BlackHat Webinar

Dan Cornell
CTO, Denim Group
@danielcornell
The Problem

• Application security testing typically uses automated static and dynamic test results as well as manual testing results to assess the security of an application
• Each test delivers results in different formats
• Different test platforms also can describe the same flaws differently, creating multiple duplications
• Security teams end up using spreadsheets to keep track manually
• It is extremely difficult to prioritize the severity of flaws as a result
• Software development teams receive unmanageable reports and only a small portion of the flaws get fixed
The Result

• Application vulnerabilities persist in applications
  – The average number of serious vulnerabilities found per website per year is 79
  – The average number of days a website is exposed to at least one serious vulnerability is 231 days
  – The overall percentage of serious vulnerabilities that are fixed annually is only 63%
• Part of that problem is there is no easy way for the security team and application development teams to work together on these issues
• Remediation quickly becomes an overwhelming project
• Trending reports that track the number of reduced vulnerabilities are impossible to create

Sources: [https://www.whitehatsec.com/assets/WPstats_summer12_12th.pdf](https://www.whitehatsec.com/assets/WPstats_summer12_12th.pdf), pages 2 & 3
http://www.veracode.com/reports (registration required)
ThreadFix

Consolidates reports so managers can speak intelligently about the status and trends of security within their organization.
Vulnerability Import

• Pulls in static and dynamic results
• Eliminates duplicate results
• Allows for results to be grouped
Real-Time Protection

Virtual patching helps protect organizations during remediation
Defect Tracking Integration

- ThreadFix can connect to common defect trackers
- Defects can be created for developers
- Work can continue uninterrupted
Product Demonstration
The Dashboard

- Lists all the development teams in the organization including number of apps for each team and a summary of the security status of those apps.
- Clicking on a team reveals the details on the apps that team is working on.

```
<table>
<thead>
<tr>
<th>Team Name</th>
<th>No. of Apps</th>
<th>Open Vulns</th>
<th>Critical</th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information Technology Services (HQ)</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Mobile Innovation R&amp;D</td>
<td>1</td>
<td>13</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Offshore - Mumbai</td>
<td>2</td>
<td>39</td>
<td>10</td>
<td>0</td>
<td>8</td>
<td>21</td>
</tr>
<tr>
<td>Offshore - Pune</td>
<td>3</td>
<td>10</td>
<td>9</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>eCommerce</td>
<td>4</td>
<td>73</td>
<td>29</td>
<td>9</td>
<td>10</td>
<td>25</td>
</tr>
</tbody>
</table>
```
Viewing The Applications By Team

- Now all of the applications managed by the eCommerce team are revealed.
- The security analyst now wants to upload new vulnerability scan data for the "Replacement Part Auctions" application and clicks on that link.
Fixing an eCommerce Team “Auction” Application –

- Vulnerability data from AppScan, Arachani, Netsparker and W3af scans are uploaded into ThreadFix.
Large Range of Tool Compatibility

- Veracode
- HP Webinspect
- PortSwigger web security
- W3af
- Imperva
- Atlassian JIRA
- FindBugs
- Snort
- Skipfish
- Ounce Labs
- Bugzilla
- Acunetix
- Nessus
- HP Enterprise Security
- WhiteHat Security
- Arachni
- Netsparker
Compatible Tool Categories

**Dynamic Scanners**
- Burp Suite
- HP WebInspect
- IBM Rational AppScan
- Mavituna Security Netsparker
- Tenable Nessus
- Acunetix
- OWASP Zed Attack Proxy
- Arachni
- Skipfish

**Static Scanners**
- HP Fortify SCA
- Microsoft CAT.NET
- FindBugs
- Ounce IBM Security AppScan Source

**SaaS Testing Platforms**
- WhiteHat
- Veracode
- QualysGuard WAS 2.0

**Defect Trackers**
- Mozilla Bugzilla
- Atlassian JIRA

**IDS/IPS and WAF**
- F5
- Deny All
- Snort
- mod_security
- Imperva
The ThreadFix Consolidation

- All of the vulnerability scans have been aggregated into ThreadFix providing a centralized view of the security status of the Auction application.
Web Application Firewall Rules Are Generated

• ThreadFix now uses the vulnerability data to automatically generate additional Web Application Firewall (WAF) “virtual patch” rules designed to protect those specific applications and their vulnerabilities.

• Since the additional WAF rules are created based on real vulnerabilities, they greatly strengthen the protection offered by the firewall system.
Protecting the Application While It Is Vulnerable

- The WAF and Intrusion Detection Systems use the ThreadFix generated “virtual patch” rules to isolate application attacks.
- The ThreadFix user can analyze this attack data to further fine-tune the WAF to actively block application exploit attempts while the application is being fixed.
- Applications are susceptible to fewer risks as a result.
Attack Data Is Also Aggregated in ThreadFix

- The attack data is also imported into ThreadFix to present a more complete picture of the organization’s security profile.

### ThreadFix Table

<table>
<thead>
<tr>
<th>Merged</th>
<th>Vulnerability Name</th>
<th>Severity</th>
<th>Path</th>
<th>Parameter</th>
<th>Defect</th>
<th>Defect Status</th>
<th>WAF Rule</th>
<th>WAF Events</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Data Handling</td>
<td>Critical</td>
<td>/stuff/</td>
<td></td>
<td>No Defect</td>
<td></td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')</td>
<td>Critical</td>
<td>/demo/OCommandInjection2.php</td>
<td>fileName</td>
<td>No Defect</td>
<td></td>
<td>Yes</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>Improper Sanitization of Special Elements used in a Command ('Command Injection')</td>
<td>Critical</td>
<td>/demo/OCommandInjection2.php</td>
<td>fileName</td>
<td>292</td>
<td>RESOLVED</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>Improper Sanitization of Special Elements used in an OS Command ('OS Command Injection')</td>
<td>Critical</td>
<td>/demo/OCommandInjection2.php</td>
<td>fileName</td>
<td>292</td>
<td>RESOLVED</td>
<td>Yes</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>Critical</td>
<td>/demo/PathTraversal.php</td>
<td>action</td>
<td>No Defect</td>
<td></td>
<td>Yes</td>
<td>136</td>
</tr>
<tr>
<td>3</td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>Critical</td>
<td>/demo/XPathInjection2.php</td>
<td>username</td>
<td></td>
<td></td>
<td>Yes</td>
<td>35</td>
</tr>
<tr>
<td>3</td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>Critical</td>
<td>/demo/XSS-reflected2.php</td>
<td>username</td>
<td></td>
<td></td>
<td>Yes</td>
<td>30</td>
</tr>
<tr>
<td>3</td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>Critical</td>
<td>/demo/XPathInjection2.php</td>
<td>password</td>
<td></td>
<td></td>
<td>Yes</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>Critical</td>
<td>/demo/EvalInjection2.php</td>
<td>command</td>
<td></td>
<td></td>
<td>Yes</td>
<td>20</td>
</tr>
</tbody>
</table>
The Negotiations Begin

• The ThreadFix aggregated data report for the Auction application provides the basis needed to decide what is to be fixed and by who.

• The security analyst and the eCommerce development team leader use the report which includes both vulnerability and attack data to decide which vulnerabilities will get fixed and which vulnerabilities represent an acceptable risk to the organization.

• Next, the two team leaders agree on how to best package the targeted vulnerabilities for the development team:
  – By type (i.e. Cross Site Scripting vulnerabilities because it’s more efficient to fix a class of vulnerabilities regardless of where they are located in the application.)
  – By developer (i.e. Joe created the user interface and is the only developer that knows how to work in that part of the application)
  – By severity (i.e. the critical vulnerabilities that need to be fixed now.)
  – Or any combination of the above.
Agreeing On The Workload

• An example of bundling the Critical Severity identified vulnerabilities into a single defect to prioritize the remediation of the application.
The Defect Tracking System

- The security analyst exports vulnerabilities with Critical Severity to the Defect Tracking System which is Bugzilla in this example.
- The eCommerce development team then uses Bugzilla to keep track of the outstanding bugs and management tasks still to be done.

![Bugzilla - Bug List](image)
Vulnerabilities Now Become Defects

- All the vulnerabilities to be fixed are packaged in a manner that makes sense to the development team’s work process.
- These vulnerabilities, which are now recognized as defects to software developers, are transferred to Bugzilla, the platform the development team is used to using.
The Defect Categories & Status Inside of ThreadFix

- At the same time, the security analyst can see all of the open vulnerabilities as well as the defects they are linked to.
- Currently none of the bugs have been resolved by the development team.

<table>
<thead>
<tr>
<th>First Defect</th>
<th>Second Defect</th>
<th>Third Defect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Improper Sanitization of Special Elements used in an OS Command ('OS Command Injection')</td>
<td>Failure to Control Generation of Code ('Code Injection')</td>
<td>Improper Control of Resource Identifiers ('Resource Injection')</td>
</tr>
<tr>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>Information Leak Through Include Source Code</td>
</tr>
<tr>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td></td>
</tr>
<tr>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td></td>
</tr>
<tr>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td></td>
</tr>
</tbody>
</table>

- **First Defect**
  - **Vulnerability Name**: Improper Sanitization of Special Elements used in an OS Command ('OS Command Injection')
  - **Severity**: Critical
  - **Path**: /demo/OSCommandInjection2.php
  - **Parameter**: fileName
  - **Defect**: 289
  - **Defect Status**: OPEN
  - **WAF Rule**: No
  - **WAF Events**: 0

- **Second Defect**
  - **Vulnerability Name**: Failure to Control Generation of Code ('Code Injection')
  - **Severity**: Critical
  - **Path**: /demo/EvalInjection2.php
  - **Parameter**: command
  - **Defect**: 289
  - **Defect Status**: OPEN
  - **WAF Rule**: No
  - **WAF Events**: 0

- **Third Defect**
  - **Vulnerability Name**: Improper Control of Resource Identifiers ('Resource Injection')
  - **Severity**: High
  - **Path**: /demo/OSCommandInjection2.php
  - **Parameter**: fileName
  - **Defect**: 291
  - **Defect Status**: OPEN
  - **WAF Rule**: No
  - **WAF Events**: 0

- **Third Defect**
  - **Vulnerability Name**: Information Leak Through Include Source Code
  - **Severity**: Medium
  - **Path**: /demo/OSCommandInjection2.php
  - **Parameter**: fileName
  - **Defect**: 291
  - **Defect Status**: OPEN
  - **WAF Rule**: No
  - **WAF Events**: 0
A Defect (Security Vulnerability) Is Fixed (Or is it?)

- The developers look into the bug containing the Critical vulnerabilities.
- They work with representatives from security to resolve the issue and then mark the bug as fixed in Bugzilla.
Bugzilla Updates Are Synchronized With ThreadFix

- When a ThreadFix update is performed, Bugzilla’s developer notes regarding bug status are synchronized with ThreadFix.
- The security team then performs additional scans to confirm that the bugs have, indeed, been fixed.

<table>
<thead>
<tr>
<th>If Merged</th>
<th>Vulnerability Name</th>
<th>Severity</th>
<th>Path</th>
<th>Parameter</th>
<th>Defect</th>
<th>Defect Status</th>
<th>WAF Rule</th>
<th>WAF Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Improper Sanitization of Special Elements used in an OS Command ('OS Command Injection')</td>
<td>Critical</td>
<td>/demo/OSCommandInjection2.php</td>
<td>fileName</td>
<td>289</td>
<td>Resolved</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Failure to Control Generation of Code ('Code Injection')</td>
<td>Critical</td>
<td>/demo/EvalInjection2.php</td>
<td></td>
<td></td>
<td>Resolved</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>High</td>
<td>/demo/XPathInjection2.php</td>
<td>password</td>
<td>290</td>
<td>New</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>High</td>
<td>/demo/EvalInjection2.php</td>
<td>command</td>
<td>290</td>
<td>New</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Failure to Preserve Web Page Structure ('Cross-site Scripting')</td>
<td>High</td>
<td>/demo/XSS-reflected2.php</td>
<td>username</td>
<td>290</td>
<td>New</td>
<td>No</td>
<td>0</td>
</tr>
</tbody>
</table>
Trending Reports Help Improve Quality

By repeating this process over time, the security teams can start to collect trending data about vulnerabilities as well as statistics of how long it is taking to resolve security issues.
ThreadFix Feature Summary

• **Vulnerability Import**
  - Imports dynamic, static and manual testing results from a variety of sources (both commercial and freely-available scanning tools as well as SaaS testing providers)
  - Correlates and normalizes application vulnerabilities across different sources

• **Defect Tracking Integration**
  - Allows application security teams to group vulnerabilities into individual defects

• **Real-Time Protection Generation**
  - Virtual patching provides protection while code-level fixes are in development
  - Application-specific rules based upon identified vulnerabilities

• **Application Portfolio Management**
  - Tracks security status of applications across the enterprise
  - Enables critical communication with developers in tools they are already using

• **Maturity Evaluation**
  - Store and report on software security program progress
  - Benchmarks security improvement against industry standards
ThreadFix Benefits

- Reduces the time required to fix vulnerable applications.
- Dramatically simplifies the effort required
- Compares the relative performance and test coverage of application vulnerability scanning technologies.
- Provides centralized visibility into current security state of applications as well as trending
- Facilitates communication between security analysts and development teams
- Provides enterprise-wide software security metrics in support of benchmarking and budget justification efforts
- No licensing fees
- Open community support
Where to Get ThreadFix

- Click on the Threadfix.bat icon in Windows, or, in Linux, navigate to the folder and execute bash threadfix.sh.
- Go on the wiki and open the “Getting Started” file for more step by step directions.
- For more information, go to [http://www.denimgroup.com/threadfix](http://www.denimgroup.com/threadfix)
Contact Information

Dan Cornell
Principal and CTO
dan@denimgroup.com
Twitter @danielcornell
(210) 572-4400

www.denimgroup.com
www.threadstrong.com
blog.denimgroup.com