Secure DevOps: A Puma’s Tail

SANS Secure DevOps Summit
Tuesday, October 10th 2017
Eric Johnson (@emjohn20)
Eric Johnson, CISSP, GSSP, GWAPT

Cypress Data Defense
• Principal Security Consultant
• Static code analysis
• Web & mobile app dynamic assessments
• Secure Development Lifecycle
• Tools development
  - Puma Scan .NET
  - SHIM

SANS Institute
• Certified Instructor
  - DEV541: Secure Coding in Java
  - DEV534: Secure DevOps
• Course Author
  - DEV531: Mobile App Security Essentials
  - DEV540: Secure DevOps & Cloud Application Security
  - DEV544: Secure Coding in .NET

©2017 – Puma Security, LLC
Roadmap

• The DevOps Problem
• Security Unit / Integration Testing
• Static Analysis Options
• Pre-Commit
• Continuous Integration
• Conclusion
Case Study | Travel Industry Breaches

• Airline Company
• 5,000 employees
• 50 .NET C# applications
• 20 software engineers
• 0 application security engineers
• 1 deployment / week

©2017 – Puma Security, LLC
Case Study | State of DevOps

• Continuous Integration via Jenkins
• Continuous Delivery via Jenkins pipeline plugin
Case Study | State of Security

- External vendor performing annual code assessments
- Internal security team receives 1,000 page PDF report
- Internal security team performs dynamic pen testing, fuzzing, etc.
Case Study | The Problem

- Security was not invited to the DevOps party
- Internal security team does not have development background
- Frequent deployments invalidate assessment results
- Missing a huge opportunity for app sec in the pipeline
Published October 2016
Release frequency up 30x
Silos still exist between Sec and DevOps
What is SecDevOps

SecDevOps / DevSecOps / DevOpsSec is about breaking down walls between security and:

• Development
• Operations
• Business

©2017 – Puma Security, LLC
SecDevOps Security Controls

- Security controls in a Continuous Integration (CI) / Continuous Delivery (CD) pipeline:

- Security stories
- Threat modeling
- IDE SAST
- Code reviews
- Security unit tests
- CI SAST
- High risk code alerts
- Supply chain scans
- Security acceptance tests
- CI DAST
- Automated security attacks
- Manual pen testing
- Automate deployment
- Security monitoring
- Security smoke testing
- Virtual patching

©2017 – Puma Security, LLC
SecDevOps Quick Wins

- Narrowing the scope and identifying some quick wins for our case study:

**PRE-COMMIT**
- SECURITY STORIES
- THREAT MODELING
- IDE SAST
- CODE REVIEWS

**COMMIT**
- SECURITY UNIT TESTS
- CI SAST
- HIGH RISK CODE ALERTS
- SUPPLY CHAIN SCANS

**ACCEPTANCE**
- SECURITY ACCEPTANCE TESTS
- CI DAST
- AUTOMATED SECURITY ATTACKS
- MANUAL PEN TESTING

**DEPLOYMENT**
- AUTOMATE DEPLOYMENT
- SECURITY MONITORING
- SECURITY SMOKE TESTING
- VIRTUAL PATCHING

©2017 – Puma Security, LLC
Roadmap

- The DevOps Problem
- **Security Unit / Integration Testing**
- Static Analysis Options
- Pre-Commit
- Continuous Integration
- Conclusion
Security Unit / Integration Testing

Built on top of standard unit and integration tests to enforce security requirements:

• Create abuse cases and evil user stories
• Focus on high risk code and business logic flaws
• Fast execution in the IDE / CI pipeline
Engineers often stay on the "happy path"
Prove the code works under normal usage
Example: positive validation testing for a normal user's name

```csharp
[Fact]
public void NameValidationPositiveTest()
{
    bool isValid = Validator.IsNameValid("Bobby Tables");
    Assert.Equal(isValid, true);
}
```
Security works with engineers to write test cases
Prove the code is secure under abnormal usage
Example: Negative validation w/ evil injection characters

```csharp
[Fact]
public void ValidateNameNegativeTest()
{
    string EVIL = "&<>""\]()|;!=~*/{}#";
    foreach (char s in EVIL.ToArray())
        Assert.Equal(Validator.IsNameValid(s.ToString()), false);
}
```
High Risk Code Examples

Security identifies high risk code performing the following functionality:

- Authentication
- Access control
- Output encoding
- Input validation
- High risk business logic
- Data entitlement checks
- Handling confidential data
- Cryptography
Security Payload Unit Test Repository (SPUTR)

- Automated and configurable security unit testing framework written by @sethlaw
- Payload lists for XSS and Injection
  - SQLi, NoSQL, LDAP, XML, XPath, OS vulnerabilities
- Supports Java Spring, ASP.NET MVC, and Django
- https://github.com/sethlaw/sputr
Roadmap

• The DevOps Problem
• Security Unit / Integration Testing
• **Static Analysis Options**
• Pre-Commit
• Continuous Integration
• Conclusion
Limited opportunity for static analysis in CI & CD pipelines:

- Speed matters (< 10 minutes)
- High accuracy rules
- Low false positive rates
Free / Open Source .NET Options

• CAT.NET
• FxCop
• Visual Studio Code Analysis
• Web Config Security Analyzer
• Custom Roslyn Analyzers
The Target

- Purposely vulnerable eCommerce application
- Contains over 50 different vulnerabilities
- Across two different versions:
  - Web Forms
  - .NET MVC 5
- Contributors:
  - Louis Gardina
  - Eric Johnson
CAT.NET v1.1 Security Benchmark

- Widget Town scan results:
  - 2 XSS, 1 Unvalidated Redirect issues
- CAT.NET is a very limited security scanner
FxCop / Code Analysis Security Benchmark

• Rule target results from the “Microsoft Security Rules” rule set

• Widget Town scan results:
  • 2 SQL Injection instances, 1 is a false positive
Scan Result Summary

- Widget Town combined CAT.NET and VS Code analysis scan results:

<table>
<thead>
<tr>
<th>Category</th>
<th>Valid</th>
<th>False Positive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross-Site Scripting</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>SQL Injection</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Unvalidated Redirect</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

©2017 – Puma Security, LLC
Widget Town combined CAT.NET and VS Code analysis scan results:

<table>
<thead>
<tr>
<th>Category</th>
<th>Valid</th>
<th>False Positive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross-Site Scripting</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>SQL Injection</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Unvalidated Redirect</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Scan Result Summary

FAIL
Introducing Roslyn

• Open-source C# and VB compilers with code analysis APIs
• Capable of producing warnings in source code as you type:

```csharp
// Configure validation logic for passwords
manager.PasswordValidator = new PasswordValidator
{
    RequiredLength = 6,
};
```
Roslyn Diagnostic Warnings

- Roslyn diagnostics are also reported during MSBuild compilation:
Building Security Analyzers 101

Session recorded at OWASP AppSec USA 2016:

• Continuous Integration: Live Code Analysis using Visual Studio and the Roslyn API
  - https://youtube.com/watch?v=Y8JKVjY-7T0

• Demonstration analyzers from the presentation:
  - https://github.com/ejohn20/puma-scan-demo
Introducing the Puma Scan

• Open source security source code analyzer built using Roslyn
• 50+ application security-specific rules
• Version 1.0.6 is available via NuGet & VS Marketplace
• Install guide, rule docs, source code:
  - https://www.pumascan.com/community
  - https://github.com/pumasecurity
  - @puma_scan

©2017 – Puma Security, LLC
Roadmap

• The DevOps Problem
• Security Unit / Integration Testing
• Static Analysis Options
• Pre-Commit
• Continuous Integration
• Conclusion
Requirements for static scanning in the IDE.

- Display vulnerabilities inside the IDE
- Provide documentation on how to fix the issue
- Allow engineers to suppress false positives
IDE Static Analysis Checklist

Display vulnerabilities inside the IDE

// DELETE api/contest/5
public void Delete(string id)
{
    using (WidgetTownDBEntities context = new WidgetTownDBEntities())
    {
        string q = string.Format("DELETE FROM Contests WHERE Id = {0}\", id);
        Context.Database.ExecuteSqlCommand(q);
    }
}

[.Sql] (local variable) WidgetTownDBEntities context

SQL Injection - EF method executes dynamic SQL without parameters
Provide documentation on how to fix the issue

SEC0016 - Cleartext Machine key

The machine key element defines keys to use for encryption and decryption of authentication cookies, view state, and verification of session state identification. The validation and decryption key values should not be stored in configuration files in cleartext. Encrypt the machineKey section of the configuration file using aspnet_regis.exe.


Insecure Configuration:

Secrets stored in the web.config file are not encrypted by default.

```csharp
<system.web>
  ...
  <machineKey validationKey="<key value>" decryptionKey="<key value>" valid=
  ...
</system.web>
```

Secure Configuration:

Use the aspnet_regis.exe utility to encrypt the machineKey element of the configuration file. Run this command from the root of your app.

```
C:\Windows\Microsoft.NET\Framework\4.0.30219\aspnet_regis.exe -p "system.web/machineKey" -p "DataProtectionConfigurationProvider"
```
Allow engineers to suppress false positives
Roadmap

- The DevOps Problem
- Security Unit / Integration Testing
- Static Analysis Options
- Pre-Commit
- **Continuous Integration**
- Conclusion
Requirements for static scanning in the CI pipeline:

- Pipeline executes accurate static analysis rules
- Process and capture results in pipeline
- Configure build thresholds to mark builds as unhealthy or failed
- Notify security when issues are suppressed by engineers
Pipeline executes accurate static analysis rules
CI Static Analysis Checklist

Process and capture results in pipeline
Configure build thresholds to mark builds as unhealthy or failed
Roadmap

• The DevOps Problem
• Security Unit / Integration Testing
• Static Analysis Options
• Pre-Commit
• Continuous Integration
• Conclusion
Welcoming contributions!
Gather feedback and address edge cases
Continue to build out additional rule categories:
  - Cleartext secrets, insecure XML processing, etc.
Further refine results using data flow analysis to eliminate false positives
Identify rules that can apply suggested code fixes
Acknowledgements

- Eric Mead - Puma Security Engineer
- Scott Sauber - Puma Security Engineer
- James Kashevos – CDD Security Engineer
- Josh Brown-White - Microsoft
- Gitter Rosyln Channel
Questions?

Contact Info:
eric.johnson@pumascan.com
@emjohn20