secuvera:
BSI-zertifizierter IT-Sicherheitsdienstleister und Prüfstelle

jtsec
BEYOND IT SECURITY

Disclaimer! This document may be outdated!

If you are looking for the last version of the ISO SC27 WG3 Technical Report “Towards Creating an Extension for Patch Management for ISO/IEC 15408 and ISO/IEC 18045”, you shall obtain it through the ISO community.

Towards Creating an Extension for Patch Management for ISO/IEC 15408 and ISO/IEC 18045

Sebastian Fritsch and Javier Tallón

Index

1 SCOPE

Common Criteria (CC) or ISO/IEC 15408 allows the certification of the assurance of IT products. The standard has proven to be flexible for high-security use-cases especially for secure elements, security hardware devices and e-government project related components.

But as good as the standard can be used for the base certification, the standard does not support re-certifications of updates or security-patched products. ISO/IEC 15408 nor ISO/IEC 18045 (or CEM) contain dedicated methods or evaluation activities which would support the evaluation of minor changes or minor updates.

Some of these aspects were addressed by users of the standards especially certification bodies but also the mutual recognition agreements (e.g. CCRA). But in a lot of real-world use-cases the developers provide updated or patched TOEs but the effort to re-certify these versions is mostly avoided.

One of the most important aspect in the context of patch management and security updates is therefore the characterization of the changes in minor and major updates. The [CCRA-AC] offers a general guideline on the differences between major and minor updates.

Typically, a security bug-fix will not fall in any of the three samples that the CCRA mutual recognition agreement [CCRA-AC] provides to characterize a major change:

- It is not a change to the set of claimed assurance requirements
- It is not a change to the set of claimed functional requirements
- It is not necessarily a set of minor changes that together have a major impact upon the security

The only hint that we have to characterize a bug fix in major or minor is if it will or not affect to the assurance evidence. For example, if a TOE has been certified to EAL1, a change to the source code and/or hardware schematics would not have an impact upon the assurance documentation.

Therefore, it shall be easy to have a maintenance report for low assurance levels than for high assurance (EAL4 or higher), as the bugfix will always affect source code. This however shall always be analysed in detail as part of the IAR and the decision of classifying as major or minor is made by the applicable CB.

In this document we will follow a different approach and try categorize patches or updates from the developer's perspective. In the following the focus is on minor updates which will be defined as one of the following:

- functional patch (functional bug-fixes, but no functional enhancements)
- security patch (only security bug-fixes)
- minor functional enhancement (new functionality)

This document defines additional building blocks (i.e. SFRs for patch functionality and one additional ALC family, see Chapter 5) which can be integrated into PPs and STs to provide additional assurance for the TOE's patching functionality and the developer's patch management process. It also encourages developer to consistently generate evidences which can be used in the future re-certification process.

Finally, this document contains recommendations to certification bodies (or mutual recognition agreement) how to utilize the additional assurance and additional evidence in their processes to support developer to consistently re-certify also their updated or patched TOEs to the benefit of the users of these TOEs.

2 NORMATIVE REFERENCES

- ISO 15408
- ISO 18045
- [CCRA-AC]

3 TERMS AND DEFINITIONS

Term	Definition
Additional code	Source Code or Binary Code which represents the Patch which will be installed onto the TOE to fulfill the update.
Assurance Continuity	Concept to maintain and extend assurance on patched TOE version.
Atomic Activation	Activation of the Patch simultaneously leads to change of identification and other meta data
Base Evaluation	Complete evaluation of the TOE, i.e. the existing Common Criteria evaluation.
TOE Assurance	Certified TOEs which are trusted in strict application of the evaluated patch management process.
TOE End-of-life	Date until when the user may expect to receive new security patches. It shall be greater than the period of validity of the certificate (which can be increased through standard AC).
Final TOE	Initial TOE with the patch applied.
Initial TOE	TOE without any patches.
Identification Data	Identifies the Initial TOE or patches.
ITSEF	Information Technology Security Evaluation Facility
Major Updates	Substantial functional enhancements of the TOE.
Minor Updates	One of the following: - functional patch (functional bug-fixes, but no functional enhancements) - security patch (only security bug-fixes) - minor functional enhancement (new functionality)
Patch	Set of changes (functional, security or both) compared to the original TOE.
Fix	A patch addressing a defect.
Security Impact Analysis Report	Developer's self-assessment of security relevance of a planned patch, used in ALC_PAM.1.2D.
Transport	Transport of patches from the developer to the user who applies the patch, either logical or physical.

4 CONCEPT

4.1 PROBLEM DESCRIPTION

The following figure shows the Product Security Risk Model for the case after a new vulnerability was detected and became publicly known. Until the developer will release an update that removes the vulnerability the product will be insecure, this status is shown in red below.

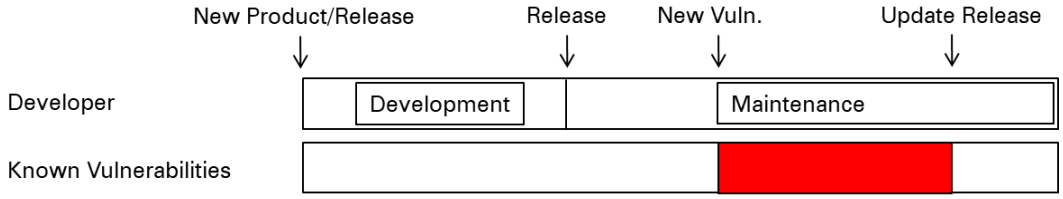


Figure 1 Product Security Risk Model

In consequence developers have the responsibility to build and release those updates in a short period of time after the vulnerability became public. Developers who obtained a CC certificate before, are implicitly or directly requested to maintain their certificate and the corresponding assurance statements. But in real-world scenarios in most cases the certificate maintenance will be avoided.

In consequence this transfers the decision to use either a previously certified or a recently patched version to the user of the TOE. In other words, the risk owner (or user of the TOE) has to decide and run his own risk assessment which version of the TOE to use.

Figure 2 Not applied CC re-certifications after minor updates

Today we only see re-certification if this is required due to regulation requirements. In these cases, the risk owner is not allowed to install the recently patched version of the TOE before the re-certification is finished.

As described above the risk owner has to decide which version to use. In case the risk owners are forced to use only certified versions, they have to accept known vulnerabilities in the TOE and further risk mitigation has to be done, i.e. additional compensating countermeasures against the new vulnerabilities have to be implemented. By supporting re-certification methods those extra activities could be avoided.

Regulatory bodies often mandate risk users to use versions of software which was approved to be secure and certified some time ago. The use of easier re-certification gives those bodies a modernized tool to mandate the use of certified but still secure software also in the later product lifecycle.

Figure 3 Availability of patch and the corresponding new certificate might be not in sync

Also, in case new re-certification are applied frequently the process of re-certification process might take a long time. In consequence end-users might have the patch ready for installation but the re-certification will take some time so the reassurance of the patch is delayed. The concept tries to prepare necessary developer evidences consistently during development time and therefore accelerates the re-certification process.

This proposed patch management extension has the following advantages for the different stakeholders:

- risk owners are supported to remove existing, known vulnerabilities in TOEs in a fast and efficient way
- regulatory body can request risk owners more effectively to install available updates to remove existing vulnerabilities
- developers get a new tool to use CC more active as a qualification strategy because more frequent certification results can be aligned with much more frequent product version releases

4.2 SOLUTION

The solution described in the following relies on two pillars:

- Add additional functional requirements (FPT_PAM) which address the patch or update functionality of the base TOE (see Chapter 5.1)
- Add additional lifecycle requirements (ALC_PAM) to get commitment from developers to consistently monitor for flaws or issues after release of the base TOE, but also encourage developers to consistently generate evidences for future re-certifications (see Chapter 5.3)

The following figure shows the application of ALC_PAM which supports the timely delivery of the patch or update but also the maintenance of the internal and also external assurance activities.

Figure 4 Application of ALC_PAM

5 PATCH MANAGEMENT COMPONENTS (FPT_PAM AND ALC_PAM)

5.1 UPDATE SFR PACKAGE

5.1.1 SPD

5.1.1.1 ASSUMPTIONS

None

5.1.1.2 THREATS

T.PAM.INSECURE_TOE: An attacker is able to subvert the update mechanisms so the TOE is not able to receive a security patch, remaining in a certified but insecure state.

T.PAM.INSECURE_PATCHING: An attacker forges an rogue malicious patch that is installed or processed by the TOE, altering the intended TSF functionality.

5.1.2 OPERATIONAL ENVIRONMENT SECURITY OBJECTIVES

OE.PAM.KEY_STRENGTH: The cryptographic keys used to provide authenticity, integrity and protection against replay or misuse of new patches shall have a strength commensurate with the Evaluation Assurance Level and follow a documented life cycle.

OE.PAM.PATCH_GENERATION: The use of cryptographic keys to provide authenticity of new patches shall be carried out in a secure and audited environment, commensurate with the Evaluation Assurance Level.

OE.PAM.NOTIFICATION: The developer shall notify the end-user of the availability of new security patches.

OE.PAM.PATCH_AVAILABILITY: The developer shall make available in a secure way, security patches and installation instructions until the end-of-life of the TOE.

5.1.3 TOE SECURITY OBJECTIVES

(optional) O.PAM.PATCH_CHECKING: The TOE shall regularly check for new security patches and notify TOE administrators of the availability of the updates.

(optional) O.PAM.TRANSPORT_SECURITY: The channel used to check for the availability of patch(s) and/or download the patch(s) shall provide integrity and authenticity.

O.PAM.AUTHENTICATED_INSTALL: An administrator user with full privileges shall be required to install a patch or schedule patch installation.

O.PAM.SECURE_LOAD: The Loader of the Initial TOE shall check an evidence of authenticity and integrity of the loaded Patch. During the Load Phase of a Patch, the TOE shall remain secure.

O.PAM.ATOMIC_ACTIVATION: Activation of the Patch and update of the Identification Data shall be performed at the same time in an Atomic way. All the operations needed for the code to be able to operate as in the Final TOE shall be completed before activation. If the Atomic Activation is successful, then the resulting product is the Final TOE

O.PAM.ERROR: In case of interruption or incident which prevents the forming of the Final TOE (such as tearing, integrity violation, error case…), the Initial TOE shall remain in its initial state or fail secure. i.e. shall be restored.

5.1.4 TOE SECURITY OBJECTIVE RATIONALE

	O.PAM.PATCH_CHECKING	O.PAM.TRANSPORT_SECURITY	O.PAM.AUTHENTICATED_INSTALL	O.PAM.SECURE_LOAD	O.PAM.ATOMIC_ACTIVATION	O.PAM.ERROR	OE.PAM.KEY_STRENGTH	OE.PAM.PATCH_GENERATION	OE.PAM.NOTIFICATION	OE.PAM.PATCH_AVAILABILITY
T.PAM.INSECURE_TOE	X	X							X	X
T.PAM.INSECURE_PATCHING		X	X	X	X	X	X	X		

T.PAM.INSECURE_TOE: This threat is mitigated by the operational environment **OE.PAM.NOTIFICATION** which will provide means to notify of the availability of new security patches to end users. The developer will make these patches available until the end-of-life of the TOE (**OE.PAM.PATCH_AVAILABILITY**)

(Optionally) According to **O.PAM.PATCH_CHECKING**, the TOE will check automatically for new updates, using a protected channel (**O.PAM.TRANSPORT_SECURITY**).

T.PAM.INSECURE_PATCHING: This threat is mitigated by the joint force of security objectives for the operational environment and security objectives for the TOE. **OE.PAM.PATCH_GENERATION** warranties that the security patches are signed in a secure environment following adequate procedures, while **OE.PAM.KEY_STRENGTH** provides assurance on the strength and quality of the used cryptographic keys.

The TOE itself have mechanisms to verify the signature of the patches (**O.PAM.SECURE_LOAD**). Only after successful verification of the signature, the TOE will process and install the patch in an atomic way (**O.PAM.ATOMIC_ACTIVATION**) so no dangerous TSF mediated actions are allowed. In case of error, **O.PAM.ERROR** will prevent the operation of the TOE in a failure state, restoring the TOE to its initial state. Every patch

installation must be approved by an administrative entity (**O.PAM.AUTHENTICATED_INSTALL**).

(Optionally) If the update is downloaded from an update server, this communication will be protected by **O.PAM.TRANSPORT_SECURITY**

5.1.5 REQUIREMENTS (SFRS)

Patch Management is a whole new topic that was not previously contemplated in ISO 15408. A new family is needed to describe most of the features expected to be present in an updatable TOE.

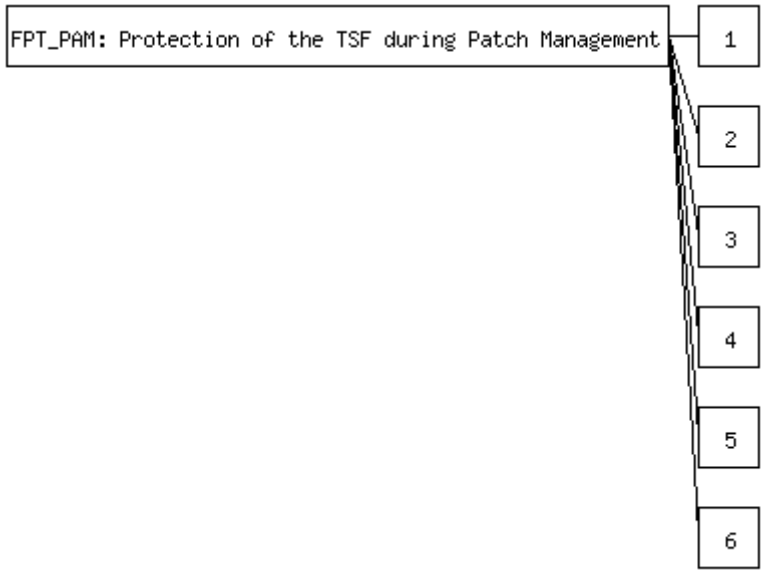
The family is included under FPT_PAM as it deals with the protection of the TSF from malicious updates and from an unnoticed unsecure state due to the presence of vulnerabilities not detected during the initial evaluation.

5.1.6 PROTECTION OF THE TSF DURING PATCH MANAGEMENT (FPT_PAM)

Family behavior

This family covers the requirements needed to protect the TSF while applying new patches to the Initial TOE.

Component levelling



User notes

This requirements and their corresponding security objectives for the TOE shall be used in joint force with ALC_PAM Security Assurance Requirements.

Patches cannot be installed without the authorization of an administrative entity

The TSF will verify every patch for integrity and authenticity before being installed

When a new patch is activated the TOE shall not be allowed to stay in an inconsistent or insecure state, preventing TSF mediated actions.

The TOE will automatically check for the availability of new security patches using an update server.

While checking for new updates or downloading new updates, the connection with the update server will allow authentication of the endpoint and integrity verification.

The TSF shall preserve a secure state while processing new patches and performing actions to avoid new errors or notify end users or update servers.

Management: FPT_PAM.1, FPT_PAM.2, FPT_PAM.3, FPT_PAM.4, FPT_PAM.5, FPT_PAM.6

There are no management activities foreseen.

Audit: FPT_PAM.1

- Approval of the patch installation
- Schedule of the patch installation

Audit: FPT_PAM.2

- Invalid digital signature

Audit: FPT_PAM.3

- Beginning of patch installation
- Success of patch installation

Audit: FPT_PAM.4

- TOE Checks for new updates
- Failure connecting to the update server
- New patch available

Audit: FPT_PAM.5

- Failure in the establishment of the channel
- Failure in the verification of the remote endpoint

Audit: FPT_PAM.6

- Failure during patch activation
- Actions taken if any

FPT_PAM.1: Administrator mediated patching

Hierarchical to:

No other components.

Dependencies:

No dependencies.

FPT_PAM.1.1: ***The TSF shall require an authenticated [selection: administrative user, administration terminal] to allow the [selection: application, scheduling] of patches***

FPT_PAM.2: Trusted patching

Hierarchical to:

No other components.

Dependencies:

No dependencies.

FPT_PAM.2.1: ***The TSF shall cryptographically verify patches prior to installation using a digital signature scheme that provides a strength of [assignment: positive integer] bits that meet the following [assignment: list of standards].***

FPT_PAM.2.2: ***The TSF shall only proceed with the installation of new patches after the integrity and authenticity has been cryptographically verified***

FPT_PAM.2.3: ***The TSF shall allow the secure update of the critical security parameters involved in the verification of the digital signature.***

FPT_PAM.3: Atomic patch activation

Hierarchical to:

No other components.

Dependencies:

No dependencies.

FPT_PAM.3.1: ***The TSF shall perform the activation of the update in an atomic way so that it will not perform any TSF mediated action but [assignment: allowed actions performed by the TSF]***

FPT_PAM.3.2: ***After atomic patch activation the TOE shall show the new version.***

FPT_PAM.4: Automatic patch checking

Hierarchical to:

No other components.

Dependencies:

FPT_PAM.5

FPT_PAM.4.1: ***The TOE shall provide the ability to check for updates connecting to [assignment: service providing updates to the TOE]***

FPT_PAM.4.2: ***The TOE shall notify end user of the availability of new updates by means of [assignment: method to notify end user]***

FPT_PAM.5: Trusted connection to update server

Hierarchical to:

No other components.

Dependencies:

No dependencies.

FPT_PAM.5.1: ***The TSF shall provide a communication channel between itself and the developer's update server that provides authentication of the server and protection of the channel data from modification or disclosure.***

FPT_PAM.5.2: ***The TSF shall permit [selection, choose one of: the TSF, the update server] to initiate communication via the trusted channel.***

FPT_PAM.5.3: ***The TSF shall initiate communication via the trusted channel for [selection: download new patches, check for new patches]***

FPT_PAM.6: Preservation of secure state during patching

Hierarchical to:

No other components.

Dependencies:

No dependencies.

FPT_PAM.6.1: ***The TSF shall preserve a secure state when the following types of failures occur during patching***

a) ***Failure according to FPT_PAM.2 Trusted patching***

b) ***Failure according to FPT_PAM.3 Atomic patch activation***

c) ***[selection: Failure according to FPT_PAM.4 Automatic patch checking, Failure according to FPT_PAM.5 Trusted connection to update server]***

FPT_PAM.6.2: ***The following rules and actions will be performed when a number of failures are detected. [assignment: rules describing actions and conditions regarding failures.]***

SFR / TOE Security Objective	O.PAM.PATCH_CHECKING	O.PAM.TRANSPORT_SECURITY	O.PAM.AUTHENTICATED_INSTALL	O.PAM.SECURE_LOAD	O.PAM.ATOMIC_ACTIVATION	O.PAM.ERROR
FPT_PAM.1			X			
FPT_PAM.2				X		
FPT_PAM.3					X	
FPT_PAM.4	X					
FPT_PAM.5		X				
FPT_PAM.6						X

5.2 SAR: ALC_PAM

The SARs introduced in the following sections are related to different evaluation phases. During base evaluation of the TOE additional evaluation actions have to be introduced (compared to standard SAR from ISO/IEC 15408-3:2009) to establish assurance to the future patch generation process. The concept is to define ALC_PAM (Patch Management) and augment this family during base evaluation in the security target.

As Patch Management is part of the life-cycle assurance it has been introduced under the ALC class. It is not part of ALC_FLR because ALC_PAM describe how to handle patches, not how to handle flaws. However, both class are closely related and therefore the dependency with ALC_FLR.2.

5.2.1 PATCH MANAGEMENT (ALC_PAM)

Objectives

The objective of this family is to identify procedures to be implemented in the development process, which will be applied after the initial release of a TOE.

The application of these patch management processes cannot be always determined at the time of the base evaluation, but at least, it is possible to evaluate the policies and procedures that a developer has in place to perform management processes in the future, and obtain some evidence of the correct application of the procedures during the patching of the problems found during the evaluation of other assurance classes like AVA and ATE.

These procedures shall include instructions on how to securely sign, distribute and apply patches and how the life cycle of the keys used for providing authenticity of new patches is handled.

Component levelling

This family contains only one component.

Application notes

None

ALC_PAM.1 Patch Management Processes

	Dependencies: ALC_FLR.2 Flaw reporting procedures
	Developer action elements:
ALC_PAM.1.1D	**The developer shall provide a Patch Management Policy.**
ALC_PAM.1.2D	**The developer shall self-assess and confirm the application of existing policies on a regular basis saving records of its application.**
ALC_PAM.1.3D	**The developer shall provide security patches using the defined policies and procedures at least until the estimated end-of-life of the TOE.**
	Content and presentation elements:
ALC_PAM.1.1C	**The developer's patch management policies shall describe what is the criteria used for the decision that a patch has to be released.**
ALC_PAM.1.2C	**The Security Target shall contain the estimated end-of-life of the TOE.**
ALC_PAM.1.3C	**The developer's patch management policies shall describe how to self-assess the security relevance of a patch (i.e. Security Impact**

	Analysis Report, S-IAR) and which procedures have to apply due to which assessment result.
ALC_PAM.1.4C	**The developer's patch management policies shall describe how to update the evidence documentation used in the base evaluation.**
ALC_PAM.1.5C	**The developer's patch management policies shall describe how unhandled (potential) flaws are documented.**
ALC_PAM.1.6C	**The developer's patch management policies shall describe which organizational role (or group) is responsible for the patch development.**
ALC_PAM.1.7C	**The developer's patch management policies shall describe which policies have to be applied until the end of life of the TOE during the patch management.**
ALC_PAM.1.8C	**Each tool used for the patch management shall be documented.**
ALC_PAM.1.9C	**The patch management policies shall describe the mandatory structure and content of the S-IAR.**
ALC_PAM.1.10C	**Each type of documentation used to record decisions in the patch management process shall be documented.**
ALC_PAM.1.11C	**The patch management policies shall describe the mandatory content of patch release notes.**
ALC_PAM.1.12C	**The patch management policies shall describe the mandatory content for the guidance documents which have to be fulfilled to support the installation of the patch.**
ALC_PAM.1.13C	**The patch management policies shall describe the mandatory procedures during patch release.**
ALC_PAM.1.14C	**The patch management policies shall contain rules in which case the evaluation facility has to perform additional tests before the patch is released.**
ALC_PAM.1.15C	**The patch management policies shall describe how each of the patch management Security Objectives for the Operational Environment are fulfilled until the end of life of the TOE.**
	Evaluator action elements:
ALC_PAM.1.1E	**The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.**

5.2.2 (FOR CEM/18045) WORK UNITS FOR ALC_PAM

The following list is not formatted as expected for CEM/18045.

ALC_PAM.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ALC_PAM.1.1C The developer's patch management policies what is the criteria used for the decision that a patch has to be released.

ALC_PAM.1-1 The evaluator shall check for the definition of criteria and check for the implementation as a policy.

- List of criteria:
  - Complexity of backports
  - Operational stability, development teams is able to estimate effect for operational stability
  - security impact
  - customer impact (i.e. practical problems, theoretical problems)
  - timely impact, i.e. customer expect patches each quarter of a year, i.e. also minor security problems have to be fixed

ALC_PAM.1-2 The evaluator shall check the status of the implementation of the policies for patch releases and verify if the policies for patch releases have the same level of detail as other developer evidences provided for ALC.

ALC_PAM.1-3 The evaluator shall verify if the following mandatory policy content was implemented as policy:

- responsible roles for the final decision to release a patch
- unique label for each patch to identify all release items

ALC_PAM.1.2C The Security Target shall contain the estimated end-of-life of the TOE.

ALC_PAM.1-4 The evaluator shall check for estimated TOE end-of-life information in the ST and for estimated TOE end-of-life information in user information material, i.e. the guidance, release notes, product (support) website.

ALC_PAM.1.3C The developer's patch management policies shall describe how to self-assess the security relevance of a patch (i.e. Security Impact Analysis Report, S-IAR) and which procedures have to apply due to which assessment result categories.

ALC_PAM.1-6 The evaluator shall check if at least two assessment result categories were defined for patch management.

- For example
  - Category 1: no patch required
  - Category 2: patch is required

ALC_PAM.1.4C The developer's patch management policies shall describe how to update the evidence documentation used in the base evaluation.

ALC_PAM.1-7 The evaluator shall check if the patch management policies describe how to update the evidence documentation in a consistent way with the evaluation assurance level.

ALC_PAM.1.5C The developer's patch management policies shall describe how unhandled (potential) flaws are documented.

ALC_PAM.1-8 The evaluator shall check the status of the implementation of the unhandled flaw documentation policy.

ALC_PAM.1.6C The developer's patch management policies shall describe which organizational role (or group) is responsible for the patch development.

ALC_PAM.1-9 The evaluator shall check the organizational definitions and responsibilities of all roles involved in the patch development process.

- Examples for definitions of patch development responsibilities
  - patch development tasks as part of RACI matrix
  - patch development tasks as function of a product development team

ALC_PAM.1.7C The developer's patch management policies shall describe which policies have to be applied until the end-of-life of the TOE during the patch management.

ALC_PAM.1-10 The evaluator shall check for the implementation of internal policies that have to be applied during TOE maintenance.

- Examples for policies regarding 3rd Party Libraries:
  - update only libraries that are still supported as well
  - backport latest changes to used library version
  - upgrade to latest library version

ALC_PAM.1.8C Each tool used for the patch management shall be documented.

ALC_PAM.1-11 The evaluator shall check the list of tools the developer uses for patch management.

ALC_PAM.1.9C The patch management policies shall describe the mandatory structure and content of the S-IAR.

ALC_PAM.1-12 The evaluator shall check the format of the S-IAR used by the developer.

- Mandatory elements of the S-IAR are
  - Description how to use bug tracker information for the S-IAR
  - Security relevance criteria: e.g. remote execution, only product type specific
  - Category criteria: e.g. CWE (common weakness enumeration)

ALC_PAM.1.10C Each type of documentation used to record decisions in the patch management process shall be documented.

ALC_PAM.1-13 The evaluator shall check the patch management policies describe how to record decisions.

ALC_PAM.1.11C The patch management policies shall describe the mandatory content of patch release notes.

ALC_PAM.1-14 The evaluator shall check if the patch management policies contain the elements that shall be mandatory in a developer's patch release notes.

ALC_PAM.1.12C The patch management policies shall describe the mandatory content for the guidance documents which have to be fulfilled to support the installation of the patch.

ALC_PAM.1-15 The evaluator shall check the developer's patch management policies for release note or update guidance requirements.

- e.g. checklist for steps to describe during patch installation

ALC_PAM.1.13C The patch management policies shall describe the mandatory procedures during patch release.

ALC_PAM.1-16 The evaluator shall check the developer's patch management policies for mandatory patch release procedures.

- Examples
    - procedure steps for (patch) release: Build -> QA test -> HW integration test -> Release
    - process definition should contain the failure of test/validation steps and how to handle these cases

ALC_PAM.1.14C The patch management policies shall contain rules in which case the evaluation facility has to perform additional tests before the patch is released.

ALC_PAM.1-17 The evaluator shall check the developer's patch management policies for rules that require testing of the evaluation facility.

- e.g. ruleset for different acting roles in the (patch) release procedure
- relevant roles: development, QA department, product owner, etc.

ALC_PAM.1.15C The patch management policies shall describe how each of the patch management Security Objectives for the Operational Environment are fulfilled until the end-of-life of the TOE.

ALC_PAM.1-18 The evaluator shall check the developer's patch management policies for a description of how the Patch Management Security Objectives are fulfilled.

ALC_PAM.1-19 The evaluator shall verify if the patch management processes address the following requirements:

- How the cryptographic keys involved in signing and/or distributing patches are generated and managed during its entire life-cycle so they have enough strength to protect the authenticity of the updates.
    - How the cryptographic keys are created
    - How the cryptographic keys are securely stored
    - The process for revocation and loading of a new cryptographic key if it is compromised
    - How the cryptographic keys are destroyed or archived at the end-of-life of the product
- The process for approving, signing and releasing new updates in a secure and audited environment.
    - Who approves the releasing of updates
    - Who can access the cryptographic keys used for signing updates
    - How the update is moved from the development environment to the signing environment so that it is not tampered
    - How this process generates logs
    - How this logs are audited
- How the user is notified of the availability of a new patch due to a security issue
    - Through email
    - Through automatic checks to a website handled by the product
- How the patches are made available and securely distributed to the end user
    - Uploaded to a website by the developer and automatically downloaded by the TOE by using an appropriate and declared security protocol
- Sent to the end-user using delivery services and providing installation instructions where administrator rights must be implemented using password/authentication codes and/or cryptographic authentication techniques

Implied evaluator action

ALC_PAM.1.2D The developer shall self-assess and confirm the application of existing policies on a regular basis saving records of its application.

ALC_PAM.1-20 The evaluator shall verify evidences of developer's self-assessment procedures.

ALC_PAM.1-21 The evaluator shall check if results or evidences for the self-assessment can be presented.

- For example
  - publication of developer self-declaration with reference to product certification ID
  - internal (or external) audit report, in general annual audit

ALC_PAM.1-22 The evaluator shall check if existing unhandled flaw documentation exists and if these fulfil the policy requirements.

ALC_PAM.1-23 The evaluator shall check if decisions in the patch management process were documented.

ALC_PAM.1-24 The evaluator shall check the patch release notes for the content elements required by the patch management policies.

ALC_PAM.1-25 The evaluator shall select and examine a sample of evidence covering each type of relevant event (e.g. signing logs, approval of updates, S-IAR, fulfilled checklists, bug tracker evidence…) to confirm that all operations of the patch management policies and procedures are carried out in line with the documentation. The evaluator may choose to sample the evidence.

- For guidance on sampling see ISO/IEC 18045, A.2, Sampling.
- Further confidence in the correct operation of the patch management policies and procedures may be established by means of interviews with selected development staff. Note that such interviews should complement rather than replace the examination of documentary evidence, and may not be necessary if the documentary evidence alone satisfies the requirement.
- The evaluator may visit the development site in support of this activity.
- For guidance on site visits see ISO/IEC 18045, A.4, Site Visits.

Implied evaluator action

ALC_PAM.1.3D The developer shall provide security patches using the defined policies and procedures at least until the estimated end-of-life of the TOE.

ALC_PAM.1-26 The evaluator shall examine aspects of the patch management procedure to determine that the patch management procedures are being used.

- In addition to examination of the procedures themselves, the evaluator seeks some assurance that they are applied in practise. Some possible approaches are:
  - a visit to the development site(s) where practical application of the procedures may be observed;
  - observing that the process is applied in practise when the evaluator obtains new updates solving the vulnerabilities found during the Vulnerability Analysis.
- If a Site Visit is already included in the evaluation plan, the evaluator shall apply option (a) to check that the processes are applied in practice.
- For guidance on site visits see A.4, Site Visits.

6 RECOMMENDATION TO CERTIFICATION BODIES (CBS)

This chapter gives recommendation to certification bodies who operate ISO/IEC 15408-based certification schemes, also called the Common Criteria schemes.

The previous chapters defined functional and assurance enhancements that support the establishment of an advanced trust model between the different parties, i.e. the developer, the evaluation facility and the certification body.

Different schemes might have another understanding of these trust models therefore it is not expected that all schemes come to the same conclusion. This document lists with recommendation to certification bodies to close the gap between certified and secure products (TOEs).

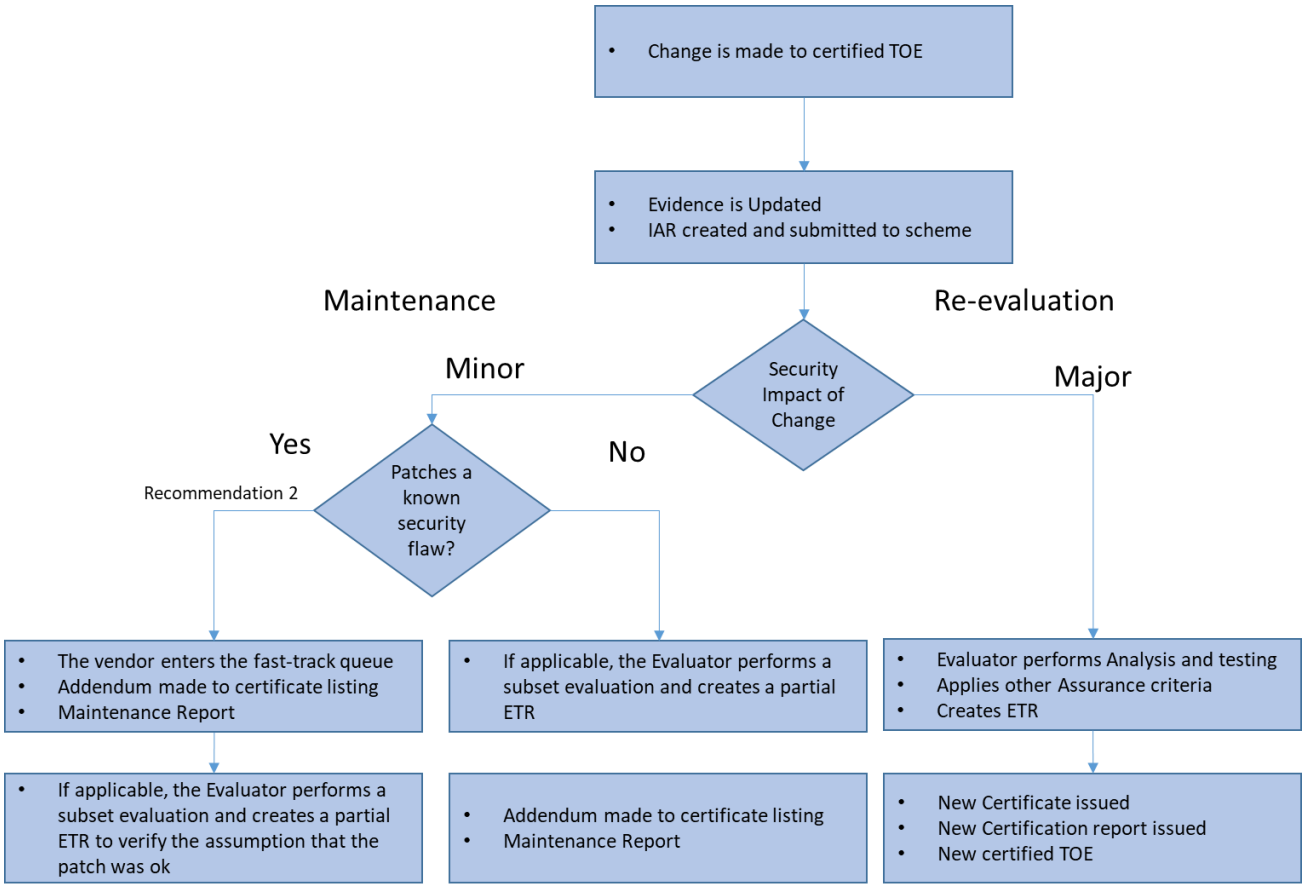


Figure 5 Maintenance and re-evaluation (based on CCRA model)

Recommendation 1: Certification bodies should support re-use of evaluation results.

Developers who claim ALC_PAM will be able to at most immediately provide evidence to future re-certifications.

For example, the CCRA already allows to re-use evaluation results. But compared to existing practices ALC_PAM encourages developers to continuously generate these evidences during product maintenance.

To support fast and plannable re-certifications the certification body should also publish scheme policies that describe how and which evaluation results can be re-used in future re-

certifications. The combination of fresh evidence from latest patch development and re-use of previous evaluation results can support efficient re-certifications.

Recommendation 2: Certification bodies should provide a fast-track re-certification for security flaws.

Developers that implement the TOE Security Objectives (with corresponding elements from FPT_PAM) and Operational Environment Security Objectives which are aligned with the requirements from ALC_PAM should be allowed to access to fast-track re-certification programs by certification bodies. Those fast-track re-certifications should only be allowed for security flaws. Certification bodies should create a fast-track priority queue for processing re-certifications.

Developers are still required to evaluate the changes with an ITSEF under the certification body, but this evaluation should start without previous authorization by the CB.

Recommendation 3: Certification bodies should provide templates to analyse the impact of changes of a patch.

This documents also provides a template as starting point for CBs in Annex B.

Recommendation 4: Certification bodies should trust by default developers in order to harmonize security and certification.

This way updates addressing security flaws should be accepted by default because of the additional assurance resulting from ALC_PAM. The patched version should be considered under the maintenance report just with the ITSEF criteria.

Recommendation 5: Certification bodies should put penalties if developers do not follow the published rules.

As part of the certificate monitoring the certification body should put penalties, e.g. suspension of the certificate.

If developers are not going under evaluation in the defined timeframe or providing incorrect or vulnerable evidence to the ITSEF the certification body should also put penalties.

If a fast-track process is available developers should be denied access to this if they do not follow the published rules.

7 ANNEX: TEMPLATE FOR THE SECURITY IMPACT ANALYSIS REPORT (S-IAR)

Definitions of the different IAR types:

IAR = changes since last evaluated version, how does it affect the product evidence, delta perspective, document for lab and CB

S-IAR (Secure IAR) = internal document, perspective of the development, impact of changes to "security" and to TSF

7.1 S-IAR STRUCTURE TEMPLATE

Flaw or issue	**Description**	**Options for mitigation** e.g. change product/TOE, new guideline (special configuration)	**Related Change** Relation to CM	**Security impact** e.g. security bug-fix, functional correction, new feature
includes reference to bug tracking ID	*security relevance consideration, e.g. remote code execution, or only product type specific flaw* *category criteria: e.g. CWE (common weakness enumeration)*			

8 ANNEX: ALC_PAM POLICY EXAMPLES

The patch management of CC product/TOE developers has to implement some core principles and policies which follow from the requirements in ALC_PAM. This Annex gives an example of an outline of such a policy.

The policy has to include these aspects:

1. Monitoring of flaws and issues
2. S-IAR result categories
3. Assessment of flaws and issues, or Patch integration (or change) criteria
4. Policies to maintain CC/ALC development process
5. Policies for patch releases
6. Updated Guidance
7. Self-assessment and confirmation of the application of existing policies the on a regular basis.

1. Monitoring of flaws and issues

Developers have to monitor all different types of sources to monitor flaws and issues. All security relevant flaws and issues have to be analysed by the developer. The result has to be documented in the S-IAR report.

The roles and responsibilities for the different sources and the assessment has to be defined.

Example

The following flaw and issue sources are monitored:

- security@company E-Mail inbox
- internally detected flaws, e.g. by QA team
- flaw and issues reported by customers
- 3rd party library related flaws, e.g. open source libraries

The product security officer is responsible for the monitoring of incoming candidate flaws and issues.

2. S-IAR result categories

At least two categories have to be defined, i.e. first category no patch is required, second category for patch is required.

But developers are encouraged to define the categories which describe their business perspective, i.e. specific policies based on customer contracts or based on requirements for regulated use-cases.

Example

In the following, the definition for the two types of categories is given:

> Category 1 “internal QA”: e.g. functional corrections not affecting the TSF, security bugfixes that do not require for an update of the ADV evidences. **If the whole patch**

has been qualified for this category the testing of the patch has to be done by the QA team.

Category 2 "re-certification": e.g. functional correction or security update of the TSF which requires for updates of the ADV evidences. **If at least one change is quailed for this category the developer has to start the re-certification immediately.**

3. Assessment of flaws and issues

For the product (or TOE) lifetime the developer has to define his internal criteria to assess flaws and issues.

The criteria have to be used to decide if the flaw remediation will be one of the following types:

- Technical correction, i.e. release of a patch, or
- Publication of additional guidance, i.e. configuration or procedural workaround, or
- Recommendation to change the product setup, e.g. the installation of technical compensating countermeasures (e.g. additional firewall packet filter)

The developer is able to handle multiple flaws by clustering the required changes into one single patch.

The handling of the flaws has to be documented as part of the S-IAR.

Example

The developer defines a policy that uses e.g. the following criteria:

- Complexity of backports
- Operational stability, development teams is able to estimate effect for operational stability
- Security impact
- Customer impact (i.e. practical problems, theoretical problems)
- Timely impact, i.e. customer expect patches each quarter of a year, i.e. also minor security problems have to be fixed
- 3rd-party library related flaws and issues:
    - update only libraries that are still supported as well, or
    - backport latest changes to used library version, or
    - upgrade to latest library version

The product security officer is responsible for the assessment of incoming candidate flaws and issues.

4. Policies to maintain CC/ALC development process

The developer has to define how the CC/ALC development process is maintained during the product (TOE) lifetime.

Note: The baseline certification has shown the developer's capability to develop and produce a product according the CC requirements. This policy aspect requires the developer to setup maintenance procedures how all CC evidences are generated in parallel to the default product (TOE) maintenance.

Example

The product security officer is responsible to maintain the evaluation input like design documents (ADV). The QA team is responsible to re-run the developer tests (ATE_FUN).

5. Policies for patch releases

The policies below have to be followed before the next patch is released:

- definition of internal release stages and policies
- process definition with failure/cancel criteria for validation tests and follow-up procedures for these cases
- definition of cases if the external evaluation facility has to be contacted and to perform additional tests before patch release

  Note: These cases do not include the re-certification scenario but are related to involvement of the evaluation facility without (full) re-certification.
- definition of ruleset for roles (e.g. development, QA department, product owner) in the patch release process
- responsible role for the final patch release decision
- unique label for each patch to identify all release items

The policies can differentiate between the different S-IAR result categories.

6. Updated Guidance

For each patch release the developer has to verify if the guidance has to be updated. The details have to be defined in a policy. The following reasons have to be considered for the policy definition:

- exceptions to let flaws or issues unhandled but guidance how to mitigate these flaws, e.g. with procedural changes
- update or installation pre-conditions, e.g. hardware requirements, have to documented

7. Self-assessment and confirmation of the application of these policies

The developer has to show in periodic manner that the policies are applied. This has to be shown by (partly) published results of the self-assessment.

The commitment of the developer has to be documented as part of the policy.

Example

The summary of the results of the annual self-assessment is published on the developer's website with reference to the related product certification IDs. The self-assessment is supported by an external audit team leader to ensure independence from the development team's perspective.