# Derived Test Requirements for the New Requirements on Software Setup Validation in the VVSG version 1.1

Version 1.1
July 23, 2009

**Preliminary Draft**



**This document and associated files have been prepared by the National Institute of Standards and Technology (NIST) and represent draft test materials for the Election Assistance Commission's version 1.1 of the VVSG. It is a preliminary draft and does not represent a consensus view or recommendation from NIST, nor does it represent any policy positions of NIST.**

## Product Disclaimer

Certain commercial entities, equipment, or material may be identified in the document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that these entities, materials, or equipment are necessarily the best available for the purpose.

**TABLE OF CONTENTS**

**LIST OF DTR**

# 1 Introduction

## 1.1 Background

By authorization of the 2002 Help America Vote Act (HAVA), the Election Assistance Commission (EAC) was given the responsibility for implementing and maintaining the Voluntary Voting System Guidelines (VVSG). As part of the maintenance process for the VVSG, the EAC is updating the VVSG 2005 by modifying and adding some requirements to the guidelines resulting in the VVSG version 1.1. The new and modified requirements are based on the requirements found in the next VVSG developed by the EAC's Technical Guidelines Development Committee (TGDC). However, not all of the requirements found in the next VVSG were included as part of the VVSG version 1.1. The EAC plans to issue the VVSG version 1.1 after receiving and reviewing public comments.

As part of the VVSG update, the EAC asked NIST to develop a set of uniform public test suites for the modified and new requirements, which will be used as part of the EAC's Testing and Certification Program. Test Labs will be able to use these freely available test suites to help determine that modified and new requirements of the VVSG version 1.1 are met by voting systems. Use of the public test suites will produce consistent results and promote transparency of the testing process. The test suites can also assist manufacturers in the development of conforming products by providing precise test specifications. Also, they will help reduce the cost of testing since each test lab would no longer need to develop its own test suites. Finally, a uniform set of public test suites can increase election officials' and voters' confidence that voting systems conform to VVSG version 1.1 requirements covered by the test suites.

## 1.2 Purpose

The purpose of this document is to develop detailed test procedures for the updated and new security requirements found in the VVSG version 1.1. In this document, detailed test procedures derived from a requirement found in the VVSG version 1.1 are contained in structure known as a derived test requirement (DTR). (See Section 1.5.1 Derived Test Requirement Structure for details). This document contains the set of derived test requirements (DTRs) for the requirements found in the VVSG version 1.1, Volume 1, Section 2.4.4 Electronic Reports. By providing detailed derived test requirements, the following objectives are achieved:

1. In-depth guidance to test laboratories to ensure high quality testing
2. Repeatability from tester to tester as well as test laboratory to test laboratory
3. Predictability of the effort involved for a testing campaign
4. Cost savings by not having to analyze and develop tests for different implementations of a voting system

## 1.3 Scope

The scope of this document is limited to functional testing of the updated and new security requirements found in the VVSG version 1.1. Testing requirements in VVSG version 1.1 other than updated and new security requirements are outside the scope of this document. Specifically, the derived test requirements (DTRs) found in this document only cover the requirements found in the VVSG version 1.1, Volume 1, Section 2.4.4 Electronic Reports.

## 1.4 Approach

In developing the set of derived test requirements (DTRs) the following approach was taken:

1. If at all possible, the test laboratory shall test compliance with a VVSG requirement by stimulus → response testing[1] on the voting system. The exceptions to this shall be rare and shall be justified only on the basis of extremely prohibitive cost.
2. The stimulus → response testing shall include nominal, boundary and outlier values as implied by the VVSG requirement and the voting system's interface(s) that implement and enforce the requirement.
3. When stimulus → response testing is not possible given the design of the voting system, the test laboratory shall examine the applicable source code.
4. When performing review of the manufacturer provided documentation, the test laboratory shall focus on gaining an understanding of the voting system and how it implements security. Priority shall be given to identification of potential security concerns based on the review and analysis of manufacturer documents with next priority to substantive inconsistencies. In addition, the test laboratory shall ensure that there is sufficient clarity to the documentation so that the security controls can be appropriately configured.

## 1.5 Derived Test Requirement Structure

A derived test requirement (DTR) is a structured used to contain detailed test procedures associated with a specific requirement. This section describes the components, nomenclature, and notation used in this document to describe the structure of a derived test requirement (DTR).

A derived test requirement consists of the following components:

1. A requirement is labeled with the literal "RE, " followed by a number based on the section of the VVSG version 1.1 containing the requirement and a title for the requirement to provide traceability back to the VVSG version 1.1. When a requirement is tested by another derived test requirement (DTR), that requirement's derived test requirement (DTR) will contain a reference to the appropriate derived test requirement (DTR).

2. A requirement may have one or more tester activities associated with it. Test activities are the detailed test procedures used to test the voting system for conformance to the VVSG version 1.1 security requirements. When no tester activity is found in a derived test requirement (DTR), it means the requirement was tested under the test procedures of another (DTR) that is specifically referenced in "Analysis" text. The tester activities are labeled with the literal "TE", followed by the requirement label without "RE", followed by period ("."), followed by sequential numbers staring with 1. For example, test activities for requirement RE 7.9.1-A are numbered RE 7.9.1-A.1, RE 7.9.1-A.2, and so on. Each tester activity title is refined based on the associated VVSG version 1.1 requirement title.

3. The label "Analysis:" precedes text that is used to provide additional information related to requirements and tester activities. In general, analysis text follows the associated requirement and tester activities being discussed. For example, analysis text following a requirement may cross-reference the test activities of another requirement that verifies the requirement or provide context of the test activities of the requirement.

## 1.6 Electronic File Features for Word Versions of the Document

An electronic version of this document was prepared using Microsoft Word and the Word Style feature. The Word Style feature provides the ability to separate text based on the Style associated with the text. The following Styles were used in this document to allow material to be subsetted in or out:

1. "reheader" Style is used to list the requirement title.
2. "teheader" Style is used to list the test procedure title.

[1] Stimulus → response testing refers to a testing method where an IT system is stimulated by providing some input and the IT system's response/output is observed and analyzed. (See Definitions section)

3. “Test Procedure” Style is used to list the test procedures test laboratories must carry out in order to test the voting system for compliance with VVSG Version 1.1 requirements.
4. “Normal” Style is used for the requirement text and text associated with analysis and rationale. “

## 1.7 General Testing Assumptions

The tester shall use each DTR to test the voting system under test and when appropriate develop more detailed test procedures and test cases based on implementation dependant characteristics such as specific configuration requirements, specific user account names, specific file names, etc.

The tester shall document test procedures and test cases.

After conducting the tests, the tester shall document the test results with sufficient detail to demonstrate that the test succeeded or failed. In the case of failure, the documentation shall be detailed enough to provide the nature of failure.

The tester may execute the DTRs in any order as long as the precedence requirements specified in individual DTR are met.

The tester shall note the start and end time in date, hours, and minutes when each DTR is executed to help in several ways including but not limited to: reconciling the event log, reconstructing DTR execution sequence, and determining the state of the voting system under test at any given time.

## 1.8 Asterisk Notation (*****)

A series of red asterisks (*****) next to test activities (i.e. all or part of a TE) indicates that the activity is conditional and may not need to be executed based on the implementation under test. In general, the test activities have a condition statement similar to: “If the voting device is an EMS…” or “If the voting system provides role-based authentication….”

## 2 Definitions

**Stimulus → Response Testing:** A test method where the IT system is stimulated by providing some input and the IT system's response (output) is observed and analyzed. Also see test method for other form of testing.

**Test Case:** A fully defined set of input and expected results for a test. A test case is the most detailed and lowest level of test documentation material.

**Test Method:** Description of one or more tests, procedures by which tests are derived, or a combination of these.

**Test Pre-Requisite:** System configuration prior to executing a test case or set of test cases. For example, prior to testing that identification and authentication succeeds and fails under appropriate conditions, user accounts with specific user ID and passwords will need to be set up.

**Test Procedures:** Procedures used to execute a collection of test cases. For example, test procedures typically will consist of executing a set of steps to set test pre-requisite and then steps for each test case as identified with the test case,

**Test Results:** Set of results for each of the test cases.

# 3 Software Setup Validation Derived Test Requirements

The following requirements support the security of voting systems by providing methods to verify that only authorized software is present on voting systems. It includes requirements for two software verification techniques. One method verifies digital signatures on software prior to installation on pieces of voting system equipment. This is a useful mechanism that helps prevent accidental or malicious software from being installed and could be employed by any voting system to protect against unauthorized software. The second method provides an external interface to voting system software. A separate piece of equipment could use this interface to verify the software on the voting system. However, this method merely provides a mechanism for detecting unauthorized software and, by itself, does not help prevent the installation of accidental or malicious software.

The term "Software Update Package" as stated in VVSG 1.1 has the following meaning: It consists of software that updates the SUT software and associated metadata. Examples of metadata are: Software Update Package Name, List of Software, Version Number for the Package, Version Numbers for the Software Components in the Package, Digital Signature, etc.

The terms "Software Update Package" is specifically intended to mean a digitally signed collection of software that can be verified prior to installation. Thus, the term "Software Update Package" is not appropriate when the verification occurs using external interface since that verification method need not entail digital signature.

Furthermore, a "Software Update Package" could be either the entire set of voting application software required by the SUT or could be one or more software components that need updating.

Based on the above, the following terms are used in the rest of the document:

1. The term "Entire SUT Software" is intended to convey the whole voting application software required by the SUT. This term is appropriate for both the verification prior to installation and verification using external interface paradigms. Furthermore, for the SUT using verification prior to installation, the term "Entire SUT Software" is synonymous with the term "software update package with entire SUT software in it"

2. The term "software update package with entire SUT software in it" is intended to convey the whole voting application software required by the SUT that performs verification prior to installation.

3. The term "software update package" is intended to convey one or more voting application software components for the SUT that performs verification prior to installation. The "software update package" contains metadata such as digital signature and version number required for verification prior to installation.

### RE 7.4.6-A (2005) Setup Validation Method:

Setup validation methods shall verify that no unauthorized software is present on the voting equipment.

### TE 7.4.6-A.1 (2005) Setup Validation Method – valid software:

The tester shall acquire the "Entire SUT Software" for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall install the software on the SUT using the "Entire SUT Software"

The tester shall execute the setup validation on the SUT.

The tester shall verify that the setup validation on the SUT indicates success, i.e., absence of unauthorized software.

The tester shall execute the vendor defined procedure (see RE 7.4.6-B (2005) Software Identification Method) to obtain the list of software files on the SUT.

The tester shall verify that the software list matches the list provided by the vendor as described on Section 2.6.3, Volume II of VVSG 1.1.

If there is no means to examine the SUT to determine software present on the SUT, the tester shall bypass the following steps. Otherwise, the tester shall carry out the following steps:

1. The tester shall examine the SUT storage for software present.
2. An example is to traverse the hierarchical directory structure of the SUT[2] and to look for file types or extensions such as .exe, .bin, .com, .dll, etc.
3. The tester shall verify that the software files using this approach matches the software list provided by the vendor.

The tester shall verify that the entry for each software item displayed on the SUT terminal includes the following information:

1. Name of software
2. Version number
3. If date of installation is present, it is the date TE 7.4.6-A.1 (2005) Setup Validation Method – valid software is executed.

The tester shall attempt to install one unauthorized software application to the SUT that is not part of the SUT software per RE 7.4.6-B (2005) Software Identification Method.

If the attempt fails, this TE passes.

If the attempt to install the unauthorized software application on the SUT succeeds, the tester shall perform the following activities:

1. The tester shall execute the setup validation on the SUT.
2. The tester shall verify that the setup validation on the SUT indicates failure i.e., presence of unauthorized software.
3. The tester shall execute the vendor defined procedure (see RE 7.4.6-B) to obtain the list of software files on the SUT.
4. The tester shall verify that the list includes an entry for the unauthorized software.

### TE 7.4.6-A.2 (2005) Setup Validation Method – invalid software:

The tester shall acquire the "Entire SUT Software" for the SUT from the appropriate source (e.g., vendor, local election board representative).

On a machine other than the SUT, the tester shall attempt to add to "Entire SUT Software" one unauthorized software application that is not part of the SUT software per RE 7.4.6-B (2005) Software Identification Method.

If the attempt fails, this TE passes.

If the attempt succeeds, the tester shall attempt to install the software on the SUT using the "Entire SUT Software"[3].

[2] The tester shall perform this activity as an administrator so that the tester has privilege to examine all the files. Furthermore, the tester shall use the appropriate configurations and settings to ensure hidden files are also examined.

If the attempt fails, this TE passes.

If the attempt to install the software on the SUT succeeds, the tester shall perform the following activities:

1. The tester shall execute the setup validation on the SUT.
2. The tester shall verify that the setup validation on the SUT indicates failure i.e., presence of unauthorized software.
3. The tester shall execute the vendor defined procedure (see RE 7.4.6-B) to obtain the list of software files on the SUT.
4. The tester shall verify that the list includes an entry for the unauthorized software.

Analysis: Installation of unauthorized software on a SUT that has already undergone setup validation is tested under TE 7.4.6-A.1 (2005) Setup Validation Method – valid software.

---

### RE 7.4.6-B (2005) Software Identification Method:

The vendor shall provide a method to comprehensively list all software files that are installed on voting systems.

Analysis: This requirement is tested under RE 7.4.6-A (2005) Setup Validation Method.

---

### RE 7.4.6-B.1 (2005) Software Version Number:

This method shall list version names and numbers for all application software on the voting system.

Analysis: This requirement is tested under RE 7.4.6-A (2005) Setup Validation Method.

---

### RE 7.4.6-B.2 (2005) Software installation date:

This method should list the date of installation for all application software on the voting system.

Analysis: This requirement is tested under RE 7.4.6-A (2005) Setup Validation Method.

---

### RE 7.4.6-C (2005) Software Verification Method:

Setup validation methods shall include a software verification method that ensures that the voting system software has not been modified illegitimately.

### TE 7.4.6-C.1 (2005) Software Verification Method – Documentation:

The tester shall examine the vendor supplied setup validation procedures.

The tester shall verify that the method includes one or more software verification procedures.

The tester shall verify that one of the software verification procedures consists of:

1. Verification of software prior to installation on the SUT; or
2. Verification of installed software on the SUT using an external interface to the SUT.

---

### RE 7.4.6-C.1 (2005) Software Verification Supporting Software and Hardware:

---

[3] The “Entire SUT Software” at this point is the vendor supplied “Entire SUT Software” plus one unauthorized software application.

The voting systems shall include any supporting software and hardware necessary to conduct the software verification method.

### TE 7.4.6-C.1.1 (2005) Software Verification Supporting Software and Hardware:

The tester shall examine the vendor documentation for the software verification method to identify any software and hardware prerequisites.

The tester shall examine the software list from TE 7.4.6-A.1 (2005) Setup Validation Method – valid software to verify that the voting system SUT includes all software prerequisites.

The tester shall examine the hardware configuration of the SUT to verify that it includes all hardware prerequisites. An example of hardware prerequisite is the device connected to the external interface for the purpose of software verification.

---

### RE 7.4.6-C.2 (2005) Software Verification Process:

The vendor shall document the process used to conduct the software verification method.

This requirement is tested under RE 7.4.6-C (2005) Software Verification Method.

---

### RE 7.4.6-C.3 (2005) Software Verification Integrity:

The software verification method shall not modify the voting system software on the voting system.

### TE 7.4.6-C.3.1 (2005) Software Verification Integrity:

The tester shall acquire the "Entire SUT Software" for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall install the software on the SUT using the "Entire SUT Software".

The tester shall execute the setup validation on the SUT.

The tester shall verify that the setup validation on the SUT passes software verification.

The tester shall list the SUT software as identified per RE 7.4.6-B (2005) Software Identification Method.

If there is no means to copy the SUT software to another media, the tester shall bypass the following steps. Otherwise, the tester shall carry out the following steps:

1. The tester shall copy all the software on the list from the SUT to removable media.
2. The tester shall take the removable media to another computer and compare the software files against the files in the "Entire SUT Software".
3. The tester shall verify that all file comparisons succeed.

Note: Alternative method of testing RE 7.4.6-C.3 (2005) Software Verification Integrity is two successive software verifications. This alternative method is executed under the following TEs:

1. TE 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content; and/or
2. TE 7.4.6-F.1 (2005) Verification After Installation, Software Verification External Interface

---

### RE 7.4.6-D.1 (2005) Software Verification Method Alternatives:

Voting systems shall include a software verification method that either verifies software prior to installation or a method that verifies software using an external interface.

This requirement is tested by TE 7.4.6-C.1 (2005) Software Verification Method – Documentation.

---

**RE 7.4.6-D.2 (2005) Number of Software Verification Method:**
Voting systems may include both software verification methods.

Analysis: This requirement is tested under TE 7.4.6-C.1 (2005) Software Verification Method – Documentation.

---

**RE 7.4.6-D.3 (2005) Vendor Specified Software Verification Method:**
Voting systems may also provide ancillary setup validation methods, including methods for verifying or identifying installed software, other than those described in this section. There are no specific requirements for ancillary setup validation methods. However, any method intended to serve as the voting system software verification method shall meet the requirements outlined in this section.

Analysis: The requirement is implicitly tested by testing all other Setup Validation and Software Verification requirements.

---

**RE 7.4.6-E.1 (2005) Verification Prior to Installation, Singular Software Installation Process:**
The voting system shall contain no more than one method for installing, updating, or removing software on a system.

*******TE 7.4.6-E.1.1 (2005) Verification Prior to Installation, Singular Software Installation Process:**
TE 7.4.6-E.1.1 (2005) Verification Prior to Installation, Singular Software Installation Process is applicable if the SUT performs software verification prior to installation.

The tester shall examine the vendor documentation and verify that there is only one method for installing software on the SUT.

The tester shall examine the vendor documentation and verify that there is no more than one method for updating software on the SUT.

The tester shall examine the vendor documentation and verify that there is no more than one method for removing software from the SUT.

The tester shall acquire the "software update package with entire SUT software in it" for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall install the software on the SUT using the "software update package with entire SUT software in it".

The tester shall examine the event log and verify that it contains the following information for an event log entry:

1. Success of software installation;
2. Date and time TE 7.4.6-E.1.1 (2005) Verification Prior to Installation, Singular Software Installation Process was executed;
3. Software update package name and version number or list of software in the software update package and their version numbers;
4. Type of action as new application/file; and

5. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the "software update package with entire SUT software in it".

Analysis:
Update and removal are tested under TE 7.4.6-E.1.3.1 (2005) Verification Prior to Installation, Software Installation Prevention While Polls Open.

---

**RE 7.4.6-E.1.1 (2005) Verification Prior to Installation, Unauthorized Software Installation Prevention:**
Voting system equipment shall prevent processes from installing software except for the one specific software installation process identified by the vendor.

Analysis: This requirement is tested by TE 7.4.6-E.1.1 (2005) Verification Prior to Installation, Singular Software Installation Process.

---

**RE 7.4.6-E.1.2 (2005) Verification Prior to Installation, Software Installation Documentation:**
The voting system vendor shall document the procedures for installing, updating, and removing voting system software, configuration files, and data files.

Analysis: This requirement is tested under TE 7.4.6-E.1.1 (2005) Verification Prior to Installation, Singular Software Installation Process.

---

**RE 7.4.6-E.1.3 (2005) Verification Prior to Installation, Software Installation Prevention While Polls Open:**
Voting system equipment shall prevent processes from installing, updating or removing software while the polls are open.

*******TE 7.4.6-E.1.3.1 (2005) Verification Prior to Installation, Software Installation Prevention While Polls Open:**
TE 7.4.6-E.1.3.1 (2005) Verification Prior to Installation, Software Installation Prevention While Polls Open is applicable if the SUT performs software verification prior to installation.

The tester shall acquire the "software update package with entire SUT software in it" for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall install the software on the SUT using the "software update package with entire SUT software in it".

The tester shall execute the vendor defined procedures to open the polls.

The tester shall attempt to install unauthorized software on the SUT. The tester shall verify that the attempt fails.

The tester shall attempt to install a voting system application software on the SUT. The tester shall verify that the attempt fails.

The tester shall examine the event log and verify that it contains the following information for an event log entry:
1. Failure of software installation;
2. Cause of failure indicates SUT state inappropriate for the action;

3. Date and time TE 7.4.6-E.1.3.1 (2005) Verification Prior to Installation, Software Installation Prevention While Polls Open was executed;
4. Voting system application software name and version number;
5. Type of action as new application/file; and
6. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the voting system application software.

If the SUT has a method for software update, the tester shall attempt to execute software update using a "software update package". The tester shall verify that the attempt fails. The tester shall examine the event log and verify that it contains the following information for an event log entry:

1. Failure of software installation;
2. Cause of failure indicates SUT state inappropriate for the action;
3. Date and time TE 7.4.6-E.1.3.1 (2005) Verification Prior to Installation, Software Installation Prevention While Polls Open was executed;
4. Software update package name and version number or list of software in the software update package and their version numbers; and
5. Type of action as update application/file; and
6. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the software update package name and version number.

If the SUT has a method to remove software, the tester shall attempt to remove a voting system application. The tester shall verify that the attempt fails. The tester shall examine the event log and verify that it contains the following information for an event log entry:

1. Failure of software installation;
2. Cause of failure indicates SUT state inappropriate for the action;
3. Date and time TE 7.4.6-E.1.3.1 (2005) Verification Prior to Installation, Software Installation Prevention While Polls Open was executed;
4. Voting system application software name and version number;
5. Type of action as removal of application/file; and
6. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the voting system application software.

---

### RE 7.4.6-E.1.4 (2005) Verification Prior to Installation, Improperly Installed Software Execution Prevention:

Voting system equipment shall prevent the execution of software not installed using the specified software installation process.

### *****TE 7.4.6-E.1.4.1 (2005) Verification Prior to Installation, Improperly Installed Software Execution Prevention:

TE 7.4.6-E.1.4.1 (2005) Verification Prior to Installation, Improperly Installed Software Execution Prevention is applicable if the SUT performs software verification prior to installation.

The tester shall acquire the "software update package with entire SUT software in it" for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall install the software on the SUT using the "software update package with entire SUT software in it".

On a machine other than the SUT, the tester shall do the following:

1. The tester shall take one of the software executables from "software update package with entire SUT software in it" and modify one byte using a hex editor.
2. The tester shall rename the file to a name other than the name assigned.
3. The tester attempt to copy the executable file on the SUT. The attempt should fail.

4. If the attempt succeeds, the tester shall attempt to execute the file. The tester shall verify that the execution fails.

---

**RE 7.4.6-E.2.1 (2005) Verification Prior to Installation, Software Installation Restriction:**
The voting system shall only allow authenticated administrators to install software on voting equipment.

*****TE 7.4.6-E.2.1.1 (2005) Verification Prior to Installation, Software Installation Restriction:**
TE 7.4.6-E.2.1.1 (2005) Verification Prior to Installation, Software Installation Restriction is applicable if the SUT performs software verification prior to installation.

The tester shall acquire the "software update package with entire SUT software in it" for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall attempt to install the software on the SUT using the "software update package with entire SUT software in it" without authentication, i.e., as an unauthenticated user. The tester shall verify that at least one of the following is true:

1. The tester can not use the SUT as unauthenticated user, i.e., performance of any function on the SUT requires user authentication;
2. The tester can not install software as unauthenticated user, i.e., while some functions may be available to unauthenticated user, the software installation is not one of those functions; or
3. The software installation attempt fails and the event log contains an event log entry with the following information:
    a) Failure of software installation;
    b) Cause of failure indicates unauthenticated user or privilege violation;
    c) Date and time TE 7.4.6-E.2.1.1 (2005) Verification Prior to Installation, Software Installation Restriction was executed;
    d) Software update package name and version number or list of software in the software update package and their version numbers;
    e) Type of action as new software update; and
    f) A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the "software update package with entire SUT software in it" name and version number.

If the SUT provides for authenticated roles or users other than administrator, the tester shall perform the following activities for each of those roles or users. Thus, if there are n roles or users other than administrator, the following steps shall be carried out n times. The tester shall ensure that one of these roles is a voter role.

1. The tester shall authenticate to the SUT in that role or user.
2. The tester shall attempt to install the software on the SUT using the "software update package with entire SUT software in it".
3. The tester shall verify that the installation attempt is either not feasible (e.g., there is no interface available) or fails. In the case of the failure an event log contains an event log entry with the following information:
    a) Failure of software installation;
    b) Cause of failure indicates unauthenticated user or privilege violation;
    c) Date and time TE 7.4.6-E.2.1.1 (2005) Verification Prior to Installation, Software Installation Restriction was executed;
    d) Software update package name and version number or list of software in the software update package and their version numbers;
    e) Type of action as new software update; and

f) A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the "software update package with entire SUT software in it" name and version number.

The tester shall authenticate to the SUT as an administrator. The tester shall attempt to install the software on the SUT using the "software update package with entire SUT software in it". The tester shall verify that the installation attempt succeeds.

The tester shall verify that during the installation or after the installation is complete, the SUT provides the following information on the terminal being used for providing installation command:

1. Lists all the software applications being installed.
2. This list matches the vendor provided list of software
3. For each application, the action of new application is listed depending if the SUT did not have voting system application installed previously. Otherwise, overwriting of existing file is listed.
4. For each application, the version number of new software shall be listed.

The tester shall verify that the event log contains an entry with the following information:

1. Success of software installation;
2. Date and time TE 7.4.6-E.2.1.1 (2005) Verification Prior to Installation, Software Installation Restriction was executed;
3. Software update package name and version number or list of software in the software update package and their version numbers;
4. Type of action as new software update; and
5. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the "software update package with entire SUT software in it" name and version number.

---

**RE 7.4.6-E.2.2 (2005) Verification Prior to Installation, Software Installation Change Reporting:**

The voting system shall present the administrator with a description of the software change being performed, including:

1. A list of all applications and/or file names being updated.
2. The type of action performed on each application and/or file (e.g., new application/file, deletion or overwriting of existing file)

*******TE 7.4.6-E.2.2.1 (2005) Verification Prior to Installation, Software Installation Change Reporting:**

TE 7.4.6-E.2.2.1 (2005) Verification Prior to Installation, Software Installation Change Reporting is applicable if the SUT performs software verification prior to installation.

TE 7.4.6-E.2.2.1 (2005) Verification Prior to Installation, Software Installation Change Reporting shall be conducted immediately after TE 7.4.6-E.2.1.1 (2005) Verification Prior to Installation, Software Installation Restriction.

If the SUT provides a capability to update software, the tester shall carry out the following steps:

1. The tester shall obtain a "software update package".
2. The tester shall verify that the software update package has an identifier associated with it.
3. The tester shall attempt to install the software on the SUT using the software update package.
4. The tester shall verify that the SUT provides the following information on the terminal being used for updating the software:
   a) Names of the software being updated;
   b) The names match the names identified in the software update package;

c) Action as new application/file or overwriting of existing file;
d) Version numbers of old software being updated;
e) Version numbers of new software; and
f) Version numbers of new software match the version numbers identified in the software update package

5. The tester shall execute the vendor defined procedure (see RE 7.4.6-B (2005) Software Identification Method) to obtain the list of software files on the SUT.
6. The tester shall verify that the list matches the list provided by the vendor.
7. The tester shall verify that the entry for each software includes the following information:
   a) Name of software
   b) Version number
   c) The tester shall verify that the version numbers for updated software match the newly updated software version numbers
8. The tester shall examine the event log and verify that it contains the following information for an event log entry:
   a) Success of software installation;
   b) Date and time TE 7.4.6-E.2.2.1 (2005) Verification Prior to Installation, Software Installation Change Reporting was executed;
   c) Software update package name and version number or list of software in the software update package and their version numbers;
   d) Type of action as new application/file or overwriting of existing file or software installation; and
   e) A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the software update package name and version number.

If the SUT provides a capability to remove software, the tester shall carry out the following steps:

1. The tester shall attempt to remove one of the software files on the SUT.
2. The tester shall verify that the SUT provides the following information on the terminal being used for removing the software:
   a) Name of the software being removed
   b) Action as deletion of software
   c) Version number of software being removed
3. The tester shall execute the vendor defined procedure (see RE 7.4.6-B (2005) Software Identification Method) to obtain the list of software files on the SUT.
4. The tester shall verify that the list matches the list provided by the vendor except for the removed software.
5. The tester shall verify that the entry for each piece of software remaining on the SUT includes the following information:
   a) Name of software
   b) Version number
6. The tester shall examine the event log and verify that it contains the following information for an event log entry:
   a) Success of software installation;
   b) Date and time TE 7.4.6-E.2.2.1 (2005) Verification Prior to Installation, Software Installation Change Reporting was executed;
   c) Software name and version removed;
   d) Type of action as removal of application/file; and
   e) A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the voting system application software removed.

---

**RE 7.4.6-E.3 (2005) Verification Prior to Installation, Installed Software Version Number:**

Voting system equipment shall store the current version identification of all software installed on the voting system equipment.

Analysis: This requirement is implicitly tested by ensuring that version number is displayed under TE 7.4.6-E.3.2.1 (2005) Verification Prior to Installation, Software Version Number Display.

---

**RE 7.4.6-E.3.1 (2005) Verification Prior to Installation, Software Version Number Reporting:**
The current version identification shall be included as part of reports created by the voting system equipment.

Analysis: This requirement is partially tested by the following TE:

1. TE 7.4.6-A.1 (2005) Setup Validation Method – valid software
2. TE 7.4.6-E.2.1.1 (2005) Verification Prior to Installation, Software Installation Restriction
3. TE 7.4.6-E.2.2.1 (2005) Verification Prior to Installation, Software Installation Change Reporting
4. TE 7.4.6-E.3.2.1 (2005) Verification Prior to Installation, Software Version Number Display

*****TE 7.4.6-E.3.1.1 (2005) Verification Prior to Installation, Software Version Number Reporting:**
TE 7.4.6-E.3.1.1 (2005) Verification Prior to Installation, Software Version Number Reporting is applicable if the SUT performs software verification prior to installation.

The tester shall examine the following printed reports and verify that each one contains the voting software version number:

1. Collection of Ballot Images Report printed under TE 2.4.4.2-D.1 (2005) DRE, collection of ballot images report
2. Collection of Ballot Images Report printed under TE 2.4.4.2-C.1 (2005) Tabulator, collection of ballot images report
3. Event log printed under TE 2.4.4.2-B.3 (2005) Tabulator, summary count report handling – Event Log
4. Event log printed under TE 2.4.4.2-E.3 (2005) Tabulator, collection of ballot images handling – Event Log
5. Tabulator Summary Count Report printed under TE 2.4.4.2-A.1 (2005) Tabulator, summary count report – Normal
6. EMS Tabulator Summary Count Report printed under TE 2.4.4.3-A.1 (2005) EMS tabulator summary count report
7. Precinct Summary Count Report printing under TE 2.4.4.3-C.1 (2005) EMS, precinct summary count reports.

---

**RE 7.4.6-E.3.2 (2005) Verification Prior to Installation, Software Version Number Display:**
The current version identification shall be displayed as part of the voting system equipment start up process.

*****TE 7.4.6-E.3.2.1 (2005) Verification Prior to Installation, Software Version Number Display:**
TE 7.4.6-E.3.2.1 (2005) Verification Prior to Installation, Software Version Number Display is applicable if the SUT performs software verification prior to installation.

The tester shall acquire the “software update package with entire SUT software in it” for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall authenticate to the SUT as an administrator.

The tester shall install the software on the SUT using the “software update package with entire SUT software in it”.

The tester shall shut down the SUT.

The tester shall start up the SUT.

The tester shall verify that the SUT terminal displays the voting system software version number.

The tester shall verify that the version number matches the version number for the voting system software identified by the SUT vendor.

---

**RE 7.4.6-E.4.1 (2005) Verification Prior to Installation, Software Update Basis:**
The process for installing, updating and removing software shall make software changes based on information contained in software update packages.

Analysis:
For installation, this requirement is tested by TE 7.4.6-A.1 (2005) Setup Validation Method – valid software.

For updating, this requirement is tested by TE 7.4.6-E.2.2.1 (2005) Verification Prior to Installation, Software Installation Change Reporting.

For removing, this requirement is tested by TE 7.4.6-E.2.2.1 (2005) Verification Prior to Installation, Software Installation Change Reporting.

---

**RE 7.4.6-E.4.2 (2005) Verification Prior to Installation, Software Update Content:**
Software update packages shall minimally contain the following information:

1. A unique identifier for the software update package.
2. Names of the applications or files modified during the update process.
3. Version numbers of the applications or files modified during the update process.
4. Any software prerequisites or dependencies for the software involved in the update.
5. A description of the type of action performed on each application and/or file (e.g., new application/file, deletion or overwriting of existing file).
6. The binary data of any new or updated files involved in the update process.

Analysis:
Items 1, 2, 3, 5, and 6 are also tested as part of TE 7.4.6-E.2.2.1 (2005) Verification Prior to Installation, Software Installation Change Reporting.

Item 4 is tested under the TE below.

*******TE 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content:**
TE 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content is applicable if the SUT performs software verification prior to installation.

The tester shall acquire the “software update package with entire SUT software in it” for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall authenticate to the SUT as an administrator. The tester shall attempt to install the software on the SUT using the “software update package with entire SUT software in it”. The tester shall verify that the installation attempt succeeds.

If the SUT provides a capability to update software, the tester shall carry out the following steps:

1. The tester shall obtain a software update package with dependencies.
2. The tester shall verify that the software update package has an identifier associated with it.
3. The tester shall remove the dependency/prerequisite software from the SUT.
4. The tester shall attempt to install the software on the SUT using the software update package and by providing the correct identifier.
5. The tester shall verify that the update fails.
6. The tester shall reinstate the dependency/prerequisite software on the SUT.
7. The tester shall make sure that any vendor specified authorizations are absent.
8. The tester shall attempt to install the software on the SUT using the software update package and by providing the correct identifier.
9. The tester shall verify that the update fails.
10. The tester shall instate vendor required authorizations.
11. The tester shall attempt to install the software on the SUT using the software update package and by providing the correct identifier.
12. The tester shall verify that the update succeeds.
13. The tester shall verify that the SUT provides the following information on the terminal being used for updating the software:
    a) Names of the software being updated
    b) The names match the names identified in the software update package.
    c) Action as new software or overwriting of software
    d) Version numbers of old software being updated
    e) Version numbers of new software
    f) Version numbers of new software match the version numbers identified in the software update package
14. The tester shall execute the vendor defined procedure (see RE 7.4.6-B (2005) Software Identification Method) to obtain the list of software files on the SUT.
15. The tester shall verify that the list matches the list provided by the vendor for the new software update package.
16. The tester shall verify the following for each piece of software in the list:
    a) Version number matches the version number for that piece of software with that in the list provided by the vendor for the new software update package.

The tester shall examine the event log and verify that it contains the following information for an event log entry:

1. Failure of software installation;
2. Cause of failure indicates dependencies not being met;
3. Date and time TE 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content was executed;
4. Software update package name and version number or list of software in the software update package and their version numbers;
5. Type of action as installation of application/file; and
6. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the software update package name and version number.

The tester shall examine the event log and verify that it contains the following information for an event log entry:

1. Failure of software installation;
2. Cause of failure indicates authorization failure;
3. Date and time TE 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content was executed;
4. software update package name and version number or list of software in the software update package and their version numbers;
5. Type of action as installation of application/file; and

6. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the software update package name and version number.

The tester shall examine the event log and verify that it contains the following information for an event log entry:

1. Success of software installation;
2. Date and time TE 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content was executed;
3. Software update package name and version number or list of software in the software update package and their version numbers;
4. Type of action as installation of application/file; and
5. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the software update package name and version number.

The tester shall execute the software verification process. The tester shall verify that the software verification succeeds. (Note: This step is conducted to gain additional assurance that the software verification does not modify the software).

---

**RE 7.4.6-E.5.1 (2005) Verification Prior to Installation, Software Update Format:**
The software update package shall be formatted in a non-restrictive, publicly-available format.

*******TE 7.4.6-E.5.1.1 (2005) Verification Prior to Installation, Software Update Format:**
TE 7.4.6-E.5.1.1 (2005) Verification Prior to Installation, Software Update Format is applicable if the SUT performs software verification prior to installation.

The tester shall verify from the vendor that the software update packages do not have any restrictions on the legitimate purchaser.

The tester shall verify from the vendor that the software update package format documentation is available to public.

---

**RE 7.4.6-E.5.2 (2005) Verification Prior to Installation, Software Update Documentation:**
Vendors shall provide a specification describing how they have implemented the format with respect to the manufacturer's specific voting devices and data, including such items as descriptions of elements, attributes, constraints, extensions, syntax and semantics of the format, and definitions for data fields and schemas.

*******TE 7.4.6-E.5.2.1 (2005) Verification Prior to Installation, Software Update Documentation:**
TE 7.4.6-E.5.2.1 (2005) Verification Prior to Installation, Software Update Documentation is applicable if the SUT performs software verification prior to installation.

The tester shall examine the vendor documentation and verify that it contains the format and schema definitions for the software update packages.

The tester shall examine the vendor documentation and verify that it contains the following information for software update package format:

1. Detailed format specification sufficient for some one to write software to process the software update packages and install them.
2. Definition, description, and semantics of each field in the software update package
3. Syntax and format for each field

4. Description of constraints on values in a field. Note that the constraints could depend on other fields and/or could be static or dynamic.
5. Description if the field is always present or is optional and if optional under what circumstances (Note extensions are viewed as an example of optional fields)

---

**RE 7.4.6-E.6.1 (2005) Verification Prior to Installation, Software Update Integrity:**
Software update packages shall be digitally signed by the National Software Reference Library (NSRL), the voting device owner, or designated notary repositories.

*******TE 7.4.6-E.6.1.1 (2005) Verification Prior to Installation, Software Update Integrity – Documentation:**
TE 7.4.6-E.6.1.1 (2005) Verification Prior to Installation, Software Update Integrity – Documentation is applicable if the SUT performs software verification prior to installation.

The tester shall examine the vendor documentation for software verification. (see RE 7.4.6-C.2 (2005) Software Verification Process). The tester shall verify that the process includes digital signature verification.

.

*******TE 7.4.6-E.6.1.2 (2005) Verification Prior to Installation, Software Update Integrity – Signature Test:**
TE 7.4.6-E.6.1.2 (2005) Verification Prior to Installation, Software Update Integrity – Signature Test is applicable if the SUT performs software verification prior to installation.

The tester shall acquire the "software update package with entire SUT software in it" for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall note the location from which the tester obtains the cryptographic reference information. The tester shall verify that this is the test lab itself, NSRL, or a notarized repository service.

The tester shall use a HEX editor to modify one of the system software files.

The tester shall authenticate to the SUT as an administrator.

The tester shall attempt to install the software on the SUT using the "software update package with entire SUT software in it".

The tester shall verify that the attempt fails.

The tester shall examine the event log and verify that it contains the following information for an event log entry:

1. Failure of software update;
2. Cause of failure indicates digital signature verification failure;
3. Date and time TE 7.4.6-E.6.1.2 (2005) Verification Prior to Installation, Software Update Integrity – Signature Test was executed;
4. Software update package name and version number or list of software in the software update package and their version numbers;
5. Type of action as software update; and
6. A cryptographic hash of 28-64 bytes that does not match the cryptographic hash for the "software update package with entire SUT software in it" name and version number.

The tester shall acquire the "software update package with entire SUT software in it" for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall attempt to install the software on the SUT using the “software update package with entire SUT software in it”.

The tester shall verify that the attempt succeeds.

The tester shall examine the event log and verify that it contains the following information for an event log entry:

1. Success of software update;
2. Date and time TE 7.4.6-E.6.1.2 (2005) Verification Prior to Installation, Software Update Integrity – Signature Test was executed;
3. Software update package name and version number or list of software in the software update package and their version numbers;
4. Type of action as software update; and
5. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the “software update package with entire SUT software in it” name and version number.

---

**RE 7.4.6-E.6.2 (2005) Verification Prior to Installation, Software Update Integrity Strength:**

Software update packages shall be digitally signed using a NIST approved algorithm with a security strength of at least 112-bits.

*******TE 7.4.6-E.6.2.1 (2005) Verification Prior to Installation, Software Update Integrity Strength:**

TE 7.4.6-E.6.2.1 (2005) Verification Prior to Installation, Software Update Integrity Strength is applicable if the SUT performs software verification prior to installation.

The tester shall examine the signature on a software update package. This can be done by double clicking on Windows or Unix systems if the software update package is signed using standard-based format.

The tester shall examine that the output displays the hashing algorithm as SHA-224 bit or greater.

The tester shall examine that the output displays the signature algorithm as one of the following:

1. 2048 Bit RSA;
2. ECDSA with primary field of 224 bits or greater;
3. ECDSA with binary field of 233 bits or greater; or
4. DSA with:
   a) Large prime 2048 bits or greater
   b) Small prime 224 bits or greater

---

**RE 7.4.6-E.7.1 (2005) Verification Prior to Installation, Software Update Integrity Verification:**

The software installation process shall verify digital signatures, software version identification, software prerequisites and dependencies, and vendor specific authorization information associated with the software before the software is installed.

Analysis:
The digital signature verification is tested under TE 7.4.6-E.6.1.2 (2005) Verification Prior to Installation, Software Update Integrity – Signature Test.

Version identification is tested 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content.

Software prerequisites and dependencies is tested under 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content.

Vendor specific authorization is tested under 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content.

---

**RE 7.4.6-E.7.2 (2005) Verification Prior to Installation, Action for Invalid Update:**
The software installation process shall not install software with invalid digital signatures, version numbers, or vendor specific authorization information, and shall not install software on systems that do not meet the update requisites.

Analysis:
The digital signature verification failure is tested under TE 7.4.6-E.6.1.2 (2005) Verification Prior to Installation, Software Update Integrity – Signature Test.

Incorrect version identification is tested 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content.

Failure of vendor specific authorization is tested under 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content.

Improper software prerequisites and dependencies is tested under 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content.

---

**RE 7.4.6-E.8 (2005) Verification Prior to Installation, Previous Version Installation Prevention:**
The voting system shall have the capability to prevent the installation of previous versions of applications or files.

*******TE 7.4.6-E.8.1 (2005) Verification Prior to Installation, Previous Version Installation Prevention:**
TE 7.4.6-E.8.1 (2005) Verification Prior to Installation, Previous Version Installation Prevention is applicable if the SUT performs software verification prior to installation.

The tester shall acquire the latest “software update package with entire SUT software in it” for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall acquire an older “software update package” for the SUT from the appropriate source (e.g., vendor, local election board representative).

The tester shall authenticate to the SUT as an administrator.

The tester shall configure the SUT to reject previous versions of software update packages.

The tester shall install the software on the SUT using the latest “software update package with entire SUT software in it”.

The tester shall verify that the installation succeeds.

The tester shall attempt to install the software on the SUT using the older “software update package”.

The tester shall verify that the attempt fails.

The tester shall examine the event log and verify that it contains the following information for an event log entry:

1. Failure of software installation;
2. Cause of failure indicates attempt to install a prior version;
3. Date and time TE 7.4.6-E.8.1 (2005) Verification Prior to Installation, Previous Version Installation Prevention was executed;
4. Software update package name and version number or list of software in the software update package and their version numbers;
5. Type of action as installation of application/file; and
6. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the "software update package" name and version number.

The tester shall attempt to install older version of one of the applications.

The tester shall verify that the attempt fails.

The tester shall examine the event log and verify that it contains the following information for an event log entry:

1. Failure of software installation;
2. Cause of failure indicates attempt to install a prior version;
3. Date and time TE 7.4.6-E.8.1 (2005) Verification Prior to Installation, Previous Version Installation Prevention was executed;
4. Software application name and version number;
5. Type of action as installation of application/file; and
6. A cryptographic hash of 28-64 bytes that matches the cryptographic hash for the software application name and version number.

---

### RE 7.4.6-E.9 (2005) Verification Prior to Installation, Software Installation Event Log Protection:

The software installation process shall result in information being stored in the voting system equipment's log such that altering or deleting log entries or the log will be detected.

### *****TE 7.4.6-E.9.1 (2005) Verification Prior to Installation, Software Installation Event Log Protection:

TE 7.4.6-E.9.1 (2005) Verification Prior to Installation, Software Installation Event Log Protection is applicable if the SUT performs software verification prior to installation.

The tester shall authenticate to the SUT as an administrator.

The tester shall attempt to use vendor supplied tools or editor to edit the event log. The tester shall verify that one of the following is true:

1. The SUT does not allow modification to the event log; or
2. The SUT uses digital signature on the event log and the following is true:
   a) Digital signature is no longer valid; and
   b) SUT will no longer modify the specific event log due to digital signature failure; and
   c) SUT will create another event log.
3. The SUT records an event of event log modification by administrator in the event log.

The tester shall attempt to delete the event log. The tester shall verify that the attempt fails.

---

### RE 7.4.6-E.10 (2005) Verification Prior to Installation, Software Installation Event Log Activity:

The minimum information to be included in the voting system equipment log shall be:

1. Success or failure of the software installation process;
2. Cause of a failed software installation (such as invalid version identification, digital signature, etc.);
3. Application or file name(s), and version number(s);
4. A description of the type of action performed on each application and/or file (e.g., new application/file, deletion or overwriting of existing file);
5. A cryptographic hash of the software update package using FIPS 140-2 level 1 or higher validated cryptographic module.

*******TE 7.4.6-E.10.1 (2005) Verification Prior to Installation, Software Installation Event Log Activity:**

TE 7.4.6-E.10.1 (2005) Verification Prior to Installation, Software Installation Event Log Activity is applicable if the SUT performs software verification prior to installation.

The tester shall examine the results of TE Crypto Module.1 through TE Crypto Module.5. The tester shall verify the following:

1. One of the cryptographic modules listed is the one that performs cryptographic hash of the software update package for storage in the event log; and
2. One of the cryptographic modules listed is the one that performs digital signature verification on the software packages.

Note: The actual event log content is verified as part of various setup validation attempt TEs such as the following:

1. TE 7.4.6-E.1.1 (2005) Verification Prior to Installation, Singular Software Installation Process
2. TE 7.4.6-E.1.3.1 (2005) Verification Prior to Installation, Software Installation Prevention While Polls Open
3. TE 7.4.6-E.2.1.1 (2005) Verification Prior to Installation, Software Installation Restriction
4. TE 7.4.6-E.2.2.1 (2005) Verification Prior to Installation, Software Installation Change Reporting
5. TE 7.4.6-E.4.2.1 (2005) Verification Prior to Installation, Software Update Content
6. TE 7.4.6-E.6.1.2 (2005) Verification Prior to Installation, Software Update Integrity – Signature Test
7. TE 7.4.6-E.8.1 (2005) Verification Prior to Installation, Previous Version Installation Prevention

---

**RE 7.4.6-F (2005) Verification After Installation, Software Verification External Interface:**

If software is verified after being installed on the voting system equipment, the voting system equipment shall provide an external interface to the location of the voting system software for software verification purposes.

*******TE 7.4.6-F.1 (2005) Verification After Installation, Software Verification External Interface:**

TE 7.4.6-F.1 (2005) Verification After Installation, Software Verification External Interface is applicable if the SUT performs software verification after installation.

The tester shall examine the vendor defined procedures for setup validation (see RE 7.4.6-C.2 (2005) Software Verification Process). The tester shall verify that the procedure includes use of an external interface on the SUT.

The tester shall physically examine the SUT and the documented interface characteristics and verify that the description matches the physical interface. Examples of interfaces are: USB, LAN, 9 pin connector (male of female), 15 pin connector (male of female), etc.

The tester shall install the software on the SUT using the "Entire SUT Software".

The tester shall execute the software verification procedures. The tester shall verify that the procedures succeed.

The tester shall execute the software verification procedures. The tester shall verify that the procedures succeed. . (Note: This step is conducted to gain additional assurance that the software verification does not modify the software).

---

**RE 7.4.6-F.1.1 (2005) Verification After Installation, External Interface Tamper Evidence:**
The external interface shall be protected using tamper evident techniques.

*****TE 7.4.6-F.1.1.1 (2005) Verification After Installation, External Interface Tamper Evidence:**
TE 7.4.6-F.1.1.1 (2005) Verification After Installation, External Interface Tamper Evidence is applicable if the SUT performs software verification after installation.

The tester shall prepare the SUT with the physical security controls as specified by the vendor.

The tester shall use routine tools available from retailed hardware store. Such tools include screw drivers, wrenches, Exacto knife, dissolving chemical, pliers, scissors, glue, adhesive, etc.

The tester shall attempt to use these tools to breach the access point where external interface used for software verification is connected. The tester's goal is to disable or replace the external interface without leaving physical evidence. The tester shall spend at least 30 minutes in an attempt to tamper with the access point without leaving physical evidence. The 30 minutes shall not include analysis, preparation and other efforts; 30 minutes shall be devoted to actual physical tampering.

TE 7.4.6-F.1.1.1 (2005) Verification After Installation, External Interface Tamper Evidence fails if the tester is able to breach the physical security of the access point for the external interface used for software verification without leaving tamper evidence within the 30 minutes time allotted. In other words, TE 7.4.6-F.1.1.1 (2005) Verification After Installation, External Interface Tamper Evidence passes under any one or more of the following:

1. The tester is unable to breach the access point;
2. The tester is able to breach the access point, but it leaves physical tamper evidence such as a scratch, old tamper evidence seal gone or visibly damaged, screw visibly damage, etc; or
3. The tester is able to breach the access point without leaving tamper evidence, but determines that such attack will take more than 30 minutes by a skilled or trained attacker.

---

**RE 7.4.6-F.1.2 (2005) Verification After Installation, External Interface Enabled Indicator:**
The external interface shall have a physical indicator showing when the interface is enabled and disabled.

*******TE 7.4.6-F.1.2.1 (2005) Verification After Installation, External Interface Enabled Indicator:**
TE 7.4.6-F.1.2.1 (2005) Verification After Installation, External Interface Enabled Indicator is applicable if the SUT performs software verification after installation.

The tester shall examine the vendor defined procedures for setup validation (see RE 7.4.6-C.2 (2005) Software Verification Process). The tester shall verify that the procedure includes a physical indicator for status (enabled or disabled) the external interface used for software verification.

The tester shall physically examine the SUT and verify that the indicator is present as indicated in the documentation.

The tester shall examine the indicator characteristics to verify that it can indicate at least two states (enabled or disabled). Examples of appropriate indicators are:

1. Light indicator
2. Color light indicator
3. Physical switch with two (enabled or disabled) or more positions.

---

**RE 7.4.6-F.1.3 (2005) Verification After Installation, External Interface Disabled During Voting:**
The external interface shall be disabled during voting.

*******TE 7.4.6-F.1.3.1 (2005) Verification After Installation, External Interface Disabled During Voting:**
TE 7.4.6-F.1.3.1 (2005) Verification After Installation, External Interface Disabled During Voting is applicable if the SUT performs software verification after installation.

The tester shall install the software on the SUT using the “Entire SUT Software”.

The tester shall use the vendor defined procedures to open the polls.

The tester shall physically examine the external interface for software verification. The tester shall verify that the physical state indicates that the external interface is disabled.

The tester shall attempt to execute software verification process. The tester shall verify that the attempt either can not be made or fails since the verification device can not access the SUT via the external interface.

---

**RE 7.4.6-F.1.4 (2005) Verification After Installation, External Interface Read-only:**
The external interface should provide a direct read-only access to the location of the voting system software without the use of installed software.

*******TE 7.4.6-F.1.4.1 (2005) Verification After Installation, External Interface Read-only:**
TE 7.4.6-F.1.4.1 (2005) Verification After Installation, External Interface Read-only is applicable if the SUT performs software verification after installation.

The tester shall examine the vendor documented software verification process.

The tester shall install the software on the SUT using the “Entire SUT Software”.

The tester shall execute the software verification process.

The tester shall document if the software verification requires execution of the software or simply have the SUT powered so that the external interface can read the SUT software.

The tester shall attempt to use this interface to write to the SUT by attempting to copy a file to the SUT.

The tester shall verify that the attempt fails.

---

**RE 7.4.6-F.2 (2005) Verification After Installation, Verification Process:**
The verification process should be able to be performed using COTS software and hardware available from sources other than the voting system vendor.

*****TE 7.4.6-F.2.1 (2005) Verification After Installation, Verification Process:**
TE 7.4.6-F.2.1 (2005) Verification After Installation, Verification Process is applicable if the SUT performs software verification after installation.

The tester shall examine the vendor documented software verification process. The tester shall list the vendor defined software required on the computer that reads the SUT software and perform the software verification.

The tester shall examine each of the software on the list developed above and write down if these are generic software available from other sources or are SUT vendor unique. Examples of general software include: unzip, zip, file compare, digital signature verification on file, etc.

---

**RE 7.4.6-F.2.1 (2005) Verification After Installation, Cryptographic module:**
If the process uses hashes or digital signatures, then the verification software shall use a FIPS 140-2 level 1 or higher validated cryptographic module.

*****TE 7.4.6-F.2.1.1 (2005) Verification After Installation, Cryptographic module:**
TE 7.4.6-F.2.1.1 (2005) Verification After Installation, Cryptographic module is applicable if the SUT performs software verification after installation.

TE 7.4.6-F.2.1.1 (2005) Verification After Installation, Cryptographic module is applicable if the SUT software verification includes hashes or digital signatures.

The tester shall confirm that at least one of the following is true:

1. The SUT vendor has identified one or more FIPS 140-2 validated cryptographic module to perform hash calculation or digital signature verification; or
2. The SUT vendor provides a FIPS 140-2 validated cryptographic module to perform hash calculation or digital signature verification for the platform on which the software verification is done.

The tester shall conduct the following TEs for the cryptographic module identified above. The tester shall ensure in conducting the TEs that the execution environment is viewed as the machine on which the software verification is executed. Note that this machine is different from the SUT.

1. TE Crypto Module.1 Cryptographic module validation information verification -- Modules
2. TE Crypto Module.2 Cryptographic module validation environment verification
3. TE Crypto Module.3 Cryptographic module validation description verification
4. TE Crypto Module.4 Cryptographic module validation configuration verification
5. TE Crypto Module.5 Cryptographic module validation algorithm verification

---

### RE 7.4.6-F.2.2 (2005) Verification After Installation, Verification Alternatives:

The verification process shall either

1. use reference information on unalterable storage media received from the repository or
2. verify the digital signature of the reference information on any other media.

### *****TE 7.4.6-F.2.2.1 (2005) Verification After Installation, Verification Alternatives:

TE 7.4.6-F.2.2.1 (2005) Verification After Installation, Verification Alternatives is applicable if the SUT performs software verification after installation.

The tester shall examine the vendor documented software verification process. The tester shall verify that the process is comparison with reference information or digital signature verification.

If the process is use of reference information, the tester shall verify that one of the following is true:

1. The reference information includes software and verification consists of file compare; or
2. The reference information is a hash of 28-64 bytes and the software verification includes hash and that hashing software is either provided by the vendor or the requirement for it is identified by the vendor.

If the process is use of digital signature, the digital signature verification software is either provided by the vendor or the requirement for it is identified by the vendor.

The tester shall use one of the following methods to modify two bytes of one of the software components of the SUT:

1. The tester may use an editor on the SUT to make the change; or
2. The tester may install an editor (e.g., hex editor or binary editor) on the SUT, make the change using the installed editor, and remove the editor from the SUT; or
3. The tester may make changes on another machine and replace the software on the SUT with the modified one.

The tester shall execute the software verification as described by the vendor documentation per RE 7.4.6-C.2 (2005) Software Verification Process.

The tester shall verify that the software verification process results in an error.

The tester shall replace the modified software with the valid software.

The tester shall execute the software verification as described by the vendor documentation per RE 7.4.6-C.2 (2005) Software Verification Process.

The tester shall verify that the SUT passes software verification.

---

### RE 7.4.6-G (2005) Setup Validation Register Values:

Setup validation methods shall verify that registers and variables of the voting system equipment contain the proper static and initial values.

### TE 7.4.6-G.1 (2005) Setup Validation Register Values:

The tester shall install the software on the SUT using the “Entire SUT Software”.

The tester shall examine the values of the static registers and variables.

The tester shall verify that these values match the vendor documentation.

The tester shall examine the values of dynamic registers and variables that are not set for specific election.

The tester shall verify that these values match the vendor documentation.

The tester shall change some static registers and variables to incorrect values.

The tester shall conduct setup validation. The tester shall verify that the validation fails.

The tester shall change the values back to proper values.

The tester shall conduct setup validation. The tester shall verify that the validation succeeds.

The tester shall change some of dynamic registers and variables that are not set for specific election to incorrect values.

The tester shall conduct setup validation. The tester shall verify that the validation fails.

The tester shall change the values back to proper values.

The tester shall conduct setup validation. The tester shall verify that the validation succeeds.

---

**RE 7.4.6-G.1 (2005) Register Values Reporting:**
The vendor should provide a method to query the voting system to determine the values of all static and dynamic registers and variables including the values that jurisdictions are required to modify to conduct a specific election.

Analysis:
This requirement is tested by TE 7.4.6-G.2.1 (2005) Register Value Documentation.

---

**RE 7.4.6-G.2 (2005) Register Value Documentation:**
The vendor shall document the values of all static registers and variables, and the initial starting values of all dynamic registers and variables listed for voting system software, except for the values set to conduct a specific election.

**TE 7.4.6-G.2.1 (2005) Register Value Documentation:**
The tester shall examine the vendor document and verify the following:

1. The documentation contains a list of static registers and variables
2. The documentation contains values for each of the static registers and variables
3. The documentation includes a list of dynamic registers and variables
4. The documentation identifies which of the dynamic registers and variables are set for specific election and the documentation identifies how the election officials can set values for each of these
5. The documentation includes initial values for the dynamic registers and variables that are not set for specific election.

The tester shall install the software on the SUT using the “Entire SUT Software”.

The tester shall examine the values of the static registers and variables.

The tester shall verify that these values match the vendor documentation.

The tester shall examine the values of dynamic registers and variables that are not set for specific election.

The tester shall verify that these values match the vendor documentation.

The tester shall use the vendor defined procedures to set dynamic registers and variables that are set for specific election. The tester shall verify that these procedures succeed.

The tester shall open the SUT for polls.

The tester shall cast some ballots.

The tester shall examine the values of the static registers and variables.

The tester shall verify that these values match the vendor documentation, i.e., they have not changed since installation.

---