POWER OF POSSIBILITIES
Automated Security Testing

No, our pen testers are not lazy
Agenda

• About me
• Five Phases of Pen Testing
• What this is?
• Why do it?
• Drawbacks to automation
• Vulnerability Correlation Tool
• Our previous process
• Our current process
• By the numbers
• What is next?
About Me

• Been doing application pen testing for 13+ years with NetSPI
  • Performed secure code reviews and pen tests for banks, retail, healthcare companies
  • Have pen tested web, mobile, thick client and mainframe applications
  • Have also worked on network pen tests
• Formerly developer for Cargill and Ecolabs
• Also have done system admin, DBA and security architect
• Formerly a product security engineer with Honeywell
  Security consultant with E&Y
Five Phases of Pen Testing

- Pen testing usually involves five phases:
  - Planning and data gathering—Define the goals of the penetration testing.
  - Scanning—Tools are used to gather more data and information on the target.
  - Gaining access—Web application attacks such as Cross-Site Scripting or SQL Injection are launched to expose vulnerabilities.
  - Maintaining access—Determine if the exposed vulnerability can be used to achieve a persistent presence in the application.
  - Covering tracks—The attacker takes care to remain undetected.
What is this?

• This is the scanning phase of a pen test

• Low cost testing process to find security vulnerabilities

• Implemented using tools that have access using an API or a CLI

• This is a set of test cases that are included in most or all of pen testing process

• Automated security testing is not intended to replace manual testing.
Why do this?

- We are constrained on the number of people we use for pen testing
- Using automation frees up pen testers time to do the more complex testing
  - Reduce the time needed to engage in routine, mundane testing
- Reduce the amount of time spent reporting findings
- A good percentage of our testing can be automated
Drawbacks to automation

• In some cases need to configure and instantiate the tool

• Not all tools can be automated

• Have to consume the results somehow

• We will get overlapping results from tools – how to we reconcile those with minimal time wasted?
Vulnerability Correlation Tool

• Designed to normalize and de-duplicate results across a variety of testing tools

• Provides a single tool that unifies output from several other tools
  • Some are already built in, but with API or CLI access we can write scripts to pull in content from others

• Vulnerability correlation is recognized by Gartner as a rising technology

Benefits of a Vulnerability Correlation Tool

- **Deduplication**—Duplicate results from the multitude of tools are automatically removed. You get one finding with a single set of results.

- **Correlation of SAST and DAST results**—SAST tools identify potential vulnerabilities, while DAST tools identify which of those vulnerabilities are actually exploitable. Combining all of these results lets you know which threats are real and of the highest priority.

- **Remediation management**—AVC tools identify the specific lines of code where vulnerabilities exist and neighboring flaws and vulnerabilities. A centralized console gives you the ability to assign, track, and monitor the progress of remediation.

- **Workflow integration**—Integration with popular environments such as Eclipse makes it easy for developers to fix problems.

- **Reporting**—A wide variety of reports make it easy to sort through testing results and track remediation efficiently. Reports on how long it takes to remediate issues, for example, help you make sure remediation is happening quickly.

- **Compliance checks**—Automatically check your codebase against regulations such as HIPAA, the DISA-STIG, and the PCI-DSS.

Taken from CodeDX: https://codedx.com/2018/09/13/the-importance-of-penetration-testing-and-vulnerability-assessments-for-web-applications/
Our previous process

• Every engineer has their own toolset running on their own laptop

• No consistent process

• Engineers copied and pasted findings from the tools into Jira

• Inconsistent findings from different engineers

• Engineer maintains their own environment
Our new process

• Using a Vulnerability Correlation Tool
  • Some tools are already built in
  • Written scripts to pull in content from other tools (NMap, Nessus, OWASP Zap, SQLMap, and MobSF)

• Using Jenkins in a Kali Linux VM for each pen tester to use

• Can report on pen testers usage for new product introduction process

• Allows engineers to focus on more complex testing not well suited to tools
By the numbers

• About 20-25% of our project time is taken up reporting via JIRA ticket creation, etc. Vulnerability Correlation tools can directly integrate with JIRA and reduce things like copy/paste, lowering this number.

If this holds true, this would have amounted to an estimated savings of $248,000 in engineering effort in 2017 alone for the pen testing team.
What is next?

• Keep adding tools to this process

• Our ultimate goal is to allow product teams to be able to conduct basic assurance penetration testing earlier in the development lifecycle

• CI/CD direct integration
  • Being able to automate fundamental penetration tests and “break builds“ when certain tests fail
QUESTIONS ANSWERED HERE EVEN THE SILLY ONES
My question to you

An anonymous user has access to sensitive data on a web site.
Is this an authentication problem or an authorization problem?