UL 2900-1

Outline of Investigation for Software Cybersecurity for Network-Connectable Products, Part 1: General Requirements

Issue Number: 1

March 30, 2016

Summary of Topics

The UL 2900-1 outline aims to provide a minimum set of requirements that developers of network-connectable products can pursue to establish a baseline of protection against vulnerabilities and software weaknesses, along with a minimum set of security risk controls and documentation to consider relative to their existing overall product risk assessments.

No Text on This Page

CONTENTS

INTRODUCTION

1 Scope

1.1 This outline applies to network-connectable products that shall be evaluated and tested for vulnerabilities, software weaknesses and malware.

1.2 This outline describes:

a) Requirements regarding the vendor's risk management process for their product.

b) Methods by which a product shall be evaluated and tested for the presence of vulnerabilities, software weaknesses and malware.

c) Requirements regarding the presence of security risk controls in the architecture and design of a product.

1.3 This outline does not contain requirements regarding functional testing of a product. This means this outline contains no requirements to verify that the product functions as designed.

1.4 This outline does not contain requirements regarding the hardware contained in a product.

2 Normative References

2.1 All references are for the latest published version of the document, unless stated otherwise.

[1] UL 2900-2-1
Outline of Investigation for Software Cybersecurity for Network-Connectable Products, Part 2-1: Particular Requirements for Network Connectable Components of Healthcare Systems

[2] UL 2900-2-2
Outline of Investigation for Software Cybersecurity for Network-Connectable Products, Part 2-2: Particular Requirements for Industrial Control Systems

[3] ITU-T X.1520,
Cybersecurity information exchange – Vulnerability/state exchange – Common vulnerabilities and exposures (CVE)

[4] ITU-T X.1521,
Cybersecurity information exchange – Vulnerability/state exchange – Common vulnerability scoring system (CVSS)

[5] ITU-T X.1524,
Cybersecurity information exchange – Vulnerability/state exchange – Common weakness enumeration (CWE)

[6] ITU-T X.1525,
Cybersecurity information exchange – Vulnerability/state exchange – Common weakness scoring system (CWSS)

[7] ITU-T X.1544,
Cybersecurity information exchange – Event/incident/heuristics exchange – Common attack pattern enumeration and classification (CAPEC)

[8] Common Weakness Risk Analysis Framework (CWRAF); retrievable from https://cwe.mitre.org/cwraf/

[9] CWE/SANS Top 25 Most Dangerous Software Errors; retrievable from cwe.mitre.org/top25

[10] CWE On the Cusp: other weaknesses to consider; retrievable from *https://cwe.mitre.org/top25/cusp.html*

[11] OWASP Top 10; latest version retrievable from *https://www.owasp.org/index.php/Top_10_2013-Top_10*

[12] ISO/IEC 11889,
Information technology – Trusted platform module library

[13] ISO/IEC 9796 (all parts),
Information technology – Security techniques – Digital signature scheme giving message recovery

[14] ISO/IEC 9797 (all parts),
Information technology – Security techniques – Message Authentication Codes (MACs)

[15] ISO/IEC 9798 (all parts),
Information technology – Security techniques – Entity authentication

[16] ISO/IEC 10118 (all parts),
Information technology – Security techniques – Hash-functions

[17] ISO/IEC 11770 (all parts),
Information technology – Security techniques – Key management

[18] ISO/IEC 14888 (all parts),
Information technology – Security techniques – Digital signatures with appendix

[19] ISO/IEC 15946 (all parts),
Information technology – Security techniques – Cryptographic techniques based on elliptic curves

[20] ISO/IEC 18033 (all parts),
Information technology – Security techniques – Encryption algorithms

[21] ISO/IEC 19772 (all parts),
Information technology – Security techniques – Authenticated encryption

[22] NIST FIPS 140-2, Annex A: Approved Security Functions

[23] NIST FIPS 140-2, Annex D: Approved Key Establishment Techniques

3 Glossary

3.1 ATTACK – The use of one or more exploit(s) by an adversary to achieve one or more negative technical impact(s).

3.2 ATTACK PATTERN – A description of a generic method for carrying out attacks.

3.3 AUTHENTICATION – The process of verifying the identity of an entity.

3.4 AUTHENTICITY – The property that data, information or software originate from a specific entity, which may or may not have been authenticated.

3.5 AUTHORIZATION – The process of giving an entity permission to access or manipulate the product, or the property that an entity has such permission.

3.6 BINARY CODE – Machine instructions and/or data in a format intended for a specific processor architecture.

3.7 BYTECODE – Instructions and/or data that are created from source code as an intermediate step before generating binary code. Bytecode is independent of a specific processor architecture and is typically handled by a virtual machine or interpreter.

3.8 COMMON ATTACK PATTERN ENUMERATION AND CLASSIFICATION (CAPEC) – Specified in ITU-T X.1544 (ref. [7]), the CAPEC is a publicly available resource providing a list and classification of a large number of attack mechanisms based on the topology of the environment.

3.9 COMMON VULNERABILITIES AND EXPOSURES (CVE) – Specified in ITU-T X.1520 (ref. [3]), the CVE is a publicly available resource providing common identifiers for known vulnerabilities and exposures.

3.10 COMMON VULNERABILITY SCORING SYSTEM (CVSS) – Specified in ITU-T X.1521 (ref. [4]), the CVSS is a publicly available resource providing a means for prioritizing vulnerabilities in terms of exploit potential.

3.11 COMMON WEAKNESS ENUMERATION (CWE) – Specified in ITU-T X.1524 (ref. [5]), the CWE is a publicly available resource providing a structured means to exchange unified, measurable sets of information providing common identifiers for software weaknesses, as well as consequences, detection methods and examples of each weakness.

3.12 COMMON WEAKNESS SCORING SYSTEM (CWSS) – Specified in ITU-T X.1525 (ref [6]), the CWSS is a publicly available resource providing a means for prioritizing CWEs based on their technical impact, ease of attack, and other factors.

3.13 COMMUNICATION PROTOCOL – A system of rules regarding syntax, semantics, synchronization and error recovery of communication, allowing two or more entities to exchange information.

3.14 CONFIDENTIALITY – The property that data, information or software is not made available or disclosed to unauthorized individuals, entities, or processes.

3.15 EXECUTABLE – A file containing instructions in binary code, which can be used by a computer to perform computational tasks.

3.16 EXPLOIT – An input or action designed to take advantage of a weakness (or multiple weaknesses) and achieve a negative technical impact.

NOTE: The existence of an exploit targeting a weakness is what makes that weakness a vulnerability.

3.17 EXTERNAL INTERFACE – An interface of the product that is designed to potentially allow access to an entity outside the product; for example user interfaces, remote interfaces, local interfaces, wireless interfaces and file inputs.

3.18 FILE – A collection of data or program instructions stored as a unit with a single name.

3.19 GENERATIONAL MALFORMED INPUT TESTING – A method of deriving malformed input test cases by using detailed knowledge of the syntax and semantics of the specifications of the protocol or file format being tested.

3.20 HARM – Physical injury or damage to the health of people, or damage to property or the environment.

3.21 INTEGRITY – The property of data, information or software not having been improperly modified.

3.22 KNOWN VULNERABILITY – A vulnerability described in the National Vulnerability Database (NVD).

NOTE: The NVD is accessible at https://nvd.nist.gov.

3.23 LOCAL INTERFACE – An external interface potentially allowing access only to individuals, entities or systems within a very acute proximity requiring physical access to the product.

NOTE: An example is a physically wired direct connection like a USB connection or RS 485 connection within physical proximity.

3.24 MALFORMED INPUT TESTING – A black-box testing technique used to reveal software weaknesses and vulnerabilities in a product by triggering them with invalid or unexpected inputs on the external interfaces of the product.

3.25 MALFORMED INPUT TEST CASE – The basic unit of malformed input testing, which consists of a single interaction with the product under test.

3.26 MALWARE – Software designed with malicious intent to disrupt normal function, gather sensitive information, and/or access other connected systems.

3.27 NETWORK – A collection of nodes and telecommunication links, allowing connected devices, software etc. to exchange data and communicate.

3.28 PENETRATION TESTING – A mechanism of evaluation of a product to exploit vulnerabilities and weaknesses discovered in the vulnerability assessment phase.

3.29 PERSONALLY IDENTIFIABLE INFORMATION – Any information belonging to an individual that can uniquely distinguish an individual or information that can be used to derive their identity.

NOTE: This can be, but is not limited to an individual's location, health records and/or financial records that when used can determine the actual individual's identity.

3.30 PRODUCT – The network-connectable device, software or system under test.

3.31 PROTOCOL – See COMMUNICATION PROTOCOL

3.32 REMOTE INTERFACE – An external interface potentially allowing access to individuals, entities or processes regardless of geographic distance to the product.

3.33 RISK – The potential for harm or damage, measured as the combination of the likelihood of occurrence of that harm or damage and the impact of that harm or damage.

3.34 RISK ANALYSIS – The systematic use of available information to identify threats and to estimate risk.

3.35 RISK CONTROL – Any action taken or feature implemented to reduce risk.

3.36 RISK MANAGEMENT – Systematic application of management policies, procedures and practices to the tasks of analyzing, evaluating, controlling and monitoring risk.

3.37 SECURE ELEMENT – A tamper-resistant platform like a chip capable of securely hosting applications and their confidential and cryptographic data and will prevent unauthorized access.

3.38 SECURITY – The state of having acceptable levels of confidentiality, integrity, authenticity and/or availability of product data and/or functionality.

3.39 SENSITIVE DATA – Sensitive data is any critical security parameter that can compromise the use and security of the product such as passwords, keys, seeds for random number generators, authentication data.

3.40 SOFTWARE – All pre-loaded data which creates, affects, and/or modifies the functionality of the product. This includes, but is not limited to, firmware, scripts, initialization files, pre-compiled code and interpreted code. This does not include software preloaded and programmed in an IC chip for small functions that require physical access and removal of the IC chip for reprogramming.

3.41 SOFTWARE WEAKNESS – A mistake in the architecture, design, coding, build process or configuration of software in the product, that may render the product vulnerable to a security exploit.

3.42 SOURCE CODE – Computer instructions written in a human-readable high-level computer language, usually as text, including possible comments.

3.43 STATIC ANALYSIS – A process in which source code, bytecode or binary code is analyzed without executing the code.

3.44 TEMPLATE MALFORMED INPUT TESTING – Also known as mutational fuzzing, template malformed input testing generates test cases by introducing anomalies into a valid message or file. Template malformed input test cases are not protocol aware and therefore will not contain items such as correct checksums and valid session IDs.

3.45 THREAT – A potentially successful attack, involving an adversary utilizing specific techniques and resources to take advantage of specific vulnerabilities or lack of risk controls within a product.

3.46 TRUSTED PLATFORM MODULE – An international standard that defines the requirements for a dedicated microprocessor with requirements for storage of cryptographic keys used to secure physical products and the software contained.

3.47 USER – A person or process using a product or accessing it over one of its external interfaces.

3.48 VENDOR – The manufacturer, reseller or supplier of a product, which takes final responsibility for the cybersecurity of that product towards the purchaser and/or user and which submits that product for testing according to this outline.

3.49 VULNERABILITY – A software weakness found in the product for which an exploit may exist, such that it can be directly used by an attacker.

3.50 WIRELESS INTERFACE – An external interface using electromagnetic waves, rather than some form of wire, to carry communication signals to and from a product.

DOCUMENTATION OF PRODUCT, PRODUCT DESIGN AND PRODUCT USE

4 Product Documentation

4.1 The vendor shall provide the following for a product evaluation:

a) A description of all functions provided by the product, including any management functions.

b) A list of all external interfaces or physical inputs or outputs of the product in its intended configuration, including:

1) All remote interfaces,

2) All local interfaces – product local internal interfaces such as SPI, I2C, JTAG and serial ports shall be included,

3) All wireless interfaces,

4) All file inputs,

5) All communication protocols supported on each of these interfaces.

c) A list of all executables and libraries in the product, including all third party and open source software. All executables and libraries shall be identified by both a software name and version number.

d) The source code of all software in the product, to the extent that source code is available to the vendor. This source code shall be the production code or fully representative of production (release) code. It shall include all scripts, libraries, makefiles, build configuration parameters and tool information. The source code provided shall be unobfuscated.

e) The binary code and/or bytecode of all software in the product, unless the vendor has no access or no rights to this binary or bytecode. The binary code and/or bytecode provided shall be unobfuscated.

f) Information on the product software build and integration process.

5 Product Design Documentation

5.1 The vendor shall provide the following for a product evaluation:

a) The security risk analysis for the product as described in 12.1 of this outline.

b) The design documentation containing sufficient details to allow an evaluation of the way each of the risk controls mentioned in Sections 7 – 11 is implemented in the product.

6 Documentation for Product Use

6.1 The vendor shall provide documentation addressing security considerations on the intended use of the product and the configuration and environment in which the product is intended to be used. The vendor shall make all documentation referred to in this section (i.e., Section 6) available to the purchasers of the product or to those authorized to have access to such product documentation. The vendor shall also provide this documentation for a product evaluation.

6.2 The vendor shall provide instructions for product use. The vendor shall document all functions provided by the product, including any management functions.

6.3 The vendor shall document all external interfaces and all communication protocols used by the product, including which external interfaces support which protocols.

6.4 The vendor shall document all versions of all software components used in the product.

6.5 The vendor shall document the list of events, including security-related events, logged by the product according to 11.3.

6.6 The vendor shall document any requirements and recommendations on the product's configuration and the environment in which the product is installed that are necessary to ensure the product's security.

NOTE: This may include requirements on network security, physical access control to the product, firewall ports and protocols, local interfaces' configuration options etc.

6.7 The vendor shall provide rationale that authentication of the product requires at least as many operations to circumvent the authentication as it does to randomly identify or guess the means of authentication.

NOTE: For example, if a key is used for authentication, then it shall require at least as many operations to circumvent the key as it is to determine the key.

RISK CONTROLS

7 General

7.1 The product (or the product's vendor, as applicable) shall comply with all of the security risk controls specified in Sections 7 – 11, unless the risk assessment performed by the vendor according to Section 12, Vendor Product Risk Management Process, shows that the risks associated with not implementing a specific control are acceptable in product use.

7.2 If the vendor chooses to not comply with one or more of these risk controls, the vendor shall document and justify this in the risk analysis per 12.1.

8 Access Control, User Authentication and User Authorization

8.1 Product operation or management services which may affect or alter the security of the product shall require user authentication prior to access.

8.2 User authentication services to the product shall implement a session time-out or other appropriate mechanism to prevent perpetual authorization. The session time-out shall be configurable.

8.3 Services that are accessible over a remote interface shall require user authentication prior to access.

Exception: Services that report status, do not provide command and control functionality or general use of the product or do not transmit sensitive data or personally identifiable data AND only output status or historical transaction data, etc., may provide unauthenticated access but will need to be documented as per Section 12, Vendor Product Risk Management Process.

8.4 Services that are accessible over a wireless interface shall require user authentication prior to access.

Exception: Services that report status, do not provide command and control functionality or general use of the product or do not transmit sensitive data or personally identifiable data AND only output status or historical transaction data, etc., may provide unauthenticated access but will need to be documented as per Section 12, Vendor Product Risk Management Process.

8.5 If the product uses a user name-and-password mechanism for authenticating users:

a) The product shall use a secure mechanism complying with the requirements in Appendix B to store the passwords.

b) Authentication error messages provided by the product shall not allow for enumerating valid user names.

c) The product shall support the possibility to set requirements regarding the length, complexity and update frequency for passwords.

NOTE: Complexity options can include special characters, minimum length, upper and lowercase and combinations of options.

d) The product shall protect against dictionary attacks and brute force attacks.

NOTE: Examples of mechanisms to do so include key stretching or preventing login attempts for the given user after a specified number of failed attempts.

e) The product shall have no hardcoded passwords that cannot be removed or altered.

8.6 For products using a role-based access mechanism:

a) The vendor shall clearly document all existing roles and their associated privileges.

b) There shall be an 'administrator' or 'system' role that has privileges exclusively related to the management of the product. Such privileges shall not be granted to other roles.

8.7 The product shall support the possibility to manage the list of valid user accounts by adding, removing and/or suspending user accounts (i.e. "whitelisting" and/or "blacklisting") or by addition, revocation or updating of authentication credentials.

8.8 The product shall enforce the principle of least privilege for every account that can be authenticated.

8.9 If a communication session over a remote interface is lost or terminated, the product shall require renewed authentication prior to allowing access over the remote interface. Stored data from the previous session shall not be used to initiate the new session.

8.10 If the product uses other mechanisms for authentication besides username and password, the mechanism used for authentication shall require as many operations to circumvent as determining the actual mechanism.

NOTE: For example, if a key is used for authentication, then it shall require at least as many operations to circumvent the key as it is to determine the key.

9 Remote Communication

9.1 The product shall ensure the integrity and authenticity of all data communicated over any remote interface. For this, the product shall use security functions complying with the requirements in Appendix C.

Exception: Remote interfaces that report status, do not provide command and control functionality or do not transmit sensitive data, etc., may not ensure integrity and authenticity but will need to be documented as per Section 12, Vendor Product Risk Management Process.

10 Cryptography

10.1 The product shall ensure the confidentiality of all sensitive data and personally identifiable data generated, stored, used or communicated by the product. The product shall use a secure mechanism complying with the requirements in Appendix B to store the sensitive data and personally identifiable data.

10.2 For the purposes of 10.1, the vendor shall identify and document which data is to be considered sensitive. Sensitive data shall include at least all personally identifiable information and any data whose disclosure could jeopardize the security properties of the product, such as cryptographic keys and passwords.

10.3 The product shall utilize only cryptographic algorithms listed in Appendix C for any security protocol. Moreover, the product shall utilize only widely accepted implementations of these algorithms and modes of operation.

10.4 The product shall use each cryptographic key only for a single intended purpose, unless a single process is able to serve multiple purposes. The vendor shall clearly document the intended purpose of each key used by the product.

NOTE: Purposes may include (but are not limited to) data encryption, providing data authenticity and integrity, key wrapping, random number generation or digital signatures.

11 Product Management

11.1 The product shall be designed and implemented such that it is possible to perform an update of the product's software, and to roll back an update to the current version during the update process if it fails.

11.2 The product shall verify the authenticity and integrity of any software update cryptographically, before installing the update. Product updates shall be possible in an offline environment. This offline product update mode should also still support validation of authenticity and integrity.

11.3 The product shall be capable of maintaining one or more log(s) of all security-related events, such as successful and unsuccessful login attempts, change of user authentication credentials, changes in the list of valid user accounts, successful and unsuccessful software updates, etc.

11.4 Unless and until they are transmitted to an external data storage, the product shall store all security-related logs in non-volatile memory and shall not allow non-privileged users to remove or change them.

11.5 Prior to its initial operation in production, the product shall require changes of any system defaults that play a role in product security, such as passwords and keys. The product shall have documentation recommending changing of system defaults.

11.6 Decommissioning of the product after its use shall allow the ability to completely erase all configuration data, sensitive data and personally identifiable data. Zeroization of this data is acceptable and can be performed as an operation or as a process procedure:

a) The operation or procedure shall at least include two steps of overwriting the configuration data, sensitive data and personally identifiable data with data that is not related.

b) The operation or procedure shall destroy the configuration data, sensitive data or personally identifiable data from all components of the product.

11.7 The following are approved integrity mechanisms for software updates:

a) A message authentication code generated on the software and firmware components.

b) A digital signature generated on the software and firmware components.

c) A hash generated on the software and firmware components, where the hash is published in such a way that it is difficult for an attacker to change.

11.8 All integrity mechanisms defined in 11.7 shall comply with Appendix C.

RISK MANAGEMENT

12 Vendor Product Risk Management Process

12.1 When designing the product, the vendor shall establish and document a security risk analysis for the product, containing:

a) An identification of all product functionalities and all data stored, processed or used by the product;

b) A list of all threats for the product, its functionalities and data;

c) An assessment of the impact of each identified threat, its operational impact and any PII or sensitive information that would be exposed, should it become a reality;

d) An assessment of the likelihood of each identified threat;

e) A determination of the resulting risk level for each threat, considering its impact and likelihood;

f) Risk acceptance criteria, (i.e. clear criteria to determine whether or not a given risk level is acceptable);

g) A determination of suitable risk controls to mitigate each threat with an unacceptable risk level. All risk controls in Sections 7 – 11 shall be considered. The vendor shall identify any additional risk controls that need to be implemented to mitigate threats. The vendor shall create a traceability matrix showing the relationship between threats and implemented risk controls.

h) An assessment of the residual risk level for each threat after application of these risk controls.

12.2 When carrying out the threat analysis resulting in the list meant in 12.1(b), the vendor shall make use of the Common Attack Pattern Enumeration and Classification (CAPEC; see ref. [7]) in order to convey the completeness of the analysis.

12.3 The vendor shall document a risk evaluation method for the possible presence of known (types of) vulnerabilities in the product. This method shall describe the criteria that the vendor will use to evaluate the level of risk for each (type of) known vulnerabilities that may be found in product. The method shall also establish the level below which a risk is acceptable to the vendor. The evaluation criteria shall be based on risk factors including, but not limited to, the CVSS score of the vulnerability, the intended use of the product and the environment in which the product would be used.

12.4 If the vendor has allowed for the presence of any known vulnerabilities in the product, the vendor's security risk analysis for the product shall contain a description of each accepted known vulnerability:

a) CVE standard vulnerability identifier,

b) The software location of the vulnerability,

c) A risk analysis, performed and documented according to the method and criteria meant in 12.3, showing that the risk level associated to the presence of this vulnerability is acceptable.

12.5 The vendor shall likewise document a risk evaluation method for the possible presence of known (types of) software weaknesses in the product. This method shall describe the criteria that the vendor will use to evaluate the level of risk for each (type of) software weakness that may be found in product. The vendor shall make use of the Common Weakness Risk Analysis Framework (CWRAF), ref. [8]. The method shall also establish the level below which a risk is acceptable to the vendor. The evaluation criteria shall be based on risk factors including, but not limited to, the CWSS score of the weakness, the intended use of the product and the environment in which the product would be used.

12.6 In case the vendor is aware of and has accepted the presence of any software weaknesses in the product, the vendor's security risk analysis for the product shall contain a description of each accepted weakness:

a) CWE standard weakness identifier,

b) The software location of the weakness,

c) A risk analysis, performed and documented according to 12.3, showing that the risk level associated with the presence of this weakness is acceptable,

d) External compensating controls to help further reduce the residual risk.

12.7 To verify compliance with 12.1 – 12.3 and 12.5, the security risk analysis for the product shall be evaluated along with the product design documentation. In particular, the following shall be determined:

a) Sufficient coverage during the security risk analysis with regard to the identification of product functionality and data and threats, impact, likelihood and resulting risk.

b) Sufficient adoption of risk controls listed in Sections 7 – 11 by either implementing each control in the product or justifying why the risk level of not implementing a control is acceptable.

c) Sufficient implementation of risk controls per the requirements in Sections 7 – 11.

NOTE: Sufficiency is established via analysis of traceability through the risk management process.

12.8 To verify compliance with 12.4 and 12.6, the vendor's risk evaluation methods for the presence of known vulnerabilities and software weaknesses shall be evaluated.

VULNERABILITIES AND EXPLOITS

13 Known Vulnerability Testing

13.1 The binary code and bytecode under test including those provided by third parties shall contain no known vulnerabilities, unless acceptable per 12.4.

13.2 To verify compliance with 13.1, all binary code and bytecode provided by the vendor according to 4.1(e) shall be evaluated for all known vulnerabilities published in the National Vulnerability Database (NVD) at the time of evaluation.

NOTE: The NVD is currently accessible at https://nvd.nist.gov.

14 Malware Testing

14.1 The binary code and bytecode in the product shall contain no malware.

14.2 To verify compliance with 14.1, all binary code and bytecode provided by the vendor according to 4.1(e) shall be inspected for malware.

15 Malformed Input Testing

15.1 The product shall continue to operate as intended when subject to invalid or unexpected inputs on its external interfaces and shall not display unexpected behavior, such as, but not limited to the following:

a) The product resets or reinitializes its configuration;

b) A process crash or assertion failure occurs without a recovery to its previous state after the test is completed in 2 minutes or less;

c) A process hangs;

d) The testing uses resources of the product and the product does not relinquish these resources after testing;

e) The product software throws an unhandled exception;

f) A storage data corruption occurs;

g) The product loses the connection to the malformed input testing tool;

h) The specified behavior of the product is interrupted and the product does not continue to operate as intended within a timeframe defined by the manufacturer;

i) The product shall not disclose any personally identifiable data or sensitive data over any interface enumerated in 4.1(b).

15.2 To verify compliance with 15.1, malformed input testing shall be performed on the product as described in this section.

15.3 During malformed input testing, the product shall be configured per vendor documentation provided per 6.1 and 6.6.

15.4 Malformed input testing shall take place within a representative environment per vendor documentation provided per 6.1 and 6.6.

15.5 The product shall be inspected to verify the presence of those and only those external interfaces specified in the vendor's documentation per 4.1(a).

15.6 The product shall be subjected to malformed input testing of all file inputs, all remote interfaces, using all protocols supported by the product on these interfaces as listed by the vendor per 4.1(a). Each protocol on an interface shall be subjected to generational malformed input testing when available. Template malformed input testing may be used if a protocol is proprietary and there are no generational malformed input tools available for that protocol.

15.7 If generational malformed input testing is used for a protocol, testing shall evenly apply to all fields of the protocol and the testing shall implement exception handling based on the context of the protocol.

NOTE: Exception handling includes checks on message lengths, message identifiers, integrity checks, correct use of cryptographic protocols and other critical protocol attributes.

15.8 If template malformed input testing is used for a protocol on an interface, the template used shall cover all fields of the protocol.

15.9 If generational malformed input testing is applied for a protocol on an interface, at least 1,000,000 unique and independent test cases or a minimum of 8 hours of test case execution shall be carried out, whichever comes first.

15.10 If template malformed input testing is applied on an interface, at least 5,000,000 unique and independent test cases or a minimum of 8 hours of test case execution shall be carried out, whichever comes first.

16 Structured Penetration Testing

16.1 The product under test shall have no vulnerabilities that can be exploited and/or cause the product to crash, degrade, or perform in an unexpected or random manner without a recovery to its previous state after the test is completed in 2 minutes or less.

16.2 To verify compliance with 16.1, penetration testing shall be carried out on the product in order to try to find and exploit any vulnerabilities in the product.

a) Circumvent the risk controls and security configuration of the product;

b) Attempt to engage the product in a denial of service;

c) Attempt to access and authenticate on the product via unauthorized means;

d) Attempt to exploit vulnerabilities acceptable in the risk analysis;

e) Attempt to elevate privilege on the product.

16.3 Attempts shall be made to identify system, application and service information via scanning of the ports, interfaces and services. Use of that information shall be considered in attempts to circumvent the security measures of the product. Exploit tools and scripts shall be used for discovered information to attempt to access the product, elevate the privilege once accessed or to gain further information about the product.

16.4 A summary shall be provided describing the plan for providing validated software updates and patches as needed throughout the lifecycle of the product to continue to assure its safety and effectiveness.

SOFTWARE WEAKNESSES

17 Software Weakness Analysis

17.1 The product under test shall contain no weaknesses in any of the evaluated source code (or derivative sources), unless acceptable per 12.4.

17.2 To verify compliance with 17.1, the following methods for finding software weaknesses shall be applied as described in Sections 17, 18 and 19: static code analysis, static binary and bytecode analysis.

18 Static Source Code Analysis

18.1 All source code provided by the vendor per 4.1(d) shall be evaluated by means of static code analysis.

18.2 The product shall be evaluated for at least all software weaknesses listed in the latest versions of the sources mentioned in Appendix A, as applicable to the product.

19 Static Binary and Bytecode Analysis

19.1 All binary and/or bytecode provided by the vendor per 4.1(e) shall be evaluated by means of static analysis.

19.2 The product shall be evaluated for at least all software weaknesses listed in the latest versions of the sources mentioned in Appendix A, as applicable to the product.

APPENDIX A

A1 Sources for Software Weaknesses

a) CWE/SANS Top 25 Most Dangerous Software Errors, ref. [9];

b) CWE/SANS on the cusp list, ref. [10];

c) OWASP Top 10, ref. [11].

No Text on This Page

APPENDIX B

B1 Requirements for Secure Mechanisms for Storing Sensitive Data and Personally Identifiable Data

B1.1 Below is a list of acceptable mechanisms:

a) Any approved security function defined in Appendix C;

b) Secure element as some tamper resistant physical media like a universally integrated circuit card or microSD;

c) Trusted platform module per the requirements of the Standard for Information technology – Trusted Platform Module Library, ISO/IEC 11889 ref. [12].

B1.2 All mechanisms shall be validated through a compliance program.

NOTE: An example of compliance is the FIPS 140 Cryptographic Algorithm Validation Program.

No Text on This Page

APPENDIX C

C1 Requirements for Security Functions

C1.1 Below is a list of acceptable security functions:

a) ISO/IEC 9796 (all parts), Information technology – Security techniques – Digital signature scheme giving message recovery, ref. [13];

b) ISO/IEC 9797 (all parts), Information technology – Security techniques – Message Authentication Codes (MACs), ref. [14];

c) ISO/IEC 9798 (all parts), Information technology – Security techniques – Entity authentication, ref. [15];

d) ISO/IEC 10118 (all parts), Information technology – Security techniques – Hash-functions, ref. [16];

e) ISO/IEC 11770 (all parts), Information technology – Security techniques – Key management, ref. [17];

f) ISO/IEC 14888 (all parts), Information technology – Security techniques – Digital signatures with appendix, ref. [18];

g) ISO/IEC 15946 (all parts), Information technology – Security techniques – Cryptographic techniques based on elliptic curves, ref. [19];

h) ISO/IEC 18033 (all parts), Information technology – Security techniques – Encryption algorithms, ref. [20];

i) ISO/IEC 19772 (all parts), Information technology – Security techniques – Authenticated encryption, ref. [21];

j) NIST FIPS 140-2, Annex A: Approved Security Functions, ref. [22];

k) NIST FIPS 140-2, Annex D: Approved Key Establishment Techniques, ref. [23].

A1.2 All security functions shall be validated through a compliance program.

NOTE: An example of compliance is the FIPS 140 Cryptographic Algorithm Validation Program.

No Text on This Page