KTM JS #13
Agenda: Security and QA Testing

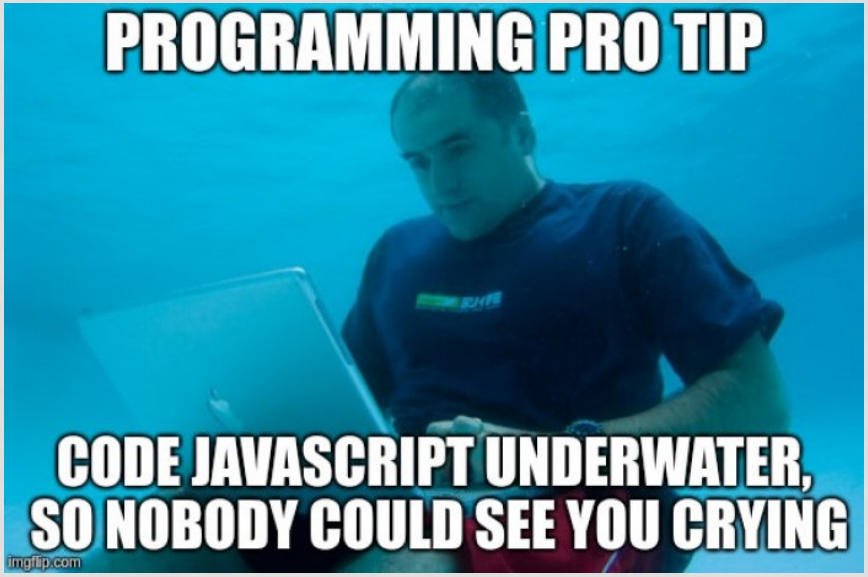
SECURITY PRACTICES IN SDLC & BEYOND

Presenter: Sabin Ranjit

05-25-2019

ABOUT ME

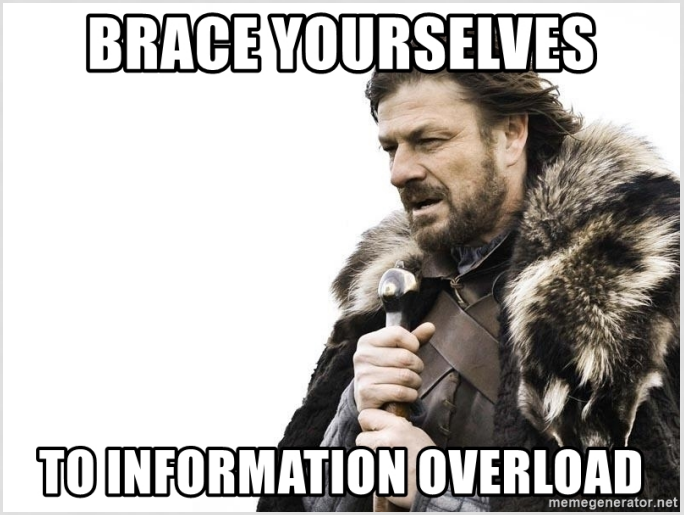
- Good looking guy :D
- DevSecOps
- Incident Handler



- https://np.linkedin.com/in/sabinranjit

TAKE WAYS

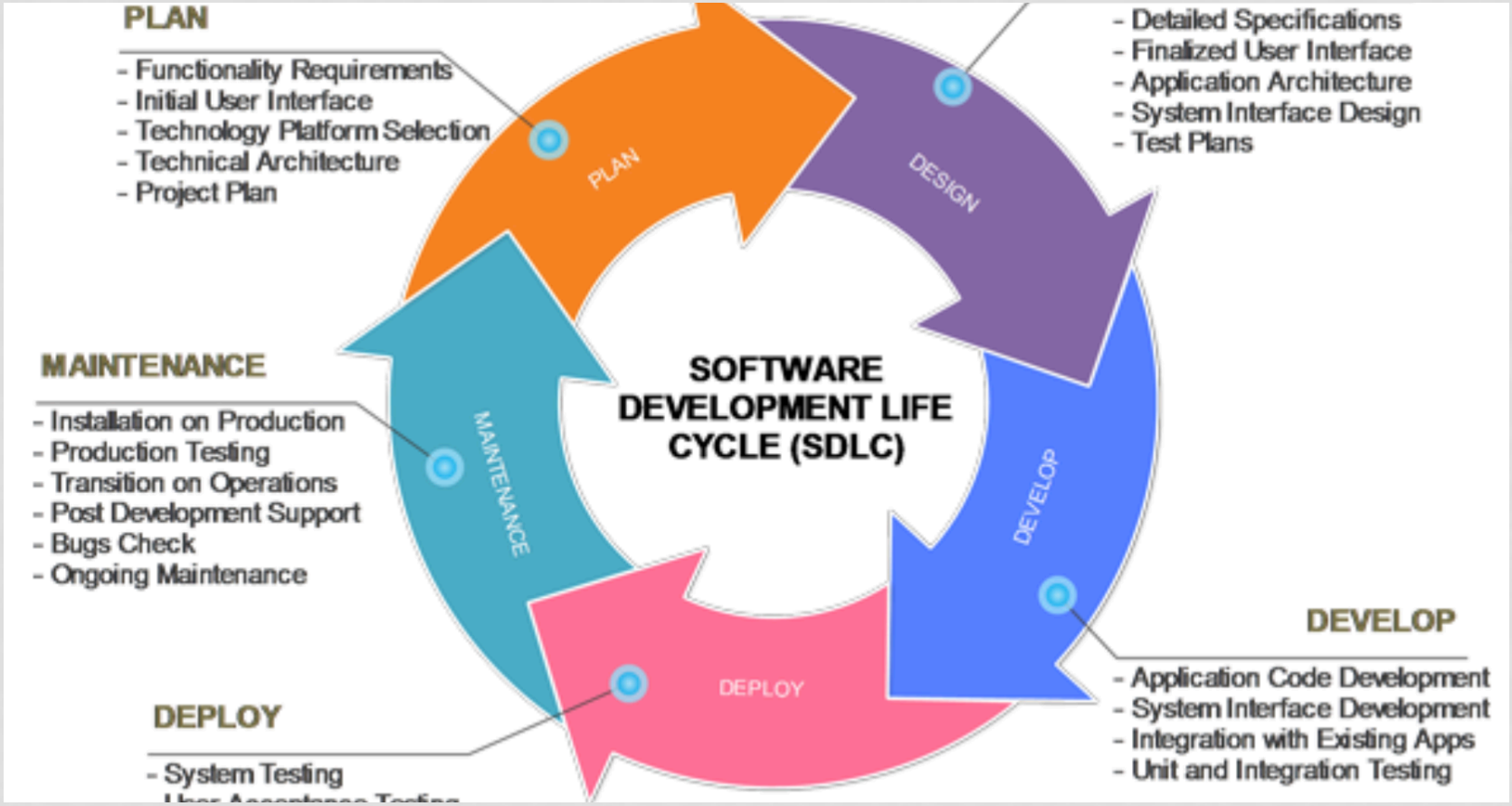
- A good example of poor slides
- Security stuffs from moon
- Few different tools
- A bit of threat modeling



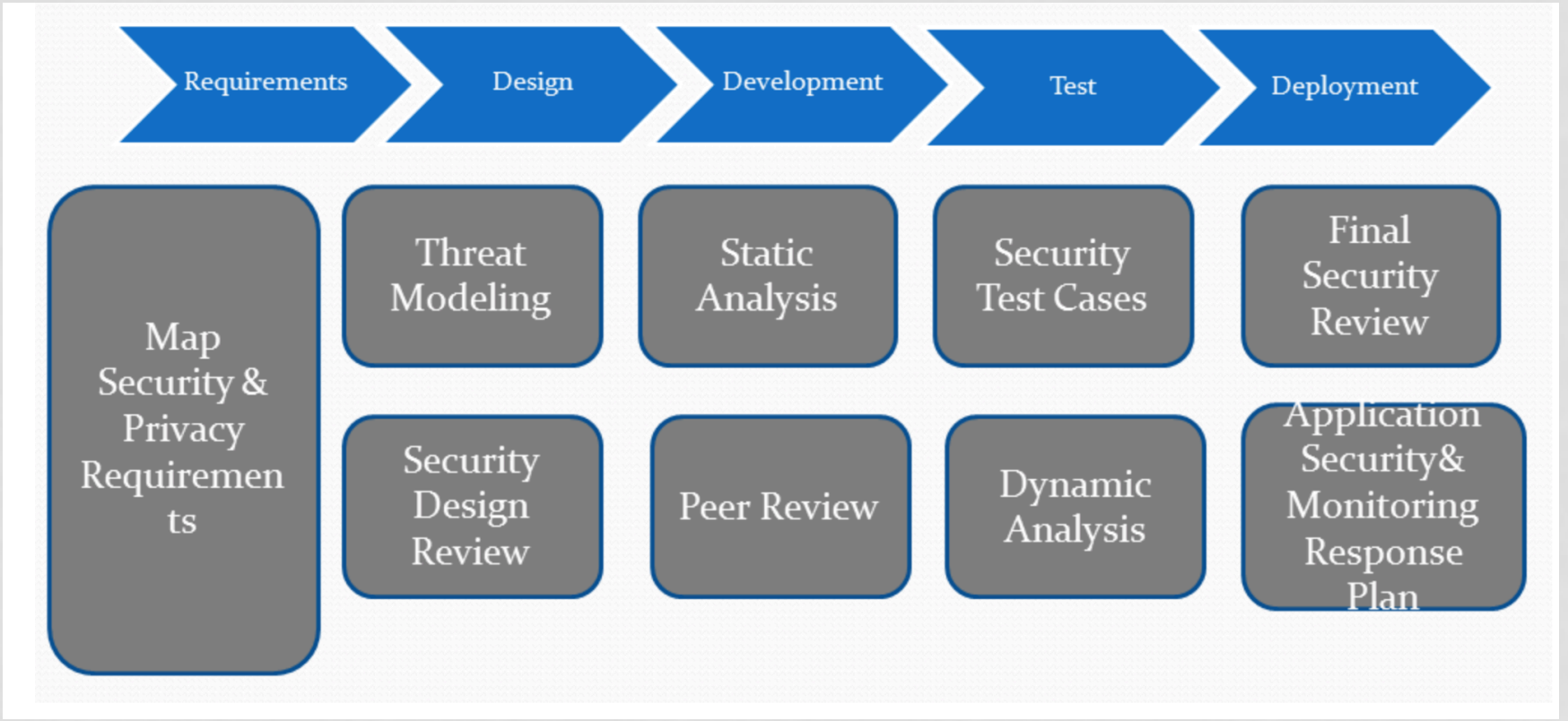
SESSION FLOW

- Sdlc
- Secure sdlc
- A bit deep into secure sdlc
- Tools
- Few words
- Questions

SDLC


PLAN
- Functionality Requirements
- Initial User Interface
- Technology Platform Selection
- Technical Architecture
- Project Plan
- Detailed Specifications
- Finalized User Interface
- Application Architecture
- System Interface Design
- Test Plans
MAINTENANCE
- Installation on Production
- Production Testing
- Transition on Operations
- Post Development Support
- Bugs Check
- Ongoing Maintenance
SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)
PLAN
DESIGN
DEVELOP
DEPLOY
MAINTENANCE
DEVELOP
- Application Code Development
- System Interface Development
- Integration with Existing Apps
- Unit and Integration Testing
DEPLOY
- System Testing

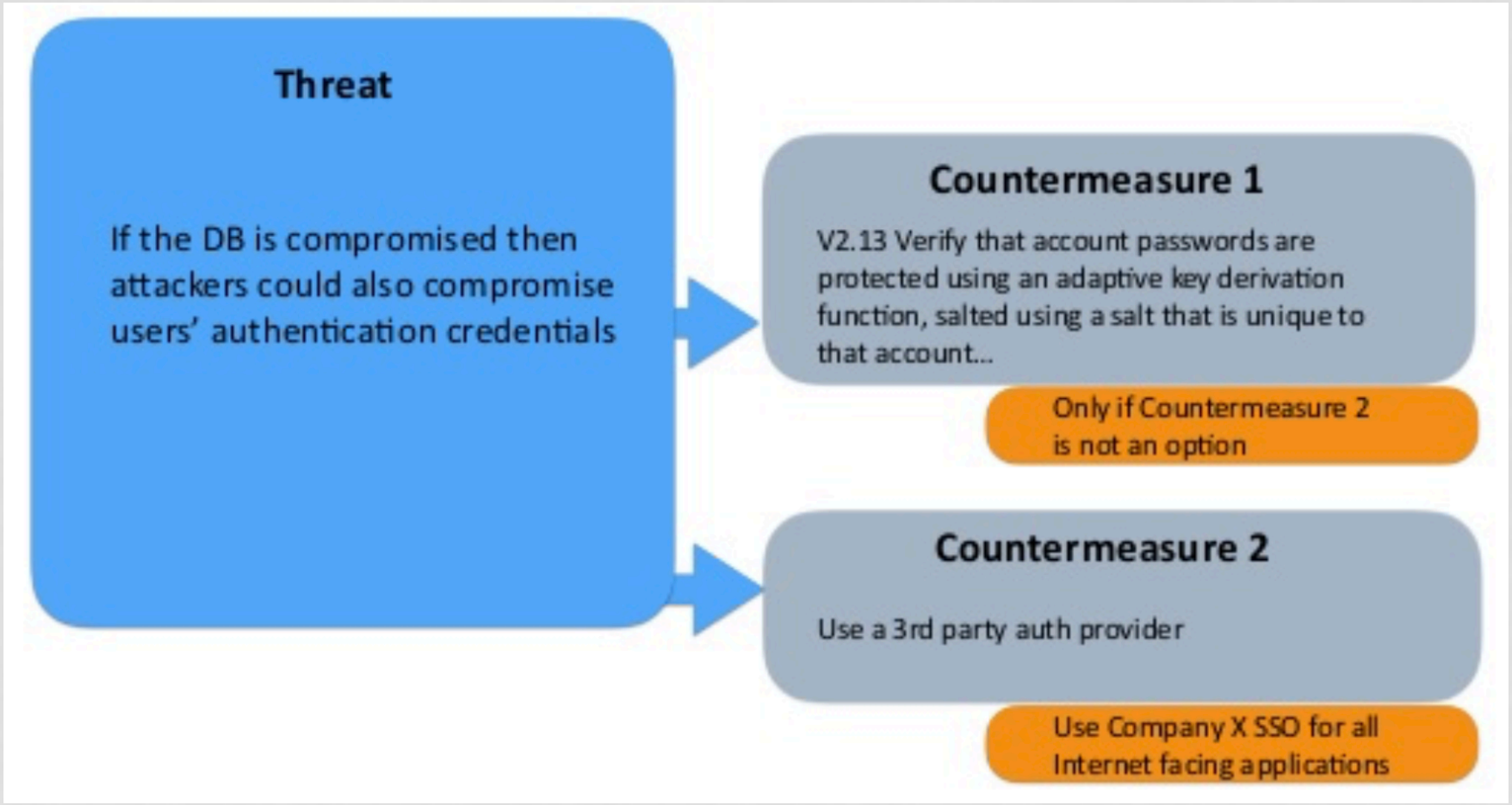
SECURE-SDLC


Requirements
Design
Development
Test
Deployment
Map Security & Privacy Requirements
Threat Modeling
Security Design Review
Static Analysis
Peer Review
Security Test Cases
Dynamic Analysis
Final Security Review
Application Security& Monitoring Response Plan

- Phase 1
  - map out and document the nonfunctional requirements.
  - Organizational security policies and standards
  - Organizational privacy policy
  - Regulatory requirements and other
  - Tools: NIST Framework, FCC cyberplanner, SANS templates, CIS benchmarks, etc.

- Phase 2
  - Threat modeling
  - Review design
  - Address implementation related vulnerabilities
  - Perform risk analysis and threat ranking
  - Many more ..
  - Tools: OWASP ASVS v4, Threatspec, Threatplaybook, Threatdragon, goSDL, etc.

- Threat modeling - generally
  - What we are building?
  - What can do wrong?
  - What are we going to do about it?
  - Did we do a good job?


Threat
If the DB is compromised then attackers could also compromise users' authentication credentials
Countermeasure 1
V2.13 Verify that account passwords are protected using an adaptive key derivation function, salted using a salt that is unique to that account...
Only if Countermeasure 2 is not an option
Countermeasure 2
Use a 3rd party auth provider
Use Company X SSO for all Internet facing applications

ASVS v4 : Token based session management

#	Description
3.5.1	Verify the application does not treat OAuth and refresh tokens — on their own — as the presence of the subscriber and allows users to terminate trust relationships with linked applications.
3.5.2	Verify the application uses session tokens rather than static API secrets and keys, except with legacy implementations.
3.5.3	Verify that stateless session tokens use digital signatures, encryption, and other countermeasures to protect against tampering, enveloping, replay, null cipher, and key substitution attacks.

- Phase 3
  - Static code analysis
  - Static application security testing (SAST)
  - Shift security Left
  - Detect early and fix early
  - Security should come from developers

  - Tools: sonarqube, snyk, graudit, etc.

SonarQube

"alert(...)" should not be used	JavaScript	Vulnerability	cwe, owasp-a3, user-experience
Code should not be dynamically injected and executed	JavaScript	Vulnerability	cwe, owasp-a7
Console logging should not be used	JavaScript	Vulnerability	owasp-a3, user-experience
Cross-document messaging domains should be carefully restricted	JavaScript	Vulnerability	html5, owasp-a7
Debugger statements should not be used	JavaScript	Vulnerability	cwe, owasp-a3, user-experience
Function constructors should not be used	JavaScript	Vulnerability	clumsy, owasp-a1
Local storage should not be used	JavaScript	Vulnerability	owasp-a3
Untrusted content should not be included	JavaScript	Vulnerability	cwe, owasp-a1, sans-top25-risky
Web SQL databases should not be used	JavaScript	Vulnerability	html5, owasp-a3, owasp-a9

Snyk

Authentication Bypass

Vulnerable module: passport-saml
Introduced through: passport-saml@0.35.0

Fix this vulnerability

Detailed paths and remediation

- **Introduced through**: console@1.0 › passport-saml@0.35.0

  Remediation: Upgrade to passport-saml@1.0.0.

Vulnerable functions

```
lib.passport-saml.saml.SAML.prototype.requestToUrl.requestToUrlHelper()
```

Overview

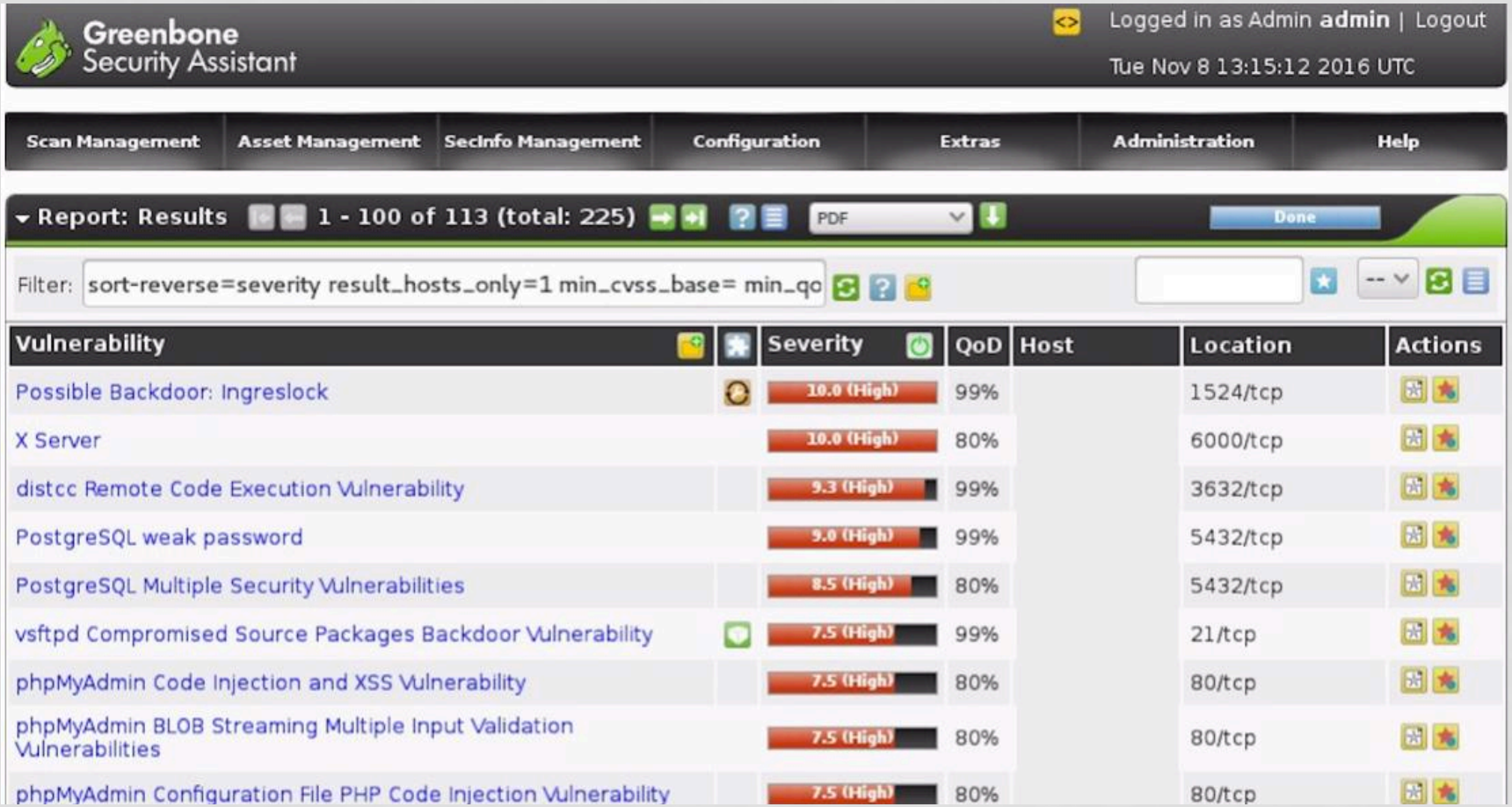
passport-saml is an authentication provider for Passport, the Node.js authentication library.

- Phase 4
  - Manual security testings
  - Dynamic Application Security Testing (DAST)

  - Tools: OWASP ZAP, Arachni, Burp suite, many more.

- Phase 5
  - Finalize changes
  - Finalize security controls and fixes
  - Secure environment: OS, VM, Containers, ..
  - Implement security controls: firewall, ids, siem, AV, etc.
  - DR plans and Incident Response Plans

  - Tools: OpenVAS, Nmap, DeepViolet, Aqua Microscanner, Vuls, Lynis, OSSEC, Snort, Auditd, etc.


Greenbone
Security Assistant
Logged in as Admin admin | Logout
Tue Nov 8 13:15:12 2016 UTC
Scan Management
Asset Management
SecInfo Management
Configuration
Extras
Administration
Help
Report: Results
1 - 100 of 113 (total: 225)
PDF
Done
Filter: sort-reverse=severity result_hosts_only=1 min_cvss_base= min_qo
Vulnerability
Severity
QoD
Host
Location
Actions
Possible Backdoor: Ingreslock
10.0 (High)
99%
1524/tcp
X Server
10.0 (High)
80%
6000/tcp
distcc Remote Code Execution Vulnerability
9.3 (High)
99%
3632/tcp
PostgreSQL weak password
9.0 (High)
99%
5432/tcp
PostgreSQL Multiple Security Vulnerabilities
8.5 (High)
80%
5432/tcp
vsftpd Compromised Source Packages Backdoor Vulnerability
7.5 (High)
99%
21/tcp
phpMyAdmin Code Injection and XSS Vulnerability
7.5 (High)
80%
80/tcp
phpMyAdmin BLOB Streaming Multiple Input Validation Vulnerabilities
7.5 (High)
80%
80/tcp
phpMyAdmin Configuration File PHP Code Injection Vulnerability
7.5 (High)
80%
80/tcp

Vuls (vuls-repo web dashboard)

ity ▾ CVSS Score ▾

						CVSS Severity
						CVSS Score
	ServerName	Container	CveID	Packages	PackageVer	
			CVE-2009-5080	groff-base	1.22.3-7-	
			CVE-2010-4664	consolekit	0.4.6-5-	
			CVE-2011-1784	keepalived	1:1.2.24-1ubuntu0.16.04.1-	
			CVE-2011-5325	busybox-initramfs	1:1.22.0-15ubuntu1-	
				busybox-static	1:1.22.0-15ubuntu1-	

BRING THE CHANGE

- Its all about Cultural change
- Pick one phase at a time, one tool at a time.
- Figure out right tool for you.
- Automate easy stuffs, focus on hard ones.
- Security trainings


THANKS FOR THE ATTENTION!
ANY QUESTIONS?