stackArmor

We provide full-stack cloud solutions for security focused customers

IMPLEMENTING SECURE DEVOPS ON PUBLIC CLOUD PLATFORMS

White Paper

stackArmor DevOps Solutions Team

stackArmor

stackArmor

Contents

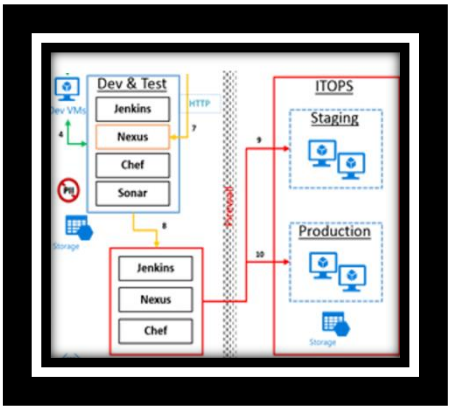
stackArmor

Abstract

Businesses are looking to accelerate the delivery of production quality software with fewer defects, and better security. Continuous Integration/Continuous Deployment (CI/CD) also known as DevOps is a rapidly maturing practice for reducing the time and effort it takes to test and deploy code into production. The rapid automation of the integration and deployment activities is common especially on cloud-based platforms. Adding security testing into the DevOps pipeline can help address the needs of regulated, compliance and public sector focused organizations. This white paper describes the use of open source technologies and commercial packages to design and deploy a Secure DevOps pipeline. Tools such as Yasca, SonarQube, and OpenSCAP amongst others when integrated with vulnerability scanners such as Tenable Nessus, HP Fortify and others provide a robust SecDevOps implementation.

What is Secure DevOps or SecDevOps?

Secure DevOps is the integration of security scans and reviews as part of the application development and deployment process. The figure on the right shows a high-level view of a CI/CD Environment. The CI/CD or DevOps pipeline includes common enabling components such as Jenkins, Nexus, Chef and SonarQube amongst others. This white paper is primarily focused on the security aspects of DevOps and therefore does not go into the details of a DevOps pipeline.



Implementing a CI/CD Security Process

The CI/CD or DevOps Security lifecycle begins with code development and integration. As the code is committed for deployment, the CI/CD security processes are activated. Common action items including static code analysis, vulnerability scanning, anti-virus scans and other similar integrity functions. The results from the security scans are provided to project management and the Chief Information Security Officer (CISO) within the organization.

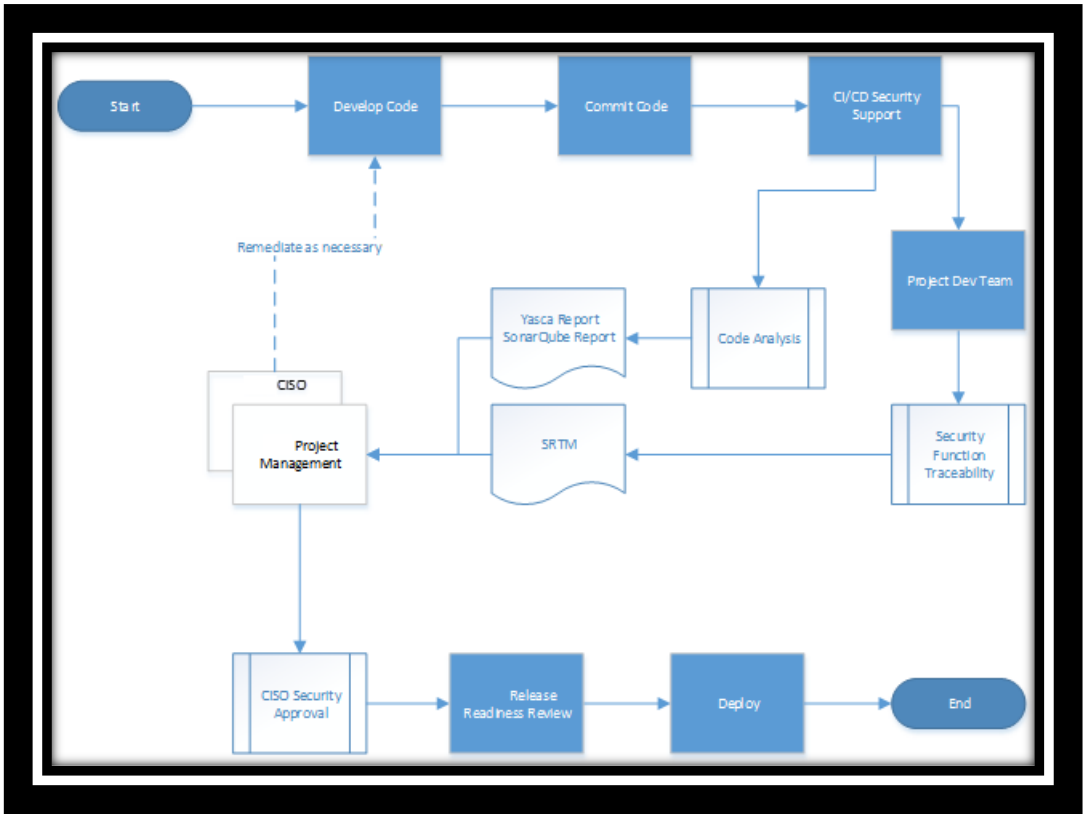


Figure: CI/CD Security Processes integrating within the overall code deployment and integration processes as an integral part of the overall pipeline

Key aspects of the CI/CD security pipeline are described in greater detail below.

Develop Code

Development is performed on an independent virtual machine (VM) within the CI/CD environment by the application development teams. In order to comply with NIST requirements for applying secure engineering principles, application developers should utilize code analysis utilities to ensure safe coding practices are followed. Project teams should leverage code analysis utilities as early as possible in the development lifecycle.

By leveraging code analysis capabilities, and correcting identified issues prior to submission to the formal Security process, the project will experience fewer delays and incidents of rework due to flaws and other security concerns.

At a minimum, code analysis should be performed as code modules are completed, but it is not necessary for modules to be completely finished for code review to be useful. Any supported language source file can be individually scanned or scanned within a directory along with other source files.

Commit Code to CI/CD

As application code is committed to the CI/CD branch in the git repository CI/CD performs a security review utilizing automated static code analysis tools. The commit step formally triggers the security checks and scans are described in greater detail below.

Tools for a Secure DevOps Pipeline

SecDevOps includes the execution of automated scanning tools and manual security reviews of results by the Security Team in order to facilitate the application deployment process.

YASCA Static Code Analysis

Yasca is a static source code analysis tool that performs a number of tests to identify actual and potential coding issues, to include those identified in the OWASP Top 10 listed in Section 3. It should be noted that Yasca, an open source tool is only one of tools to support secure coding practices. Other code analysis tools include HP Fortify, IBM AppScan, and others. Yasca utilizes individual plugins to perform scanning of targeted files. The Yasca implementation may include the following plugins (depending on the development environment):

- ***Grep Plugin.*** Uses external GREPfiles to scan target files for simple patterns.
- ***PMD Plugin.*** Uses PMD to parse and scan Java (and JSP) source code for issues.
- ***JLint Plugin.*** Uses J-Lint to scan Java .class files for issues.
- ***antiC Plugin.*** Uses antiC to scan Java and C/C++ source code for issues.
- ***FindBugs Plugin.*** Uses FIndBugs to scan Java class and Jar files for issues.
- ***Lint4J Plugin.*** Uses Lint4J to scan Java .class files for issues.

Yasca Severity Levels

Yasca plugins implement five (5) severity levels:

- 1 – Critical
- 2 – High
- 3 – Warning
- 4 – Low
- 5 – Informational

When code has been committed to the CI/CD Git repository the associated Jenkins job builds the code base. The Jenkins build invokes a Yasca scan of the committed code, which creates a Yasca report in HTML format as well as CSV format. The Yasca results CSV file is further processed and formatted into an xml document. After the Yasca file is processed, Sonar Scanner is invoked to analyze the created XML file using custom rules to map the Yasca results into the SonarQube dashboard.

SonarQube

SonarQube (formerly known as Sonar) is an open source tool suite to measure and analyse to quality of source code. SonarQube provides reporting and management oversight for the CISO and Security team to collect and monitor security issues as part of the CI/CD pipeline.

Yasca-SonarQube Severity Mapping

SonarQube implements five (5) severity levels:

- Blocker
- Critical
- Major
- Minor
- Info

Yasca severity levels are mapped to SonarQube severity levels in accordance with the table below:

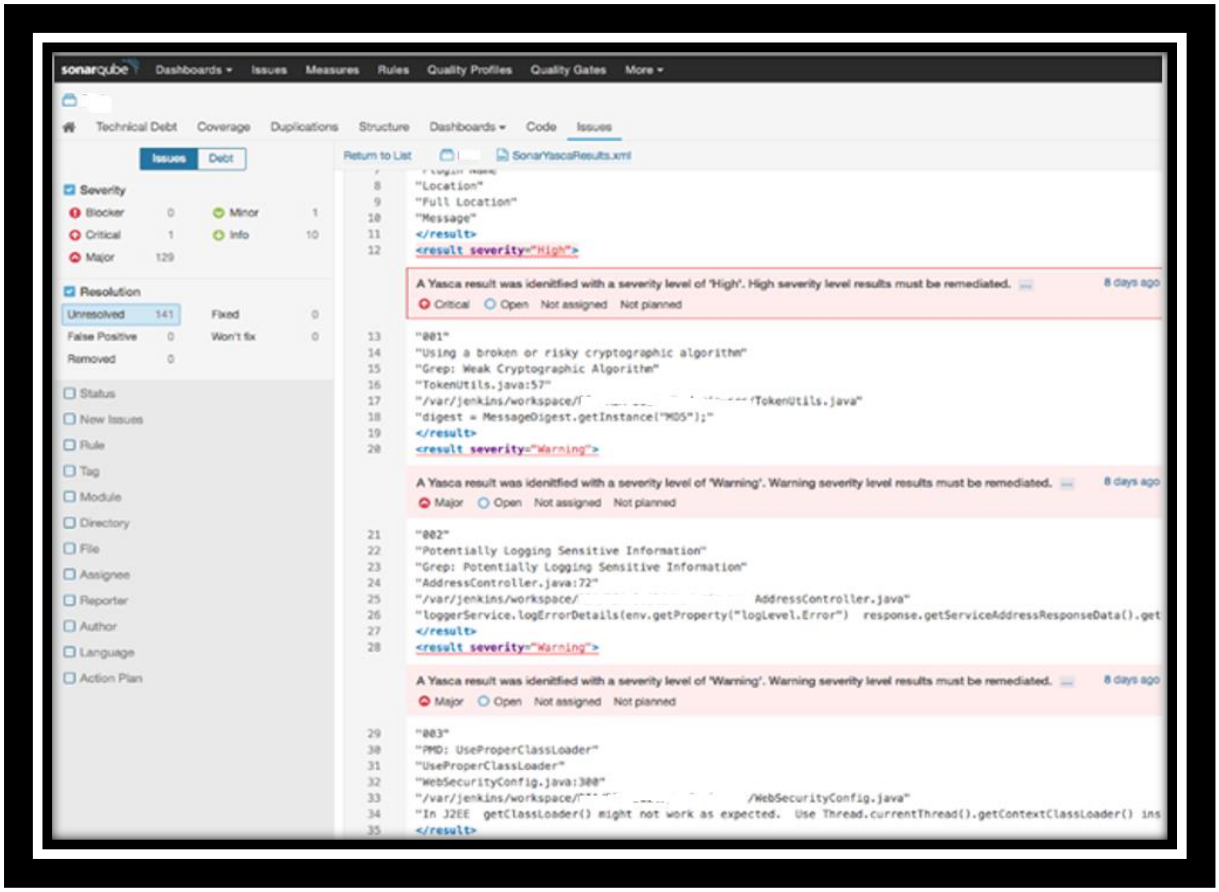
Yasca Severity Level	SonarQube Severity Level
1 – Critical	Blocker
2 – High	Critical
3 – Warning	Major
4 – Low	Minor
5 – Informational	Info

Yasca Sample Reports

Once the mappings are established, Yasca scans performed as part of the CI/CD build process are configured to generate a detailed report of findings and piped into SonarQube. An example of a Yasca report finding is shown.

CSV formatted reports are condensed versions providing Finding #, Plugin Name, Severity, Location, and Message fields. These CSV files are converted to an XML file that is imported to SonarQube.

An example SonarQube report is provided below.



HPE Fortify Static Code Analyzer (SCA)

Depending on the security needs of the organization additional security checks can be added. Commercial packages such as HPE Fortify Static Code Analyzer (SCA) provide static application security testing (SAST). It is used to analyse the source code of an application for security vulnerabilities. It reviews code and helps developers identify and resolve issues during development and testing.

Fortify Severity Levels

Fortify SCA implements four (4) severity levels:

- Critical
- High
- Medium
- Low

Fortify-SonarQube Severity Mapping

SonarQube implements five (5) severity levels:

- Blocker
- Critical
- Major
- Minor
- Info

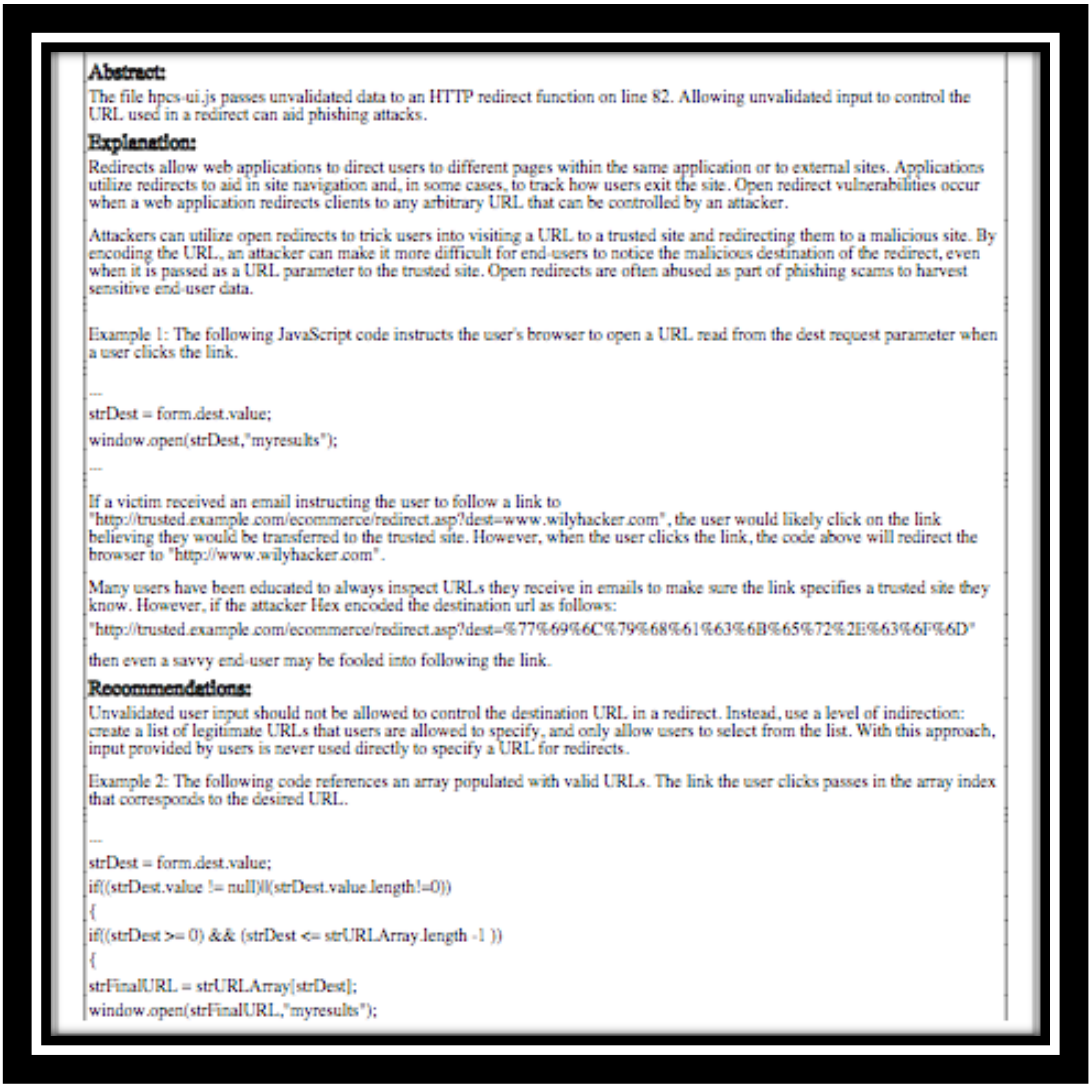
Fortify severity levels are mapped to SonarQube severity levels in accordance with the table below:

Fortify Severity Level	SonarQube Severity Level
Critical	Blocker
High	Critical
Medium	Major
Low	Minor
--	Info

Fortify Sample Reports

By default HPE Fortify SCA natively produces a proprietary result file with an FPR extension. Fortify SCA may also be configured to produce a text (TXT) or an xml-based FVDL file. Fortify SCA also provides a Report Generator utility to produce PDF or XML files. For issues related to the flow of data, Fortify identifies a *Source*, the code that collects and sends input, and a *Sink*, the code that receives/processes the input.

An example of the PDF report is shown below:



Abstract:

The file hpcs-ui.js passes unvalidated data to an HTTP redirect function on line 82. Allowing unvalidated input to control the URL used in a redirect can aid phishing attacks.

Explanation:

Redirects allow web applications to direct users to different pages within the same application or to external sites. Applications utilize redirects to aid in site navigation and, in some cases, to track how users exit the site. Open redirect vulnerabilities occur when a web application redirects clients to any arbitrary URL that can be controlled by an attacker.

Attackers can utilize open redirects to trick users into visiting a URL to a trusted site and redirecting them to a malicious site. By encoding the URL, an attacker can make it more difficult for end-users to notice the malicious destination of the redirect, even when it is passed as a URL parameter to the trusted site. Open redirects are often abused as part of phishing scams to harvest sensitive end-user data.

Example 1: The following JavaScript code instructs the user's browser to open a URL read from the dest request parameter when a user clicks the link.

```
...
strDest = form.dest.value;
window.open(strDest,"myresults");
...
```

If a victim received an email instructing the user to follow a link to "http://trusted.example.com/ecommerce/redirect.asp?dest=www.wilyhacker.com", the user would likely click on the link believing they would be transferred to the trusted site. However, when the user clicks the link, the code above will redirect the browser to "http://www.wilyhacker.com".

Many users have been educated to always inspect URLs they receive in emails to make sure the link specifies a trusted site they know. However, if the attacker Hex encoded the destination url as follows:

"http://trusted.example.com/ecommerce/redirect.asp?dest=%77%69%6C%79%68%61%63%6B%65%72%2E%63%6F%6D"

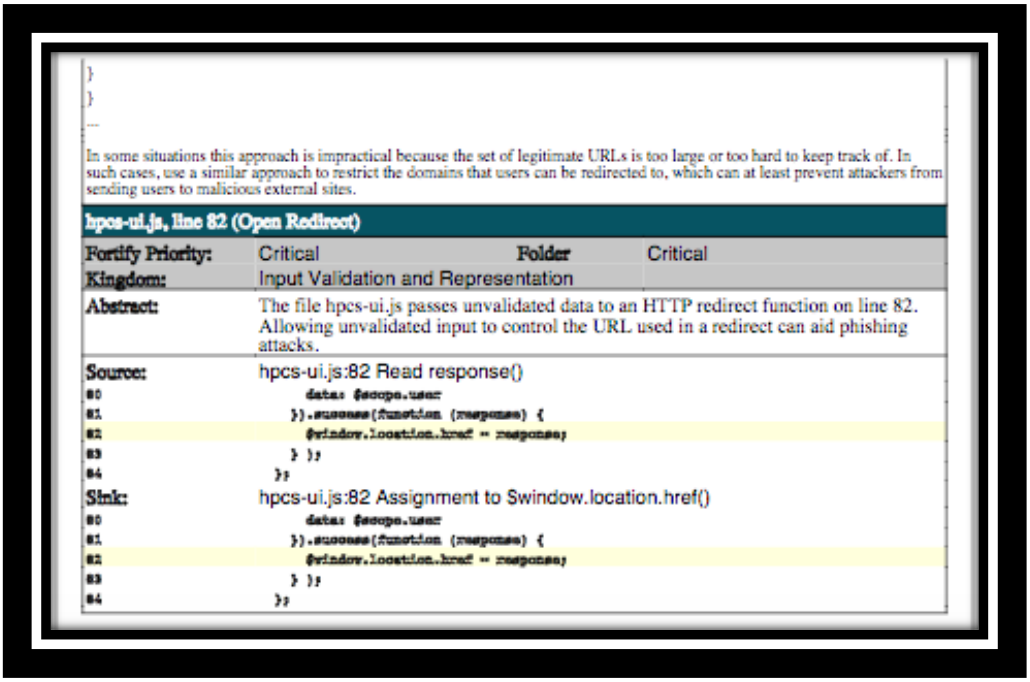
then even a savvy end-user may be fooled into following the link.

Recommendations:

Unvalidated user input should not be allowed to control the destination URL in a redirect. Instead, use a level of indirection: create a list of legitimate URLs that users are allowed to specify, and only allow users to select from the list. With this approach, input provided by users is never used directly to specify a URL for redirects.

Example 2: The following code references an array populated with valid URLs. The link the user clicks passes in the array index that corresponds to the desired URL.

```
...
strDest = form.dest.value;
if((strDest.value != null)&&(strDest.value.length!=0))
{
if((strDest >= 0) && (strDest <= strURLArray.length -1 ))
{
strFinalURL = strURLArray[strDest];
window.open(strFinalURL,"myresults");
```



}
}

In some situations this approach is impractical because the set of legitimate URLs is too large or too hard to keep track of. In such cases, use a similar approach to restrict the domains that users can be redirected to, which can at least prevent attackers from sending users to malicious external sites.

hpcs-ui.js, line 82 (Open Redirect)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Input Validation and Representation		
Abstract:	The file hpcs-ui.js passes unvalidated data to an HTTP redirect function on line 82. Allowing unvalidated input to control the URL used in a redirect can aid phishing attacks.		

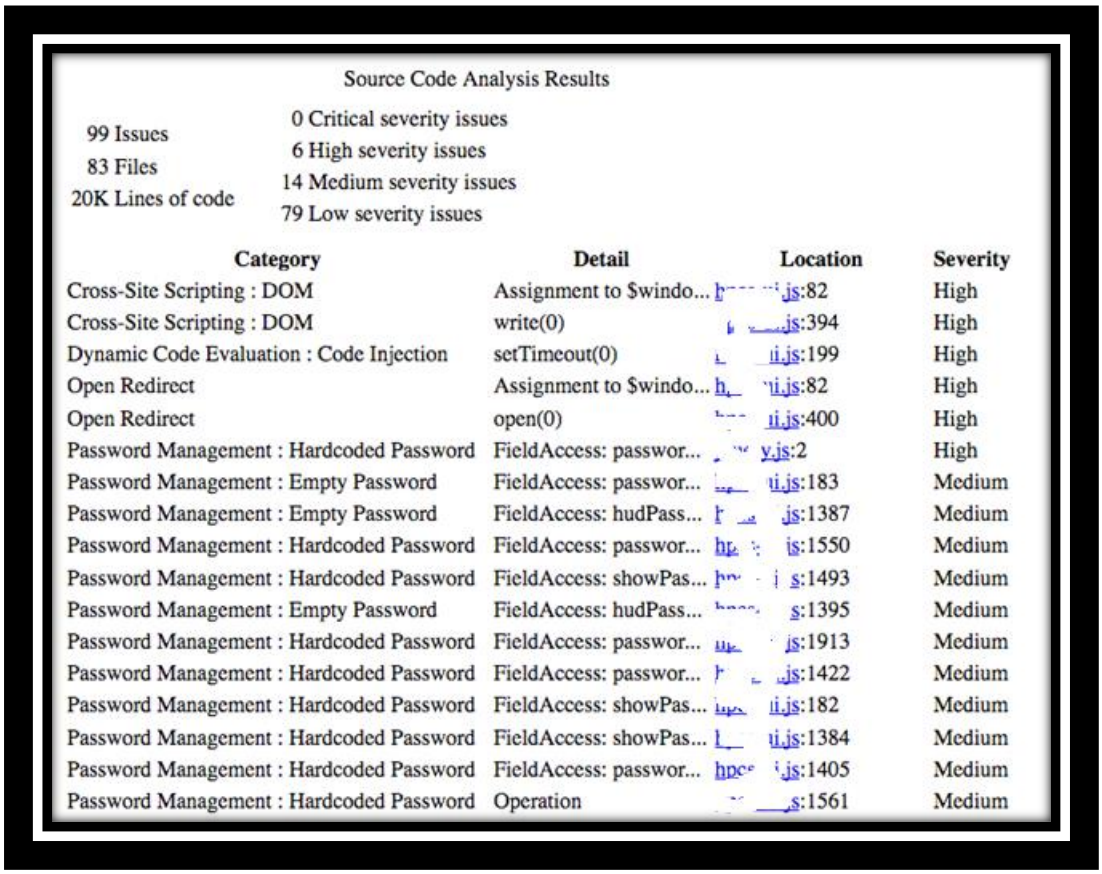
Source: hpcs-ui.js:82 Read response()

```
80          data: $scope.user
81        }).success(function (response) {
82          $window.location.href = response;
83        } );
84      };
```

Sink: hpcs-ui.js:82 Assignment to $window.location.href()

```
80          data: $scope.user
81        }).success(function (response) {
82          $window.location.href = response;
83        } );
84      };
```

An example of the HTML Summary Report is shown below:



Source Code Analysis Results

99 Issues
83 Files
20K Lines of code

0 Critical severity issues
6 High severity issues
14 Medium severity issues
79 Low severity issues

Category	Detail	Location	Severity
Cross-Site Scripting : DOM	Assignment to $windo...	h[illegible]js:82	High
Cross-Site Scripting : DOM	write(0)	[illegible].js:394	High
Dynamic Code Evaluation : Code Injection	setTimeout(0)	[illegible]ui.js:199	High
Open Redirect	Assignment to $windo...	h[illegible]ui.js:82	High
Open Redirect	open(0)	[illegible]ui.js:400	High
Password Management : Hardcoded Password	FieldAccess: passwor...	[illegible]y.js:2	High
Password Management : Empty Password	FieldAccess: passwor...	[illegible]ui.js:183	Medium
Password Management : Empty Password	FieldAccess: hudPass...	[illegible]js:1387	Medium
Password Management : Hardcoded Password	FieldAccess: passwor...	[illegible]js:1550	Medium
Password Management : Hardcoded Password	FieldAccess: showPas...	[illegible]s:1493	Medium
Password Management : Empty Password	FieldAccess: hudPass...	[illegible]s:1395	Medium
Password Management : Hardcoded Password	FieldAccess: passwor...	[illegible]js:1913	Medium
Password Management : Hardcoded Password	FieldAccess: passwor...	[illegible].js:1422	Medium
Password Management : Hardcoded Password	FieldAccess: showPas...	[illegible]ui.js:182	Medium
Password Management : Hardcoded Password	FieldAccess: showPas...	[illegible]ui.js:1384	Medium
Password Management : Hardcoded Password	FieldAccess: passwor...	[illegible].js:1405	Medium
Password Management : Hardcoded Password	Operation	[illegible]s:1561	Medium

Nessus

Nessus Vulnerability Scanner is a vulnerability scanner by Tenable. Nessus identifies system vulnerabilities, missing patches, and non-compliant system configurations. Scans can be performed on a periodic basis and the results are to the CI/CD Project Manager.

Consistent with the DevOps culture, the application development teams are responsible for mitigating findings related to hosted applications. The CI/CD team is responsible for mitigating findings related to the underlying platform (OS, Database, Web Server). The CI/CD team coordinates with application development teams and/or the security team to address platform findings that may affect hosted applications.

OpenSCAP

OSCAP utilizes XCCDF checklist profiles to evaluate system configurations for the operating system against an established checklist profile. The CI/CD pipeline utilizes OSCAP to evaluate the system configurations for the instances supporting the CI/CD development pipeline.

OpenSCAP Severity Levels

OpenSCAP implements four (4) severity levels:

- High
- Medium
- Low
- Other

OpenSCAP-SonarQube Severity Mapping

SonarQube implements five (5) severity levels:

- Blocker
- Critical
- Major
- Minor
- Info

OpenSCAP severity levels are mapped to SonarQube severity levels in accordance with the table below:

OpenSCAP Severity Level	SonarQube Severity Level
High	Blocker
	Critical
Medium	Major
Low	Minor
Other	Info

OpenSCAP Sample Report

System Settings 41x fail 3x error 56x notchecked		
Installing and Maintaining Software 5x fail 1x notchecked		
Disk Partitioning 1x fail		
Ensure /tmp Located On Separate Partition	low	fail
Ensure /var Located On Separate Partition	low	notselected
Ensure /var/log Located On Separate Partition	low	notselected
Ensure /var/log/audit Located On Separate Partition	low	notselected
Ensure /home Located On Separate Partition	low	notselected
Encrypt Partitions	low	notselected
Updating Software 1x fail 1x notchecked		
Ensure Red Hat GPG Key Installed	high	fail
Ensure gpgcheck Enabled In Main Yum Configuration	high	pass
Ensure gpgcheck Enabled For All Yum Package Repositories	high	pass
Ensure Software Patches Installed	high	notchecked
Software Integrity Checking 3x fail		
Verify Integrity with AIDE 3x fail		
Install AIDE	medium	fail
Disable Prelinking	low	fail
Build and Test AIDE Database	medium	notselected
Configure Periodic Execution of AIDE	medium	fail
Verify Integrity with RPM		
Verify and Correct File Permissions with RPM	low	notselected
Verify File Hashes with RPM	low	notselected
Additional Security Software		
Install Intrusion Detection Software	high	notselected
Install Virus Scanning Software	low	notselected
File Permissions and Masks 3x fail 23x notchecked		
Restrict Partition Mount Options 9x notchecked		
Add nodev Option to Non-Root Local Partitions	low	notselected
Add nodev Option to Removable Media Partitions	low	notchecked
Add noexec Option to Removable Media Partitions	low	notchecked
Add nosuid Option to Removable Media Partitions	low	notchecked
Add nodev Option to /tmp	low	notchecked
Add noexec Option to /tmp	low	notchecked
Add nosuid Option to /tmp	low	notchecked
Add nodev Option to /dev/shm	low	notchecked
Add noexec Option to /dev/shm	low	notchecked

ClamAV

ClamAV is an antivirus scanner for Linux operating systems. ClamAV will be installed on Linux servers supporting application development. ClamAV is configured to scan local directories and files for known malicious code on a nightly schedule.

The application development teams are responsible for mitigating findings related to hosted applications. The CI/CD team is responsible for mitigating findings related to the underlying platform (OS, Database, Web Server).

Windows Defender

Windows Defender is an antivirus scanner for Windows operating systems. Windows Defender will be configured on Windows servers and workstations supporting application development. Windows

Defender is configured to scan local directories and files for known malicious code on a nightly schedule.

The application development teams are responsible for mitigating findings related to hosted applications. The CI/CD team is responsible for mitigating findings related to the underlying platform (OS, Database, Web Server).

Conclusion

The rapid adoption of cloud platforms such as Amazon Web Services (AWS) and use of Continuous Integration/Continuous Deployment (CI/CD) practices presents a unique opportunity to deliver more secure code by integrating security practices into the pipeline. There is a wide variety of open source and commercial tools that allow the creation SecDevOps pipelines that assist with the security and information assurance function. By integrating the performance of security testing and scanning as part of the build and deploy process, SecDevOps allows the ability to deliver Continuous Information Assurance.

About stackArmor

stackArmor is a AWS Certified partner with experienced cybersecurity and AWS solution architects with an experience deploying compliant applications for Healthcare, Financial Services, Public Sector, Department of Defense and Commercial customers including Non-profits. We help customers in the following areas:

- AWS Cloud Architecture and Migration Services
- DevOps and Automation Architecture and Implementation Services
- AWS Managed Services and Cloud Operations
- AWS Value-Added Resale and Hosting Support Services
- Cybersecurity Compliance and Penetration Scanning Services

Additionally, we have an out-of-the-box solution - stackArmor StackBuilder™ is a “Turbo Tax” like wizard for helping application owners quickly configure a fully functional AWS environment. The wizard walks the user through a series of simple questions through a 5 step process. Upon submission of the request, the user is presented with login credentials to a fully configured and operational environment ready to go.

StackBuilder™ provides a rich and easy to use consumer-grade experience for non-technical users to jumpstart their projects by answering a series of simple questions. StackBuilder’s intelligent provisioning and capacity estimation engine leverages the rich set of services provided by the AWS cloud platform including wide variety of EC2 instances, Virtual Private Cloud (VPC), Auto Scaling Groups, Clustering and Elastic Load Balancers (ELB) amongst others. The user of StackBuilder™ does not have to go through the various steps associated with configuring and setting up the AWS infrastructure as they are handled automatically. This allows the user to focus on his project without waiting for costly consultants or the need for cloud infrastructure expertise.

Please contact us at solutions@stackarmor.com or call at 888-964-1644.

Resources

1. White paper on Cloud Security Best Practices and Common Errors.

   https://www.stackarmor.com/resources/

2. https://www.stackarmor.com/is-your-business-ready-for-the-coming-cybersecurity-tsunami/