Network Device Interpretation # 26

Testing the absence of a published hash for a TOE update

Status: ☒ *Active* ☐ *Inactive*

Date: *30-May-2016*

Type of Document: ☒ *Technical Decision* ☐ *Technical Recommendation*

Approved by: ☒ *Network iTC Interpretations Team* ☐ *Network iTC*

Affected Document(s): *ND cPP V1.0, FW cPP V1.0, ND SD V1.0*

Affected Section(s): *Multiple, see Resolution section for details*

Superseded Interpretation(s): *None*

Issue:

2a. We foresee two different options for mechanisms for the TOE to validate a published hash

(1) The TOE receives the update package which contains the hash for validating. The TOE produces a hash of the update and compares it against the hash contained within the update. The TOE will then perform the update or not perform the update based upon the hash check. This could all be done with or without admin action.

(2) The TOE received the update package. The TOE produces a hash of the update and presents it to the admin. The admin can then check the hash value against a published value. The admin can then perform an action of the TOE to instruct it to update or not update based upon their manual hash check.

We would understand that the use of "published" hash to mean a value that is documented either on a website, guidance documentation, etc. However, this only seems to be needed for option (2) whereas option (1) would not require the admin to view the hash value for confirmation.

Our question, is option (2) allowed? This seems to have been allowed for the NDPP based upon TD0026. However, the supporting document has Test 2 - 2) which requires the testing of an update for "an image without published hash". If our understanding of "published" hash is correct, this would basically be a non-test for option (2) because without a hash value documented there is nothing for the admin to validate the hash value against.

If option (2) is allowed, then what is the expectation of the test for Test 2 - 2)?

If option (2) is not allowed, then we recommend removing or clarifying "published" hash as it is confusing.

Resolution:

The NIT is in favor of disallowing option 1 as described above because it is critical to the usefulness of a published hash that it is obtained from a trusted source. The NIT recommends to modify the wording of the application notes to FPT_TUD_EXT.1.2 and FPT_TUD_EXT.1.3 as well as the section on FPT_TUD_EXT.1 in the Supporting Document to enhance clarity.

Changes to the cPPs:

Addition to the Application Note for FPT_TUD_EXT.1.2:

The TSS explains what actions are involved in the TOE support when using the 'support automatic checking for updates' or 'support automatic updates' selections.

When published hash values (see FPT_TUD_EXT.1.3) are used to protect the trusted update mechanism, the TOE must not automatically download the update file(s) together with the hash value (either integrated in the update file(s) or separately) and automatically install the update without any active authorization by the Security Administrator, even when the calculated hash value matches the published hash value. When using published hash values to protect the trusted update mechanism, the option 'support of automatic updates' must not be used (automated checking for updates is permitted, though). The TOE may automatically download the update file(s) themselves but not to the hash value. For the published hash approach, it is intended that a Security Administrator is always required to give active authorisation for installation of an update (as described in more detail under FPT_TUD_EXT.1.3) below. Due to this, the type of update mechanism is regarded as 'manually initiated update', even if the update file(s) may be downloaded automatically. A fully automated approach (without Security Administrator intervention) can only be used when 'digital signature mechanism' is selected in FPT_TUD_EXT.1.3 below.

Addition to the Application Note for FPT_TUD_EXT.1.3:

When published hash values are used to secure the trusted update mechanism, an active authorization of the update process by the Security Administrator is always required. The secure transmission of an authentic hash value from the developer to the Security Administrator is one of the key factors to protect the trusted update mechanism when using published hashes and the guidance documentation needs to describe how this transfer has to be performed. For the verification of the trusted hash value by the Security Administrator different use cases are possible. The Security Administrator could obtain the published hash value as well as the update file(s) and perform the verification outside the TOE while the hashing of the update file(s) could be done by the TOE or by other means. Authentication as Security Administrator and initiation of the trusted update would in this case be regarded as 'active authorization' of the trusted update. Alternatively, the Administrator could provide the TOE with the published hash value together with the update file(s) and the hashing and hash comparison is performed by the TOE. In case of successful hash verification the TOE can perform the update without any additional step by the Security Administrator. Authentication as Security Administrator and sending the hash value to the TOE is regarded as 'active authorization' of the trusted update (in case of successful hash verification), because the administrator is expected to load the hash value only to the TOE when intending to perform the update. As long as the transfer of the hash value to the TOE is

performed by the Security Administrator, loading of the update file(s) can be performed by the Security Administrator or can be automatically downloaded by the TOE from a repository.

Replacement for the Application Note for FMT_MOF.1.1/AutoUpdate:

FMT_MOF.1/AutoUpdate is only applicable if the TOE supports automatic checking for updates and/or automatic updates and allows to enable and disable them. Enable and disable of automatic checking for updates and/or automatic updates is restricted to Security Administrators. The option "automatic update" may only be selected if digital signatures are used to validate the trusted update.

Changes to the SD

Addition to FPT_TUD_EXT.1/TSS:

If a published hash is used to protect the trusted update mechanism, then the evaluator shall verify that the trusted update mechanism does involve an active authorization step of the Security Administrator, and that download of the published hash value, hash comparison and update is not a fully automated process involving no active authorization by the Security Administrator. In particular, authentication as Security Administration according to FMT_MOF.1/ManualUpdate needs to be part of the update process when using published hashes.

Addition to FPT_TUD_EXT.1/Guidance documentation:

If a published hash is used to protect the trusted update mechanism, the evaluator shall verify that the guidance documentation describes how the Security Administrator can obtain authentic published hash values for the updates.

Replacement of description for FPT_TUD_EXT.1/Tests, Test 2:

Test 2 (if digital signatures are used): The evaluator first confirms that no updates are pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test. The evaluator obtains or produces illegitimate updates as defined below, and attempts to install them on the TOE. The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:

1. A modified version (e.g. using a hex editor) of a legitimately signed update.

2. An image that has not been signed

3. An image signed with an invalid signature (e.g. by using a different key as expected for creating the signature or by manual modification of a legitimate signature).

4. If the TOE allows a gap between the installation of an update and a required reboot or activation to execute the updated code, the TOE must be able to display both the currently executing version and most recently installed version. The handling of version information of the most recently installed version might differ between different TOEs depending on the point in time when an attempted update is rejected. The evaluator shall verify that the TOE handles the most

recently installed version information for that case as described in the guidance documentation. After the TOE has rejected the update the evaluator shall verify, that both, current version and most recently installed version, reflect the same version information as prior to the update attempt.

Test 2 (if published hash is verified on the TOE): If the published hash is provided to the TOE by the Security Administrator and the verification of the hash value over the update file(s) against the published hash is performed by the TOE, then the evaluator shall perform the following tests. The evaluator first confirms that no update is pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test.

1. The evaluator obtains or produces an illegitimate update such that the hash of the update does not match the published hash. The evaluator provides the published hash value to the TOE and calculates the hash of the update either on the TOE itself (if that functionality is provided by the TOE), or else outside the TOE. The evaluator confirms that the hash values are different, and attempts to install the update on the TOE, verifying that this fails because of the difference in hash values (and that the failure is logged). Depending on the implementation of the TOE, the TOE might not allow the user to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE.

2. The evaluator uses a legitimate update and tries to perform verification of the hash value without storing the published hash value on the TOE. The evaluator confirms that this attempt fails. Depending on the implementation of the TOE it might not be possible to attempt the verification of the hash value without providing a hash value to the TOE, e.g. if the hash value needs to be handed over to the TOE as a parameter in a command line message and the syntax check of the command prevents the execution of the command without providing a hash value. In that case the mechanism that prevents the execution of this check shall be tested accordingly, e.g. that the syntax check rejects the command without providing a hash value, and the rejection of the attempt is regarded as sufficient verification of the correct behaviour of the TOE in failing to verify the hash. The evaluator then attempts to install the update on the TOE (in spite of the unsuccessful hash verification) and confirms that this fails. Depending on the implementation of the TOE, the TOE might not allow to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE.

3. If the TOE allows a gap between the installation of an update and a required reboot or activation to execute the updated code, the TOE must be able to display both the currently executing version and most recently installed version. The handling of version information of the most recently installed version might differ between different TOEs. Depending on the point in time when the attempted update is rejected, the most recently installed version might or might not be updated. The evaluator shall verify that the TOE handles the most recently installed version information for that case as described in the guidance documentation. After the TOE has rejected the update the evaluator shall verify, that both, current version and most recently installed version, reflect the same version information as prior to the update attempt.

<u>Addition to the description for FPT_TUD_EXT.1/Tests:</u>

If the verification of the hash value over the update file(s) against the published hash is not performed by the TOE, Test 2 shall be skipped.

Rationale:

Regarding option 1: the TOE cannot just trust a hash delivered with the update package because an attacker would be able to calculate a correct hash for an invalid update package, and therefore the 'publication' of the hash by a trusted source is part of the countermeasure.

Description for Test 2 has been split for the use cases of using digital signatures and using published hash to protect the trusted update.

Further Action:

Update the ND cPP, FW cPP and ND SD V1.0 as proposed.

Action by Network iTC:

Click here to enter text.