# **DCS Product Security Testing Process**

# Scope

This process covers product security testing process for DCS products and solutions.

Product security testing and validation coverage include:

1. Functional Testing of new and updated Security Features and requirements.
2. Periodic vulnerability Scanning of interim and final releases.
3. Report issues following the vulnerability management procedures.

(YN: Mitigation is part of vulnerability management not part of testing)

1. Penetration tests performed by either internaly or by a 3rd part vendors.
2. **Functional Testing of Security Features**

Functional testing of the security features involves the black box testing of the security features and security protocol implemented in the system.

For example, covering the following features during this phase

## SSH Protocol Tests

* 1. Run ‘SSHredder’ test suite to validate multiple vulnerabilities in SSH protocol implementation
  2. Execute ‘SSH\_Scan’ open-source tool for analyzing SSH protocol

## SSL Protocol Tests

* 1. Run ‘testssl.sh’ and ‘sslyze’ tool to check the support of TLS/SSL ciphers, protocols as well as some cryptographic flaws and verify the SSL vulnerabilities (Heartbleed, poodle, beast, breach)
  2. Run ‘sslscan’ to check the SSL protocol implementation

1. Identify unauthorized access.
2. Identify unprotected private data
3. Ensure password are in compliance with the requirements (YN: add link)

# Vulnerability Scanning

Vulnerability scanning can help identify outdated software versions, missing patches, and misconfigurations, and validate compliance with or deviations from security policy. This is done by identifying the operating systems and major software applications running on the hosts and matching them with information on known vulnerabilities stored in the scanners’ vulnerability databases.

To accomplish this, we would be using either a commercial or open-source Vulnerability scanners

These Vulnerability scanners will

* + Identifies hosts and open ports
  + Identifies known vulnerabilities
  + Often provides advice on mitigating discovered vulnerabilities

The following are some examples of open-source tools that can be used:

(YN: what about Vcode, and Tenable.io)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tool Tested** | **Protocol** | **Feature** | **OS Support** | | | **Info** |
| **Linux** | **Windows** | **Android** |
| SSHredder | SSH | Validate multiple vulnerabilities in SSH | ü | ü | û | <https://www.cvedetails.com/vulnerability-list/vendor_id-471/Putty.html> |
| ssh\_scan & ssh-audit | SSH | SSH security scanners | ü | ü | û | [https://github.com/mozilla/ssh\_scan https://github.com/arthepsy/ssh-audit](https://linuxsecurity.expert/tools/ssh_scan/) |
| testssl\_sh | SSL | SSL/TLS scanner and configuration audit tool | ü | ü | û | <https://github.com/drwetter/testssl.sh> |
| Lynis | OS platform | Auditing, system hardening, compliance testing | ü | û | û | <https://cisofy.com/lynis/> |
| Sn1per | Miscallaneous | Automated scanner that can be used during a penetration test to enumerate and scan for vulnerabilities | ü | ü | û | <https://github.com/1N3/Sn1per> |
| ZAPScanner | Web Application | Web app scanner | NA | NA | NA | <https://www.zaproxy.org/> |
| nmap\_vulners | Miscallaneous | NSE script using some well-known service to provide info on vulnerabilities | ü | ü | û | <https://github.com/vulnersCom/nmap-vulners> |
| Vulscan | Miscallaneous | Vulscan is a module which enhances nmap to a vulnerability scanner | ü | ü | û | <https://github.com/scipag/vulscan> |
| OpenVAS | Miscallaneous | Vulnerability Assessment Scanner | ü | ü | û | <https://www.openvas.org/> |
| Archery | Miscallaneous | opensource vulnerability assessment and management tool | ü | ü | û | <https://github.com/archerysec/archerysec> |

# Vulnerability Reporting

Critical and HIGH vulnerabilities (defects) found in the system is recommended to be reported within 24 hours of the discovery.

All the vulnerabilities to be tracked using the standard tooling (JIRA\*) and the fixes should be applied as per the vulnerability management process.

Mitigation recommendations, includes the outcome of the root cause analysis, should be developed for each finding. Providing a technical recommendation like updating a particular patch. Examples of mitigation actions include process and procedure modifications; security architecture changes; deployment of new security technologies; and deployment of OS and application patches.

Upon completion of analysis, a report should be generated that identifies system and network vulnerabilities and their recommended mitigation actions. Security testing results can be used in the following ways:

* As a reference point for corrective action
* In defining mitigation activities to address identified vulnerabilities
* A benchmark for tracking the product’s progress in meeting security requirements