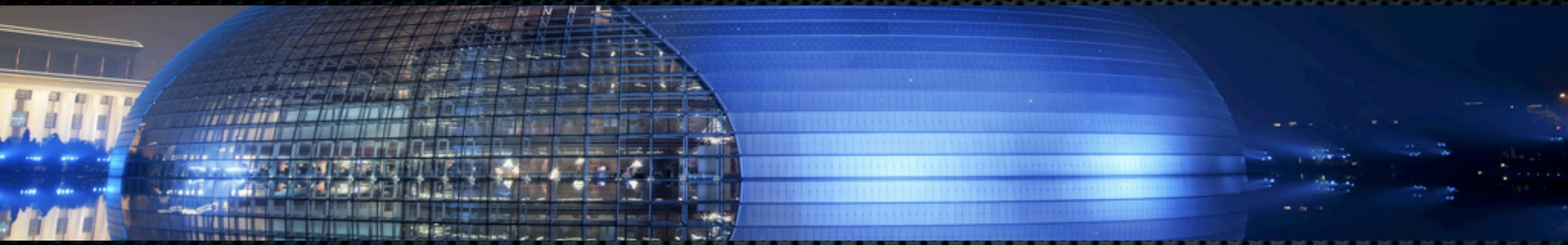


Java Web Application Security

Develop. Penetrate. Protect. Relax.

Matt Raible

http://raibledesigns.com

@mraible

Images by Stuck in Customs - http://www.flickr.com/photos/stuckincustoms

Introductions

- Your experience with web development?
- Your experience with implementing security?
- Have you used Java EE 6, Spring Security or Apache Shiro?
- What do you want to get from this talk?

Blogger on
raibledesigns.com

Father, Skier,
Cyclist

Founder of AppFuse

**Web Framework
Connoisseur**

Who is **Matt Raible**?

Why am I here?

- **Purpose**
  - To learn more about Java webapp security and transform myself into a security expert.
- **Goals**
  - Show how to implement Java webapp security.
  - Show how to penetrate a Java webapp.
  - Show how to fix vulnerabilities.

Session Agenda

- Security Development
  - Java EE 6, Spring Security, Apache Shiro
  - SSL and Testing
- Verifying Security
  - OWASP Top 10 & Zed Attack Proxy
- Commercial Tools and Services
- Conclusion

Develop Penetrate Protect Relax

Develop

Dynamic Language Support?

- If it deploys on Tomcat, it has a web.xml.
  - Grails
  - JRuby on Rails
  - Lift
  - Play! Framework

Java EE 6

- Security constraints defined in web.xml
  - web resource collection - URLs and methods
  - authorization constraints - role names
  - user data constraint - HTTP or HTTPS
- User Realm defined by App Server
- Declarative or *Programmatic* Authentication
- Annotations Support

Java EE 6 Demo

http://www.youtube.com/watch?v=8bXBGU7uo4o

Servlet 3.0

- HttpServletRequest
  - authenticate(response)
  - login(user, pass)
  - logout()
  - getRemoteUser()
  - isUserInRole(name)

Servlet 3.0 and JSR 250

- Annotations
  - @ServletSecurity
  - @HttpMethodConstraint
  - @HttpConstraint
  - @RolesAllowed
  - @PermitAll
  - @DenyAll

Java EE Security Limitations

- No error messages for failed logins
- No Remember Me
- Container has to be configured
- Doesn’t support regular expressions for URLs

Spring Security

- Filter defined in web.xml
- Separate security context file loaded by Spring
  - Defines URLs, Roles and Authentication Providers
  - Defines UserService (provided or custom)
- Password Encoding
- Remember Me

Spring Security Demo

http://www.youtube.com/watch?v=poc5dyImbig

Securing Methods

<global-method-security secured-annotations="enabled"/>

```
@Secured("IS_AUTHENTICATED_ANONYMOUSLY")
public Account readAccount(Long id);

@Secured("IS_AUTHENTICATED_ANONYMOUSLY")
public Account[] findAccounts();

@Secured("ROLE_TELLER")
public Account post(Account account, double amount);
```

<global-method-security jsr250-annotations="enabled"/>

Securing Methods 3.0

```
<global-method-security pre-post-annotations="enabled"/>
```

```
@PreAuthorize("isAnonymous()")
public Account readAccount(Long id);

@PreAuthorize("isAnonymous()")
public Account[] findAccounts();

@PreAuthorize("hasAuthority('ROLE_TELLER')")
public Account post(Account account, double amount);
```

Spring Security Limitations

- Authentication mechanism in WAR
- Securing methods only works on Spring beans
- My remember me example doesn't work

Apache Shiro

- Filter defined in web.xml
- shiro.ini loaded from classpath
  - [main], [urls], [roles]
- Cryptography
- Session Management

Apache Shiro Demo

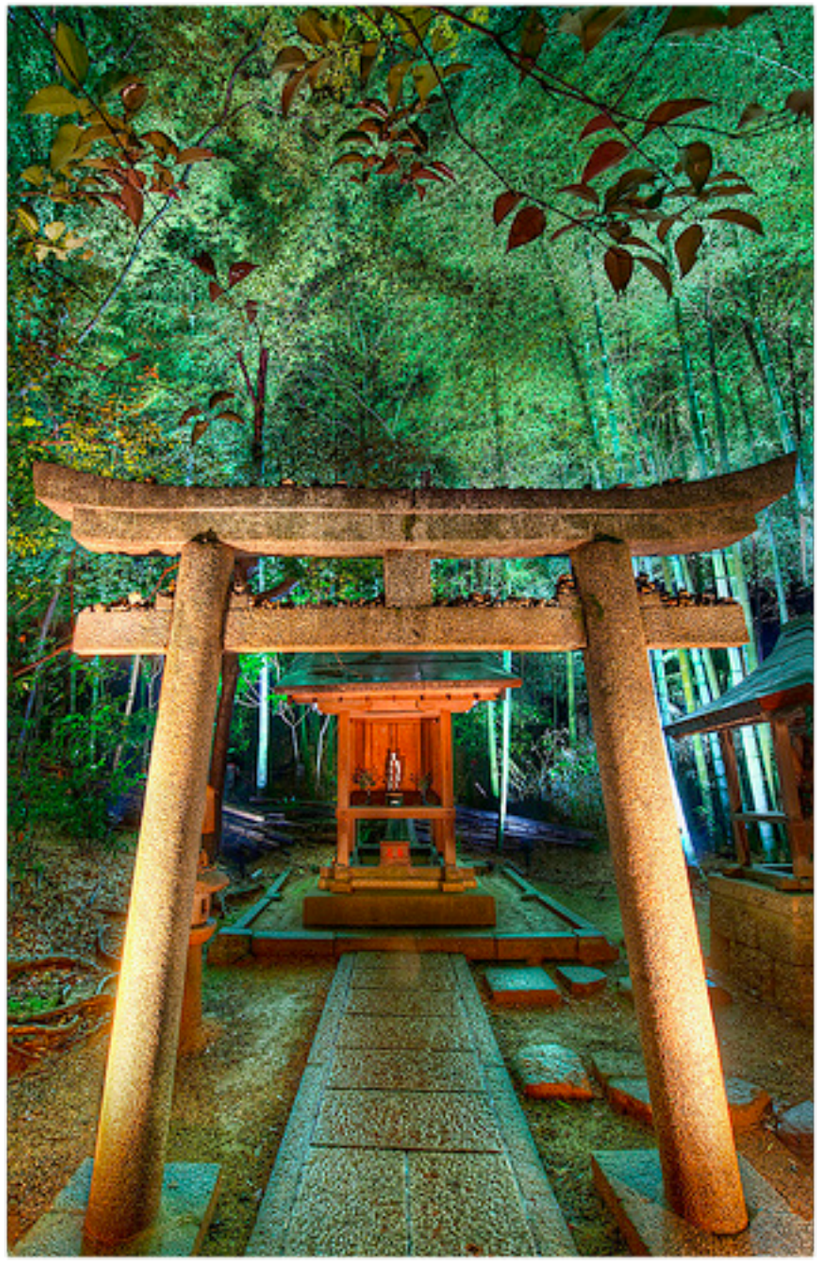
http://www.youtube.com/watch?v=YJByiDvOhsc

Apache Shiro Limitations

- Limited Documentation
- Getting Roles via LDAP not supported
- No out-of-box support for Kerberos
- REST Support needs work

Testing with SSL

- Cargo doesn't support http and https at same time
- Jetty and Tomcat plugins work for both
- Pass javax.net.ssl.trustStore & javax.net.ssl.trustStorePassword to maven-failsafe-plugin as <systemPropertyVariables>



Ajax Login

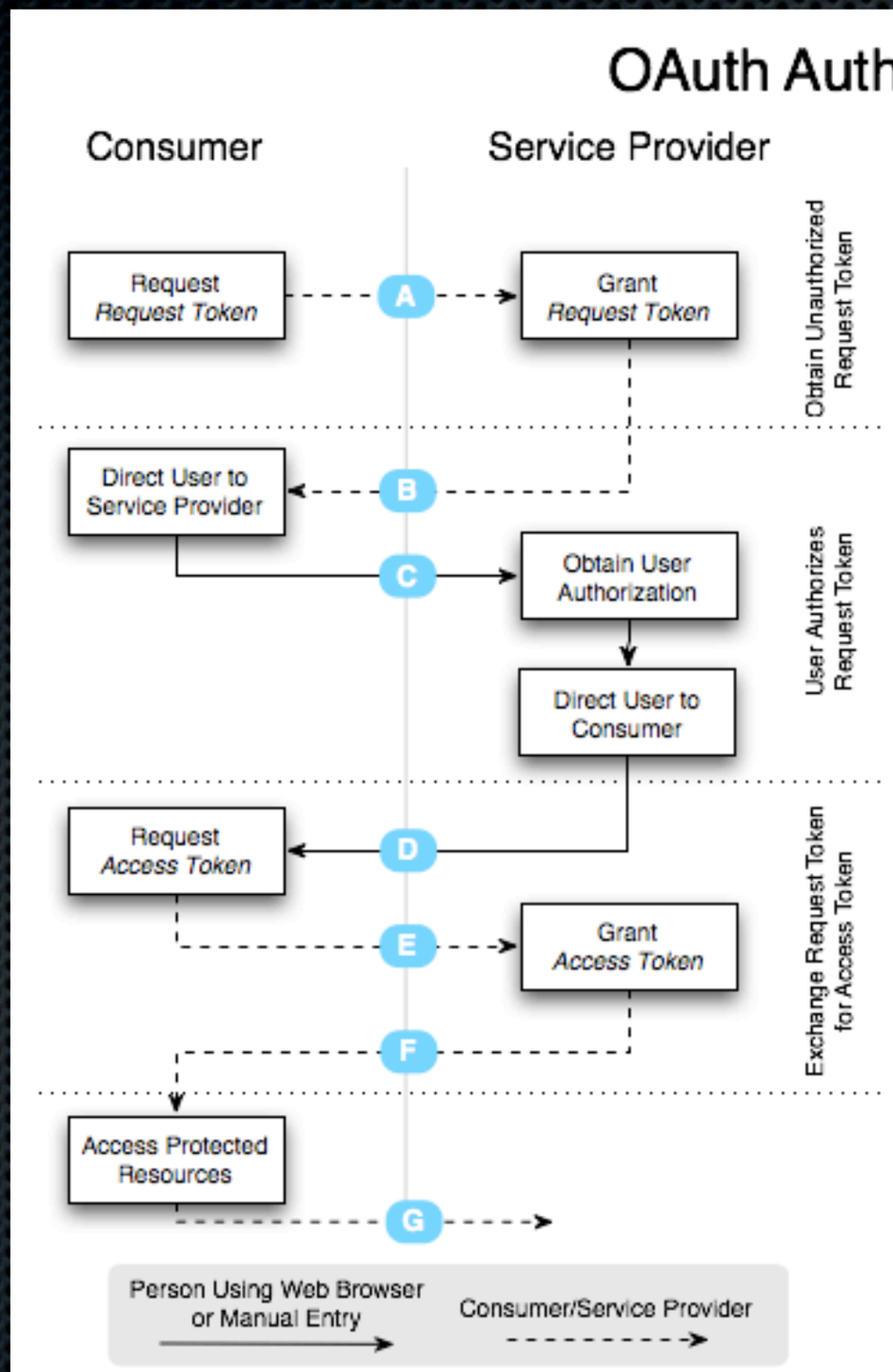
```
package org.appfuse.examples.webapp.security;

import javax.servlet.*;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

public class OptionsHeadersFilter implements Filter {

    public void doFilter(ServletRequest req, ServletResponse res, FilterChain chain)
            throws IOException, ServletException {
        HttpServletResponse response = (HttpServletResponse) res;

        response.setHeader("Access-Control-Allow-Origin", "http://" + req.getServerName());
        response.setHeader("Access-Control-Allow-Methods", "GET,POST");
        response.setHeader("Access-Control-Max-Age", "360");
        response.setHeader("Access-Control-Allow-Headers", "x-requested-with");
        response.setHeader("Access-Control-Allow-Credentials", "true");

        chain.doFilter(req, res);
    }

    public void init(FilterConfig filterConfig) {
    }

    public void destroy() {
    }
}
```

http://raibledesigns.com/rd/entry/implementing_ajax_authentication_using_jquery

Securing a REST API

- Use Basic or Form Authentication
- Use Developer Keys
- Use OAuth

OAuth



A Consumer Requests *Request Token*

Request includes
oauth_consumer_key,
oauth_signature_method,
oauth_signature,
oauth_timestamp,
oauth_nonce,
oauth_version (optional).

B Service Provider Grants *Request Token*

Response includes
oauth_token,
oauth_token_secret.

C Consumer Directs User to Service Provider

Request includes
oauth_token (optional),
oauth_callback (optional).

D Service Provider Directs User to Consumer

Request includes
oauth_token (optional).

E Consumer Requests *Access Token*

Request includes
oauth_consumer_key,
oauth_token,
oauth_signature_method,
oauth_signature,
oauth_timestamp,
oauth_nonce,
oauth_version (optional).

F Service Provider Grants *Access Token*

Response includes
oauth_token,
oauth_token_secret.

G Consumer Accesses Protected Resources

Request includes
oauth_consumer_key,
oauth_token,
oauth_signature_method,
oauth_signature,
oauth_timestamp,
oauth_nonce,
oauth_version (optional).

REST Security and OAuth Demo

http://raibledesigns.com/rd/entry/implementing_oauth_with_gwt

http://raibledesigns.com/rd/entry/grails_oauth_and_linkedin_apis

REST Security Resources

- Implementing REST Authentication
  - http://www.objectpartners.com/2011/06/16/implementing-rest-authentication/
- OAuth2's "Client Credentials" API Key Grant Type
  - http://stackoverflow.com/questions/6190381/how-to-keep-the-client-credentials-confidential-while-using-oauth2s-resource-ow (http://bit.ly/k5LqsH)
  - Thanks to @kdonald for the link!

Penetrate

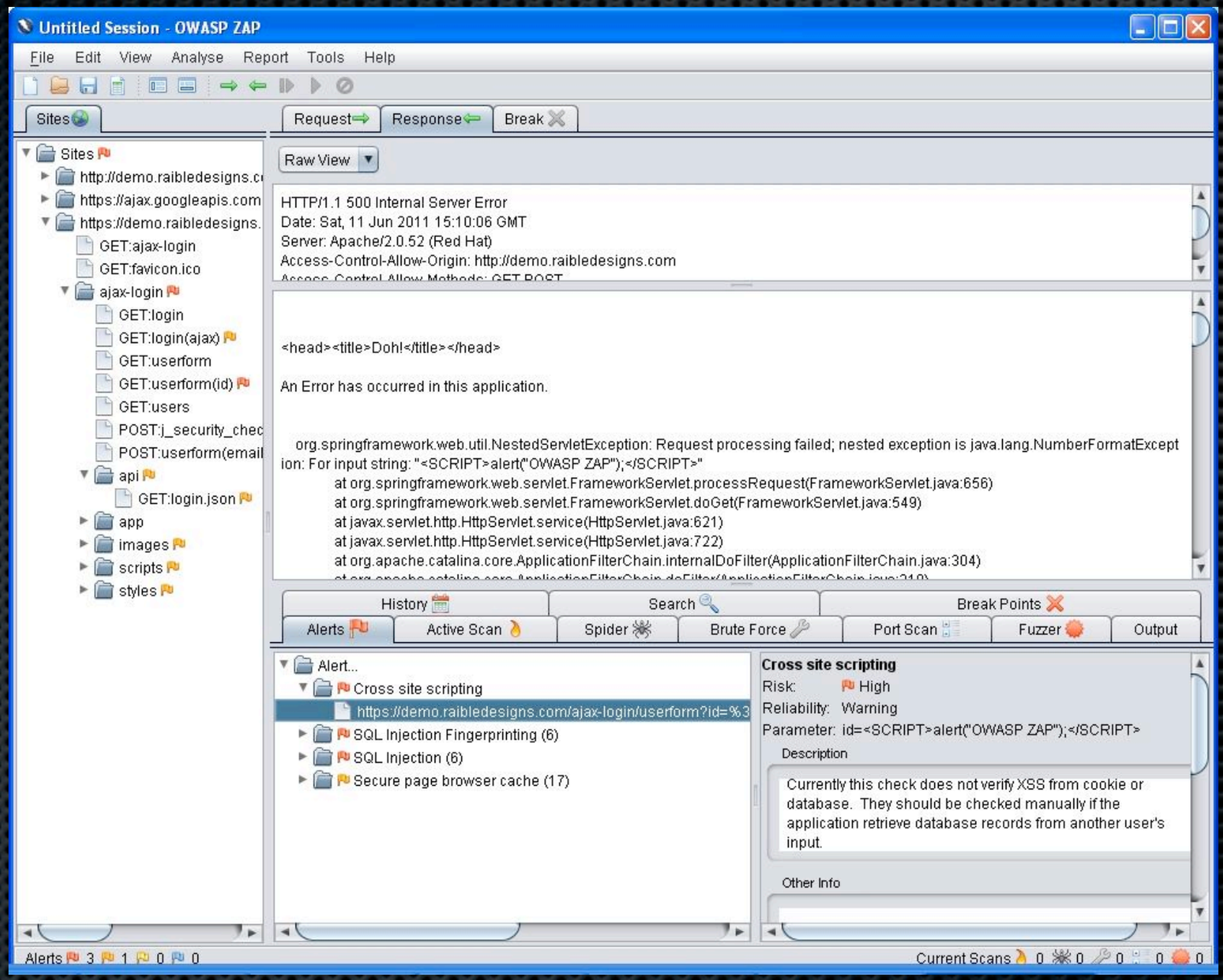
- OWASP Testing Guide and Code Review Guide
- OWASP Top 10
- OWASP Zed Attack Proxy
- Burp Suite
- OWASP WebGoat

OWASP

- The Open Web Application Security Project (OWASP) is an open community dedicated to enabling organizations to develop, purchase, and maintain applications that can be trusted. At OWASP you'll find free and open ...
  - Application security tools, complete books, standard security controls and libraries, cutting edge research
  - http://www.owasp.org

OWASP
The Open Web Application Security Project

Penetration Testing Demo



http://raibledesigns.com/rd/entry/java_web_application_security_part4

7 Security (Mis) Configurations in web.xml

1. Error pages not configured
2. Authentication & Authorization Bypass
3. SSL Not Configured
4. Not Using the Secure Flag

http://software-security.sans.org/blog/2010/08/11/security-misconfigurations-java-webxml-files

7 Security (Mis)Configurations

5. Not Using the HttpOnly Flag
6. Using URL Parameters for Session Tracking
7. Not Setting a Session Timeout

http://software-security.sans.org/blog/2010/08/11/security-misconfigurations-java-webxml-files

Protecting Ajax Login

```
<session-config>
    <session-timeout>15</session-timeout>
    <cookie-config>
        <http-only>true</http-only>
        <secure>true</secure>
    </cookie-config>
    <tracking-mode>COOKIE</tracking-mode>
</session-config>
```

```
<form action="${ctx}/j_security_check" id="loginForm"
      method="post" autocomplete="off">
```

OWASP Top 10 for 2010

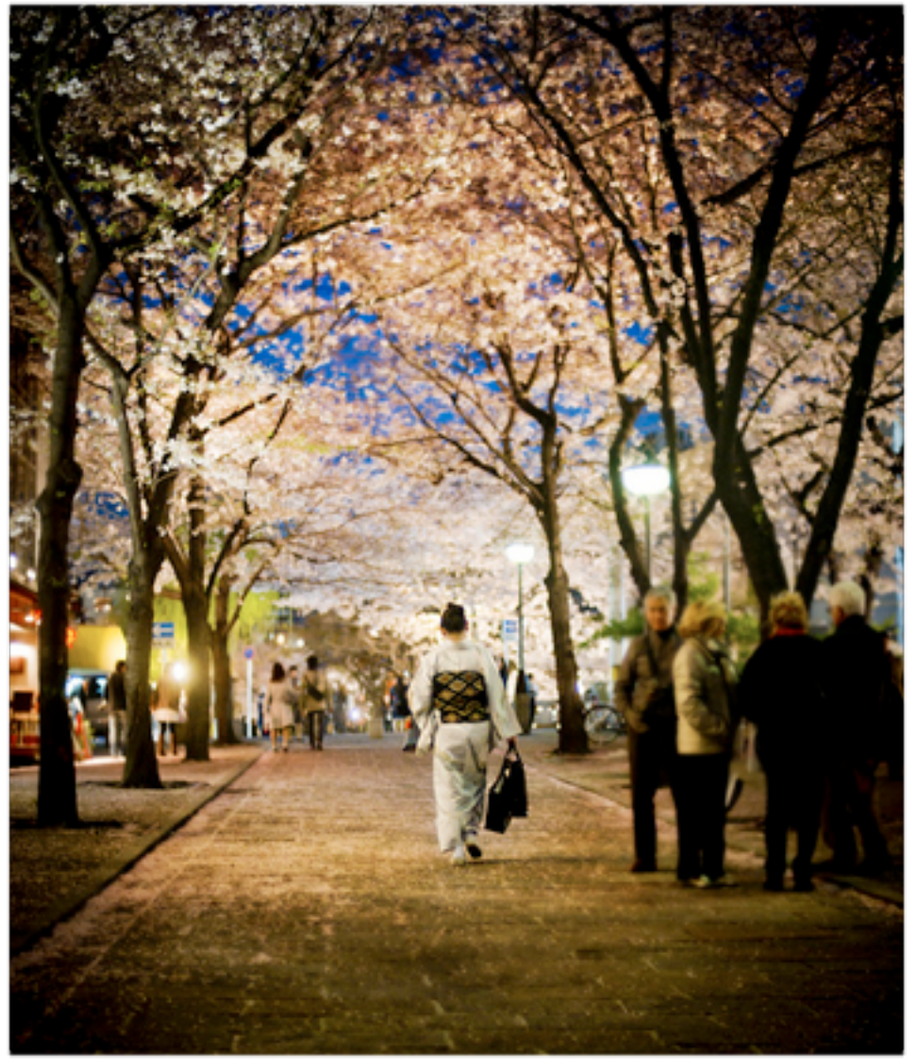
1. Injection
2. Cross-Site Scripting (XSS)
3. Broken Authentication and Session Management
4. Insecure Direct Object References
5. Cross-Site Request Forgery (CSRF)

OWASP Top 10 for 2010

6. Security Misconfiguration
7. Insecure Cryptographic Storage
8. Failure to Restrict URL Access
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects and Forwards

Protect

- Firewalls
- IDS and IDPs
- Audits
- Penetration Tests
- Code Reviews with Static Analysis Tools

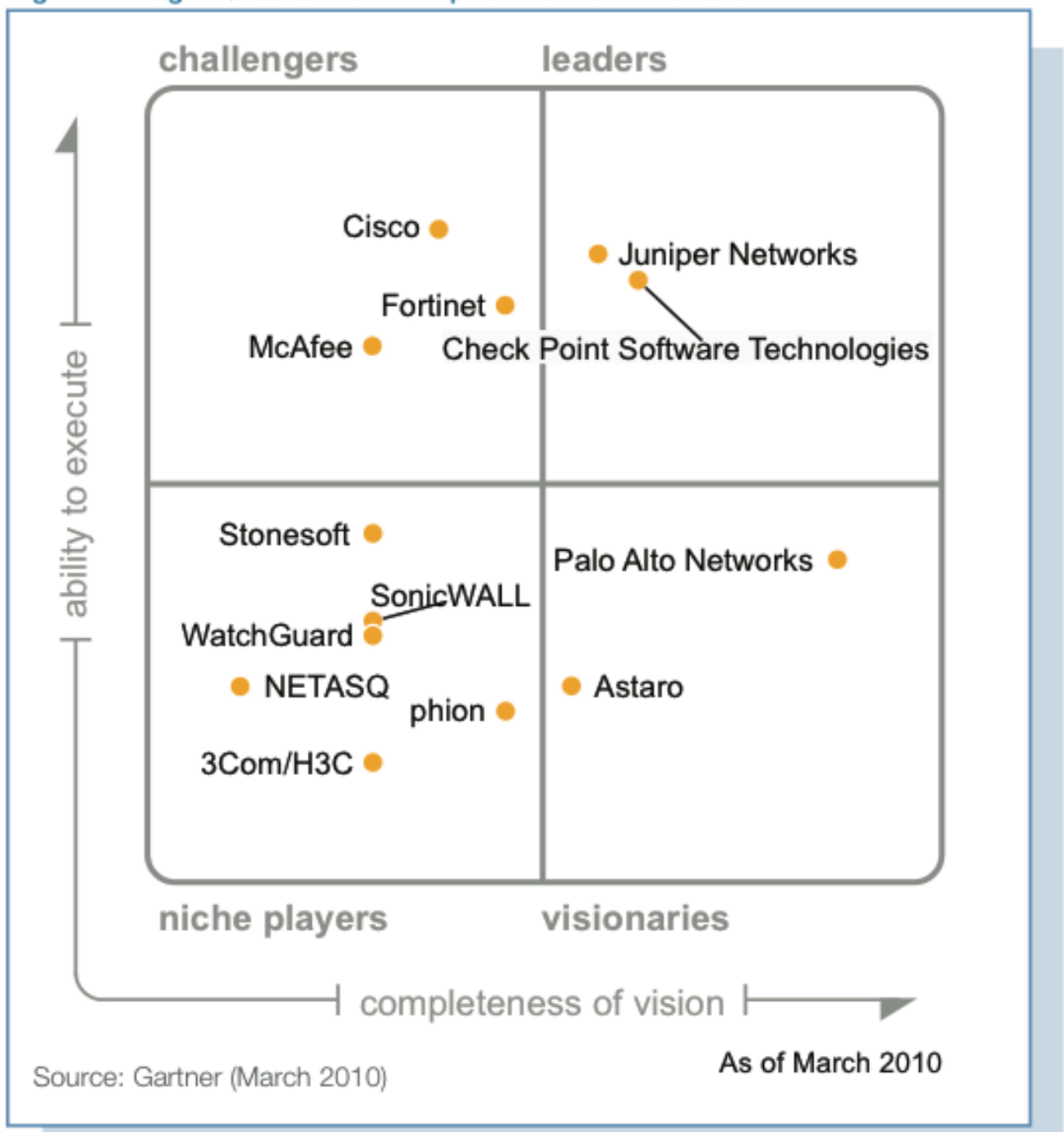


Firewalls

- Stateless Firewalls
- Stateful Firewalls
  - Invented by Nir Zuk at Check Point in the mid-90s
- Web App Firewalls
  - Inspired by the 1996 PHF CGI exploit
  - WAF Market $234m in 2010

Gartner on Firewalls

Figure 1. Magic Quadrant for Enterprise Network Firewalls



Relax

- **Web App Firewalls**: Imperva, F5, Breach
  - **Open Source**: WebNight and ModSecurity
- **Stateful Firewalls**: Juniper, Check Point, Palo Alto
- **IDP/IDS**: Sourcefire, TippingPoint
  - **Open Source**: Snort
- **Audits**: ENY, PWC, Grant Thornton
- **Pen Testing**: WhiteHat, Trustwave, Electric Alchemy
  - **Open Source**: OWASP ZAP
- **Static Analysis**: Fortify, Veracode

Remember...

"Security is a quality, and as all other quality, it is important that we build it into our apps while we are developing them, not patching it on afterwards like many people do." -- *Erlend Oftedal*

From: http://bit.ly/mjufjR

Action!

- Use OWASP and Open Source Security Frameworks
  - Don’t be afraid to contribute!
- Follow the Security Street Fighter Blog
  - http://software-security.sans.org/blog
- Use OWASP ZAP to pentest your apps
- Don’t be afraid of security!

Questions?

Contact Information

http://raibledesigns.com
@mraible

My Presentations

http://slideshare.net/mraible