



# OWASP

The Open Web Application Security Project

## **Hybrid Analysis Mapping:**

## **Making Security and Development Tools Play Nice Together**

**Dan Cornell**

**CTO, Denim Group** [@danielcornell](mailto:danielcornell@denimgroup.com)

**This presentation contains information about DHS-funded research:**

Topic Number: H-SB013.1-002 - Hybrid Analysis Mapping (HAM)

Proposal Number: HSHQDC-13-R-00009-H-SB013.1-002-0003-1



# OWASP

The Open Web Application Security Project

## Dan Cornell

- Dan Cornell, founder and CTO of Denim Group
- Software developer by background (Java, .NET, etc)
- OWASP San Antonio
- 15 years experience in software architecture, development and security
- Heads Denim Group's application security team





# OWASP

The Open Web Application Security Project

## Denim Group Overview

- **Headquarters in San Antonio, TX**
  - *Remote offices: San Francisco, Seattle, New York City, Dallas and Austin*
  - *Founded in 2001, 90 employees, profitable since inception*
  - *Inc. Magazine 5000 Fastest Growing Company (5 consecutive years)*
- **Secure software services and products company**
  - *Builds secure software*
  - *Helps organizations assess and mitigate risk of existing software*
  - *Provides e-Learning and classroom training so clients can build secure software*
- **Customer base spans Fortune 500 and DoD**
  - *Market Focus: Financial Services, Banking, Insurance, Healthcare, and U.S. Air Force*
- **Software-centric view of application security**
  - *Application security experts are practicing developers delivering a rare combination of expertise in today's industry*
  - *Development pedigree translates to rapport with development managers*
  - ***Business impact: shorter time-to-fix application vulnerabilities***



# OWASP

The Open Web Application Security Project

## Agenda

- Security versus Development
  - *Teams*
  - *Tools*
- Hybrid Analysis Mapping (HAM)
  - *ThreadFix Overview*
  - *DHS SBIR Phase 1 Work*
- Demonstrations
- Next Steps
- Questions



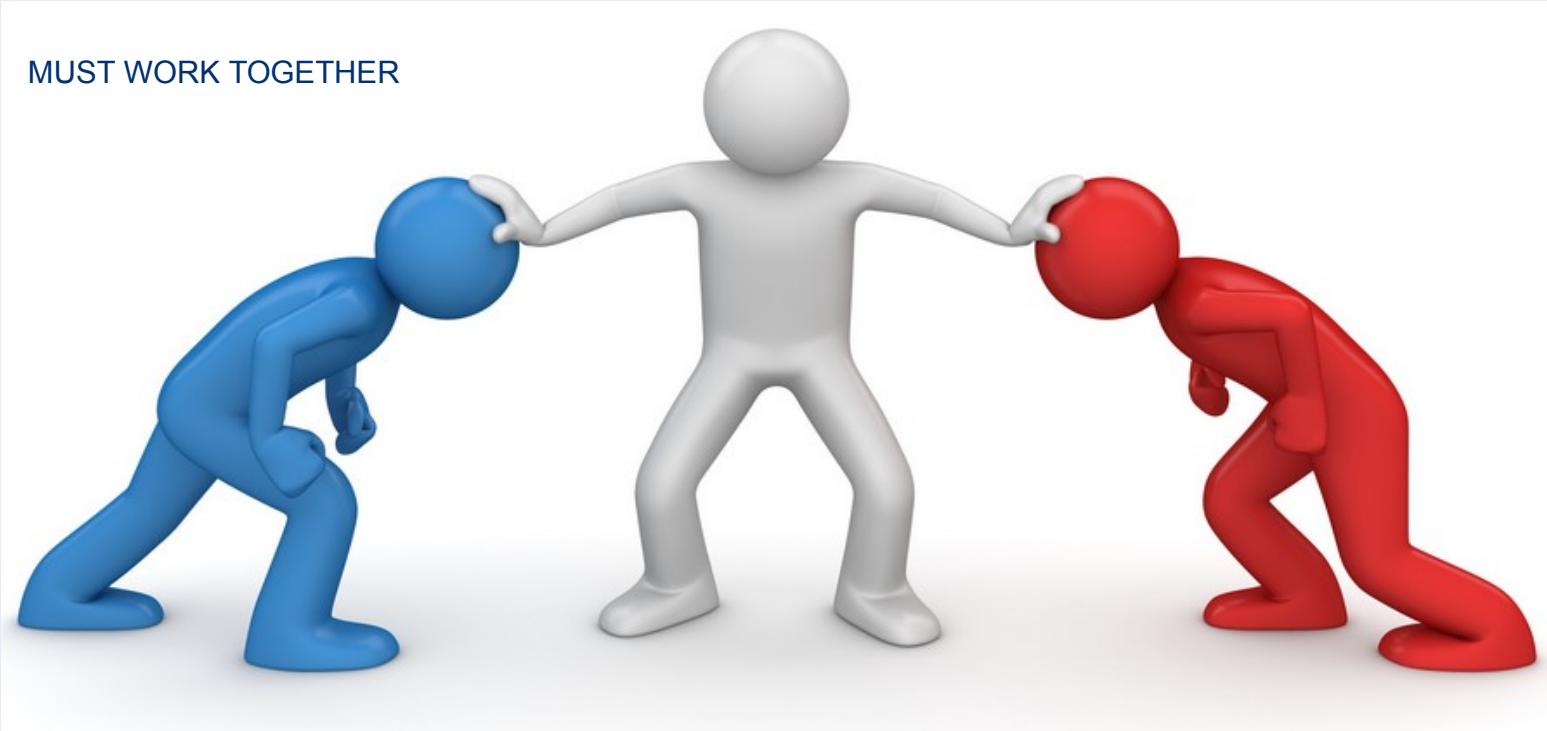
# OWASP

The Open Web Application Security Project

## Development and Security

- Have different goals
- Use different tools

- MUST WORK TOGETHER





# OWASP

The Open Web Application Security Project

## Development

- Features, functions timelines
  - *Get it done!*
- Manage workload: Defect tracking systems
- Write code: Integrated Development Environment (IDE)



# OWASP

The Open Web Application Security Project

## Security

- Policies, standards, risks
  - *Keep the world safe!*
- Network and infrastructure tools/paradigms adapted for applications
  - *Network scanners -> Application scanners*
  - *Network firewalls -> Web application firewalls*



# OWASP

The Open Web Application Security Project

## Problems with Security Tools (if you are in Security)

- Code analysis (static) tools can be challenging
  - *Often require coding experience to use effectively*
  - *Provide LOTS of results (often including false positives)*
- Application scanning (dynamic) tools have their own challenges
  - *Have to be properly configured to get good coverage*
  - *Have to learn about the application*





# OWASP

The Open Web Application Security Project

## Problems with Security Tools (if you are a Developer)

- [Developers Don't Speak PDF](#)
  - *Developers also don't speak Excel*
- Taking action based on dynamic scan results can be challenging
  - *What code needs to be changed and how?*



# OWASP

The Open Web Application Security Project

## How Do We Make Things Better?

- Developers need to learn more about security
  - *To better understand what impact their actions have on their organization's security posture*
- Security teams need to learn more about development
  - *What tools do they use?*
  - *How do they manage their workload*
- These interactions need to become more natural



# OWASP

The Open Web Application Security Project

## So What?

- Genesis of this presentation was some DHS-funded research
  - *Looking into “Hybrid Analysis Mapping” (HAM) via their SBIR program*
- Looking into better integration between dynamic and static scanners
  - *Found interesting applications above and beyond our original goals*

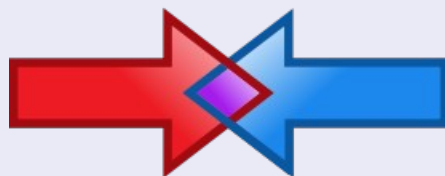


# OWASP

The Open Web Application Security Project

## Hybrid Analysis Mapping – Phase 1 Goal

- Determine the feasibility of developing a system that can reliably and efficiently correlate and merge the results of automated static and dynamic security scans of web applications.



*IBM AppScan Standard*



# OWASP

The Open Web Application Security Project

## Dynamic Application Security Testing

- Spider to enumerate attack surface
- Fuzz to identify vulnerabilities based on analysis of request/response patterns



# OWASP

The Open Web Application Security Project

## Static Application Security Testing

- Use source or binary to create a model of the application
  - *Kind of like a compiler or VM*
- Perform analysis to identify vulnerabilities and weaknesses
  - *Data flow, control flow, semantic, etc*

```
String username = request.getParameter("username");  
String sql = "SELECT * FROM User WHERE username = '" + username + "'";  
  
Statement stmt;  
stmt = con.createStatement();  
stmt.execute(sql);
```



# OWASP

The Open Web Application Security Project

## Hybrid Analysis Mapping – Phase 1 Sub-Goals

- Standardize vulnerability types
- Match dynamic and static locations
- Improve static parameter parsing



# OWASP

The Open Web Application Security Project

## Hybrid Analysis Mapping

### Phase 1 - Technical Objectives

- **Technical Objective 1: Create common data structure standards for both automated static and dynamic security scanning results.**
  - *Task 1: Create a Data Structure for Automated Dynamic Security Scanning Results*
  - *Task 2: Create a Data Structure for Automated Static Security Scanning Results*
- **Technical Objective 2: Research and prototype methods of mapping the results of automated static and dynamic security scanning.**
  - *Task 1: Create a Structured Model for Hybrid Analysis Mapping*
  - *Task 2: Investigate Approaches for Vulnerability Type Mapping*
  - *Task 3: Investigate Approaches for Mapping Source Code Files to URLs*
  - *Task 4: Investigate Approaches for Determining Injection Points*





# OWASP

The Open Web Application Security Project



## ThreadFix

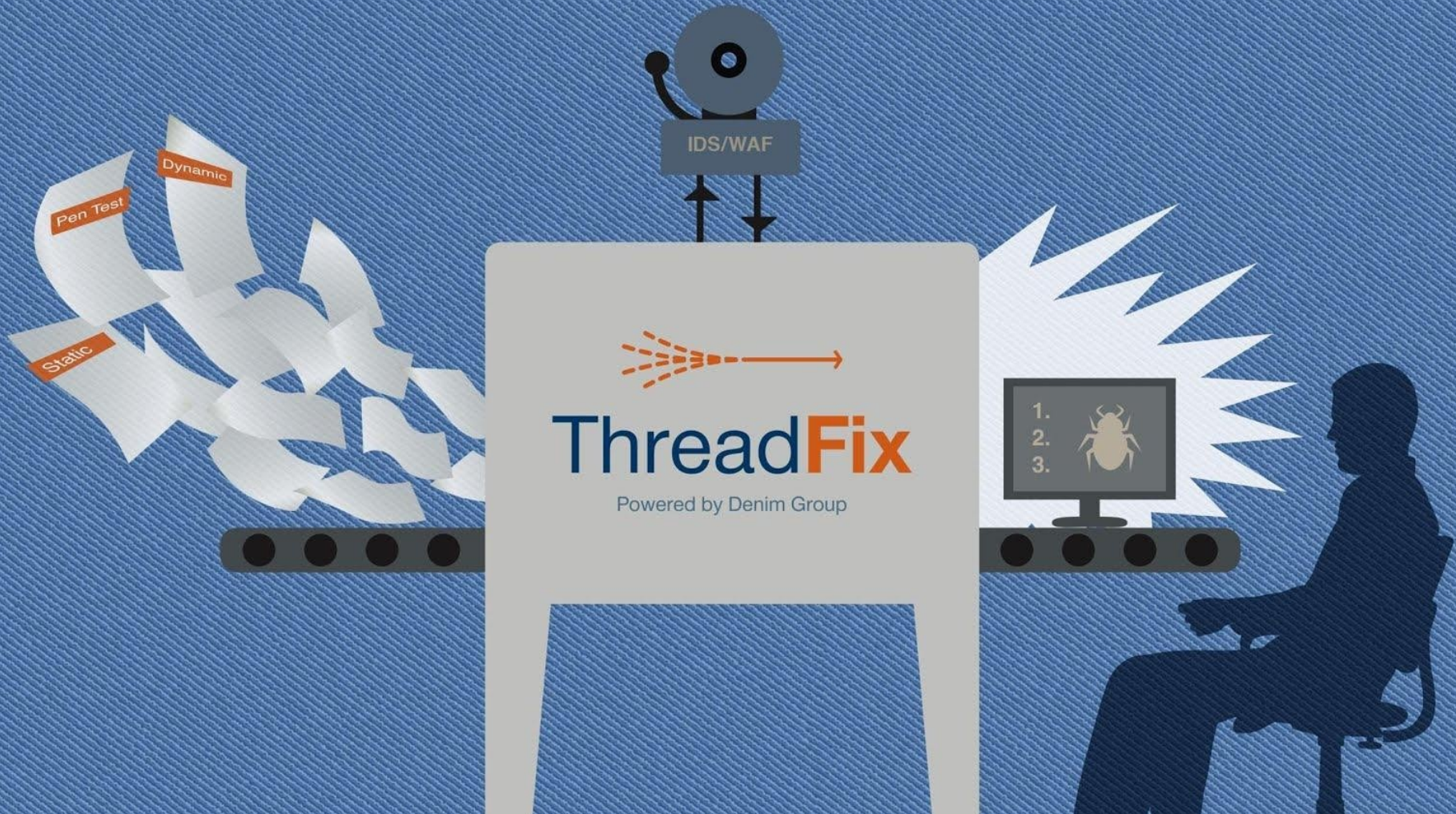
Powered by Denim Group

- **Open source vulnerability management and aggregation platform:**
  - *Allows software security teams to reduce the time to remediate software vulnerabilities*
  - *Enables managers to speak intelligently about the status / trends of software security within their organization.*
- **Features/Benefits:**
  - *Imports dynamic, static and manual testing results into a centralized platform*
  - *Removes duplicate findings across testing platforms to provide a prioritized list of security faults*
  - *Eases communication across development, security and QA teams*
  - *Exports prioritized list into defect tracker of choice to streamline software remediation efforts*
  - *Auto generates web application firewall rules to protect data during vulnerability remediation*
  - *Empowers managers with vulnerability trending reports to pinpoint issues and illustrate application security progress*
  - *Benchmark security practice improvement against industry standards*
- Freely available under the Mozilla Public License (MPL) 2.0
- Download available at: [www.denimgroup.com/threadfix](http://www.denimgroup.com/threadfix)

# ThreadFix

Accelerate Software Remediation

ThreadFix is a software vulnerability aggregation and management system that helps organizations aggregate vulnerability data, generate virtual patches, and interact with software defect tracking systems.





# OWASP

The Open Web Application Security Project

## List of Supported Tools / Technologies:

### Dynamic Scanners

*Acunetix*

*Arachni*

*Burp Suite*

*HP WebInspect*

*IBM Security AppScan Standard*

*IBM Security AppScan Enterprise*

*Mavituna Security Netsparker*

*NTO Spider*

*OWASP Zed Attack Proxy*

*Tenable Nessus*

*Skipfish*

*w3aF*

### Static Scanners

*FindBugs*

*IBM Security AppScan Source*

*HP Fortify SCA*

*Microsoft CAT.NET*

*Brakeman*

### SaaS Testing Platforms

*WhiteHat*

*Veracode*

*QualysGuard WAS*

### IDS/IPS and WAF

*DenyAll*

*F5*

*Imperva*

*Mod\_Security*

*Snort*

### Defect Trackers

*Atlassian JIRA*

*Microsoft Team Foundation Server*

*Mozilla Bugzilla*

### Known Vulnerable Component Scanner

*Dependency Check*





# OWASP

The Open Web Application Security Project

## Large Range of Tool Compatibility



HP WebInspect



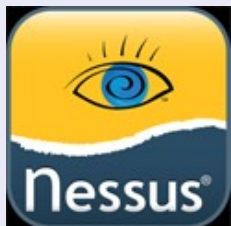
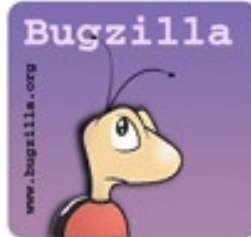
w3af  
Web Application Attack and Audit Framework



FindBugs<sup>TM</sup>  
because it's easy



OUNCE LABS



Enterprise Security



arachni  
web application security scanner framework



# OWASP

The Open Web Application Security Project

## What is a Unique Vulnerability?

- (CWE, Relative URL)
  - *Predictable resource location*
  - *Directory listing misconfiguration*
  
- (CWE, Relative URL, Injection Point)
  - *SQL injection*
  - *Cross-site Scripting (XSS)*
  
- Injection points
  - *Parameters – GET/POST*
  - *Cookies*
  - *Other headers*



# OWASP

The Open Web Application Security Project

## Why Common Weakness Enumeration (CWE)?

- Every tool has their own “spin” on naming vulnerabilities
- OWASP Top 10 / WASC 24 are helpful but not comprehensive
- CWE is exhaustive (though a bit sprawling at times)
- Reasonably well-adopted standard
- Many tools have mappings to CWE for their results
- Main site: <http://cwe.mitre.org/>



# OWASP

The Open Web Application Security Project

## Information Used

- Source Code (Git URL)
- Framework Type (JSP, Spring)
- Extra information from Fortify (if available)



# OWASP

The Open Web Application Security Project

## Vulnerability Types

- Successful CWE standardization
- Investigation into trees and Software Fault Patterns
  - *Meant to correct for human errors*
  - *Hard to do in an automated fashion*





# OWASP

The Open Web Application Security Project

## Unified Endpoint Database (Static and Dynamic)

- EndpointQuery
  - *dynamicPath*
  - *staticPath*
  - *Parameter*
  - *httpMethod*
  - *codePoints [List<CodePoint>]*
  - *informationSourceType*
- EndpointDatabase
  - *findBestMatch(EndpointQuery query): Endpoint*
  - *findAllMatches(EndpointQuery query): Set<Endpoint>*
  - *getFrameworkType(): FrameworkType*



# OWASP

The Open Web Application Security Project

## Parsing Attack Surface Locations

- JSP: Start with root JSP folder
- Spring: Parse @Controller classes



# OWASP

The Open Web Application Security Project

## Parsing Parameters

- JSP: Look for `request.getParameter()` calls
  - *Coupled with lightweight dataflow analysis*
- Spring: Parse `@RequestParam`, `@PathVariable`, `@Entity` annotations

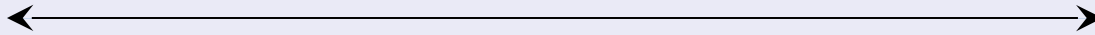


# OWASP

The Open Web Application Security Project

## HAM Bridge

Static



Dynamic

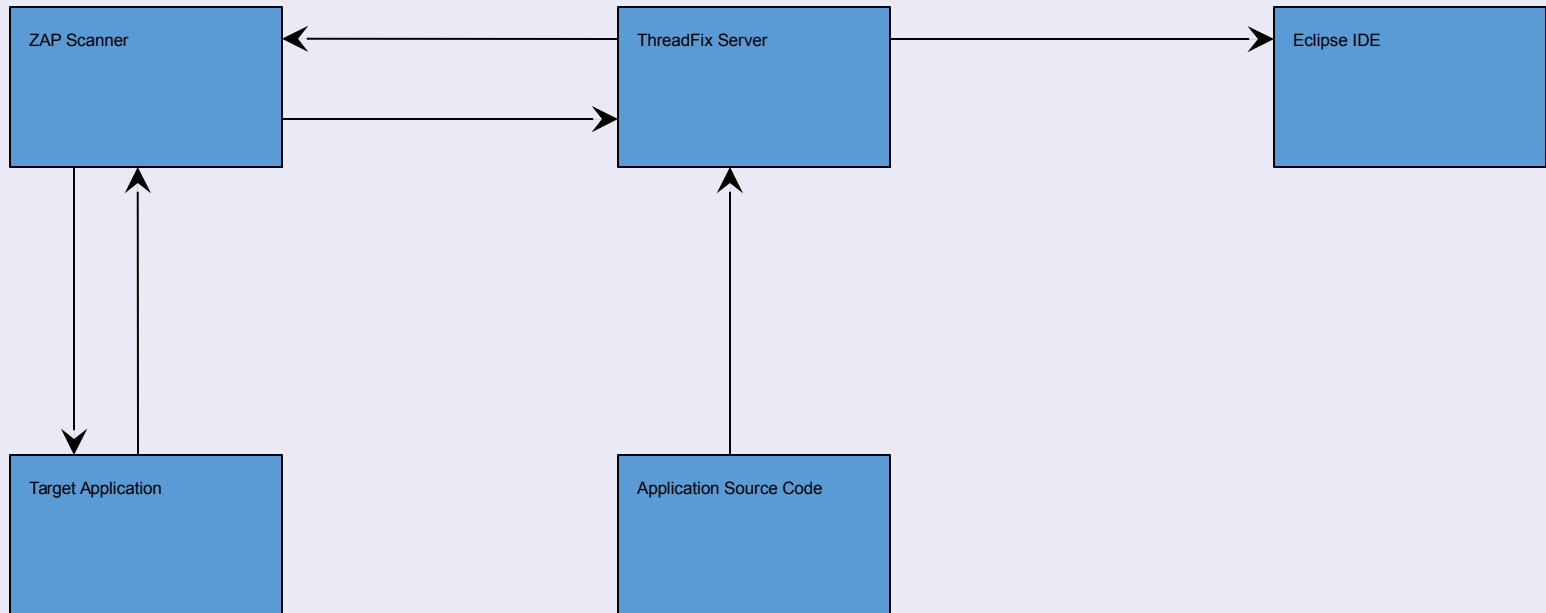
- EndpointDatabase enables more than merging
- Scanner integration allows smarter scanning
- IDE plugin shows all vulnerabilities inline



# OWASP

The Open Web Application Security Project

## System Structure





# OWASP

The Open Web Application Security Project

## Demonstrations

- Show me my application's attack surface
- Merge static and dynamic scanner results
- De-duplicate dynamic RESTful scanner results
- Pre-load my scanner with attack surface data
- Map my dynamically-identified vulnerabilities to their source code location



# OWASP

The Open Web Application Security Project

## Application Attack Surface (CLI)

```
ham — bash — 124x40
Dans-MacBook-Pro:ham dcornell$ java -jar endpoints-experimental-20131106.jar ~/git/bodgeit/root/
INFO [main] MergeConfigurationGenerator.getDatabase(20) | Attempting to calculate framework type based on project contents.
INFO [main] MergeConfigurationGenerator.getType(16) | Attempting to guess Framework Type from source tree.
INFO [main] MergeConfigurationGenerator.getType(17) | File: /Users/dcornell/git/bodgeit/root
INFO [main] ServletMappings.guessApplicationType(175) | About to guess application type from web.xml.
INFO [main] ServletMappings.guessApplicationType(217) | Determined that the framework type was JSP
INFO [main] MergeConfigurationGenerator.getType(34) | Source tree framework type detection returned: JSP
INFO [main] MergeConfigurationGenerator.getDatabase(24) | Calculated framework : JSP.
INFO [main] GeneratorBasedEndpointDatabase.<init>(54) | Using generic EndpointGenerator-based translator.
INFO [main] GeneratorBasedEndpointDatabase.buildMappings(69) | Building mappings.
INFO [main] GeneratorBasedEndpointDatabase.buildMappings(82) | Done building mappings. Static keys: 0, dynamic keys: 16
[POST, GET] ./about.jsp, []
[POST, GET] ./admin.jsp, []
[POST, GET] ./advanced.jsp, [q, debug]
[POST, GET] ./basket.jsp, [update, productid, quantity, debug]
[POST, GET] ./contact.jsp, [anticsrf, debug, comments]
[POST, GET] ./footer.jsp, []
[POST, GET] ./header.jsp, [debug]
[POST, GET] ./home.jsp, [debug]
[POST, GET] ./init.jsp, []
[POST, GET] ./login.jsp, [username, debug, password]
[POST, GET] ./logout.jsp, []
[POST, GET] ./password.jsp, [password1, password2]
[POST, GET] ./product.jsp, [typeid, prodid, debug]
[POST, GET] ./register.jsp, [password1, username, password2, debug]
[POST, GET] ./score.jsp, [debug]
[POST, GET] ./search.jsp, [q, debug]
Dans-MacBook-Pro:ham dcornell$
```



# OWASP

The Open Web Application Security Project

## Merging Static and Dynamic Scanner Results

The screenshot shows a web browser window displaying a list of vulnerabilities. The browser address bar shows the URL: `localhost:8080/threadfix/organizations/1/applications/1?nonce=8B1A8A51836EE9F593CC7575C55B9497`. The list of vulnerabilities includes several 'Critical' items related to 'Race Condition' and 'Cross-site Scripting'.

Severity	Description	URL	Parameter	Action
Critical	Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	/product.jsp		View More
Critical	Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	/register.jsp		View More
Critical	Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	/score.jsp		View More
Critical	Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	/search.jsp		View More
Critical	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	/basket.jsp	productid	View More
Critical	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	/contact.jsp	anticsrf	View More
Critical	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	/contact.jsp	comments	View More

**Scan History**

Channel	Scan Date	User
IBM Rational AppScan	6/19/13 9:18:47 AM	user
Fortify 360	6/19/13 12:41:39 AM	user

**Comments**

User	Date	Comment
		No comments found.

Add Comment

The screenshot also shows a continuation of the vulnerability list below the tables, including items for `/contact.jsp` (parameter: `user1@thebodgeitstore.com`), `/login.jsp` (parameter: `username`), and `/product.jsp` (parameter: `prodid`).





# OWASP

The Open Web Application Security Project

## Merging Static and Dynamic Scanner Results

The screenshot shows the ThreadFix web interface. The browser address bar indicates the URL: localhost:8080/threadfix/organizations/1/applications/1/vulnerabilities/28?nonce=7FEF0E60ECF6CABF857C29D3D5C6... The page title is 'Vulnerability Details | ThreadFix'. The navigation menu includes Dashboard, Applications, Scans, Reports, and a user profile dropdown. The breadcrumb trail is: Applications Index / Team: HAM Demo / Application: Bodgeit / Vulnerability 28.

### Vulnerability Details

[Toggle More Info](#)

#### Basic Information

Type	Severity	Path	Parameter
Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') ( <a href="#">CWE Entry</a> )	Critical	/bodgeit/contact.jsp	comments

#### Findings

Scanner Name	Severity	Vulnerability Type	Path	Calculated URL Path	Calculated Source Path	Parameter	Native ID
IBM Rational AppScan	High	Stored Cross-Site Scripting	/bodgeit/contact.jsp	/contact.jsp	/contact.jsp	comments	aa113d21429a4e5fdb6a189c3ab23c79 <a href="#">View Finding</a>
Fortify 360	Critical	Cross-Site Scripting: Reflected	root/contact.jsp	/contact.jsp		comments	4AC2C441CFC0081776F519B90EBE6650 <a href="#">View Finding</a>

#### Data Flow

File Name root/contact.jsp  
Line number 37  
Line text String comments = (String) request.getParameter("comments");  
File Name root/contact.jsp  
Line number 37



# OWASP

The Open Web Application Security Project

## De-Duplicate Dynamic RESTful Scanner Results

The screenshot shows a web browser window with the URL `localhost:8080/threadfix/organizations/1/applications/4?nonce=CDB343B3C8A8F5428F447AAFDDA51CE8`. The page displays a list of vulnerabilities with columns for Severity, Type, Path, and Parameter. A comment section is visible, containing a table with Scan History and a single comment.

Severity	Type	Path	Parameter	
Critical (11)				
Critical	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	/owners	lastName	<a href="#">View More</a>
Critical	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	/owners/{id}/pets/{id}/visits/new	new	<a href="#">View More</a>
Critical	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	/owners/{id}/pets/{id}/edit	name	<a href="#">View More</a>
Critical	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	/owners/{id}/pets/new	birthDate	<a href="#">View More</a>
Critical	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	/owners/{id}/pets/{id}/visits/new	description	<a href="#">View More</a>

### Scan History

Channel	Scan Date	User
IBM Rational AppScan	6/4/13 10:57:15 AM	user
IBM Rational AppScan	6/4/13 10:57:15 AM	user

### Comments

User	Date	Comment
1 user	10:44:04 11/20/2013	The app uses a REST-style URL scheme so AppScan is multi-reporting vulnerabilities. They seem to have been properly de-duped, however.

[Add Comment](#)

Critical	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	/owners/{id}/edit	firstName	<a href="#">View More</a>
Critical	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	/owners/{id}/edit	lastName	<a href="#">View More</a>
Improper Neutralization of Special Elements used in an				



# OWASP

The Open Web Application Security Project

## De-Duplicate Dynamic RESTful Scanner Results

The screenshot shows the ThreadFix web interface. The browser address bar displays the URL: localhost:8080/threadfix/organizations/1/applications/4/vulnerabilities/10982?nonce=C1E9E2E479E2D82DA5A827724... The interface includes a navigation bar with 'Dashboard', 'Applications', 'Scans', 'Reports', and a user profile 'user'. The breadcrumb trail is: Applications Index / Team: HAM Demo / Application: PetClinic (vulnerable) / Vulnerability 10982.

### Vulnerability Details

Toggle More Info

#### Basic Information

Type	Severity	Path	Parameter
Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') <a href="#">(CWE Entry)</a>	Critical	/petclinic/owners/10/pets/12/visits/new	description

#### Findings

Scanner Name	Severity	Vulnerability Type	Path	Calculated URL Path	Calculated Source Path	Parameter	Native ID
IBM Rational AppScan	High	Blind SQL Injection	/petclinic/owners/10/pets/12/visits/new	/owners/{id}/pets/{id}/visits/new	/src/main/java/org/springframework/sample/petclinic/web/VisitController.java	description	1a07468c95c11f782dee9ab5b7fb6c05 <a href="#">View Finding</a>
IBM Rational AppScan	High	Blind SQL Injection	/petclinic/owners/4/pets/5/visits/new	/owners/{id}/pets/{id}/visits/new	/src/main/java/org/springframework/sample/petclinic/web/VisitController.java	description	4054c867b8edad825d4e4468b3e7ac9a <a href="#">View Finding</a>

Comments



# OWASP

The Open Web Application Security Project

## Seed Scanner with Attack Surface

The screenshot shows the OWASP ZAP interface during a scan of a local host. The 'Sites' tree on the left shows the scanned site structure. The main area displays the 'Request' and 'Response' details. The 'History' tab at the bottom shows a list of processed URIs with their methods and flags.

Processed	Method	URI	Flags
●	GET	http://localhost:8081	SEED
●	GET	http://localhost:8081/bodgeit/	SEED
●	GET	http://localhost:8081/bodgeit	SEED
●	GET	http://localhost:8081/bodgeit/about.jsp	SEED
●	GET	http://localhost:8081/bodgeit/admin.jsp	SEED
●	GET	http://localhost:8081/bodgeit/home.jsp	SEED
●	GET	http://localhost:8081/bodgeit/advanced.jsp	SEED
●	GET	http://localhost:8081/bodgeit/advanced.jsp?q=true	SEED
●	GET	http://localhost:8081/bodgeit/contact.jsp	SEED
●	GET	http://localhost:8081/bodgeit/advanced.jsp?debug=true	SEED
●	GET	http://localhost:8081/bodgeit/login.jsp	SEED



# OWASP

The Open Web Application Security Project

## Map Dynamic Scan Results to LoC in IDE

The screenshot shows the Spring Tool Suite IDE with the following components:

- Project Explorer:** Shows a project structure with folders like 'bodgeit', 'lib', 'root', 'src', and 'petclinic'.
- Editor:** Displays the code for `JpaVisitRepositoryImpl.java`. A line of code is highlighted: `Query query = this.em.createQuery("SELECT visit FROM Visit v where v.pets.id= " + petId);`. The variable `petId` is enclosed in a red box.
- ThreadFix Vulnerabilities:** A pane at the bottom showing a list of detected vulnerabilities. The entry for the highlighted code is:

Resource	Location	Parameter	CWE	CWE Name	Defect Url
JpaOwnerR...	line 64	null	566	Authorization Bypass Through User-Controlled SQL Primary Key	
JpaOwnerR...	line 64	null	89	Improper Neutralization of Special Elements used in an SQL...	
JpaOwnerR...	line 64	ownerId	566	Authorization Bypass Through User-Controlled SQL Primary Key	
JpaOwnerR...	line 64	ownerId	566	Authorization Bypass Through User-Controlled SQL Primary Key	
JpaOwnerR...	line 64	ownerId	566	Authorization Bypass Through User-Controlled SQL Primary Key	
JpaVisitRep...	line 60	null	566	Authorization Bypass Through User-Controlled SQL Primary Key	
JpaVisitRep...	line 60	null	89	Improper Neutralization of Special Elements used in an SQL...	
JpaVisitRep...	line 60	pet	566	Authorization Bypass Through User-Controlled SQL Primary Key	
OwnerCont	line 84	dri	550	Information Exposure Through Server Error Message	



# OWASP

The Open Web Application Security Project

## Framework Support

Level of effort to support current frameworks:

JSP:

---

Language	files	blank	comment	code
----------	-------	-------	---------	------

---

Java	7	167	176	698
------	---	-----	-----	-----

---

SUM:	7	167	176	698
------	---	-----	-----	-----

---

Spring:

---

Language	files	blank	comment	code
----------	-------	-------	---------	------

---

Java	15	370	412	1491
------	----	-----	-----	------

---

SUM:	15	370	412	1491
------	----	-----	-----	------

---



# OWASP

The Open Web Application Security Project

## What's Next?

- Started SBIR Phase 2 contract
- More frameworks:
  - *ASP.NET MVC*
  - *ASP.NET*
  - *Python/Django*
  - *Java/Struts*
  - *Java/JSF*
  - *Ruby on Rails*
- More IDEs:
  - *Visual Studio*



# OWASP

The Open Web Application Security Project

## What's Next?

Expand the application model:

- Authentication
- Authorization
- Injection points: Cookies and HTTP headers





# OWASP

The Open Web Application Security Project

## Call To Action

- Download the endpoint calculator and run on your code
  - *What works? What doesn't work?*
  - *Currently support: Java/JSP and Java/Spring*
  - *Others in the works*
- Download the ThreadFix ZAP plugin
  - *Pre-seed some scans*
  - *Store scan data in ThreadFix*
- Download the ThreadFix Eclipse plugin
  - *Pull dynamic scan results into your IDE*
- What tools and frameworks do YOU want to see supported?





# OWASP

The Open Web Application Security Project

## Resources

- ThreadFix downloads (please don't download the GitHub ZIPs)
  - <http://www.threadfix.org/download/>
- Endpoint generator
  - <https://github.com/denimgroup/threadfix/wiki/Endpoint-CLI>
- ZAP plugin
  - <https://github.com/denimgroup/threadfix/wiki/Zap-Plugin>
- Eclipse plugin
  - <https://github.com/denimgroup/threadfix/wiki/Eclipse-IDE-Plugin>



# OWASP

The Open Web Application Security Project

## Questions?

Dan Cornell

[dan@denimgroup.com](mailto:dan@denimgroup.com)

Twitter: [@danielcornell](https://twitter.com/danielcornell)

[www.denimgroup.com](http://www.denimgroup.com)

[www.denimgroup.com/blog](http://www.denimgroup.com/blog)

(210) 572-4400

[www.threadfix.org](http://www.threadfix.org)

[www.denimgroup.com/threadfix](http://www.denimgroup.com/threadfix)

[github.com/denimgroup/threadfix](https://github.com/denimgroup/threadfix)