OWASP AppSec EU Belfast 8-12 May, 2017

# Monitoring Attack Surface and Integrating Security into DevOps Pipelines

Dan Cornell, CTO Denim Group

# Agenda

- Background
- Importance of Attack Surface
- What Does Attack Surface Have to Do with DevOps?
- Hybrid Analysis Mapping (HAM) Background
- Installation Instructions
- Use Cases
- Questions

# My Background

- Dan Cornell, founder and CTO of Denim Group
- Software developer by background (Java, .NET, etc)
- OWASP San Antonio

# Denim Group Background

- Secure software services and products company
  - Builds secure software
  - Helps organizations assess and mitigate risk of in-house developed and third party software
  - Provides classroom training and e-Learning so clients can build software securely
- Software-centric view of application security
  - Application security experts are practicing developers
  - Development pedigree translates to rapport with development managers
  - **Business impact: shorter time-to-fix application vulnerabilities**
- Culture of application security innovation and contribution
  - Develops open source tools to help clients mature their software security programs
    - *Remediation Resource Center, ThreadFix*
  - OWASP national leaders & regular speakers at RSA, SANS, OWASP, ISSA, CSI
  - World class alliance partners accelerate innovation to solve client problems

# OWASP ZAP

- Open source web proxy and dynamic application security testing tool

- https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

# Example Codebases

- BodgeIt Store
  - Example vulnerable web application
  - https://github.com/psiinon/bodgeit
- Java Spring Petstore
  - Example Spring application
  - https://github.com/spring-projects/spring-petclinic
- Railsgoat
  - Example vulnerable web application
  - https://github.com/OWASP/railsgoat

## ThreadFix Community Edition

- Application vulnerability management
  - And some other stuff
- https://github.com/denimgroup/threadfix

## Downloads

- https://dl.dropboxusercontent.com/u/737351/endpoints-json.jar
- https://dl.dropboxusercontent.com/u/737351/threadfix-release-2.zap
- https://github.com/denimgroup/threadfix-examples/tree/master/web_app_attack_surface

## Importance of Attack Surface

# Importance of Attack Surface

- This is where an attacker can “reach out and touch” your application
  - Web: Mostly in the HTTP request: URL, parameters, headers (cookies)
  - Mobile, IoT: More complicated
  - We will focus on web today
- Target for dynamic testing
  - Automated DAST
  - Manual assessment/penetration testing

## What Does Attack Surface Have to Do With DevOps?

- ~~If you want your talk to be accepted, it has to have DevOps in the title~~

- Let’s look at what we want from security in the DevOps pipeline

# Security in the DevOps Pipeline

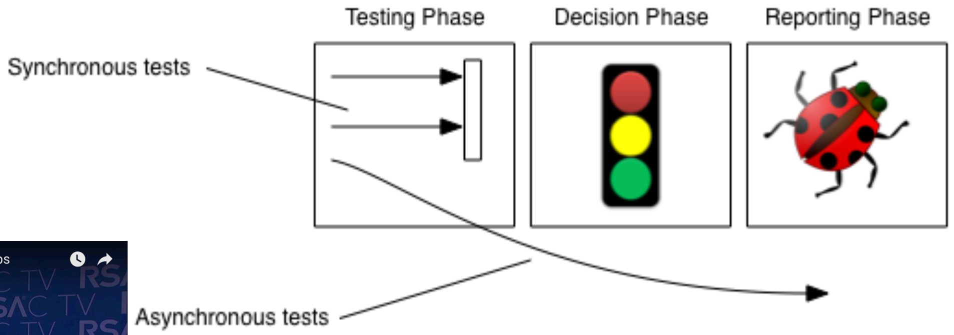
Organizations like Etsy and Netflix are doing *amazing* things to secure apps via their DevO

# Security in the DevOps Pipeline

- Testing
  - Synchronous
  - Asynchronous
- Decision
- Reporting

Blog Post: Effective Application Security Testing in DevOps Pipelines

http://www.denimgroup.com/blog/2016/12/effective-application-security-testing-in-devops-pipelines/



https://www.denimgroup.com/resources/effective-application-security-for-devops/

# Focus on Testing in DevOps Pipeline

- Many security tools run too long to include in many pipeline builds
  - Full SAST, DAST
- Security testing also includes manual testing
  - Which is *way* too slow for most pipeline builds

- Tracking attack surface changes over time can help us:
  - Focus testing activities
  - Trigger testing activities

## Hybrid Analysis Mapping

- Goal: Merge the results of SAST and DAST testing
- Funded via DHS S&T SBIR contracts
- Facilitated the creation of our attack surface modeling engine

## Department of Homeland Security Support

- Currently in Phase 2 of a DHS S&T CSD SBIR
- Acronyms!
  - DHS = Department of Homeland Security
  - S&T = Directorate of Science and Technology
  - CSD = CyberSecurity Division
  - SBIR = Small Business Innovation Research
- Geared toward developing new technologies for Federal customers

- Hybrid Analysis Mapping (HAM)
- Technology has been included with ThreadFix
- Has also resulted in some other released components we will talk about today

- **Please do not assume this talk is endorsed by DHS**
  - **This is just me talking about what we have done**

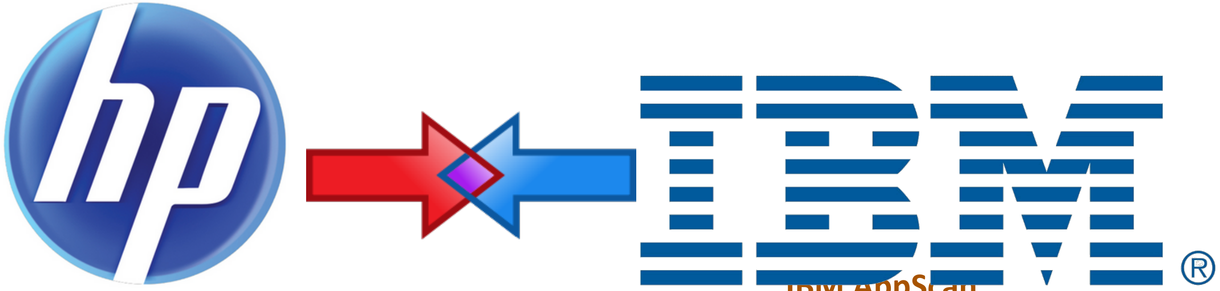
## Hybrid Analysis Mapping (HAM)

- Initial goal: Correlate and merge results from SAST and DAST

- After we made that work, we found other stuff we could do with the technology

# Hybrid Analysis Mapping (HAM)

- **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**

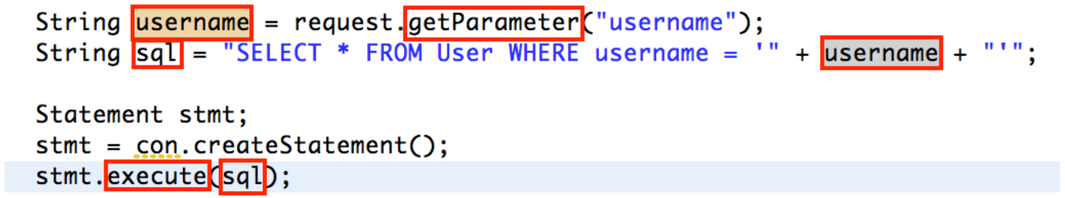
**Standard**

## Dynamic Application Security Testing (DAST)

- Spider to enumerate attack surface
  - Crawl the site like Google would
  - But with authentication / session detection
- Fuzz to identify vulnerabilities based on analysis of request/response patterns
  - If you send a SQL control character and get a JDBC error message back, that could indicate a SQL injection vulnerability
- A finding looks like (CWE, relative URL, [entry point])

## Static Application Security Testing (SAST)

- 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
- A finding looks like (CWE, code/data flow)



```
String username = request.getParameter("username");
String sql = "SELECT * FROM User WHERE username = '" + username + "'";

Statement stmt;
stmt = con.createStatement();
stmt.execute(sql);
```

## Hybrid Analysis Mapping Sub-Goals

- Standardize vulnerability types
  - Settled on MITRE Common Weakness Enumeration (CWE)
- Match dynamic and static locations
  - Use knowledge of language/web framework to build attack surface database
- Improve static parameter parsing
  - Parse out of source code to match with DAST result

## Information Used

- Source Code
  - Git, Subversion, Local Copy

- Framework Type
  - Java: JSP, Spring, Struts
  - C#: .NET WebForms, .NET MVC
  - Ruby: Rails
  - PHP: in progress

- Extra information from SAST results (if available)

## Unified Endpoint Database

- EndpointQuery
  - dynamicPath
  - staticPath
  - Parameter
  - httpMethod
  - codePoints [List<CodePoint>]
  - informationSourceType

- EndpointDatabase
  - findBestMatch(EndpointQuery query): Endpoint
  - findAllMatches(EndpointQuery query): Set<Endpoint>
  - getFrameworkType(): FrameworkType

# Merging SAST and DAST Results

- I have a DAST result:
  - (“Reflected XSS”, /login.jsp, “username” parameter)
- Query the Endpoint Database:
  - Entry point is com.something.something.LoginController.java, line 62
- Search the other findings for SAST results like:
  - (“Reflected XSS”, source at com.something.something.LoginController.java, line 62)

- If you find a match – correlate those two findings
- Magic!

# That’s Great But I Want More

- So our research produced a successful/valuable outcome
  - Hooray
- But – given these data structures, what else can we do?

- From an EndpointDatabase we can:
  - Get *all* of the application’s attack surface
  - Map DAST results to a specific line of code

- Given those capabilities we can:
  - Pre-seed scanners with attack surface
  - Map DAST results to lines of code in a developer IDE
  - Map DAST results to lines of code in SonarQube

## Final Thoughts on SBIR Work with DHS S&T

SBIR
SMALL BUSINESS
INNOVATION RESEARCH
Science & Technology

- Great use of the SBIR program
  - In my humble and *totally* unbiased opinion

- Proved to be the tipping point to developing HAM
  - HAM was *interesting*, but required material investment

- Research produced a successful outcome (we think)
- We found other things we could do with the technology
- Released much of it open source to increase adoption

# Scanner Seeding

- What if we could give the DAST spidering process a head start?
- Pre-seed with *all* of the attack surface
  - Landing pages that link in to the application
  - Hidden directories
  - Backdoor or “unused” parameters

- Currently have plugins for OWASP ZAP and BurpSuite
  - Plugin for IBM Rational AppScan Standard is in progress

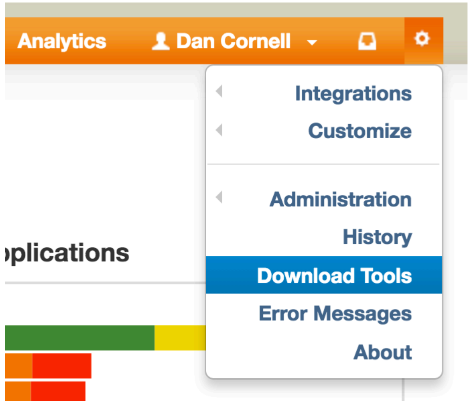
Burp suite

AppScan
IBM. Rational.

https://github.com/denimgroup/threadfix/wiki/Scanner-Plugins

# Getting the Plugins

- Main ThreadFix site
  - https://github.com/denimgroup/threadfix/
- ThreadFix build instructions
  - https://github.com/denimgroup/threadfix/wiki/Development-Environment-Setup
  - “Running ThreadFix Without an IDE”
- Download plugins from ThreadFix



## Plugin Installation

- OWASP ZAP plugin installation instructions
  - https://github.com/denimgroup/threadfix/wiki/Zap-Plugin

- Plugins also available for:
  - Portswigger BurpSuite Professional
  - IBM Rational AppScan (soon)

## Attack Surface Enumeration

- Find *all* of the attack surface
  - URLs
  - Parameters that will change application behavior
  - Future: Cookies, other HTTP headers
- Why is this a problem?
  - Hidden landing pages
  - Multi-step processes that automated crawls don’t traverse
  - Unknown parameters
  - Debug/backdoor parameters (will discuss this further)
- Great for REST APIs support single-page web applications and mobile applications

## Attack Surface Enumeration Benefits

- Reduce false negatives from scanners
  - Better coverage for standard fuzzing

- Pen test *all of* the application

# Endpoints CLI Notes

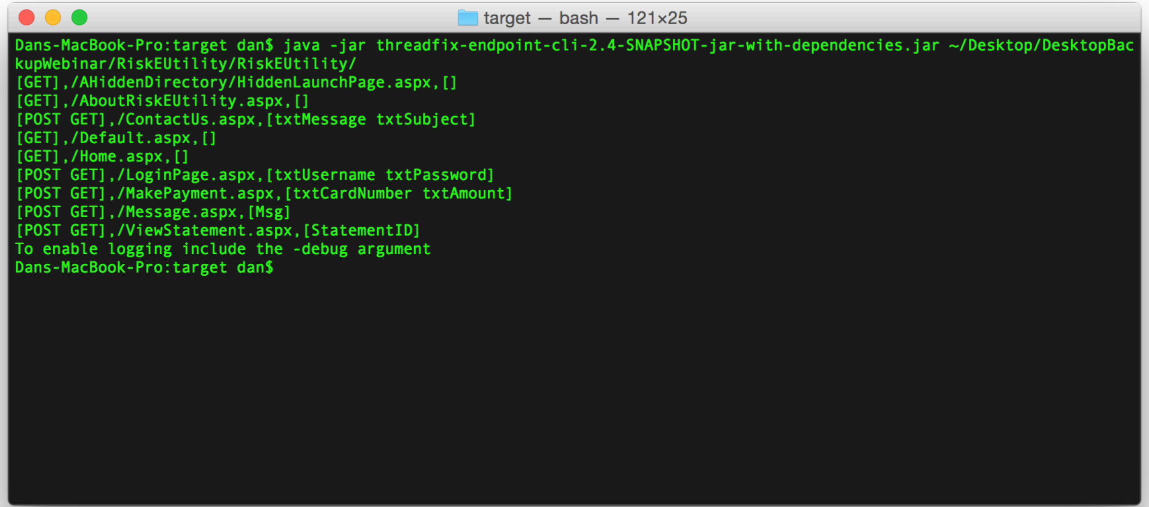
- Syntax: java –jar [jar-name].jar /path/to/source

- JAR name will change based on build ID
- After Maven build, can also be found in: $GIT/threadfix/threadfix-cli-endpoints/target/
- You want the ”-jar-with-dependencies” JAR

- Will output list of HTTP methods, URLs and parameters based on analysis of the source code
- Attack surface!

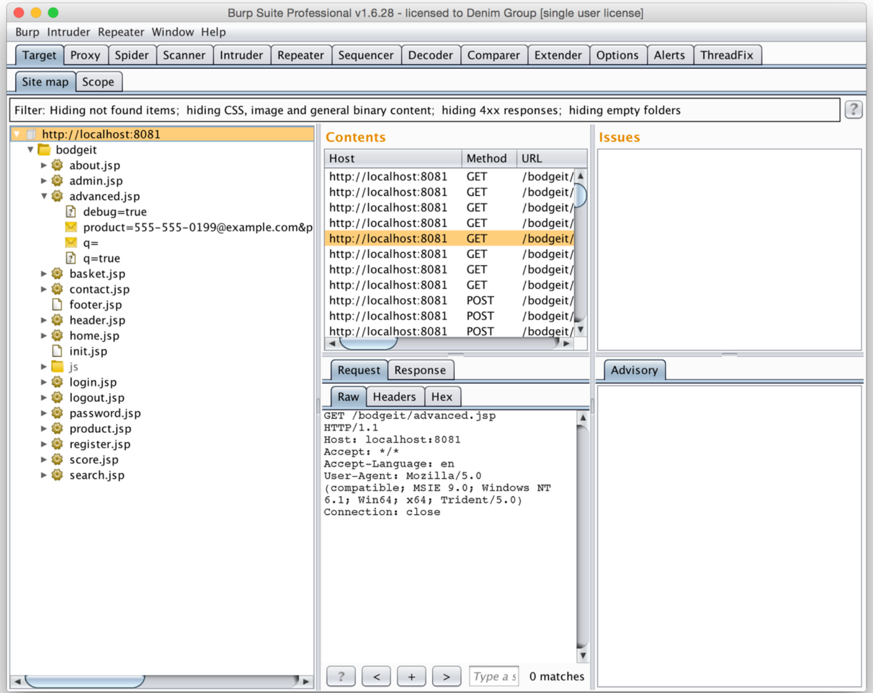
- Add “-json” to the end of the command to get output in JSON format
  - Easier to manipulate

# Command Line Demo



```
Dans-MacBook-Pro:target dan$ java -jar threadfix-endpoint-cli-2.4-SNAPSHOT-jar-with-dependencies.jar ~/Desktop/DesktopBac
kupWebinar/RiskEUtility/RiskEUtility/
[GET],/AHiddenDirectory/HiddenLaunchPage.aspx,[]
[GET],/AboutRiskEUtility.aspx,[]
[POST GET],/ContactUs.aspx,[txtMessage txtSubject]
[GET],/Default.aspx,[]
[GET],/Home.aspx,[]
[POST GET],/LoginPage.aspx,[txtUsername txtPassword]
[POST GET],/MakePayment.aspx,[txtCardNumber txtAmount]
[POST GET],/Message.aspx,[Msg]
[POST GET],/ViewStatement.aspx,[StatementID]
To enable logging include the -debug argument
Dans-MacBook-Pro:target dan$
```

# Scanner Attack Surface Seeding Demo


Burp Suite Professional v1.6.28 - licensed to Denim Group [single user license]
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts ThreadFix
Site map Scope
Contents
Issues
Request Response
Raw Headers Hex
Advisory
? < + > Type a s 0 matches

Filter: Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

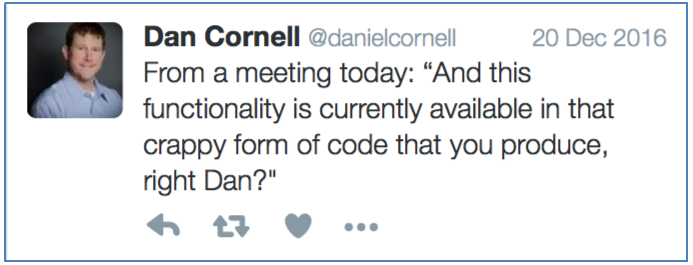
```
http://localhost:8081
  bodgeit
    about.jsp
    admin.jsp
    advanced.jsp
      debug=true
      product=555-555-0199@example.com&p
      q=
      q=true
    basket.jsp
    contact.jsp
    footer.jsp
    header.jsp
    home.jsp
    init.jsp
    js
    login.jsp
    logout.jsp
    password.jsp
    product.jsp
    register.jsp
    score.jsp
    search.jsp
```

| Host | Method | URL |
|---|---|---|
| http://localhost:8081 | GET | /bodgeit/ |
| http://localhost:8081 | GET | /bodgeit/ |
| http://localhost:8081 | GET | /bodgeit/ |
| http://localhost:8081 | GET | /bodgeit/ |
| http://localhost:8081 | GET | /bodgeit/ |
| http://localhost:8081 | GET | /bodgeit/ |
| http://localhost:8081 | GET | /bodgeit/ |
| http://localhost:8081 | GET | /bodgeit/ |
| http://localhost:8081 | POST | /bodgeit/ |
| http://localhost:8081 | POST | /bodgeit/ |
| http://localhost:8081 | POST | /bodgeit/ |

```
GET /bodgeit/advanced.jsp
HTTP/1.1
Host: localhost:8081
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0
(compatible; MSIE 9.0; Windows NT
6.1; Win64; x64; Trident/5.0)
Connection: close
```

# attack_surface_lib.py

- Warning!



Dan Cornell @danielcornell 20 Dec 2016

From a meeting today: “And this functionality is currently available in that crappy form of code that you produce, right Dan?"

- What’s the opposite of ”Pythonic?”
- Race conditions, sloppy file handling, etc
- Possibly even some command injection
  - That you can currently exploit from … the command line
  - Some mitigations in place, but…
- Please be careful what you attach this to

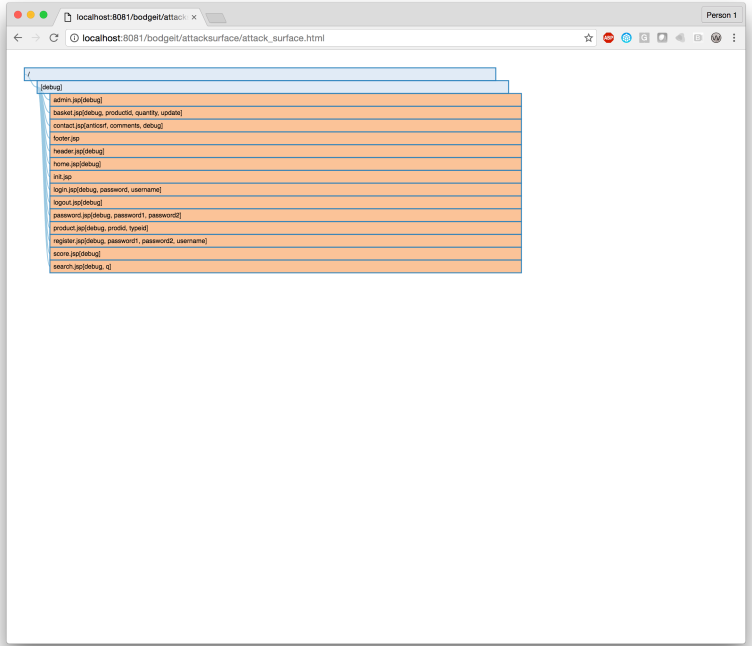
# attack_surface_lib.py

- What does it do?
  - Takes JSON output of cli-endpoints
    - Creates attack surface tree data structure
    - Calculates differences between trees
  - Some git utility tasks

- Used as the basis for upcoming examples

- https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/attack_surface_lib.py

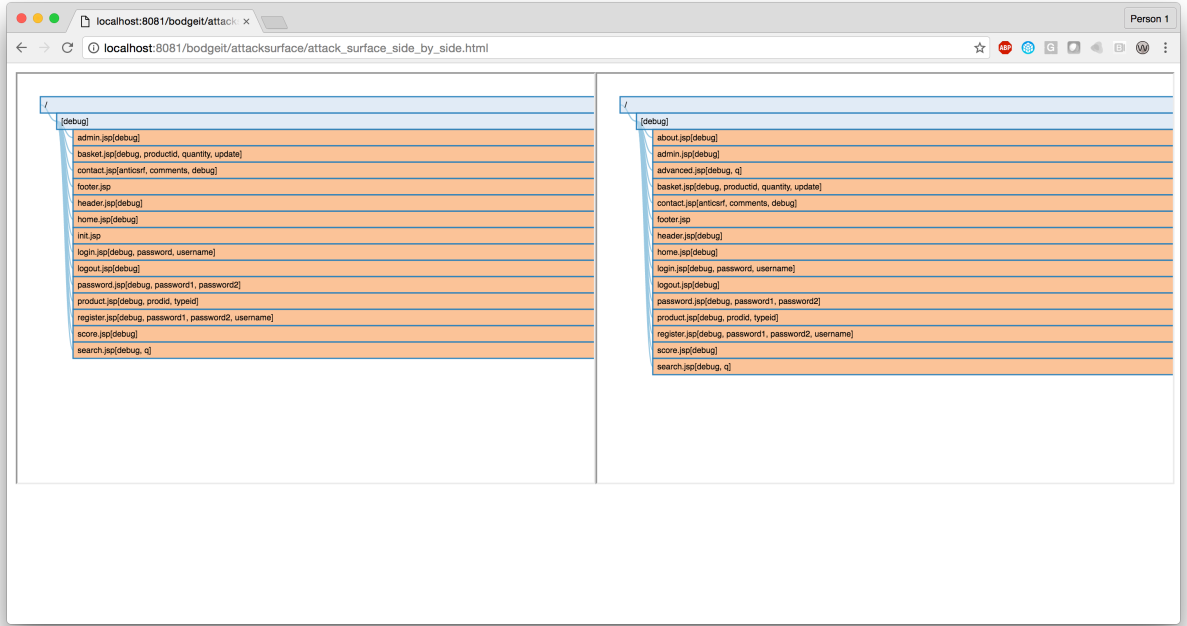
# Attack Surface Visualization Demo


localhost:8081/bodgeit/attack
Person 1
localhost:8081/bodgeit/attacksurface/attack_surface.html

```
/
  [debug]
    admin.jsp[debug]
    basket.jsp[debug, productid, quantity, update]
    contact.jsp[anticsrf, comments, debug]
    footer.jsp
    header.jsp[debug]
    home.jsp[debug]
    init.jsp
    login.jsp[debug, password, username]
    logout.jsp[debug]
    password.jsp[debug, password1, password2]
    product.jsp[debug, prodid, typeid]
    register.jsp[debug, password1, password2, username]
    score.jsp[debug]
    search.jsp[debug, q]
```

HTML framework: https://github.com/denimgroup/threadfix-examples/tree/master/web_app_attack_surface/html
Code: https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/make_d3_tree_json.py

# Attack Surface Comparison Visualization Demo



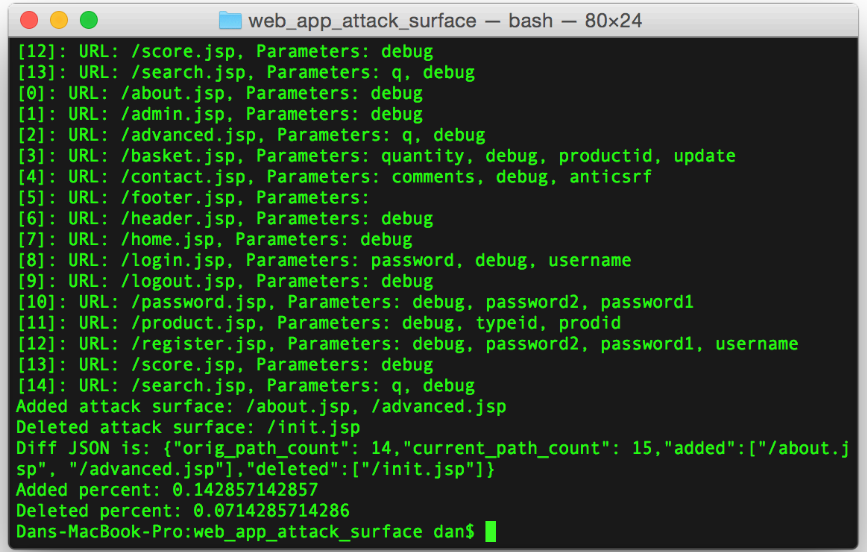
localhost:8081/bodgeit/attacksurface/attack_surface_side_by_side.html

```
/
  [debug]
    admin.jsp[debug]
    basket.jsp[debug, productid, quantity, update]
    contact.jsp[anticsrf, comments, debug]
    footer.jsp
    header.jsp[debug]
    home.jsp[debug]
    init.jsp
    login.jsp[debug, password, username]
    logout.jsp[debug]
    password.jsp[debug, password1, password2]
    product.jsp[debug, prodid, typeid]
    register.jsp[debug, password1, password2, username]
    score.jsp[debug]
    search.jsp[debug, q]
```

```
/
  [debug]
    about.jsp[debug]
    admin.jsp[debug]
    advanced.jsp[debug, q]
    basket.jsp[debug, productid, quantity, update]
    contact.jsp[anticsrf, comments, debug]
    footer.jsp
    header.jsp[debug]
    home.jsp[debug]
    login.jsp[debug, password, username]
    logout.jsp[debug]
    password.jsp[debug, password1, password2]
    product.jsp[debug, prodid, typeid]
    register.jsp[debug, password1, password2, username]
    score.jsp[debug]
    search.jsp[debug, q]
```

HTML framework: [https://github.com/denimgroup/threadfix-examples/tree/master/web_app_attack_surface/html](https://github.com/denimgroup/threadfix-examples/tree/master/web_app_attack_surface/html)
Code: [https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/make_d3_tree_json.py](https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/make_d3_tree_json.py)

## Diffing Attack Surface Demo



```
[12]: URL: /score.jsp, Parameters: debug
[13]: URL: /search.jsp, Parameters: q, debug
[0]: URL: /about.jsp, Parameters: debug
[1]: URL: /admin.jsp, Parameters: debug
[2]: URL: /advanced.jsp, Parameters: q, debug
[3]: URL: /basket.jsp, Parameters: quantity, debug, productid, update
[4]: URL: /contact.jsp, Parameters: comments, debug, anticsrf
[5]: URL: /footer.jsp, Parameters:
[6]: URL: /header.jsp, Parameters: debug
[7]: URL: /home.jsp, Parameters: debug
[8]: URL: /login.jsp, Parameters: password, debug, username
[9]: URL: /logout.jsp, Parameters: debug
[10]: URL: /password.jsp, Parameters: debug, password2, password1
[11]: URL: /product.jsp, Parameters: debug, typeid, prodid
[12]: URL: /register.jsp, Parameters: debug, password2, password1, username
[13]: URL: /score.jsp, Parameters: debug
[14]: URL: /search.jsp, Parameters: q, debug
Added attack surface: /about.jsp, /advanced.jsp
Deleted attack surface: /init.jsp
Diff JSON is: {"orig_path_count": 14,"current_path_count": 15,"added":["/about.j
sp", "/advanced.jsp"],"deleted":["/init.jsp"]}
Added percent: 0.142857142857
Deleted percent: 0.0714285714286
Dans-MacBook-Pro:web_app_attack_surface dan$
```

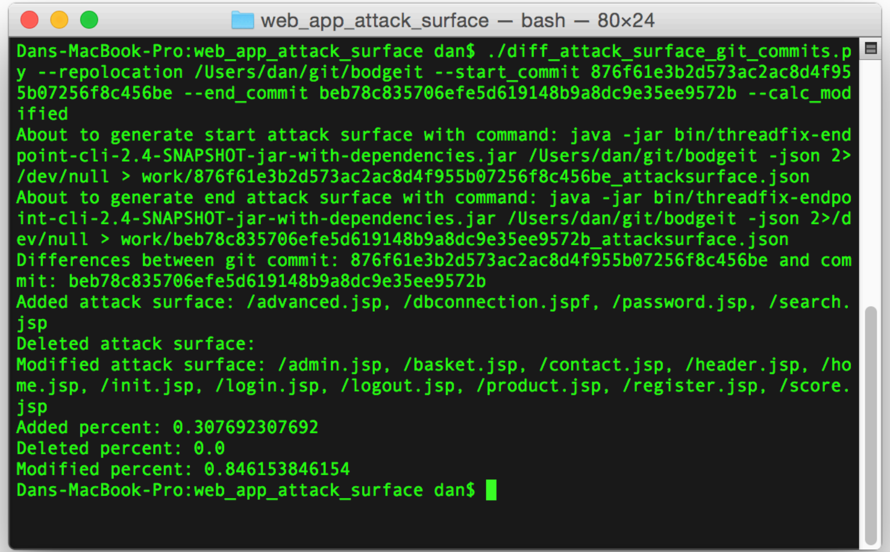
Code: https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/diff_attack_surface_git_commits.py

## What About Behavior Changes?

- Identify files that have changed that are associated with attack surface
- Mark that attack surface as possibly having changed behavior

- Is this perfect? No.
- Does it provide additional information with potential value? Yes.

# Potential Behavior Modified Demo



```
Dans-MacBook-Pro:web_app_attack_surface dan$ ./diff_attack_surface_git_commits.p
y --repolocation /Users/dan/git/bodgeit --start_commit 876f61e3b2d573ac2ac8d4f95
5b07256f8c456be --end_commit beb78c835706efe5d619148b9a8dc9e35ee9572b --calc_mod
ified
About to generate start attack surface with command: java -jar bin/threadfix-end
point-cli-2.4-SNAPSHOT-jar-with-dependencies.jar /Users/dan/git/bodgeit -json 2>
/dev/null > work/876f61e3b2d573ac2ac8d4f955b07256f8c456be_attacksurface.json
About to generate end attack surface with command: java -jar bin/threadfix-endpo
int-cli-2.4-SNAPSHOT-jar-with-dependencies.jar /Users/dan/git/bodgeit -json 2>/d
ev/null > work/beb78c835706efe5d619148b9a8dc9e35ee9572b_attacksurface.json
Differences between git commit: 876f61e3b2d573ac2ac8d4f955b07256f8c456be and com
mit: beb78c835706efe5d619148b9a8dc9e35ee9572b
Added attack surface: /advanced.jsp, /dbconnection.jspf, /password.jsp, /search.
jsp
Deleted attack surface:
Modified attack surface: /admin.jsp, /basket.jsp, /contact.jsp, /header.jsp, /ho
me.jsp, /init.jsp, /login.jsp, /logout.jsp, /product.jsp, /register.jsp, /score.
jsp
Added percent: 0.307692307692
Deleted percent: 0.0
Modified percent: 0.846153846154
Dans-MacBook-Pro:web_app_attack_surface dan$ 
```

# Applications for DevOps Pipelines

- Target DAST testing to focus on new attack surface in latest build
  - “Run an authenticated ZAP scan against the three new URLs added in the last commit”

- Set thresholds for when manual assessment/penetration testing is triggered
  - “Schedule a manual penetration test when the attack surface has increased by 10 URLs”
  - “Schedule a manual penetration test when the attack surface has increased by 5%”
  - Focus those efforts on new attack surface

- ChatOps: Attack surface delta notifications on commit
  - “Commit beb78c835706efe5d619148b9a8dc9e35ee9572b added attack surface: /advanced.jsp, /preferenes.jsp”

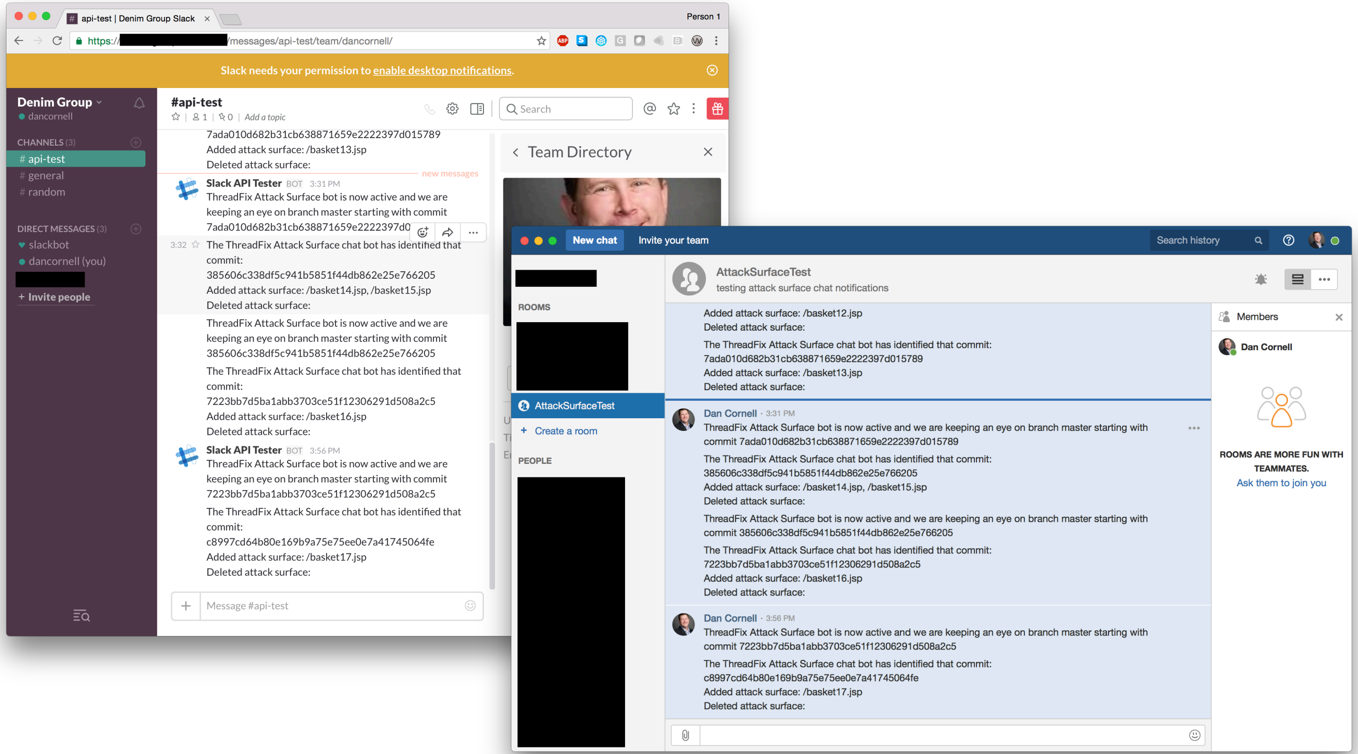
# attacksurface_notifier.py

- Watch a git repository for new commits
- When there are commits, check for attack surface changes
- On attack surface changes – do stuff

- In production: would be done via CI/CD server
- BUT for demo purposes…

- https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/attacksurface_notifier.py

# Attack Surface ChatOps Demo



7ada010d682b31cb638871659e2222397d015789
Added attack surface: /basket13.jsp
Deleted attack surface:

**Slack API Tester** BOT 3:31 PM
ThreadFix Attack Surface bot is now active and we are keeping an eye on branch master starting with commit 7ada010d682b31cb638871659e2222397d0
3:32 The ThreadFix Attack Surface chat bot has identified that commit:
385606c338df5c941b5851f44db862e25e766205
Added attack surface: /basket14.jsp, /basket15.jsp
Deleted attack surface:

ThreadFix Attack Surface bot is now active and we are keeping an eye on branch master starting with commit 385606c338df5c941b5851f44db862e25e766205
The ThreadFix Attack Surface chat bot has identified that commit:
7223bb7d5ba1abb3703ce51f12306291d508a2c5
Added attack surface: /basket16.jsp
Deleted attack surface:

**Slack API Tester** BOT 3:56 PM
ThreadFix Attack Surface bot is now active and we are keeping an eye on branch master starting with commit 7223bb7d5ba1abb3703ce51f12306291d508a2c5
The ThreadFix Attack Surface chat bot has identified that commit:
c8997cd64b80e169b9a75e75ee0e7a41745064fe
Added attack surface: /basket17.jsp
Deleted attack surface:

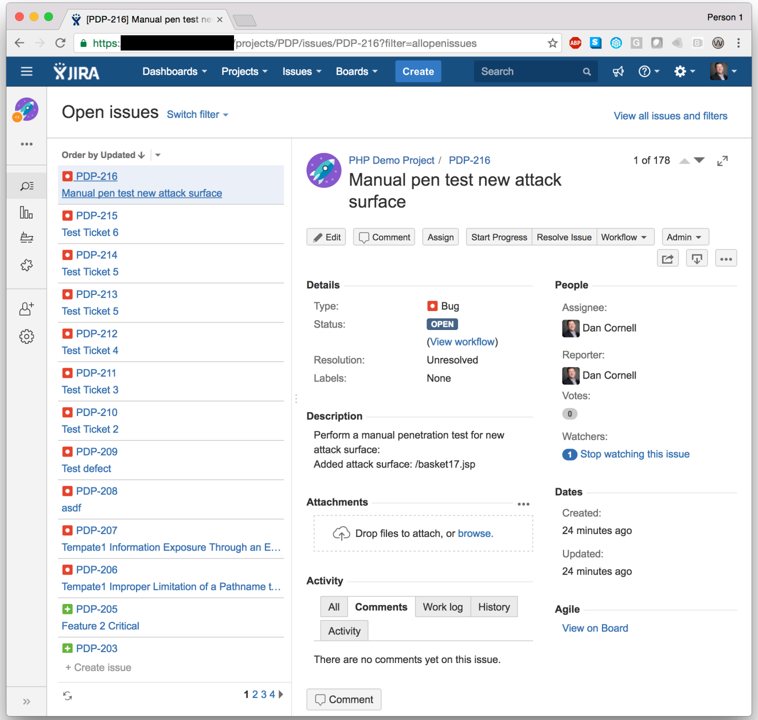
Added attack surface: /basket12.jsp
Deleted attack surface:
The ThreadFix Attack Surface chat bot has identified that commit:
7ada010d682b31cb638871659e2222397d015789
Added attack surface: /basket13.jsp
Deleted attack surface:

**Dan Cornell** · 3:31 PM
ThreadFix Attack Surface bot is now active and we are keeping an eye on branch master starting with commit 7ada010d682b31cb638871659e2222397d015789
The ThreadFix Attack Surface chat bot has identified that commit:
385606c338df5c941b5851f44db862e25e766205
Added attack surface: /basket14.jsp, /basket15.jsp
Deleted attack surface:
ThreadFix Attack Surface bot is now active and we are keeping an eye on branch master starting with commit 385606c338df5c941b5851f44db862e25e766205
The ThreadFix Attack Surface chat bot has identified that commit:
7223bb7d5ba1abb3703ce51f12306291d508a2c5
Added attack surface: /basket16.jsp
Deleted attack surface:

**Dan Cornell** · 3:56 PM
ThreadFix Attack Surface bot is now active and we are keeping an eye on branch master starting with commit 7223bb7d5ba1abb3703ce51f12306291d508a2c5
The ThreadFix Attack Surface chat bot has identified that commit:
c8997cd64b80e169b9a75e75ee0e7a41745064fe
Added attack surface: /basket17.jsp
Deleted attack surface:

Code: [https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/attacksurface_notifier.py](https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/attacksurface_notifier.py)

# Manual Test JIRA Ticket Demo



[PDP-216] Manual pen test ne

Person 1

https:[redacted]/projects/PDP/issues/PDP-216?filter=allopenissues

Open issues Switch filter

View all issues and filters

Order by Updated

| | |
|---|---|
| PDP-216 | Manual pen test new attack surface |
| PDP-215 | Test Ticket 6 |
| PDP-214 | Test Ticket 5 |
| PDP-213 | Test Ticket 5 |
| PDP-212 | Test Ticket 4 |
| PDP-211 | Test Ticket 3 |
| PDP-210 | Test Ticket 2 |
| PDP-209 | Test defect |
| PDP-208 | asdf |
| PDP-207 | Tempate1 Information Exposure Through an E… |
| PDP-206 | Tempate1 Improper Limitation of a Pathname t… |
| PDP-205 | Feature 2 Critical |
| PDP-203 | |

+ Create issue

1 2 3 4

PHP Demo Project / PDP-216

1 of 178

Manual pen test new attack surface

**Details**

| | |
|---|---|
| Type: | Bug |
| Status: | OPEN (View workflow) |
| Resolution: | Unresolved |
| Labels: | None |

**Description**

Perform a manual penetration test for new attack surface:
Added attack surface: /basket17.jsp

**Attachments**

Drop files to attach, or browse.

**Activity**

There are no comments yet on this issue.

**People**

Assignee:
Dan Cornell

Reporter:
Dan Cornell

Votes:
0

Watchers:
1 Stop watching this issue

**Dates**

Created:
24 minutes ago

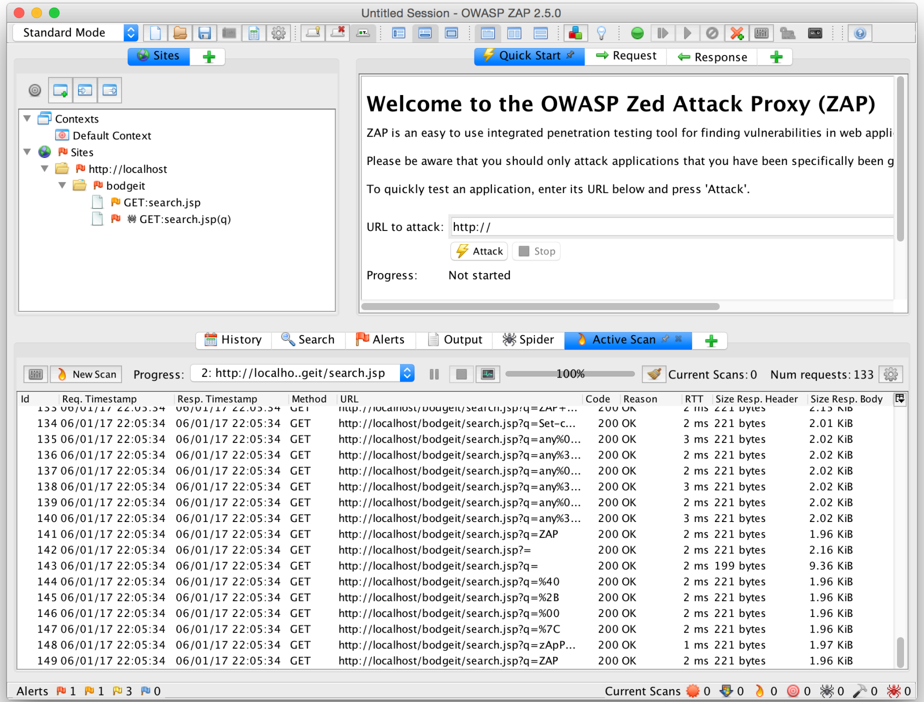
Updated:
24 minutes ago

**Agile**

View on Board

Code: https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/attacksurface_notifier.py

# Differential ZAP Scan Demo



**Welcome to the OWASP Zed Attack Proxy (ZAP)**

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web appli

Please be aware that you should only attack applications that you have been specifically been g

To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://

Attack  Stop

Progress: Not started

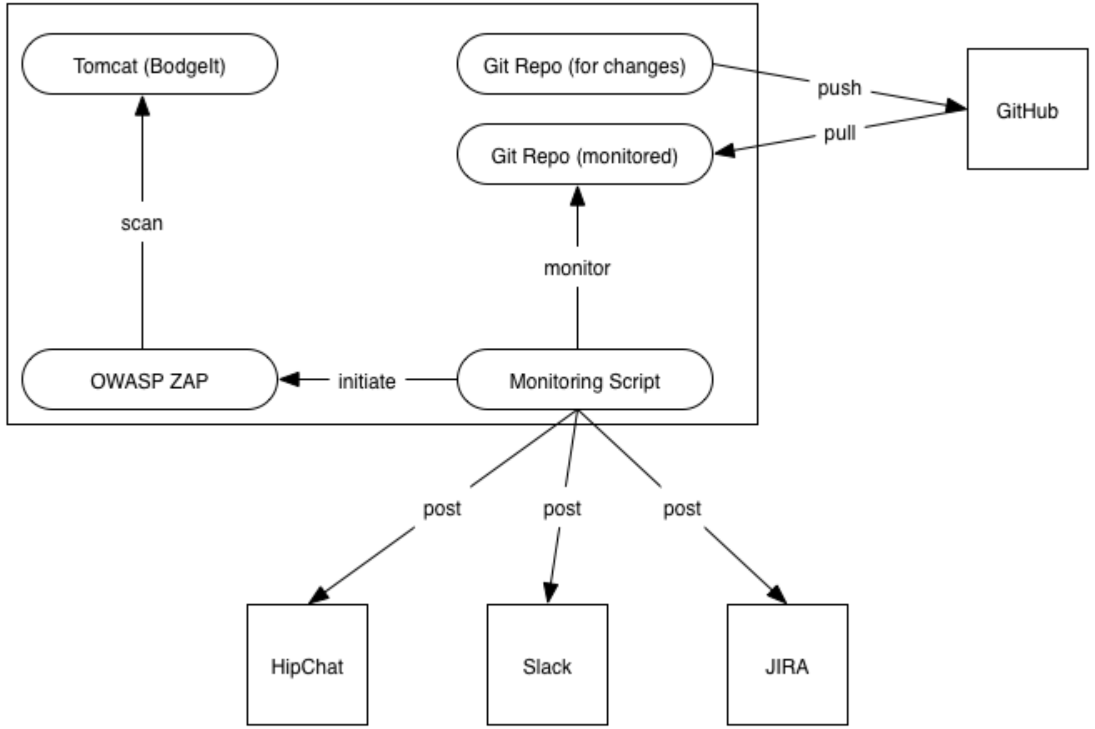
Progress: 2: http://localho..geit/search.jsp  100%  Current Scans: 0  Num requests: 133

| Id | Req. Timestamp | Resp. Timestamp | Method | URL | Code | Reason | RTT | Size Resp. Header | Size Resp. Body |
|---|---|---|---|---|---|---|---|---|---|
| 133 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=ZAP+... | 200 | OK | 2 ms | 221 bytes | 2.15 KiB |
| 134 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=Set-c... | 200 | OK | 2 ms | 221 bytes | 2.01 KiB |
| 135 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=any%0... | 200 | OK | 3 ms | 221 bytes | 2.02 KiB |
| 136 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=any%3... | 200 | OK | 2 ms | 221 bytes | 2.02 KiB |
| 137 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=any%0... | 200 | OK | 2 ms | 221 bytes | 2.02 KiB |
| 138 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=any%3... | 200 | OK | 3 ms | 221 bytes | 2.02 KiB |
| 139 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=any%0... | 200 | OK | 2 ms | 221 bytes | 2.02 KiB |
| 140 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=any%3... | 200 | OK | 3 ms | 221 bytes | 2.02 KiB |
| 141 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=ZAP | 200 | OK | 2 ms | 221 bytes | 1.96 KiB |
| 142 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?= | 200 | OK | 2 ms | 221 bytes | 2.16 KiB |
| 143 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q= | 200 | OK | 2 ms | 199 bytes | 9.36 KiB |
| 144 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=%40 | 200 | OK | 2 ms | 221 bytes | 1.96 KiB |
| 145 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=%2B | 200 | OK | 2 ms | 221 bytes | 1.96 KiB |
| 146 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=%00 | 200 | OK | 2 ms | 221 bytes | 1.96 KiB |
| 147 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=%7C | 200 | OK | 2 ms | 221 bytes | 1.96 KiB |
| 148 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=zApP... | 200 | OK | 1 ms | 221 bytes | 1.97 KiB |
| 149 | 06/01/17 22:05:34 | 06/01/17 22:05:34 | GET | http://localhost/bodgeit/search.jsp?q=ZAP | 200 | OK | 2 ms | 221 bytes | 1.96 KiB |

Alerts 1 1 3 0  Current Scans 0 0 0 0 0 0 0

Code: https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/attacksurface_notifier.py

# Demo Architecture



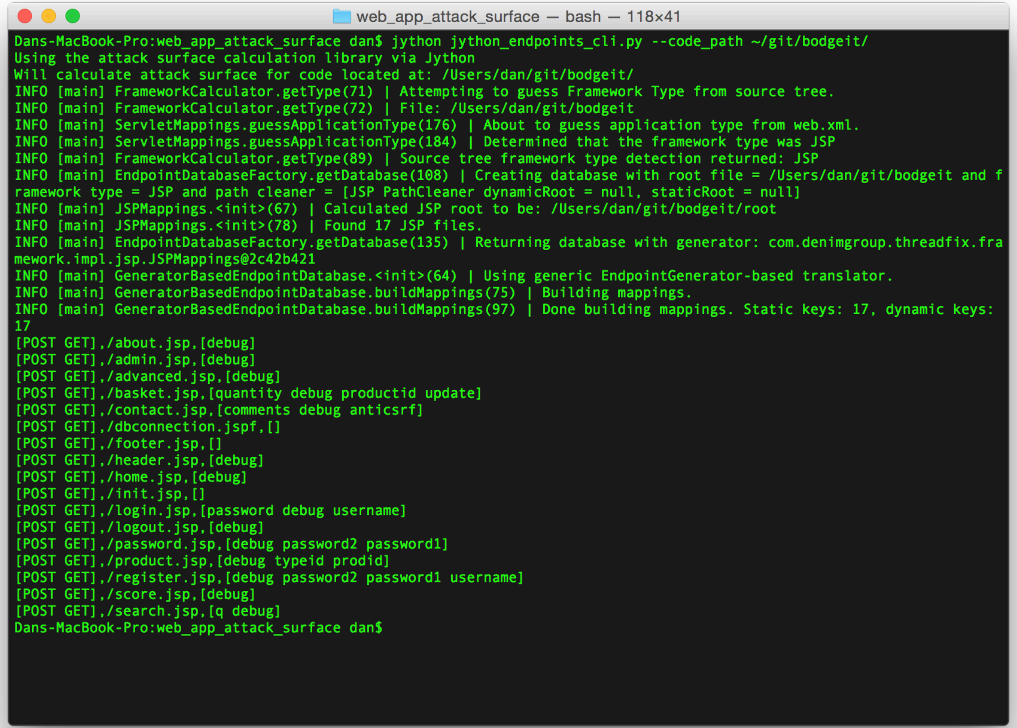
## Scripting Attack Surface Interactions

- [Anywhere]
  - Script using endpoints-cli.jar JSON outputs
    - That’s most of what we’ve seen here
  - Script using JSON output from ThreadFix API
    - Can be useful in environments with limited access to source code

- Java: Use endpoints-cli.jar as a library
  - We need to do a better job of documenting the APIs

- Jython: Use endpoints-cli.jar as a library

# Jython Use of HAM Library Demo



web_app_attack_surface — bash — 118×41

```
Dans-MacBook-Pro:web_app_attack_surface dan$ jython jython_endpoints_cli.py --code_path ~/git/bodgeit/
Using the attack surface calculation library via Jython
Will calculate attack surface for code located at: /Users/dan/git/bodgeit/
INFO [main] FrameworkCalculator.getType(71) | Attempting to guess Framework Type from source tree.
INFO [main] FrameworkCalculator.getType(72) | File: /Users/dan/git/bodgeit
INFO [main] ServletMappings.guessApplicationType(176) | About to guess application type from web.xml.
INFO [main] ServletMappings.guessApplicationType(184) | Determined that the framework type was JSP
INFO [main] FrameworkCalculator.getType(89) | Source tree framework type detection returned: JSP
INFO [main] EndpointDatabaseFactory.getDatabase(108) | Creating database with root file = /Users/dan/git/bodgeit and f
ramework type = JSP and path cleaner = [JSP PathCleaner dynamicRoot = null, staticRoot = null]
INFO [main] JSPMappings.<init>(67) | Calculated JSP root to be: /Users/dan/git/bodgeit/root
INFO [main] JSPMappings.<init>(78) | Found 17 JSP files.
INFO [main] EndpointDatabaseFactory.getDatabase(135) | Returning database with generator: com.denimgroup.threadfix.fra
mework.impl.jsp.JSPMappings@2c42b421
INFO [main] GeneratorBasedEndpointDatabase.<init>(64) | Using generic EndpointGenerator-based translator.
INFO [main] GeneratorBasedEndpointDatabase.buildMappings(75) | Building mappings.
INFO [main] GeneratorBasedEndpointDatabase.buildMappings(97) | Done building mappings. Static keys: 17, dynamic keys:
17
[POST GET],/about.jsp,[debug]
[POST GET],/admin.jsp,[debug]
[POST GET],/advanced.jsp,[debug]
[POST GET],/basket.jsp,[quantity debug productid update]
[POST GET],/contact.jsp,[comments debug anticsrf]
[POST GET],/dbconnection.jspf,[]
[POST GET],/footer.jsp,[]
[POST GET],/header.jsp,[debug]
[POST GET],/home.jsp,[debug]
[POST GET],/init.jsp,[]
[POST GET],/login.jsp,[password debug username]
[POST GET],/logout.jsp,[debug]
[POST GET],/password.jsp,[debug password2 password1]
[POST GET],/product.jsp,[debug typeid prodid]
[POST GET],/register.jsp,[debug password2 password1 username]
[POST GET],/score.jsp,[debug]
[POST GET],/search.jsp,[q debug]
Dans-MacBook-Pro:web_app_attack_surface dan$
```

Code: https://github.com/denimgroup/threadfix-examples/blob/master/web_app_attack_surface/jython_endpoints_cli.py

# Next Steps

- Expand the model of application attack surface
  - Currently: Parameters, HTTP verbs
  - Working on: HTTP headers (cookies)
  - Future: Other application types:  Mobile, IoT

- Better visualization
  - More details
  - Better granularity
  - Track changes over time

- Native integrations: Jenkins, Slack, HipChat, JIRA, etc
  - This is very “scripty” right now

## Questions / Contact Information

**Dan Cornell**

Principal and CTO

dan@denimgroup.com

Twitter @danielcornell

(844) 572-4400

www.threadfix.it