

SECURITY BOOT CAMP .NET DEVELOPERS



Philippe Arteau
Security Researcher for GoSecure

ConFoo.CA
DEVELOPER CONFERENCE



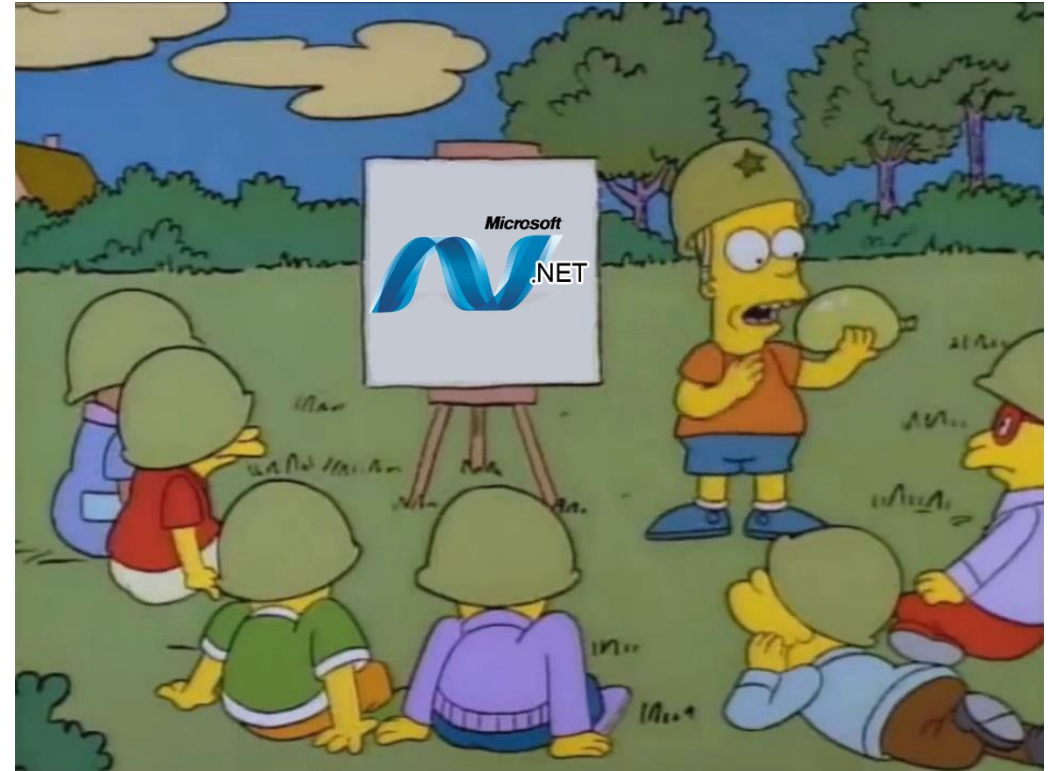
12/03/2018

WHO AM I ?

- Philippe Arteau
- Security Researcher at GoSecure
- Open-source developer
 - ~~.NET Security Guard~~ Security Code Scan (Roslyn – Static Analysis for .NET)
 - Find Security Bugs (SpotBugs - Static Analysis for Java)
 - Burp and ZAP Plugins (Retire.js, CSP Auditor)
- Volunteer for the nsec conference and former trainer

AGENDA

- Introduction
- Vulnerabilities in .NET Context
 - Path Traversal
 - XSS
 - Cryptography
 - Hardcoded secret
- Automating Checks
 - Visual Studio / MsBuild
- Recent Trends
 - Deserialization
 - Double Parsing
- Methodology for Code Review
- Conclusion



An abstract network diagram with red nodes and lines. The nodes are represented by circles and hexagons, some of which are highlighted with a darker red color. The lines are thin and red, connecting the nodes in a complex, interconnected pattern. The background is white, and the overall style is modern and technical.

INTRODUCTION

SECURITY CODE REVIEW

- Code review is the systematic examination of source code^[1] with the specific goal of findings security bugs.
- Security Bugs?
 - Injections
 - XSS
 - Cryptographic weakness
 - Logic flaw
 - And many more...

[1] [Wikipedia: Code review](#)

WHY SECURITY CODE REVIEW?

- Complementary to dynamic techniques (penetration testing, fuzzing, etc.)
- Every technique has its advantages
- Code review advantages:
 - Coverage
 - Finding all instances of a vulnerability
 - Accessible activity for developer
 - Excellent for doing defense in depth

The background of the slide features a complex, abstract network diagram. It consists of numerous red nodes, some of which are hexagons and others are circles, interconnected by a web of thin red lines. The nodes are distributed across the slide, with a higher density in the lower half. The overall aesthetic is technical and digital.

VULNERABILITIES IN .NET CONTEXT

PATH TRAVERSAL

- SQL injection are *easy* to manage with **prepare statement**
- **Path traversal** is a source of injection that is often overlooked
- When does it matter?
 - File upload (writing to filesystem)
 - Document loading (reading from filesystem)
 - Can be applied in rare cases to URL ^[1]

[1] Example: https://sakurity.com/blog/2015/03/15/authy_bypass.html

DEMO



CROSS-SITE SCRIPTING (XSS)

- Encoding with Razor template is usually secure
 - HTML entities are escaped by default

Special cases

- Use of `@Html.Raw()`
- Placing values in JavaScript `/!\`
- JavaScript client-side template

DEMO



PADDING ORACLE AND INTEGRITY

- .NET Framework provided symmetric encryption primitive
 - Namespace [System.Security.Cryptography](#)
 - Include the mode: CBC, ECB, OFB, ...
 - Does not provided integrity

DEMO



HARDCODED PASSWORD

- Password
- Service account
- API keys
- Store value in configuration
- Encrypt the value

Identity Server

```
new Client
{
    ClientId = "client",
    AllowedGrantTypes = GrantTypes.ClientCredentials,
    ClientSecrets =
    {
        new Secret("secret".Sha256())
    },
    AllowedScopes = { "api1" }
}
```



AUTOMATING CHECKS

AUTOMATE CODE ANALYSIS

- Identifying bugs and vulnerabilities is nice but...

0 references | Philippe Arteau, 3 days ago | 1 author, 1 change

```
public class HomeController : Controller
```

```
{
```

0 references | 0 changes | 0 authors, 0 changes | 0 requests | 0 exceptions

```
public ActionResult Index(string input)
```

```
{
```

```
    if (input == "") {
```

```
        return View();
```

```
    }
```

Error List

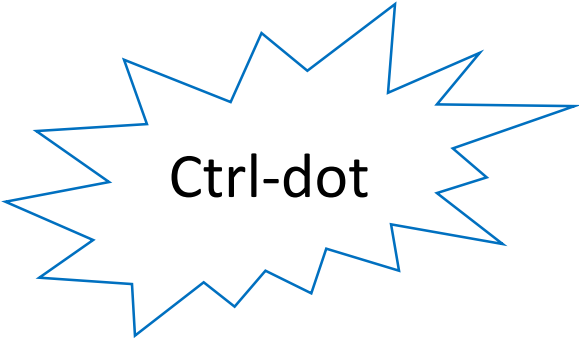
Entire Solution 0 Errors 36 Warnings 0 Messages Build + IntelliSense

Code	Description
CA1820	Replace the call to 'string.operator ==(string, string)' in 'HomeController.Index(string)' with a call to 'String.IsNullOrEmpty'.
CA2210	Sign 'DemoDevTeach.dll' with a strong name key.
	The 'packages' element is not declared.
CA1822	The 'this' parameter (or 'Me' in Visual Basic) of 'CookieSample.createCookie()' is never used. Mark the member as static (or Shared in Visual Basic) or at least one property accessor, if appropriate.
CA1822	The 'this' parameter (or 'Me' in Visual Basic) of 'HardcodePassword.test(string)' is never used. Mark the member as static (or Shared in Visual Basic) or at least one property accessor, if appropriate.
CA1822	The 'this' parameter (or 'Me' in Visual Basic) of 'MvcApplication.Application_Start()' is never used. Mark the member as static (or Shared in Visual Basic) or at least one property accessor, if appropriate.
SG0009	The cookie is missing security flag HttpOnly
SG0008	The cookie is missing security flag Secure

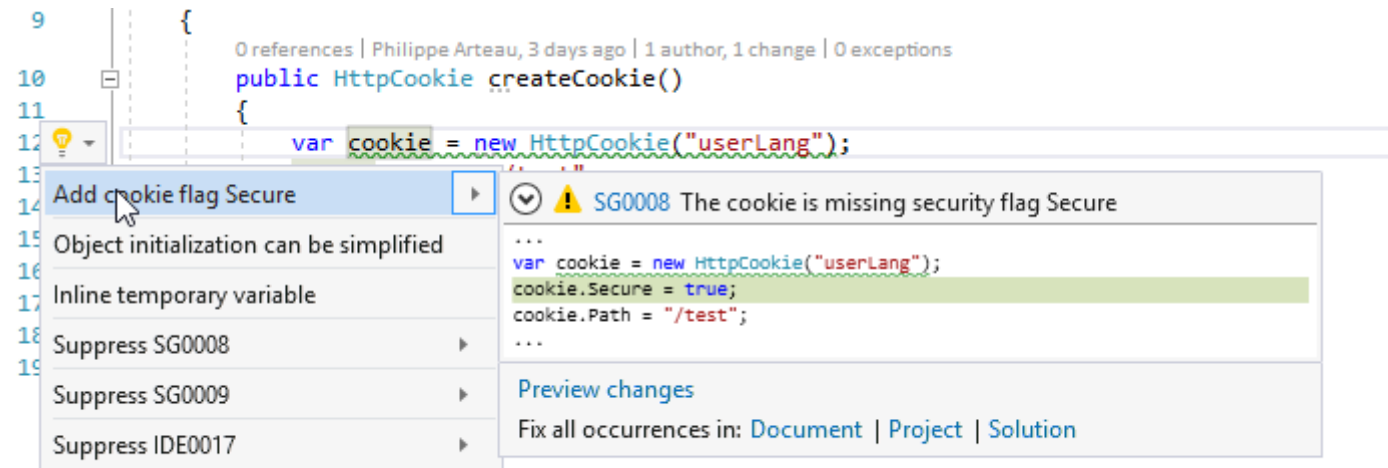
Code Metrics Results CodeLens Error List Command Window Output

AUTOMATE CODE REFACTORING

- Remediation is even better!



Ctrl-dot



- Some vulnerabilities require high-level understanding of the application

Security Code Scan Demo



DEMO



The background of the slide features a complex, abstract network diagram. It consists of numerous red lines of varying thicknesses connecting various nodes. The nodes are represented by red hexagons of different sizes, some of which are enclosed within larger, faint red circles. The network is most dense in the lower half of the image, with a more sparse, web-like structure in the upper half. A solid dark grey horizontal bar runs across the bottom of the slide, and a thin red line is visible at the very bottom edge.

RECENT 'TRENDS

JSON DESERIALISATION

- History repeats itself
 - 2016: Numerous Java application were found vulnerable to native deserialization
 - 2017: Researchers ^[1] found issues in .NET JSON serializer
 - Some libraries have issued updates
 - The vulnerability was called: JSON Friday 13th
- Two ingredients needed for a successful attack
 - Gadgets
 - Unsafe deserialization



[1] Alvaro Muñoz, Oleksandr Mirosh and James Forshaw

JSON DESERIALIZATION

Affected librairies

- FastJSON
- Json.NET (use of TypeNameHandling.All)
- FSPickler
- Sweet.Jayson
- JavascriptSerializer
- DataContractJsonSerializer

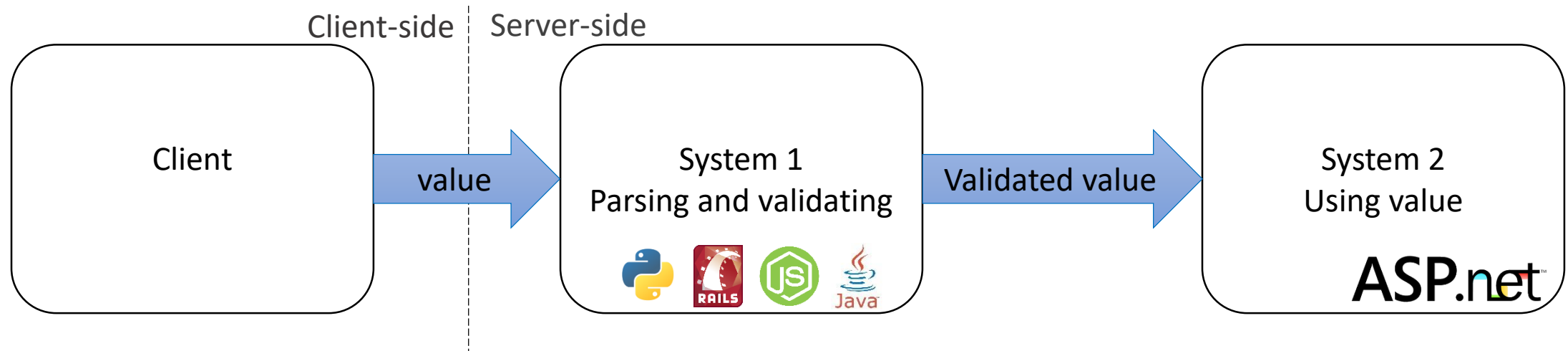


DEMO



Ref: <https://www.blackhat.com/docs/us-17/thursday/us-17-Munoz-Friday-The-13th-JSON-Attacks-wp.pdf>

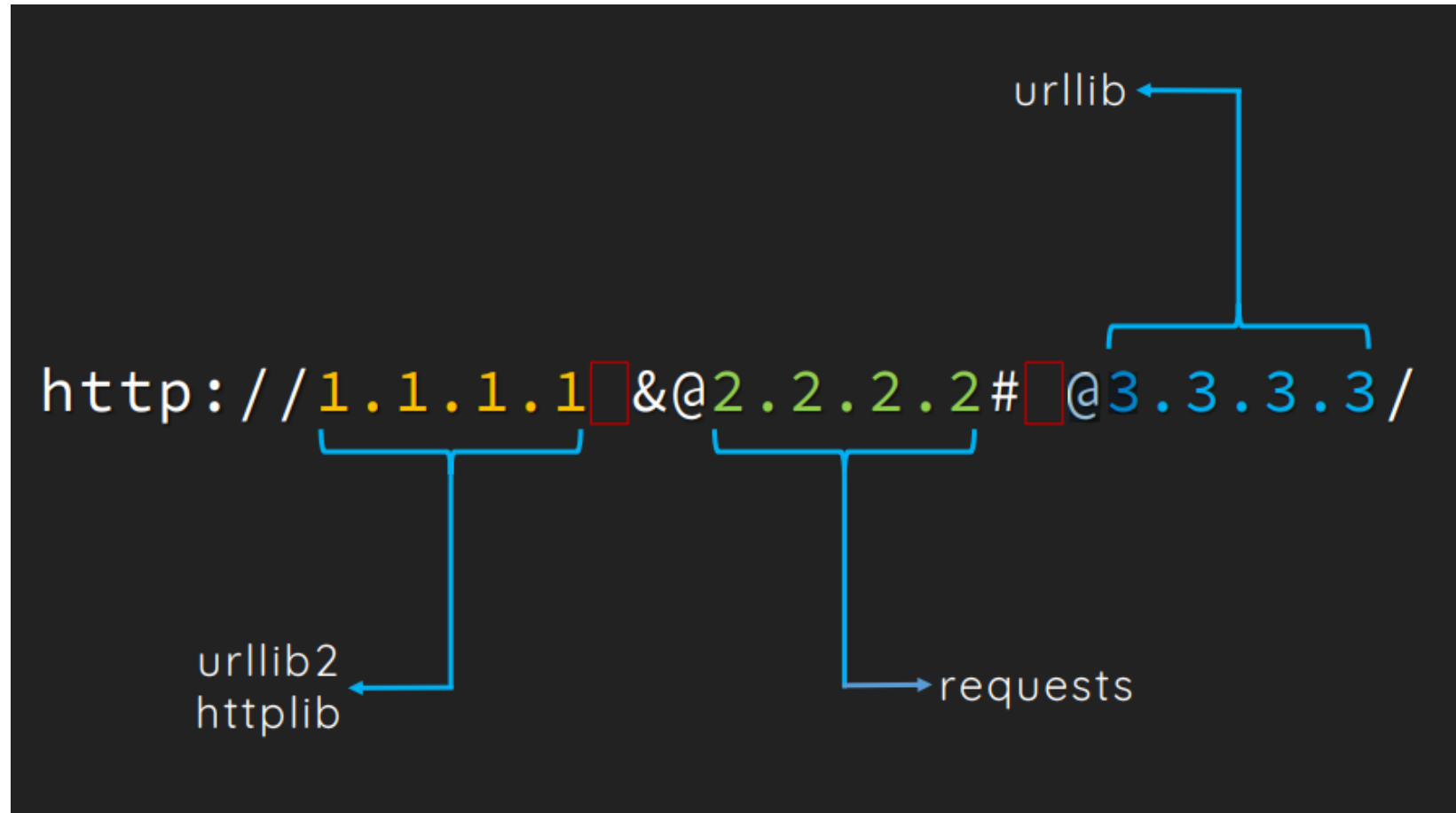
DOUBLE PARSING



- What if the **system validating** and **using the value** was not the same

DOUBLE PARSING: URLS

When parsing the following URL, what is the host?



Reference: [A New Era of SSRF - Exploiting URL Parser in Trending Programming Languages!](#)

DOUBLE PARSING: URIS

- Less likely to happen in .NET
 - Small numbers of URI parser
- High probability when interacting **in other systems**
- DNS rebinding needed to be considered for host whitelisting
- Conclusion
 - Do not trust validated input that was parsed differently

FIND THE BUGS

```
{  
  "username": "philippe",  
  "fullname": "Philippe A.",  
  "newPassword": "C0nf00"  
}
```

IdentityValidator.cs

```
boolean IsValidRequest(json) {  
  var jsonReader = JsonReaderWriterFactory.CreateJsonReader(json,[...]);  
  var root = XElement.Load(jsonReader);  
  
  return root.XPathSelectElement("//username ").Value == HttpContext.Current.User.Identity.Name;  
}
```

UpdateUser.cs

```
void ProcessUpdateUser(json) {  
  if(IsValidRequest(json)) {  
    JObject user = JObject.Parse(json);  
  
    var usersToUpdate = context.User.Where(u => u.username == user.GetValue("username")).ToList();  
    usersToUpdate.ForEach(u => u.Password = user.GetValue("newPassword"));  
    context.SaveChanges();  
  }  
}
```

*Pseudocode is highly simplified

JSON PARSER IN .NET

```
{  
  "username": "philippe",  
  "username": "yannlarrivee",  
  "fullname": "hihihi",  
  "newPassword": "C0nf00"  
}
```



using System.Runtime.Serialization.Json;

```
{  
  "username": "philippe",  
  "username": "yannlarrivee",  
  "fullname": "hihihi",  
  "newPassword": "C0nf00"  
}
```

using Newtonsoft.Json;

```
{  
  "username": "philippe",  
  "username": "yannlarrivee",  
  "fullname": "hihihi",  
  "newPassword": "C0nf00"  
}
```

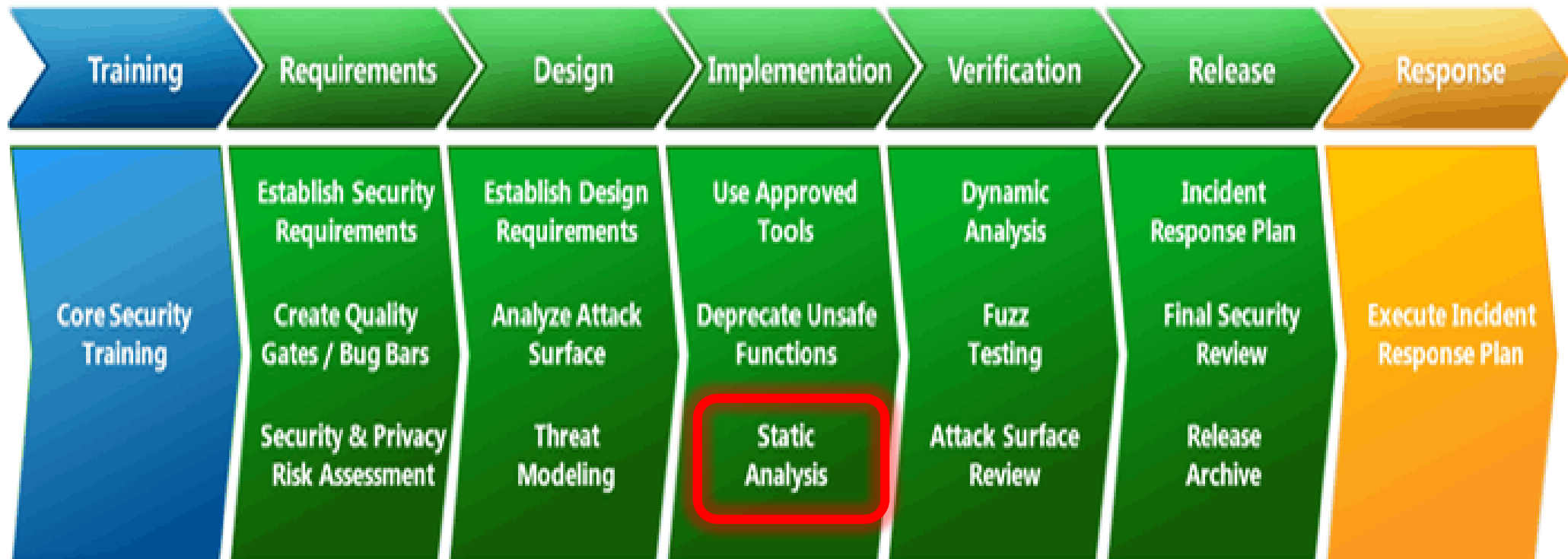
Inspired by: <https://justi.cz/security/2017/11/14/couchdb-rce-npm.html>

The background of the slide features a complex, abstract network diagram. It consists of numerous red nodes, some of which are hexagons and others are circles, interconnected by a web of thin red lines. The nodes are distributed across the slide, with a higher density in the lower half. The overall aesthetic is technical and digital.

METHODOLOGY FOR CODE REVIEW

CODE REVIEW IN SDLC

- First thing first, code review is ONE of the security activities that need to be integrated in the development lifecycle.



CODE REVIEW STEPS

1. Threat modeling ^[1]
2. Analysis
3. Reporting (Document or Opening ticket) *
4. Bug fixing *

[1] [OWASP Code Review Guide](#) v2 p.32

* Not covered in this presentation

'THREAT' MODELING : DECOMPOSING THE APPLICATION

- Identify assets to protect
 - Personal information
 - Documents
 - Passwords
- Identify entry points
 - MVC Controller
 - Web Services
 - Forms
- Identify external dependencies
- Imagine possible threats (STRIDE : **S**poofing, **T**ampering, **I**nformation **D**isclosure, **D**enial of Service, **E**levation of privilege)

ANALYSIS

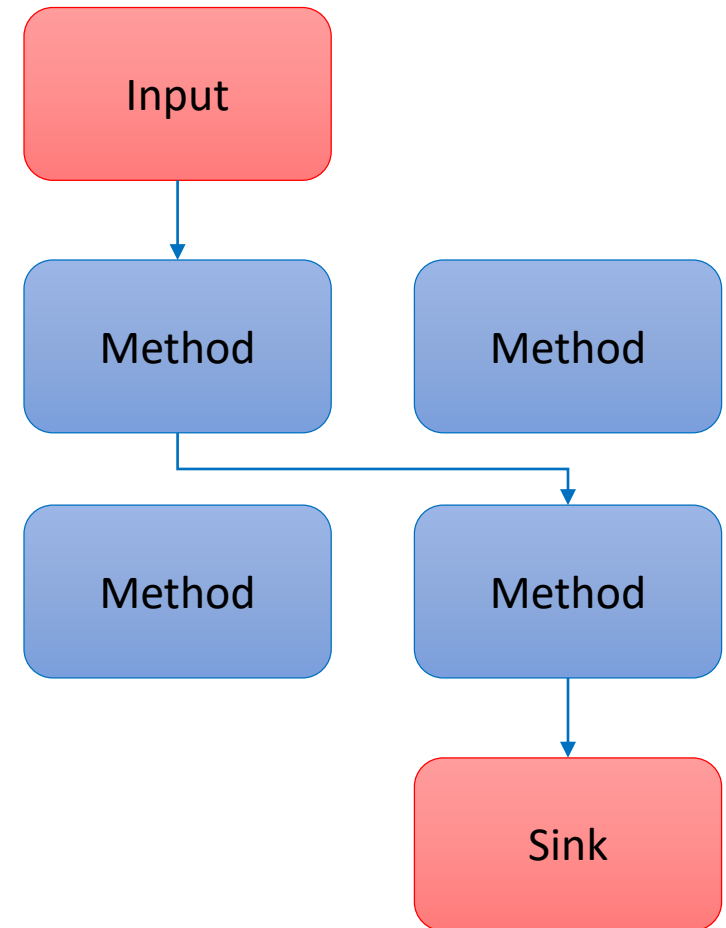
1. Tools configuration
2. Automate scan
 - Review potential issues
3. Manual review



Static analysis tools can also be run in parallel with the manual review.

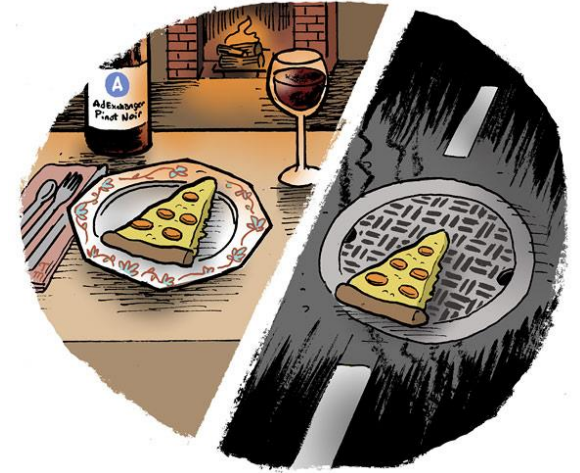
DATAFLOW

- Mapping between inputs and APIs
- Categories of bugs
 - Injection
 - Path traversal
 - Static IV (Cryptography)
 - Deserialization



CONTEXT

- APIs are not vulnerable by default
- APIs are designed to be used in a certain context
- Categories of bugs
 - Random number generation
 - Oracle Padding Attack or any other active attack
 - Control based on Host header
 - Insecure communication (internal communication vs network communication)
 - Configuration files vs Upload files



Context Matters

CHECKLIST

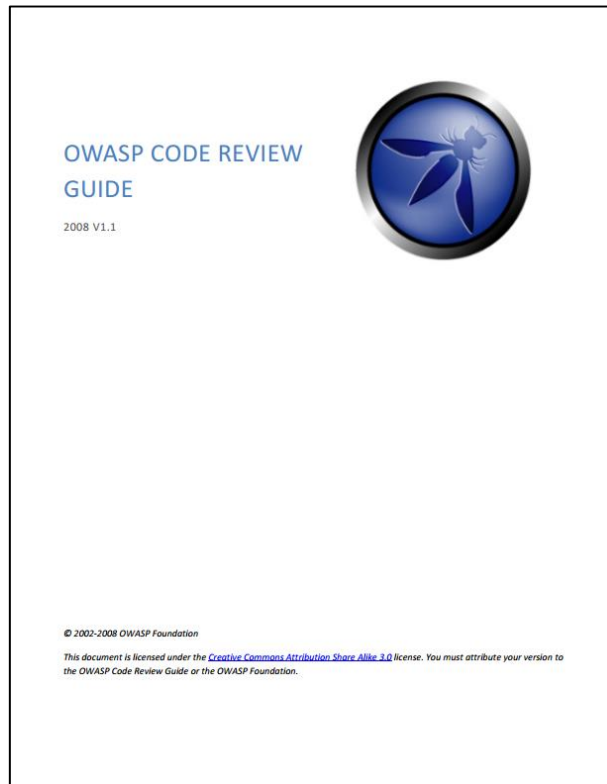
- Intended for baseline verifications
- Guidelines
- Reproducibility



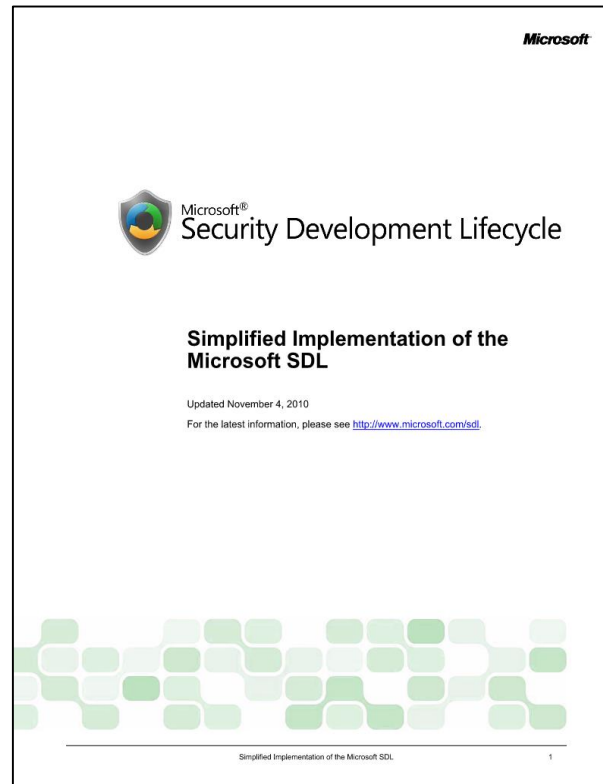
#	Description	1	2	3	Since
5.1	Verify that the runtime environment is not susceptible to buffer overflows, or that security controls prevent buffer overflows.	✓	✓	✓	1.0
5.3	Verify that server side input validation failures result in request rejection and are logged.	✓	✓	✓	1.0
5.5	Verify that input validation routines are enforced on the server side.	✓	✓	✓	1.0

Listing taken from: [OWASP Application Security Verification Standard Project](#)

GOOD RESOURCES



[Code Review Guide](#)



[Development Lifecycle](#)



[Verification List](#)

The background of the slide features a complex, abstract network diagram. It consists of numerous red nodes, some of which are hexagons and others are circles, interconnected by a web of thin red lines. The nodes are distributed across the slide, with a higher density in the lower half. A solid dark grey horizontal bar runs across the bottom of the slide, and a thin red line is positioned just above it.

CONCLUSION

CONCLUSION

- Code review is a powerful technique to find security bugs
 - But don't forget to do dynamic tests as well
- Use tools when possible
 - Build or extends tools when needed
- Recent trends affecting .NET
 - Angular XSS (Client-side template injection)
 - Deserialization vulnerabilities
 - Double parsing

The background of the slide features a complex, abstract network diagram. It consists of numerous red lines of varying thicknesses connecting various nodes. The nodes are represented by different shapes, including circles, hexagons, and polygons, some of which are filled with red or have red outlines. The network is dense in the center and more sparse towards the edges. A solid red horizontal bar runs across the bottom of the slide.

REFERENCES

REFERENCES

- OWASP .NET Project

[https://www.owasp.org/index.php/Category:OWASP .NET Project](https://www.owasp.org/index.php/Category:OWASP_.NET_Project)

- .NET Security Cheat Sheet

[https://www.owasp.org/index.php/.NET Security Cheat Sheet](https://www.owasp.org/index.php/.NET_Security_Cheat_Sheet)

- Security Code Scan

<https://security-code-scan.github.io/>

ROSLYN REFERENCES

- .NET Compiler Platform ("Roslyn"): Analyzers and the Rise of Code-Aware Libraries

<https://www.youtube.com/watch?v=lp6wrpYFHhE>

- Roslyn Wiki

<https://github.com/dotnet/roslyn/wiki>

- Learn Roslyn Now: Part 10 Introduction to Analyzers by Josh Varty

<https://joshvarty.wordpress.com/2015/04/30/learn-roslyn-now-part-10-introduction-to-analyzers/>

- .NET Compiler Platform SDK

<https://marketplace.visualstudio.com/items?itemName=VisualStudioProductTeam.NETCompilerPlatformSDK>

.NET JSON DESERIALIZATION

- Ysoserial.net : Payload generator

<https://github.com/pwntester/ysoserial.net>

- Friday The 13th JSON Attack - White Paper

<https://www.blackhat.com/docs/us-17/thursday/us-17-Munoz-Friday-The-13th-JSON-Attacks-wp.pdf>



QUESTIONS ?

Contact

✉ parteau@gosecure.ca
🌐 gosecure.net/blog/
🐦 @h3xStream @GoSecure_Inc