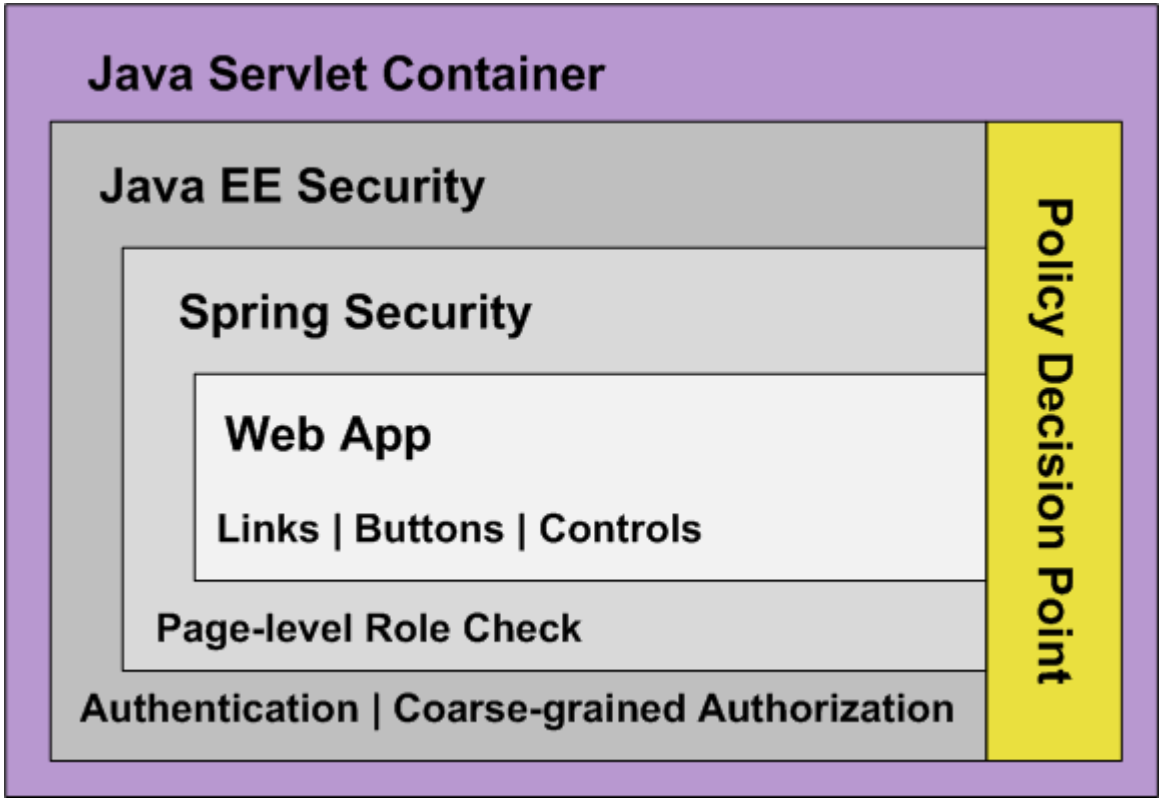
Example #4

Apache

Fortress

ABAC

Demo



https://github.com/shawnmckinney/fortress-abac-demo

ABAC Demo

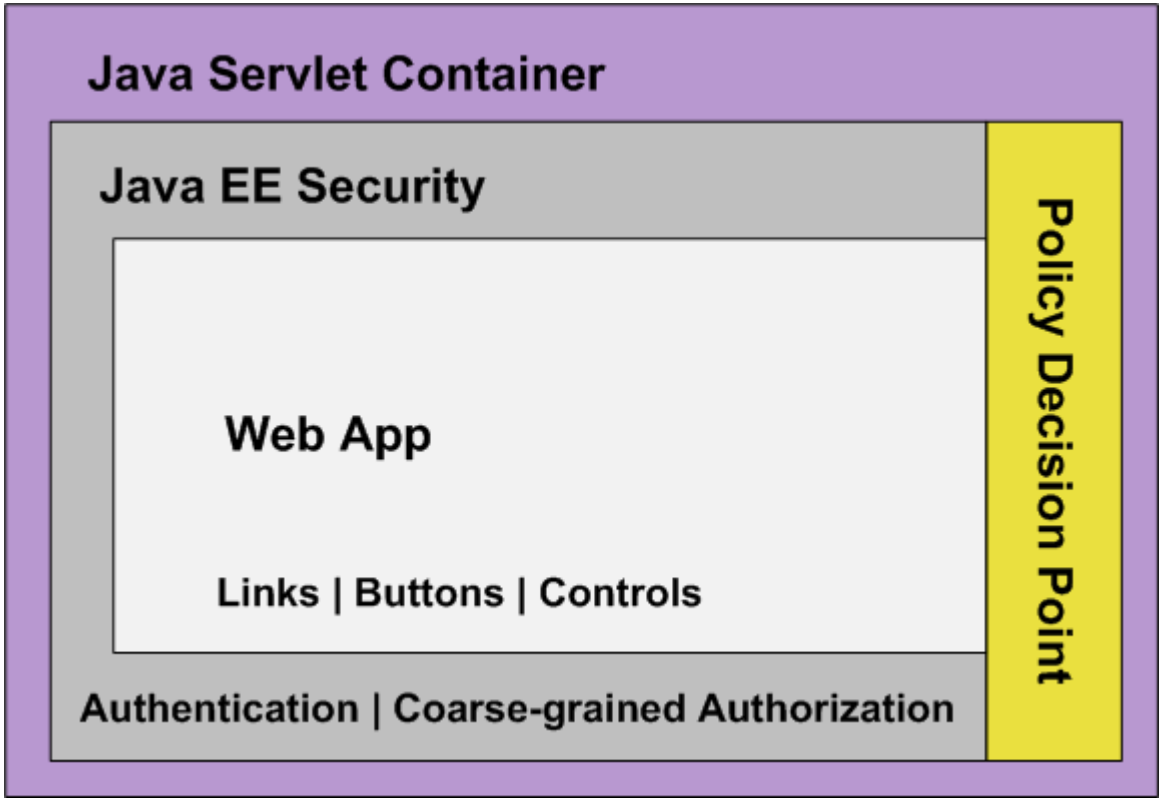

his is the first time you've seen this Stop error screen,
art your computer. If this screen appears again, follow
e steps:
to make sure any new hardware or software is properly installed.
his is a new installation, ask your hardware or software manufactu
any Windows updates you might need.
roblems continue, disable or remove any newly
lled hardware
oftware. Disable BIOS memory options such as
or shadowing.
ou need to use Safe Mode to remov
disable
restart
computer, press F8 to select Ad
Startup
then
ct Safe Mode.
nical information:
STOP: 0x00000050 (0xFD3094C2,0x00000
000)
SPCMDCON.SYS - Address FBFE7617 base at
eStamp 3d6d

Example #5

RBAC

ABAC

Demo



https://github.com/shawnmckinney/rbac-abac-sample

Closing Thoughts

1. Never allow users more than they need to do their jobs
   - *Principle of Least Privilege*
2. Apply security controls across many layers
   - *Defense in Depth*
3. RBAC may be combined with ABAC
   - *Fine-grained Authorization*

Examples

1. https://github.com/shawnmckinney/serial-exploit-sample
2. https://github.com/shawnmckinney/apache-fortress-demo
3. https://github.com/shawnmckinney/role-engineering-sample
4. https://github.com/shawnmckinney/fortress-abac-demo
5. https://github.com/shawnmckinney/rbac-abac-sample

Bonus:

6. https://github.com/shawnmckinney/fortress-saml-demo

Contact Info