CALIFORNIA STATE UNIVERSITY
SAN BERNARDINO

CSUSB Vulnerability Management Guidelines

CSUSB, Information Security &

Emerging Technologies Office

Last Revised: **09/17/2015**

Final

REVISION CONTROL

Document Title: CSUSB Vulnerability Management Guidelines

Author: Javier Torner

File Reference:

Date	By	Action	Pages
09/15/2015	J Torner/J Macdonell	Created Standard	All
9/17/2015	L Carrizales	Guidelines approved by ISET Subcommittee on 9/16/15. Made changes to the document based on recommendations from ISET Subcommittee.	All

Review/Approval History

Date	By	Action	Pages
9/16/2015	ISET Subcommittee	Approved Guidelines	All

Contents

Guidelines - Vulnerability Management

1. Introduction

An essential component of risk management for information technology (IT) infrastructure is the inventory and remediation of known vulnerabilities. To reduce this risk, vulnerabilities should be inventoried and remediated in a timely manner.

Inventory may be conducted using a variety of techniques: vulnerability scanning software (e.g. Nessus, Retina, Qualys), systems management software (e.g. SCCM), patch management software, code review, and physical inventory. Essentially, each technique probes systems and applications to identify and categorize known vulnerabilities.

Remediation, the process of correcting or mitigating a vulnerability, may include a combination of: unattended automatic updates, group policies, scheduled maintenance, code patching, deployment of network or application firewalls, physical relocation, etc.

2. Inventory Methodologies

The CSUSB Vulnerability Management Standard requires creating an inventory of systems, applications, and their associated vulnerabilities.

Additionally, vulnerability metrics need to be tracked to facilitate vulnerability reporting and risk management. Example metrics include: number of vulnerable systems, number of classified vulnerabilities (critical, high, medium, low), time to remediation, etc. Most metrics are created and tracked using vulnerability scanning software.

2.1. General (Blackbox) Scanning

General scans are appropriate for Internet-facing systems. They are meant to report back vulnerabilities found within the public-facing services. They are more aggressive as compared to credentialed scans as the services are repeatedly subjected to known exploits.

2.2. Credentialed Scanning

Credentialed scans are appropriate for most CSUSB network-connected systems. These scans are programmed with administrative credentials. The vulnerability scanning agent will crawl filesystems and registries to inventory software versions, policy settings, etc. They are less aggressive as compared to general scans, as they do not include probing with exploits.

2.3. Application-Specific Scanning

Application specific vulnerability tools are most appropriate for custom-developed applications. For example, web vulnerability scanners accommodate forms-based logins, and can coordinate with server-side modules that monitor the runtime and system logs generated during the scan.

2.4. Systems Management Inventory

A complement or possible alternative to vulnerability scanning is the use of software inventory from systems management software, which can be used to inventory and track at-risk systems by querying for out-of-date software.

2.5. Questionnaires and Benchmarks

For information assets and repositories where a network vulnerability scan is inappropriate, such as a paper repository, tools similar to the sensitive data questionnaire or a checklist provided for a regulated data such as electronic health records.

3. Scanning Tools

In addition to being used routinely for vulnerability inventory and metrics, vulnerability scanning is a form of security testing and may be used during software development. Vulnerability scanning may also be required as part of the Quality Assurance (QA) process depending on the criticality of an application. Applications in production should be periodically scanned. Web application specifically should be scanned according to a schedule defined in the CSUSB Web Application Standard.

3.1. Development Environments

Consider first testing the behavior of vulnerability scanning software against development environments. Vulnerability scanners take precautions to prevent unwanted side effects. However, they are known to crash sensitive devices like printers and videophones and may also corrupt data of particularly vulnerable web applications.

3.2. Use-case Considerations

When scanning applications, consider multiple use cases: submitting a form, creating a report, changing permissions, logged in, not logged in, etc. Also consider multiple user roles: guest, trusted user, administrator, etc.

3.3. Production Workflow

Scans of production systems and applications should be scheduled to minimize impact. Also consider availability of administrators to correct any issues that may arise as an unintended result of vulnerability scanning. System administrators and developers may perform preliminary or on-demand unofficial scans. To gain access to the scanning tool, send an email to security@csusb.edu.

4. Vulnerability Remediation

4.1. Typical Action

For system vulnerabilities, most often a corrective action is included in the vulnerability scan report (e.g. "upgrade openssl to version 1.0.1e-2+deb7u17" or "see Microsoft Security Bulletin MS15-097" or "Apply Group Policy option …"). The expected action is to apply a software update, or modify a configuration option.

Similarly, for application vulnerabilities, the vulnerability scan report most often references a best practice or references an article to help a developer identify and correct the vulnerability.

4.2. False Positives

In some cases, the scanner will flag a possible issue that, upon examination, is found to not actually be a security issue. For example, if a Web page contains a benign string of digits that just-so-happens to match the pattern of a credit card number, the scanner may raise an alert on that pattern. However, the digits might not actually be a credit card number. This is called a False Positive.

Often false positives can be excluded from future reports through the report editing feature of the vulnerability scanning software.

4.3. Reclassification

In some cases, the scanner will flag a possible vulnerability that, upon examination, is found to indeed be a correctly identified issue, but for business or other reasons the concern is not relevant in the specific situation.

As an example, the scanner may find a pattern matching an email address on a web page and flag it as a risk. Upon review, it is seen that the pattern actually is an email address. The scanner is correct. This is not a false positive. However, this may not be a pertinent concern. Perhaps the webpage is a "Contact Us" page. In situations like this, the security posture being evaluated by the scanner does not match the security posture desired by CSUSB.

4.4. Mitigation

Occasionally, a vulnerability is discovered for which no corrective action is available. As an example, historically applications have linked against a Microsoft XML library that is no longer supported by Microsoft. In such cases, mitigating controls, such as limiting firewall access or requiring a second factor for authentication, may be appropriate.

5. References

Sensitive Data Questionnaire -- https://iso.csusb.edu/policies
Example checklist for regulated data: http://www.healthit.gov/providers-professionals/security-risk-assessment-tool