Commercial Product Assurance for Smart Metering Equipment

Updates to the v1.2 GSME and ESME Security Characteristics

Approved by the SEC Security Sub-Committee (SSC)

Version 0.9 – to be incorporated into Version 1.3 of the ESME and GSME Security Characteristics on 26th September 2019

CONTENTS

OVERVIEW

The SC updates were initiated as lessons learned from observations by the CPA Authority in operating the CPA Scheme in practice and have been consulted on with BEIS, SSC and industry. The general themes include:

1. Clarifications to SC wording where it is potentially open to interpretation. Updated wording reflects interpretations ultimately followed in CPA evaluations.
2. Some areas of product security functionality exist that could benefit from being covered by more meaningful assurance activities during a CPA evaluation.
3. Alignment with wording changes already applied to the updated Comms Hub Security Characteristic (as part of the v1.2 to v1.3 change).

The following sections describe the changes approved by the SSC for the current set of detailed requirements in the GSME and ESME Security Characteristics (nearly all requirements are identical in these two SCs), along with the associated rationale in each instance.

The tracked changes and highlighting explains the development and subsequent amendment in response to industry comments.

[Page intentionally blank]

APPROVED CHANGES TO SC WORDING

DEV.M926: Protected software environment (CHANGE)

Change summary:

Improve clarity of SC wording and remove potential ambiguities, based on observations arising from CPA evaluations.

Rationale for changes:

In this instance, the wording updates target the following areas in this SC requirement:

- Freshen MISRA-related wording
- Stack protection expectations in product
- Assessment of third-party libraries, especially where that software is helping the product to meet one or more SC requirements.

Specific changes:

SC wording change: (modified/new wording change-marked)

DEV.M926: Protected software environment

This mitigation is required to counter exploitation of a software implementation/logic error

At Foundation Grade the product **is required to** implement software protection measures as part of the design process.

The device design information shall describe the process environment in the device in order to allow the evaluator to identify any defensive or robustness mechanisms provided by the platform or OS.

The developer shall provide evidence to demonstrate device firmware compliance with MISRA ~~2012~~ rules for C (or equivalent for the target language), by application of an appropriately configured static analysis tool. Where the target language is C, MISRA 2012 or later must be used (and, where supported by the static analysis tool, include the additional rules introduced in MISRA:C 2012 Amendment 1). Where the target language is not C, the developer shall demonstrate equivalence by mapping each rule onto the equivalent criterion for the target language, accompanied by the method of demonstrating that the criterion has been met.

The developer shall provide a rationale for how the device firmware protects against stack and heap corruption. Stack protection is typically expected to be provided via a compiler option that uses canaries to protect against a function's return address being overwritten. ~~Where~~ Whether or not such a compiler option is ~~not available, an alternative stack protection mechanism may be used. Either way~~, used the stack protection implementation needs to comply with the "Stack Protection Expectations" appendix in this document.

The developer shall demonstrate that they review all device firmware against a checklist of security flaws, including known vulnerabilities, in other versions of the product or its components (e.g. where 3rd party software/hardware is used), and known vulnerabilities in similar devices. Note: Aspects of this requirement should be covered by the developer's ongoing Build Standard compliance obligations.

~~Note: Device firmware in all contexts of this requirement includes any third-party libraries that are relied on by the product to meet one or more requirements in this Security Characteristic.~~

The SC requirement wording changes regarding stack protection expectations will be accompanied by the introduction of a new appendix: DEV.M926: Protected software environment (CHANGE)

Appendix E – Stack Protection Expectations

A sufficiently robust level of stack protection is expected that provides the following features as a minimum (which are typically on a par with those provided via a stack protection compiler option):

- Detect corruption of a function return address before the function returns to that address. i.e. The corrupted return address will not be used, and appropriate remediation action will be performed instead, such as rebooting the product into a good known state.
- ~~Detect corruption of other variables in the stack frame, again before the function completes.~~
- Be present in functions that have one or more arrays declared in the function's stack frame (this includes third party library code within the same runtime environment as the application code).
- If canaries are used to detect corruption, then:
  - The size of the canaries must be at least that of a memory pointer for the device's platform (e.g. canary size would need to be at least 32 bits for a 32-bit architectural)
  - The values used for the canaries must vary across different devices in a non-predictable manner (not necessarily reliant on the same RNG function used to generate cryptographic key material).
  - Additionally, the canary value should also change in a specific device each time the product (re)boots, though this is not mandatory.

Note: Although it would be desirable to detect overflow of one stack variable into another, this is not mandatory.

Update 25/02/2019 (w.r.t. 08/10/2018 wording):

- Slight relaxation of wording for MISRA 2012 amendment 1 – intention for the additional rules to be checked, where supported by standard industry static analysis tooling, but acceptance that the such static tooling may not yet cover all those rules.
- Reworded stack protection requirement, which includes a relaxation of the sub-requirement to also protect against overflow of one local variable on the stack into another.
- Last proposed paragraph of M926 has been questioned as having potentially significant impact on vendors. The new text relates to libraries that help the product meet other **requirements** in the SC, which are important from a security point of view and the default interpretation in CPA evaluations to date. This said, if vendors can indicate where the choice of third-party libraries (required to help the product meet SC requirements) might be problematic, this can be discussed.
- Query that a requirement for MISRA compliance is disproportionate to the security risks – no rationale to substantiate this view has been provided and the MISRA compliance requirement was agreed with industry back in 2014.

Update 28/02/2019:

- MISRA compliance in third party libraries proving a stumbling point with a number of parties – it is therefore now proposed NOT to add any new wording here at this time and instead open a longer-term action to develop appropriate new wording for future SC versions or future lab guidance documentation. NCSC view here remains that where the product is relying heavily on a third-party library to prevent product security from being materially compromised, the associated code should not have been hacked together. Hopefully, this is never the case, but getting some measure of MISRA compliance in such libraries acts as a useful smoke test. Having said this, industry concerns are acknowledged that choices for third-party libraries may be limited in some circumstances.
- Slight tweaks made to stack protection expectation wording, including a clarification that the product's main RNG does not necessarily need to be used for canary generation.
- Suggestion that potential impact on product w.r.t. new MISRA 2012 Amendment 1 compliance requirement can be discussed in start-up meeting for new assurance work (AM, evaluation or re-evaluation).

Update 14/06/2019:

- Amendment to wording that might have implied compiler stack protection is mandated option (updated wording is yellow-highlighted).

DEV.M946: Clock synchronisation (CHANGE)

Change summary:

Adjust wording to be compliant with multiple versions of GBCS.

Rationale for changes:

The wording was originally written to be compliant with GBCS v1.0, but aspects of the wording are considered to be mismatched to the relevant Set Clock / Clock sync functionality described in versions of GBCS later than v1.0.

Specific changes:

SC wording change: (modified wording changed marked)

DEV.M946: Clock synchronisation

This mitigation is required to counter tampering with the device clock

At Foundation Grade the product **is required to** prevent unauthorised modifications to the device clock.

The device shall synchronise time with the Communications Hub as defined in [d, 9], on receipt of a valid Set Clock command and also periodically.

On receipt of a valid Set Clock command, the time shall be retrieved from the Communications Hub. If the time retrieved is within the tolerance specified in the command, the device shall set its time to that retrieved from the Communications Hub. If it is outside the specified tolerance, the time status shall be marked as Unreliable, and ~~an alert sent as defined in [d, 9]~~ a response or alert (depending on the version of [d] GBCS being complied with) sent according to the specific requirements of [d, 9].

Additionally, once every 24 hours (but no more frequently under normal operating conditions) a synchronisation is attempted. If the attempt to retrieve a valid time fails, the synchronisation will be subsequently attempted every 30 minutes until a valid time value has been retrieved according to the specific requirements of [d, 9]. If the result is that the time retrieved from the Communications Hub is not more than 10 seconds different from the device's current time, the device shall set its time to that retrieved from the Communications Hub and mark the time status as Reliable. If it is more than 10 seconds different from the device's current time, the time status shall be marked as Unreliable and a Security Log entry shall be recorded, and an alert sent as defined in [d, 9]. Note: Depending on the version of [d] being complied with, the Security Log entry and alert actions might only be needed if the time status has changed to be Unreliable.

Design information shall identify any interface through which the device clock's time can be modified and how it is ensured that no unauthorised modifications can be made.

Update 25/02/2019:

- Amendment to wording to cover wording variations across different GBCS versions.
- Taken opportunity to include requirement to retry the clock resync if failure occurs.

Update 28/02/2019:

- Attempts to resync every 30-minutes queried. Follow-up: Slight amendment so it's clear that this is still to meet GBCS requirements.

DEV.2.M847: Minimise interfaces (CHANGE)

Change summary:

Improve clarity of SC wording, based on observations arising from CPA evaluations.

Rationale for changes:

In this instance, the proposed wording updates target vendor and lab expectations relating to any non-GBCS interfaces that may be available in the product.

Specific changes:

SC wording change: (modified wording changed marked)

DEV.2.M847: Minimise interfaces

This mitigation is required to counter exploitation of a non-operational interface through crafted input

This mitigation is required to counter exploitation of an operational interface through crafted input

At Foundation Grade the product **is required to** ensure that only necessary protocols and services are available on the device.

~~If there is any additional functionality provided in the device beyond that required to meet the functional requirements detailed in [b], [d] and [e], the developers must provide the evaluator with design documentation and a rationale to demonstrate that it doesn't impact the security requirements in this Security Characteristic.~~

Where a device provides additional functionality, beyond that required to meet the functional requirements detailed in [b], [d] and [e], via additional protocols and services, the developers shall provide details of the functionality with an associated analysis that clearly indicate where security impacting functionality can occur. Where such additional functionality is present and has the potential to be security impacting, its unauthorised use shall be protected against using security mechanisms at least as strong as those in [d] that protect against unauthorised use of critical commands, using the same RBAC model. As a guide, "security impacting functionality" here is that functionality that would have the same material impact as a GBCS "critical command" (e.g. with the SME.C.C categorisation).

~~At time of writing, there are discussions about meters generally being provided with additional functionality to that specified in [b], [d] and [e] to support triage and redeployment scenarios for meters. Such functionality, along with the specifications for any security protections required to protect against its unauthorised use, may be captured in additional documentation (to supplement [b], [d] and [e]) and be referenced in future versions of this Security Characteristic.~~

Update 25/02/2019:

- Removal of evaluator expectations (which are really VER activities).
- Added a note to capture possible future updates to the SC in acknowledgement of current discussions around support for additional interface functionality to support triage and re-deployment activities.

Update 28/02/2019:

- Removed suggested placeholder text as should now be covered by the preceding text and the outcomes of the triage discussion.

DEV.4.M138: State the Security Strength required for random numbers (CHANGE)

Change summary:

Improve clarity of SC wording and alignment with associated Comms Hub Security Characteristic requirement.

Rationale for changes:

Self-explanatory.

Specific changes:

SC wording (and title) change: (modified wording change-marked)

DEV.4.M134~~8~~: State raw entropy requirements ~~the Security Strength required for random numbers~~

This mitigation is required to counter prediction of randomly generated values due to a weak Entropy Source

At Foundation Grade the product **is required to** have clearly defined entropy requirements for all operational random number generation ~~employ an Entropy Source of sufficient Security Strength for all random number generation required in the operation of the product~~.

Device design information shall specify all cryptographic keys employed by the device (including any that are not required for normal operation), and their method of generation or installation.

The developer must state how much raw entropy is required from the product's entropy source (for example, which is used to reseed the PRNG), ~~the Security Strength required of their Entropy Source~~ based on analysis of all security-related random numbers used in the device, including any generated keys. It shall be at least 128 bits as required by the elliptic curve-based asymmetric mechanisms using the P-256 curve, assuming no other security features of the device have significant entropy requirements ~~require greater Security Strength~~.

Update 25/02/2019:

- Minor tweaks to highlight that entropy requirements relate to security aspects of the device.

Update 28/02/2019:

- Question asked about whether the RNG only applies to generation of cryptographic key material. Follow-up: The only other mention in the SC of security functionality having a random element is that associated with the canaries, and this has now been clarified as not necessarily needing a strong RNG.

DEV.4.M140: Smooth output of Entropy Source with approved PRNG (CHANGE)

Change summary:

Improve clarity of SC wording and alignment with associated Comms Hub Security Characteristic requirement. Additionally, add wording to ensure that power loss and power cycling scenarios gets covered in the RNG analysis.

Rationale for changes:

Alignment with associated Comms Hub Security Characteristic requirement self-explanatory. Power loss/cycling wording aspects proposed in line with current CPA evaluation policy.

Specific changes:

SC wording change: (modified wording change-marked)

DEV.4.M140: Smooth output of Entropy Source with approved PRNG

This mitigation is required to counter prediction of randomly generated values due to insufficient raw entropy reaching the PRNG ~~a weak PRNG~~

At Foundation Grade the product **is required to** ensure that all random data used originates from a PRNG that is seeded by one or more entropy sources. ~~employ an RNG of sufficient Security Strength for all random number generation required in the operation of the product.~~

The device design information shall include a description of how the output from all Entropy Sources and their associated PRNG can generate sufficient entropy for all keys and random numbers used in the product's regular operation including, where applicable, demonstrations of conformance to relevant standards such as the NIST SP800-90 series.

Note: The PRNG implementation may be based on standards other than the NIST SP800-90 series if it is believed they provide an equivalent level of security; if so, the rationale for this will need to have been agreed with the CPA Authority beforehand.

Device design information shall also describe how the entropy in PRNG's state will persist for any power-loss / device restart scenarios.

For more details about which key pairs need to be generated by the product ~~on RNG requirements~~, see ~~the description of key pair generation in~~ [d, 4].

Update 25/02/2019:

- No changes to suggested wording.
- Query received about what other evidence might be permissible. Given the variation possible in product implementations, guidance on a case-by-case basis currently seems the best way forward, whilst retaining the option to publish more detailed supporting guidance in future.

DEV.4.M141: Reseed PRNG as required (CHANGE)

Change summary:

Introduce a new *recommendation* to reseed periodically.

Rationale for changes:

This is in light of most CPA-certified devices having the capability to reseed an onboard PRNG (such as a NISTapproved DRBG), the usage of which would provide an additional mitigation against device's RNG implementation failing during the product's envisaged ~15+ year operational lifetime. The minimum figure of a month is suggested as this seems to be unlikely to significantly impact product functionality, though other alternative figures could be discussed.

Specific changes:

SC wording change: (modified wording change-marked, with further revisions yellow-highlighted)

DEV.4.M141: Reseed PRNG as required

This mitigation is required to counter prediction of randomly generated values due to a weak PRNG

At Foundation Grade the product **is required to** follow an approved reseeding methodology.

For instance, if a SP 800-90 series compliant DRBG is being used, the product must implement the reseed mechanism recommended by the relevant SP 800-90 series document.

Reseeds of the PRNG should be performed at least once a month or immediately prior to generating a new random number (e.g. as part of generating a new key), provided, in the latter case, this does not significantly impact on the time taken to generate the random number.

Update 25/02/2019:

- Added example of an "approved" reseed mechanism.
- Have also added alternative option for reseeding.
- Note: period of 1 month has been questioned. However, nothing tangible has been offered in terms of what material impact such functionality would have on the product:
  - Performing such an operation once a month would appear to have pretty zero impact on a product that already has timers present and, in many cases will have the necessary reseed functionality present (e.g. as part of a CAVP validated solution), but unused in practice.
  - Even the sleepy, low-power GSME would seem to lose little of its available resources to performing this function once a month.
  - NCSC consider that meeting this recommendation would improve the product's defensive stance.

Update 16/04/2019:

- Minor change to reflect that if the product is using a SP 800-90 series DRBG it must (not should) be using the appropriate reseed mechanism.

DEV.4.M290: Employ an approved Entropy Source (CHANGE)

Change summary:

Improve clarity of SC wording, based on observations arising from CPA evaluations.

Rationale for changes:

Use of improvised noise sources were noted in one or more evaluations in which the level of analysis and associated assurance activities were unclear to the associated vendor and lab.

Specific changes:

SC wording change: (modified wording change-marked)

DEV.4.M290: Employ an approved Entropy Source

This mitigation is required to counter prediction of randomly generated values due to a weak Entropy Source

At Foundation Grade the product **is required to** generate random bits using an Entropy Source whose entropy generation capability is understood.

The developer must provide a detailed description of the Entropy Source used, giving evidence that it can generate sufficient entropy for each use of random numbers in the device, including an estimate of entropy per bit. The Entropy Source should be implemented according to guidance in relevant standards such as the NIST SP800-90 series.

If a hardware noise source is used, then the manufacturer's name, the part numbers and details of how this source is integrated into the product must be supplied. If a software Entropy Source is employed, the API calls used must be provided. Where appropriate, details must be given of how the outputs of multiple Entropy Sources are combined.

The device design information shall include an analysis of each noise source used for generating cryptographic keys, detailing the amount of entropy considered ~~believed~~ to be obtained from this source. This analysis should be supported by relevant datasheets, API specifications and results from the developer testing, as appropriate.

Where devices or functions are used that are not dedicated noise sources (for instance, A-D converters), analysis will additionally need to demonstrate that the improvised device or function will reliably provide a stated level of entropy for the operational environments the product may be deployed in. Factors that should be considered vary on the type of improvised noise source but could for instance involve (a) differences in temperature and humidity between the test lab and operational environment, (b) how predictable any input sampled by the improvised device might be in practice.

Important: Entropy measurements are performed on the raw data sampled, i.e. before any subsequent processing of that data that could result in the data being scrambled in a manner that distorts the measured entropy.

Update 25/02/2019:

- Bit more of a steer given about what analysis might be appropriate for improvised devices.
- Query about whether entropy levels from the output of a Von Neumann filtering stage can be stated – yes that option is possible, because no scrambling is involved (just discarding of bits).
- Query that a third-party provider of a noise source solution might not share details, such as raw entropy per bit levels – this has not been a significant issue in most evaluations to date.
- Query about what approved solutions already exist – unfortunately NCSC cannot recommend specific solutions.

DEV.4.M853: Prevent unauthorised changes to future-dated actions (CHANGE)

Change summary:

Broaden scope of SC requirement with respect to checking future-dated actions.

Rationale for changes:

The current wording focusses on checking that a future dated action will still work correctly if the clock is changed, rather than considering more generally what guarantees that the future dated action takes place, this despite things *such as* changes to the clock.
It is also considered important to make such a change as a future dated action may be necessary to complete the actioning of a critical command.

Specific changes:

SC wording change: (modified wording change-marked)

DEV.4.M853: Protect ~~Prevent unauthorised changes to~~ future-dated actions

This mitigation is required to counter future-dated actions not being carried out at the specified time

This mitigation is required to counter unauthorised addition, modification or removal of a future action

At Foundation Grade the product **is required to** carry out relevant future-dated actions at the time specified ~~when the clock is updated~~.

The device design information will describe why a future dated command ~~can be guaranteed to~~ will execute at the future time ~~specified~~, according to [d, 9], regardless of other events that the device is expected to encounter that could conceivably impact when a future dated action may occur (such as the device rebooting or a clock change taking place).

When the device clock is updated it shall neither miss nor repeat actions previously stored for future action nor miss calendar-based events.

At Foundation Grade the product **is required to** ensure that only authentic messages can cause a future-dated command to be added.

The device shall ensure that a future-dated message can only be added by receipt of a message from a source that is authorised to send that message type.

At Foundation Grade the product **is required to** ensure that only authentic messages can cause a future-dated command to be deleted, replaced or modified.

The device shall ensure that a future-dated message can only be modified, replaced or cancelled by receipt of another message of the same type from a source that is authorised to send that message type, or on change of control of the device.

Update 25/02/2019:

- Wording amended to capture more obvious events that might impact the firing of future dated events.

Update 28/02/2019:

- Questions asked about how exact the future-dated time needs to be, with a thought during the meeting that “at the time specified” could be removed. Follow-up: On reflection, running the command when it is required (or as close to afterwards, depending on what the device is doing at the time) is an integral part of the requirement, so have tweaked wording and linked it with [d] requirements.

DEV.4.M911: Self-test of RNG (CHANGE)

Change summary:

Clarification that if a product implements its PRNG according to the NIST SP800-90 series, it will be required to perform self-checks on the PRNG output in accordance with those NIST requirements.

Rationale for changes:

There is a concern that the NIST requirements for these self-checks can be missed by developers implementing a NIST-approved DRBG from scratch.

Specific changes:

SC wording change: (modified wording change-marked)

DEV.4.M911: Self-test of RNG

This mitigation is required to counter prediction of randomly generated values due to a weak RNG

At Foundation Grade the product **is required to** carry out self-testing of RNG output.

The developer shall provide a description of the self-testing performed by the random number generator and why they consider ~~believe~~ the implemented tests are adequate. For clarity, the self-tests are expected to be applied to the output of the Entropy Source (checking the final RNG output shall be covered by Evaluation/Cryptocheck of the PRNG, under the VER requirements in this Security Characteristic).

Note: Where a NIST SP800-90 series compliant PRNG is used by the product, the self-tests required by these standards are expected to be implemented.

(Update 25/02/2019: No changes made to suggested updates [except for a minor wording change])

DEV.4.M951: Mutual authentication on the HAN (CHANGE)

Change summary:

Change rigor of lab evidence from 'DEV' to 'VER' – when demonstrating that the product will authenticate another device on the HAN before exchanging any application data with it.

Rationale for changes:

This is in recognition that this security requirement provides an important protection against eavesdropping on the HAN by less trusted devices (e.g. HAN devices that are not CPA-certified).

Specific changes:

SC wording change: (modified wording change-marked)

VER~~DEV~~.4.M951: Mutual authentication on the HAN

This mitigation is required to counter connecting an unauthorised device to the HAN
This mitigation is required to counter spoofing an existing HAN device
This mitigation is required to counter aberrant behaviour of another HAN device

At Foundation Grade the evaluator **will** confirm that the product ~~product~~ ~~**is required to**~~ authenticates a device on the HAN before securely exchanging information with it ~~(other than that required to complete the joining)~~.

The evaluator shall confirm that:

- the product will only successfully join to another HAN device according to the different scenarios permitted for the join to occur in [d, 13.7]
- the product will not attempt to join to the other HAN device in other scenarios such as (a) the other device is of a type that the product is not permitted to communicate with, (b) the other device's details are not in the product's Device Log, (c) the other device's details are mismatched with details in the product's Device Log and (d) the other device's key pair is mismatched with the security credentials held for that device in the product's Device Log.
- only once a successful join has occurred, will application data be exchanged between the product and the other HAN device, this data being encrypted using a symmetric key agreed by the two devices in accordance with [d, 13.7].

At Foundation Grade the evaluator **will** confirm that removal of the other device's entry from the product's Device Log will result in the encrypted link between the product and other HAN device being terminated (i.e. it will no longer be possible for application data to be exchanged between the two devices).

Update 25/02/2019:

- Add some suggested assurance activities for the evaluator
- Query that such testing may already be happening elsewhere – if this is the case and the nature of the testing + results can be shared with the evaluators and the evaluators can be satisfied that the evidence demonstrates conformance with this SC requirement, additional lab testing may not be needed.
- Query about changing DEV to VER rather than introducing a new VER as per other SC requirements – unclear in this instance what value gets added by retaining the DEV.

DEV.7.M894: Limit rate of UTRN validation attempts (CHANGE)

Change summary:

Change rigor of lab evidence from 'DEV' to 'VER' – when demonstrating that the product will rate limit attempts to process UTRNs.

Rationale for changes:

This is in recognition that rate limiting is an important security control protecting against unauthorised use of a critical command – critical from the viewpoint that a failure to rate-limit could result in non-authentic UTRNs being incorrectly accepted via some brute-force method.

Specific changes:

SC wording change: (modified wording change-marked)

VER~~DEV~~.7.M894: Limit rate of UTRN validation attempts

This mitigation is required to counter attempting to guess a valid UTRN

This mitigation is required to counter blocking UTRN acceptance by causing repeated validation failures by manual entry

At Foundation Grade the evaluator **will** confirm ~~product~~ ~~**is required to**~~ ~~limit~~ the rate limiting of unsuccessful UTRN validation attempts.

The mechanism for limiting the rate of validation attempts shall not result in a period of non-acceptance of longer than 60 minutes. UTRNs input by manual entry, either through the user interface or via a PPMID, during any period of non-acceptance shall be discarded without attempting to validate the UTRN.

Note: Rate-limiting via the PPMID ~~will~~ should be independent to that via the user interface. For instance, a temporary block on UTRN entry via PPMID ~~will~~ does not in itself cause UTRN entry via the user interface to also be blocked.

Update 25/02/2019:

- Query relating to a suggestion that SMETS wording implies independent rate limiting checks on the User Interface and HAN interface, but not on the PPMID interface. However, this argument seems irrelevant given that rate-limiting on the PPMID interface already appears to be present in meters already evaluated under CPA.

Update 28/02/2019:

- Relaxed requirement for independent rate limiting for PPMID versus Manual entry to recommendation.

VER.M846: Secure failure recovery (CHANGE)

Change summary:

Alignment of requirement text with the equivalent in the Comms Hub Security Characteristic.

Rationale for changes:

Self-explanatory.

Specific changes:

SC wording change: (modified wording change-marked)

VER.M846: Secure failure recovery

This mitigation is required to counter disruption of a device by electromagnetic interference
This mitigation is required to counter exploitation of a software implementation/logic error

At Foundation Grade the evaluator **will** attempt to induce failures and observe correct recovery behaviour.

The evaluator shall verify by testing that all of a representative sample of the recognised error conditions are correctly handled. This sample shall include error conditions that do not arise directly as a result of input failures (examples of such a test might be a failure of the power-up firmware integrity verification check or other self-test, or corruption of internal state values; test equipment such as an emulator may therefore be used to enable these tests). The sample shall also include tests of the device's ability to recover from a communications overload (i.e. messages arriving at a rate that exceeds the device's ability to process them), and of the device's ability to resist and/or recover from electromagnetic interference (such as electrostatic discharge).

The evaluator shall provide a rationale that the sample is sufficiently representative, based on the design information relating to error handling.

The evaluator shall also seek evidence ~~(performing additional tests where necessary)~~ that the risk of potentially exploitable bugs in product code (in particular code handling remote incoming messages) will be robustly mitigated against, for instance by one or more product features detecting anomalous code behaviour and responding with a controlled restart.

The recovery action(s) shall be executed only using code that has passed the start-up integrity check for the current execution (since the last reset or power-on). Code that has failed the start -up integrity check shall not be run.

Update 25/02/2019:

- Slight rewording of last paragraph to make more reader friendly.
- Query about what mechanisms need to be tested – those in the corresponding DEV.

Update 28/02/2019:

- Question over what constitutes a pass relating to how potentially exploitable bugs are mitigated. Follow-up: This has not been an issue in evaluations to date, including those for Comms Hubs in which equivalent wording has been placed into the same Security Characteristic. This said, a lab guidance paper might be useful to provide clarity here.
- Wording in last paragraph questioned regarding immutable. Follow-up: Such wording removed because, in practice, it is likely that any recovery actions will need to be performed by complex code, which will be updateable and hence integrity checked before being run. ***PTO***

Update 16/04/2019:

- Minor amendment to penultimate paragraph so as to also allow developer provided evidence to be reused by the lab (subject to lab checks on the developer test that would be appropriate for VER evidence, as per PPFGE).

VER.M946: Clock synchronisation (CHANGE)

Change summary:

1. Increase scope to also cover Set Clock and not just the automated 24-hour clock synchronisation.
2. Adjust wording to be compliant with multiple versions of GBCS.

Rationale for changes:

1. All circumstances resulting in clock synchronisation should be verified by the lab, thus now also including the Set Clock command (which it should be noted, is also a critical command).
2. Self-explanatory (see also rationale for proposed change to DEV.M946)

Specific changes:

SC wording changes: (modified wording change-marked)

VER.M946: Clock synchronisation

This mitigation is required to counter tampering with the device clock

This mitigation is required to counter vulnerabilities associated with significant clock inaccuracy

At Foundation Grade the evaluator **will** verify correct device behaviour in response to both a Set Clock command and the recurring 24-hour clock synchronisation event.

The evaluator shall confirm, possibly with the assistance of testing performed by the developer, that the device's clock synchronisation functionality complies with section 9 of the version of [d] that the device has been designed to comply with. More specifically, this confirmation will focus on checking that:

- the meter's time gets set to the value expected according to each of the different scenarios in [d, 9].
- the meter's time remains unchanged in all other scenarios in [d, 9].
- the meter's time status gets set to the value expected in all scenarios in [d, 9].
- any responses or alerts are constructed appropriately (containing the expected resultant time and time status where required) and sent to the appropriate destinations as required by the different scenarios in [d, 9].
- any Security Log entries are written as required by the different [d, 9] scenarios.

Additionally, the evaluator shall confirm that if the product fails to receive a response from the Comms Hub with a valid time value, or any response at all, during a recurring 24-hour clock synchronisation event, the meter will attempt to perform the clock synchronisation again, 30 minutes later.

At Foundation Grade the evaluator **will** verify that, if any other mechanisms exist to update the device time and time status, that these operate as specified and are protected from unauthorised use.

Update 25/02/2019

- Reworded in light of material differences noted between the different GBCS versions – essentially reworded to say test the different [d, 9] scenarios, checking that time and time status get changed / left alone as appropriate and the correct responses, alerts and security log entries happen when they should (relaying the correct information about time and time status, where required).
- Also added new wording to test any additional, non-GBCS, mechanisms to change the time.

Update 28/02/2019

- Some of the wording questioned as it appeared to be performing tests using a Comms Hub. Followup: Now clarified that the evaluators are to look at the meter's behaviour if it fails to receive a (valid) response from the CH.

VER.1.M347: Verify update mechanism (CHANGE)

Change summary:

Increased clarity of SC wording and increased emphasis on checking authentic firmware updates work when required to.

Rationale for changes:

Being able to successfully perform a firmware update is an important security control and an essential one for CPA-certified smart meters (e.g. enabling suppliers to tackle security and other weaknesses that may arise in a product over its envisaged ~15+ year operational lifetime).

Specific changes:

SC wording change: (modified wording change-marked, with further revisions yellow-highlighted)

VER.1.M347: Verify update mechanism

This mitigation is required to counter causing unauthorised activation of authentic firmware This mitigation is required to counter unauthorised modification to firmware in situ This mitigation is required to counter inability to load firmware updates required to patch security weaknesses

At Foundation Grade the evaluator **will** validate the developer's assertions regarding the suitability and security of their update process.

The evaluator shall confirm the following ~~validation shall include the following checks~~:

- once a complete firmware update image has been received, its cryptographic protection will be validated and, if any checks fail, this will result in the image being rejected such that it cannot subsequently be activated (note: the checks will involve validation of the image's protective signature as per requirements in [d], along with any additional cryptographic checks performed on the image),

- a successfully received firmware update image, cryptographically validated as per the previous point, will not be activated if any of the cryptographic validations required by [d] on the associated Activate Firmware command fail – this for both immediate and future dated firmware activation scenarios,

- similarly, a successfully received and cryptographically validated firmware update image will not be activated if the 'manufacturerImageHash' field in the Activate Firmware command does not match the hash in the firmware update image - this again for both immediate and future dated firmware activation scenarios,

- attempting to action an Activate Firmware command (either when the command has been received with no 'executionDateTime' specified or when it is time for a previously-received, future-dated command to be executed) will fail when there is no successfully received complete firmware update image – or one has been received but one or more cryptographic checks on that image have failed – and

- where a partially received firmware image – or a full image over which cryptographic checks have not been successfully performed – has been stored, this will not get activated if a device reboot occurs.

In addition to the above checks (that focus on ensuring a firmware update does not occur when not appropriate), the evaluator shall also confirm:

- the design for receiving and activating a firmware update, via authentic Distribute Firmware and Activate Firmware commands, is clearly documented and tested against by the developer, confirming that there are no obvious areas of uncertainty that could result in an unexpected failure to update the firmware,

~~- the activation of new firmware will thus perform reliably in an operational device (and will not for instance be impacted by heavy ZigBee traffic during the window in which the new firmware is being activated), and~~

- where a product does not incorporate anti-replay protection on the Activate Firmware message, product security is not undermined by a subsequent replaying of a valid Activate Firmware message (when used for either immediate or future-dated firmware activation).

- ~~the security of the download process to ensure authenticity and integrity of the update – atomicity of attempted updates: the update shall either completely succeed or completely fail~~
- ~~the security of the activation process (including a check that the firmware version in the activation request corresponds to a version previously received and whose integrity and authenticity has been checked)~~
- ~~confirmation that the secure process applies to all firmware that can be updated.~~

Update 25/02/2019:

- Reworded to make evaluator assurance activities clearer: two sections, one to check firmware updates are rejected where required and the other to increase confidence that they will happen when required.
- Also added additional check to cover instances in which product does not incorporate replay protection on the Activate Firmware command, which GBCS does not mandate.

Update 16/04/2019:

- Correction of typos.
- Removed note about testing of firmware update in a backdrop of heavy ZigBee traffic as it is unclear whether the benefits are proportionate to the increased developer + lab effort involved.
- Extending the coverage of the assurance activities around FW activation design and testing to the FW update process more generally.

VER.2.M80: Protocol robustness testing (CHANGE)

Change summary:

Increased clarity of SC wording based on observations made during evaluations.

Rationale for changes:

A lot of discussion about fuzz testing requirements for ESMEs and GSMEs has taken place between the CPA Authority, labs and vendors.
The main thrust of this has been to explore how much of the message structures relating to the different interfaces need to be fuzz tested, hence the motivation to improve clarity here.

NOTE:

For ZigBee fuzz testing, there is a consideration that other non-CPA-certified devices on the HAN could be exploited to send crafted input over the pairwise-encrypted channel to attempt to subvert a meter. This implies that extending fuzz testing into this channel is desirable. The risks associated with such a threat vector are unclear, hence this note, but are considered significant enough to merit inclusion in this SC requirement.

Specific changes:

SC wording change: (modified wording change-marked + text to discuss is yellow-highlighted)

VER.2.M80: Protocol robustness testing

This mitigation is required to counter exploitation of a non-operational interface through crafted input

This mitigation is required to counter exploitation of an operational interface through crafted input

At Foundation Grade the evaluator **will** perform fuzz testing of the available interfaces.

As per guidance in the Process for Performing Foundation Grade CPA Evaluations [a], structured fuzz testing is expected for all available interfaces, physical AND logical. Based on mandatory functional requirements in [d], the following two interfaces will require fuzz testing: ZigBee and GBCS application layer messages.

For ZigBee, fuzz testing shall be performed on all the messages that can be received including those that are (a) unencrypted, (b) encrypted with the network key (and thus visible to all devices on the HAN) and (c) encrypted with an APS key set up to protect comms between the product and each other type of HAN device that is not required to be CPA-certified (at time of writing, PPMID, IHD and CAD).

When fuzz testing GBCS Application layer messages (‘use cases’), mutations are expected to cover all parts of a message that the product will attempt to decode up to the point of authentication. The point of authentication for these messages (as relevant to [a], for smart metering equipment) is the point at which the protective crypt gets successfully validated (one or both of digital signature and MAC, dependent on the message type); any message decoding performed before this point (even just to check message well-formedness) will be in scope of GBCS Application layer fuzz testing.

With this in mind, some message payload fuzz testing is expected (in addition to all the other sections of a GBCS Application layer message that can be present (i.e. GBT header, grouping header, signature field, etc), the amount of payload fuzz testing depending on how much of the message’s payload gets decoded by the product before the point of authentication is reached. This minimum expectation is based on some GBCS application messages requiring content in the payload to be decoded and processed as part of the cryptographic validation process for the message type.

In addition to the ZigBee and GBCS application layer interfaces, it is possible that the device may have additional interfaces beyond those defined in [d] that might be accessible to an attacker and hence also require fuzz testing.

~~Fuzz testing is described in more detail in the Process for Performing Foundation Grade Evaluations [a]. Interfaces that are disabled and that cannot be directly accessed without physical modification involving breach of the tamper-protection boundary are not included in the scope of fuzz testing.~~

~~If the device includes separate components each with their own tamper-protection boundary, and with inter-component interfaces between the components (which can therefore be accessed without breaching the tamper-protection boundary) then these intercomponent interfaces shall be included in the scope of fuzz testing.~~

Update 25/02/2019:

- Amended suggested wording to specify that fuzz testing needs to also be performed within the pairwise ZigBee link, as per the updated Discussion Note details.
- Intention to generate a separate lab guidance paper on to provide more in-depth guidance on fuzz testing expectations. The overarching aim of this document would be to help reduce the time and effort required to perform the fuzz testing, this through providing checks that a lab can perform to determine the amount of attack surface presented in incoming messages (i.e. up to the "point of authentication"), so that the whole complexity of different types of messages does not need to be fuzzed.

VER.4.M4: Evaluation/Cryptocheck (CHANGE)

Change summary:

Minor tweak in line with CPA evaluation policy.

Rationale for changes:

DRBG is the more typical crypt primitive associated with PRNGs.

Specific changes:

SC wording change: (modified wording change-marked)

VER.4.M4: Evaluation/Cryptocheck

This mitigation is required to counter exploitation of a cryptographic algorithm implementation error

At Foundation Grade the evaluator **will** ensure all cryptographic algorithms employed for security functionality have been validated as per the "Cryptography Review" section in the CPA Foundation Process document.

The evaluator shall include in this activity a confirmation (by reference to relevant CAVP or equivalent certificates, or by activities in the course of the CPA evaluation) that ~~component~~ cryptographic algorithms used by ~~primitives of~~ the PRNG (such as DRBG ~~SHA~~) have been independently validated for correctness.

Where cryptographic algorithms claim certification under CAVP (or equivalent external certification), then the evaluator shall confirm that this certification has been achieved for the relevant hardware/firmware/software components of the product, at the relevant version for the component. For cryptographic algorithms that are not certified using an external process, the evaluator shall confirm the correctness of the implementation by means of known answer tests, as described in the CPA Foundation Process document, Reference [a].

The cryptographic primitives used by the device shall be only those specified in [d], including those primitives used by all UTRN validation functionality in the device.

Update 25/02/2019:

- Typo – now using correct terminology w.r.t. DRBG and CAVP.
- Added wording to make it clearer that this requirement also covers UTRN functionality (this considering removal of VER.7.M4 but not removal of the DEV.7 + VER.7 “UTRN” logical component).

VER.4.M855: Receiver replay check (CHANGE)

Change summary:

Clarification that evidence is required for all command types requiring replay protection and inclusion of the anti-replay protection requirements for UTRNs.

Rationale for changes:

Clarification about the coverage of evidence and cover a potentially big gap in the testing of UTRN protections.

Specific changes:

SC wording change: (modified wording change-marked)

VER.4.M855: Receiver replay check

This mitigation is required to counter interception and replay of messages

At Foundation Grade the evaluator **will** verify that messages are not actioned more than once.

The evaluator shall confirm by testing that the device correctly rejects messages with unacceptable count values relative to its current state, and that the device correctly generates count values for which it is responsible. The testing shall cover both commands for immediate execution and future-dated commands (where ~~if~~ applicable).

The mechanism for protection against replay is defined in [d, 4.3]. Only certain messages require the protection, as specified in the Use Cases in [d, 19], summarised in [d, Table 20]. However, a different anti-replay mechanism is used for Security Credential commands as defined in [d, 13], and for Pre-Payment Top-Ups as defined in [d, 14].

Notes:

- Evidence is required for ALL commands that incorporate replay protection.
- When testing the anti-replay protection for Pre-Payment Top-Ups, the evaluator shall verify that Pre-Payment Top-Up messages are rejected if its UTRN counter value (a) matches any value in the meter's UTRN counter cache or (b) is lower than the lowest value in the meter's UTRN counter cache. These tests will also cover all the interfaces over which the meter can receive a UTRN.

Update 25/02/2019: (no changes to suggested new wording)

- Query about whether a sampled approach is acceptable: No, *all* commands incorporating antireplay protection for the device need to be checked. This because the associated commands can have significant security impact if they are maliciously replayed.
- No change to proposal made, as it does highlight that evidence is needed for "ALL" commands that require replay protection.

VER.4.M939: Enable update of security credentials (CHANGE)

Change summary:

Increase coverage of assurance activities to confirm that the product correctly implements updating of security credentials.

Rationale for changes:

Update Security Credentials functionality should be viewed as an important security enforcing function. If nothing else, it would need to be relied on if KRP key compromise occurs and it is also worth noting the reputational importance of this functionality correctly supporting GBCS Change of Supplier requirements. It is envisaged that manufacturers are already testing this functionality extensively, given its importance, so it is hoped that this will lead to minimal impact on overall evaluation effort.

Specific changes:

SC wording change: (modified wording change-marked, with further revisions yellow-highlighted)

VER.4.M939: Enable update of security credentials

This mitigation is required to counter exploiting incomplete update of security credentials
This mitigation is required to counter use of compromised security credentials
This mitigation is required to counter exploitation of a software implementation/logic error
This mitigation is required to counter installation of an invalid certificate

At Foundation Grade the evaluator **will** verify that the update of a security credential is atomic.

The evaluator will test that the update of each security credential either finishes successfully with complete replacement of all parts of the relevant credential or else retains the old credential.

At Foundation Grade the evaluator **will** verify that, in addition to the general critical message validation checks described in VER.4.M927, certificate path validation (CPV) always successfully completes, where required to do so by [d], before the validated replacement remote party certificate is installed.

The specific type of CPV required by [GBCS] will vary according to the type of certificate and the operation of each type of CPV will be verified by the evaluator.

At Foundation Grade the evaluator **will** verify that, once validation checks have been successfully performed, the specified security credentials replacement will take place with subsequent product functionality confirming this.

The evaluator shall seek evidence to confirm that all the different types of remote party security credentials defined in [d, 4] (i.e. covering the different types of remote party role, keyUsage and cellUsage, appropriate for the product type) can be replaced, using all the different credentials replacement modes defined in [d]~~, this by using each of the different credentials replacement modes defined in [GBCS] (for the different and covering the different types of types of credentials (i.e. varying according to s in terms of remote party, keyUsage and cellUsage)types of certificate that key usage types and) supported that are relevant for the product according to [d, 4] by the product~~.

Checks on subsequent product functionality should, as a minimum, confirm that the new credentials will be used for the associated cryptographic mechanisms, instead of the old ones. For instance, depending on the type of credential replaced, the following tests are suggested: (a) digital signature verification, (b) MAC authentication + generation, (c) certificate path validation and (d) encryption + decryption of sensitive data.

Update 25/02/2019:

- Tweaked wording as CPV is not required for all variants of CS02b (i.e. some instances of CS02b anyByContingency do not require this.)
- For clarity:
  - the last paragraph is intended to perform tests that demonstrates the security cred updates have worked by using the new creds to validate crypt on new incoming messages.
  - It might be possible for labs to reuse evidence from GBCS interop testing to help confirm this last point.
- Removed the following wording (which had previously been the last paragraph), which is already considered to be covered by:
  "The evaluator shall also verify that remote party credentials can only be updated using the [GBCS] defined modes supported by the product and not by any other replacement mode or mechanism."
- Query received that such testing may be covered in GBCS Interop testing –if this is the case and the nature of the testing + results can be shared with the evaluators and the evaluators can be satisfied that the evidence demonstrates conformance with this SC requirement, additional lab testing may not be needed.

Update 16/04/2019:

- Minor rewording to make it clearer that the testing needs to cover replacement of all the different certificates that GBCS requires to be present in the device.

VER.6.M917: Verify logical protection of keys (CHANGE)

Change summary:

Improve clarity of wording in terms of which security data is in scope of this SC requirement and what checks need to be performed as a minimum.

Rationale for changes:

As stated in the summary. No material impact on evaluations as this is a clarification and reflective of what has been encountered in evaluations to date.

Specific changes:

SC wording change: (modified wording change-marked)

VER.6.M917: Verify logical protection of security data ~~keys~~

This mitigation is required to counter gaining access to security data in a single device via operational interfaces

This mitigation is required to counter gaining access to security data in a single device via nonoperational interfaces

At Foundation Grade the evaluator **will** confirm the protection of ~~access to~~ security data, such as cryptographic key material.

The evaluator shall confirm that:

a) no sensitive key material (private asymmetric keys and any symmetric keys) can be exfiltrated from the product, and

b) the following security related data cannot be modified, except as a result of certain authentic messages defined in [d] intended for the purpose: device security credentials, remote party security credentials, including anti-replay counters and the meter's UTRN counter cache.

Note: This confirmation shall also take into account any documented product interfaces additional to [d] that have the potential to exfiltrate sensitive key material or modify security related data.

~~The evaluator shall confirm by testing that security credentials and other cryptographic keys to which the tester does not have authorised access cannot be accessed for read or for modification. The testing should include a search of interface documentation for methods other than normal operational messages.~~

Update 25/02/2019:

- Reworded to avoid potential confusion about DEV vs VER activities and to scope the impact that any potential non-GBCS interface might have on testing.
- Suggestion that more guidance could be provided about different types of internal interface. Not really possible to do this at this time as it would be going into very product specific detail (SC wording needs to be product agnostic). However: Intention for this to be borne in mind for future SC updates and/or associated guidance papers.

VER.7.M*nnn*: Confirm correct UTRN processing (**NEW**)

***** New requirement*** summary:**

Introduction of a new SC requirement to facilitate lab verification that the protections against non-authentic UTRNs are effective, whilst authentic UTRNs get correctly processed, when required.
Note: Introduction of this SC requirement would help facilitate the removal of the SC Requirement VER.4.M4 (covered elsewhere in this document).

Rationale for new requirement:

Testing of UTRNs is currently limited to DEVs, yet the associated functionality is clearly critical, both in terms of supply impact (e.g. supply lost if authentic UTRN incorrectly rejected) and fraud (e.g. non-authentic/repeated UTRN incorrectly accepted). Impact on evaluations should be limited if manufacturers are performing testing in this area already.

*****New** requirement wording:***

New SC wording:

VER.7.M???: Confirm correct UTRN processing

This mitigation is required to counter exploitation of a UTRN validation vulnerability for fraudulent purposes
This mitigation is required to counter loss of supply due to non-acceptance or incorrect processing of a valid UTRN

At Foundation Grade the evaluator **will** seek evidence of developer testing that demonstrates the product will always reject a non-authentic UTRN value, received via the user interface and PPMID.

This is intended to supplement other testing requirements elsewhere in the Security Characteristic and confirm that the code handling incoming UTRNs is sufficiently robust that, if any exceptional circumstances are encountered during processing, the default behaviour will be to reject the UTRN (i.e. fail-safe).

At Foundation Grade the evaluator **will** confirm that the product correctly processes valid UTRN values, received via the user interface and PPMID.

The evaluator will confirm that a sample of valid UTRNs are accepted via both user interface and PPMID and, in each instance, will result in (a) the meter's balance being adjusted by the amount specified in the given UTRN and (b) an appropriate response message being sent to the Supplier to reflect the adjustment. Note: There is an expectation that evaluator may be able to reuse testing that developers are performing in this area as evidence for this requirement.

Update 25/02/2019:

- Introduction of this new VER requirement has been queried for the following reasons:
  - This is about fraud that can be detected by other means (e.g. reconciliation of credit purchases)
  - It should be covered by other testing within the wider smart metering programme
  - Introduces new tests to what is often a slow process.
- Points are noted and carefully considered but counter arguments are as follows:
  - Reconciling UTRN attempts is second line and checks on first line protections should be performed.
  - Current checking of these first line protections is limited to DEVs only, meaning in theory that a developer only needs to assert that they perform the required checks.
  - 'DEV's usually have at least one associated VER, but this is not the case for UTRNs.
  - A main aim of the SCs is to mitigate potential CNI risks but reputational risks (such as fraud), are also protected against. Also "commit fraud" is captured as a high-level attacker objective in the SC Maps from which the SC requirements are derived).
  - the developer should already be performing the sort of testing suggested here, which could then potentially be reused by the lab.

VER.7.M4: Evaluation/Cryptocheck (**DELETE**)

***** deleted requirement*** rationale:***

This requirement has not really provided any meaningful value in evaluations, as all cryptographic compliance evidence has nearly always been covered by VER.4.M4, titled the same and targeted at message protection more generally (whereas VER.7.M4 focusses more specifically on protective crypt for UTRN messages).

DEP.5.M934: Unique security data per device (CHANGE)

Change summary:

Relocate this DEP requirement to DEV.

Rationale for changes:

Based on some evaluations, it has become apparent that this requirement – that two or more instances of a device must not share the same key data – is better expressed as a DEV requirement, rather than a DEP.

Specific changes:

SC wording change: (modified wording change-marked)

DEV~~DEP~~.6~~5~~.M934: Unique security data per device

This mitigation is required to counter gaining access to security data in a single device

At Foundation Grade the product ~~deployment~~ **is required to** contain no security data that enables compromise of a different device.

Devices shall not contain data which if compromised would directly enable an attacker to compromise ~~another core device~~ one or more other devices deployed in different premises (such as shared keys that would enable the attacker to masquerade as a different device, or a different core device). This requirement applies to all life-cycle stages of the product, following manufacture, and applies to all the interfaces, including any additional to those defined in [d], both external to and within the product's tamper boundary.

Update 25/02/2019:

- Clarified that this requirement applies to all stages of life-cycle post-manufacture and to any interfaces that an attacker could get at – both external to and within the tamper boundary.
- For info: This is looking to avoid things like the same symmetric key or asymmetric private key being used in multiple devices to protect device security; It would not apply to asymmetric KRP public keys.

Update 14/06/2019:

- Clarified that the overarching intent of the requirement is to avoid compromise of sensitive/private data in one instance of the product type from weakening security on other deployed instances of the product (updated wording is yellow highlighted).