
UK IT SECURITY EVALUATION & CERTIFICATION SCHEME

UK Scheme Information Notice – SIN 092

Effective Date: 25 February 2009

**Subject: Vulnerability-centric Evaluation:
Improving Evaluations by Putting Vulnerabilities First**

REFERENCES

- [CC] CC, Version 3.1, Revision 2, September 2007
- [CEM] CEM, Version 3.1, Revision 2, September 2007
- [AIS20] Functionality classes and evaluation methodology for deterministic random number generators, Bundesamt für Sicherheit in der Informationstechnik (BSI), version 1, 2 December 1999
- [GVSA] Guidance on Vulnerability Search & Analysis, UK Certification Body, v4.1, July 2003
- [NIST] A Statistical Test Suite for Random and Pseudo random Number Generators for Cryptographic Applications, see <http://csrc.nist.gov/rng/>
- [UK007] Verdict justifications in CEM compliant evaluations (UK CC Interpretation - UK/3.1/007), UK IT Security Evaluation & Certification Scheme, March 2007

Introduction to the Vulnerability-centric Approach

Overview

1. This SIN describes the basis for a ‘new’ approach to carrying out and reporting CC evaluations. The ‘newness’ arises from adopting a structure to the evaluation and its actions that is driven by vulnerability analysis and penetration testing, rather than the individual work units or evaluator action elements in CC. This new approach is referred to here as the ‘vulnerability-centric approach’. The structure of the evaluator’s work changes, but the content is still sufficient to encompass the evaluator action elements in [CEM], and the ETR still reports at the level of [CEM] work units¹.
2. For the purposes of this discussion we choose an assurance level of approximately EAL4. The choice of level is not particularly important, except that we assume that the set of deliverables includes an implementation representation (for completeness of the picture) and that an independent vulnerability analysis is required. Notwithstanding the fact that an independent vulnerability analysis is not required below EAL2, the essence of the vulnerability-centric approach, and the related ETR structure, is still applicable (and highly desirable) at the lower level.

¹ This description relates to CCv3.1. In general, whilst there are changes in the fine detail for applying the approach to a CC v2.3 evaluation, the underlying principles and the general approach are essentially the same.

The linear approach to evaluation

3. The picture of activity that is most directly implied by reading CC is as follows²:

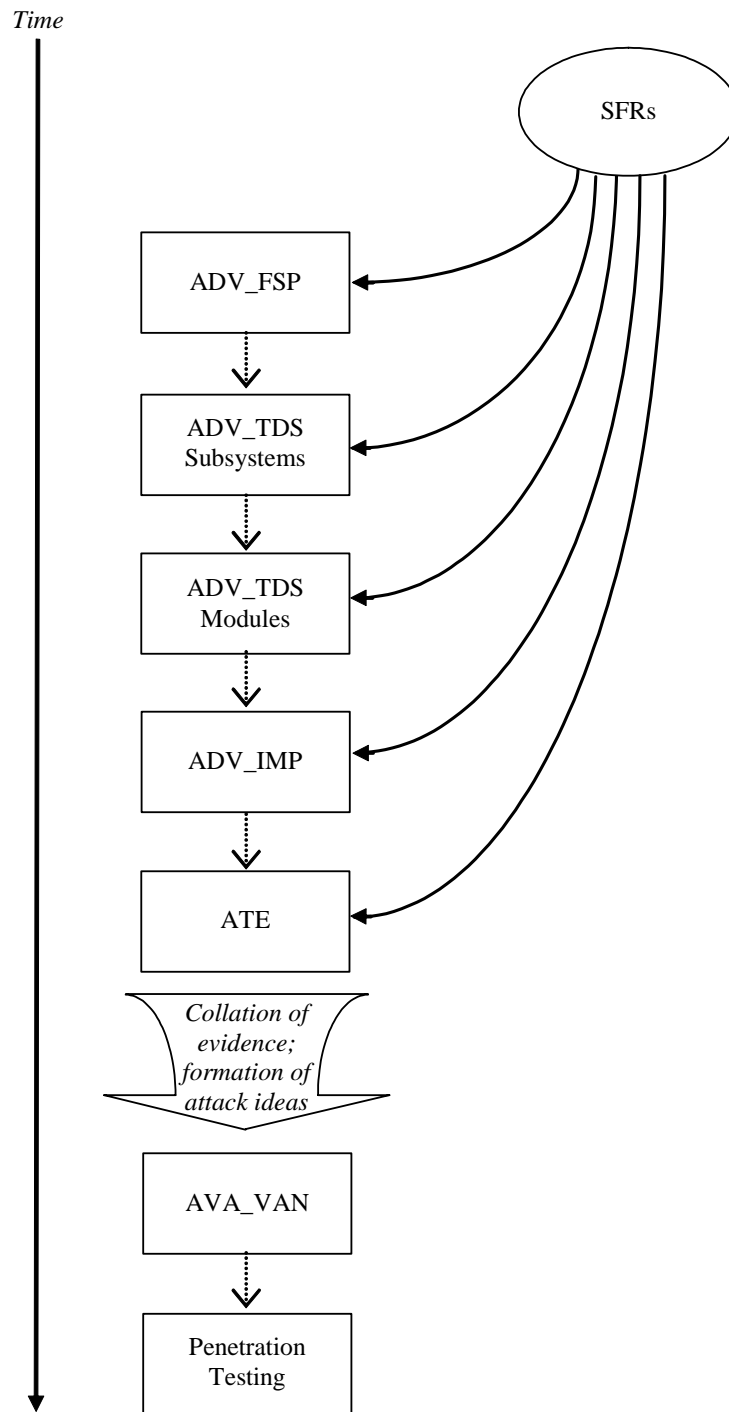


Figure 1: ‘Conventional’, or ‘linear’, family-led evaluation structure

² Figure 1 omits assurance classes other than ADV, ATE and AVA. This is because it aims to show the activities that dominate the flow and timing of an evaluation. In general, other families will be fitted into the flow in parallel, and are flexible in their timing. In the case of ADV_ARC (which is also omitted for the sake of clarity), the implication from CC is that this will be carried out in parallel with the other ADV activities (though mainly ADV_TDS), and similarly feed into the ATE and AVA_VAN activities.

4. In this picture of activity, the evaluator proceeds from the Security Functional Requirements (SFRs) through the process of tracing into successive design. In addition, each design representation is assessed as a separate activity, with separate CEM work units to be completed and checks to be made. After the design representations, the SFRs may be 'located' in the developer test evidence, as a means of assessing the developer testing, and of identifying a subset of tests that the evaluator may repeat (and in some cases variations of these tests may give rise to penetration tests).
5. At the end of the design and test assessments, the information and knowledge gained about the TOE is collated and converted into ideas for vulnerability analysis and penetration testing ([GVSA] describes ideas on how this process takes place in the linear context; similar guidance is provided below for the vulnerability-centric approach).
6. The purpose of penetration testing is viewed as 'to confirm or disprove the exploitability of any potential vulnerabilities' (according to the UK evaluator training course module on Penetration Testing). This implies that penetration testing is carried out *after* vulnerability analysis (and other evaluation activities), and that it is *confirmatory* rather than *exploratory*. In practice, penetration testing tends to be the main experience that the evaluators will gain of actually using the TOE, and usually involves adds a further dimension to the understanding gained from reading TOE design (and other) deliverables, or even observing the TOE in operation.
7. The linear picture of activity fits the separate descriptions of requirements and activities in [CC/3] and [CEM] and, particularly for new evaluators, can therefore seem like the intended way of approaching the evaluation. Indeed, CC presents no alternative picture of how an evaluation should (or could) be carried out. Furthermore, since the work items that the evaluators are required to complete are expressed in this way, and the reporting requirements on the evaluators are expressed in terms of work items, it creates an impression that the most efficient and least risky way to carry out and document the evaluation is to follow this separated structure. Following the linear approach makes it relatively easy to write a conventional ETR, but also relatively easy to miss connections between descriptions of SFRs at different levels, and between documentation and the reality of an operating TOE.
8. There is nothing inherently wrong with a linear approach in terms of the ability to complete a comprehensive and effective security evaluation. However, it is not the most efficient way to conduct, or to document an evaluation. Experience suggests that understanding the design representations of a complete new TOE is generally a difficult task for any evaluator, and is more difficult without a focus on operation (both theoretical and hands-on) of the TOE³. There is also a danger that, because vulnerability analysis and penetration testing are left to the end of the process, the amount of time actually spent on these activities will be reduced (or that they will be carried out under stressful conditions that do not favour a creative and observant approach) because of time pressures towards

³ It might be argued that the statement of the SFRs and the determination of the accuracy and completeness of their instantiation in the various TSF representations is designed to provide this focus. However, the suggestion here is that without an early focus on the actual way in which the TOE operates, or a focus on potential exploitable vulnerabilities then it is easy for the design assessment to become something of an end in itself (e.g. when assessing completeness of a representation). Such early analysis is likely to be needed to bridge the gap between CC terminology and structures (in SFRs) and TOE terminology and structures (in design documents).

the end of an evaluation. Since finding and demonstrating vulnerabilities⁴ is a critical part of an evaluation, this is a danger worth avoiding. The avoidance may also help to stop an evaluation overrunning at a late stage due to the amount of analysis and number of penetration tests being ‘suddenly’ discovered to be larger than expected.

The vulnerability-centric approach to evaluation

9. The alternative picture of an evaluation that is driven by vulnerability analysis and penetration testing, as presented in this document, is as follows:

⁴ Or perhaps, from a more optimistic point of view, we could say ‘achieving a sufficiently comprehensive analysis to generate confidence that any vulnerabilities present would in fact have been found by the evaluators’.

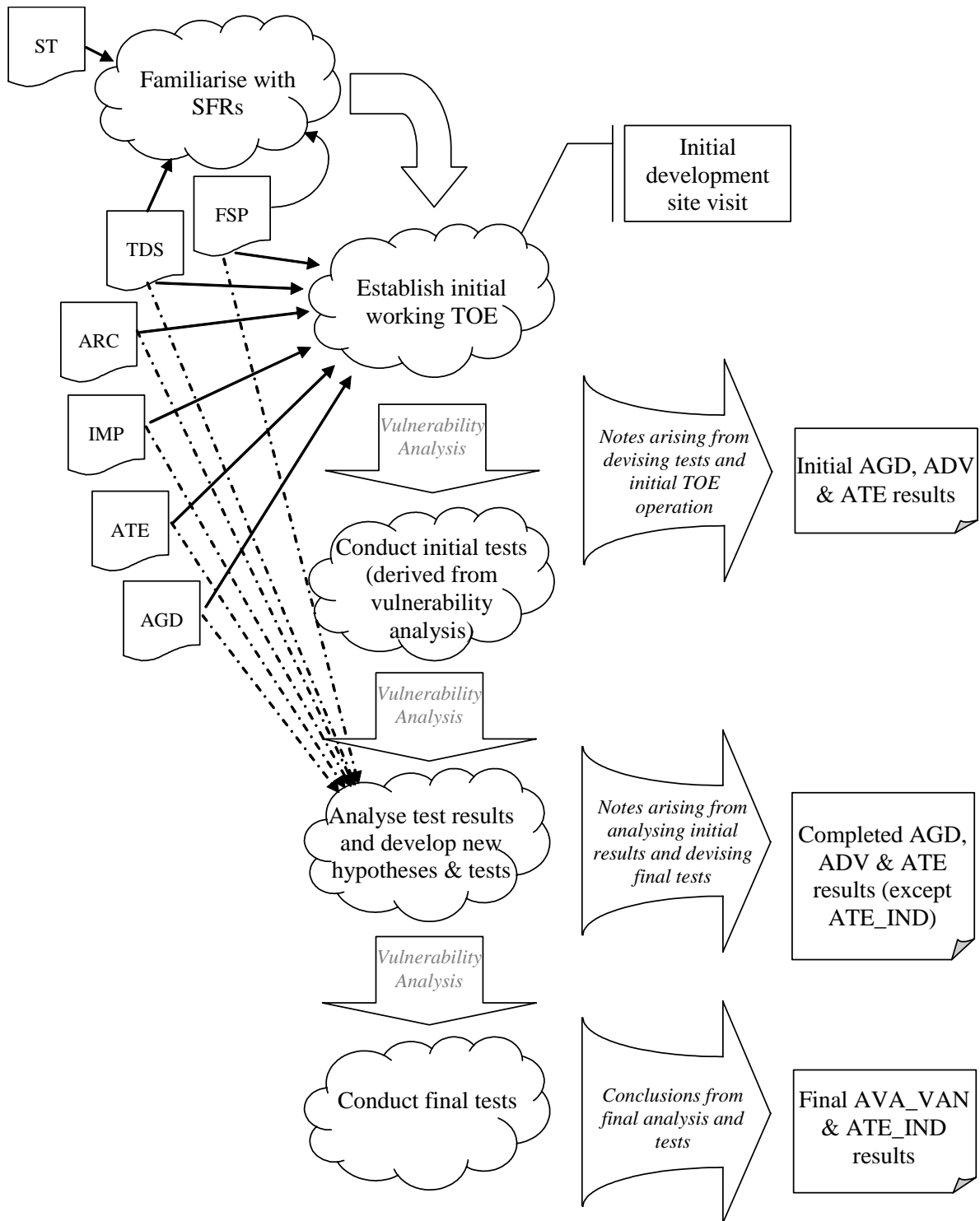


Figure 2: Vulnerability-centric evaluation structure

10. Figure 2 shows an evaluation structure that is led by the activities of analysing vulnerabilities and conducting penetration tests. This means that it focuses on the following activities:
 - a) Constructing an operational version of the TOE at an early stage
 - b) Carrying out an early visit to the developer in order to clarify the evaluators' understanding of the TOE and SFRs design and operation
 - c) Carrying out an early set of penetration tests (and possibly additional functional tests).
11. This clearly focuses on the AVA_VAN and ATE_IND activities, and results in a more exploratory approach to vulnerability analysis and penetration testing. The analysis and test activities are also more firmly grounded in experience of the TOE operation. The completion of other CC family requirements (e.g. ADV and AGD) arises as a consequence of seeking sufficient understanding to carry out a sufficiently systematic and detailed vulnerability analysis, and to devise penetration tests that support the vulnerability analysis. Part of the rationale for this approach is that it encourages the vital creative ingredient in penetration testing, which can be lost if the vulnerability analysis becomes driven by a checklist⁵ derived from earlier ADV and ATE activities.
12. Focusing on hands-on work with the TOE can be expected to be of benefit even where the final TOE and/or final versions of deliverables are not available for the early stages. In particular, early vulnerability analysis can be carried out by focusing on the security architecture. In general, normal development activities (i.e. excluding activities undertaken to correct vulnerabilities) do not usually change a TOE in ways that affect the core conclusions about the security functions. If the TOE does change in a substantial way (whether this is because of discoveries by the developer or the evaluator) then the impact will be such as to require specific replanning regardless of whether the conventional or vulnerability-centric approach is being used.
13. An early focus on establishing a working TOE and searching for potential vulnerabilities is also expected to be helpful in assessing changes when a re-evaluation is carried out, whether this is a complete re-evaluation or an evaluation of changes to a previously certified TOE. A similar argument applies to cases of composition (including those covered by the ACO assurance requirements in a composition evaluation).
14. Other areas of evaluation (such as ALC) are addressed in the section 'Relationship to CC Assurance Requirements' below.

⁵ There may or may not be an actual checklist. However, the ADV/ATE-led approach tends to produce checklist-like output, which may become the complete penetration test scope under the pressures commonly found in the later stages of an evaluation. This is not, of course, intended to prevent the use of checklists from ADV/ATE activities (or other experience). Provided the 'creative' step is made, checklists will always be useful at least as review criteria for completeness. However, the intent here is to prevent the main emphasis being on checklists.

15. The approach described here is deliberately flexible, and may therefore be seen as being insufficiently detailed or well-defined. However, the authors resist this suggestion for the following reasons:
- a) The approach uses the existing work unit requirements in CEM as its building blocks; it redeploys effort, adjusts the order of activities and adopts different priorities, but the rigour put in place in CEM is still present.
 - b) The approach is founded on the experience of evaluators, and relies on their judgement to map requirements on to each individual evaluation. This is not so different to the current situation, where evaluator experience and judgement is still a significant factor (and justified by the training process and supervision both within an ITSEF and by Certification Bodies).

Vulnerability-centric Evaluator Activities

16. This section discusses each of the evaluator activities shown in Figure 2, describing their inputs, outputs and objectives⁶. As a general comment, it should be made clear that the vulnerability-centric approach does not mean that the evaluator spends no time on ‘conventional’ activities such as tracing SFRs in the design representations. Rather it turns many of the design-related activities into means towards the end of effectively testing and analysing vulnerabilities in the TOE, instead of taking these activities as ends in themselves.
17. This section begins by presenting an ideal approach to the activities (e.g. it assumes that all deliverables are available at the beginning of the activity). Describing this ideal approach is intended to convey the spirit of the vulnerability-centric approach, but it is recognised that most evaluations will involve some compromises⁷. The later parts of the activity descriptions therefore discuss these compromises (e.g. a minimum set of initial deliverables is suggested for ‘Familiarise with SFRs’), and there is further discussion of adapting the approach in the later section on ‘Examples of Applying the Activities’.

Familiarise with SFRs

Purpose of the ‘Familiarise with SFRs’ activity

18. The essential starting point for an evaluation is for the evaluators to gain familiarity with the ST and in particular with the SFRs and how they are provided by the TOE (as described by the TOE Summary Specification). In the vulnerability-centric approach this is identified as an early activity which overlaps with the establishment of the initial TOE. (It could even be viewed as a sub-activity of ‘Establish the initial working TOE’.)
19. A vulnerability-centric evaluation places the early understanding of the SFRs, how they are implemented, and how they are used, as important early activities. The search for vulnerabilities relies on a comprehensive understanding of the full meaning and ‘rich picture’ of the SFRs, including: the assets that they protect and/or use, the threats that they protect against⁸, the relationships (e.g. dependencies) between SFRs and the security architecture, and the aspects of each SFR that can and should be tested.

⁶ A formal presentation of the activities in this way is not ideal, since the point of this approach is that the activities are largely flexible. Flexibility is necessary to deal with the variations in individual evaluations, such as when various deliverables are available. As noted elsewhere, the vulnerability-centric approach represents more of a *restructuring* of evaluation activities than a *redefinition*. However, this view of the activities is presented in order to give another view of the approach that may appeal more to evaluators who favour process definition.

⁷ At the same time it is recognised that there are cases of evaluations where compromises will reduce the efficiency of the evaluation, and this should at least be acknowledged. One of the problems that can arise with the linear approach is that it appears that any activity can be started as soon as its deliverables are available. Those evaluators who adopt the vulnerability-centric approach will probably raise more questions about whether sufficient ‘context’ is available to usefully carry out an activity.

⁸ Of course it should theoretically be obvious which assets are involved in an SFR and which threats it protects against. However, in practice this is not always the case: the progression from threats to objectives (for both the TOE and the environment) frequently loses the clarity of the threat-SFR relationship.

Actions involved in the 'Familiarise with SFRs' activity

20. The interim objectives are to produce for each SFR a description of how it is implemented and represented in the design deliverables. So the evaluator reaches a preliminary view of:
 - a) how the assets are represented in the design
 - b) how each SFR is represented in the design⁹
 - c) conditions that might enable an SFR to be bypassed or tampered with
 - d) supporting functions that may not be evident in the SFR definition but which may be critical to its operation (e.g. integrity protection functions) and hence important for vulnerability analysis and penetration testing.
21. Clearly this can lead to preliminary assessment of various aspects of deliverables for ADV work units, such as completeness, correctness and consistency of design deliverables.
22. In addition, if the ATE deliverables are available at this time, the aspects of the SFRs that are tested by the developer, as described in the ATE deliverables, are analysed so that the evaluator comes to a preliminary view of:
 - a) what aspects of each SFR have been correctly and adequately shown by the developer (with reference to the evaluator's independent analysis of where/how the SFR and its related assets are represented in the design deliverables from FSP to IMP, where feasible at this stage)
 - b) what aspects of each SFR have *not* been correctly or adequately shown by the developer
 - c) what additional tests, or variations of tests the evaluator would like to try in the initial tests (some of these may be to aid understanding and to explore TOE behaviour, some may be to demonstrate TOE behaviour under different security scenarios; some tests may achieve both of these aims).
23. The examination of tests may also help in the understanding of the implementation of SFRs, and in understanding some of the design deliverables (for example, test descriptions may make clear how parts or operations of the TOE fit together, in ways that are not so easy to describe in the 'flat' design representations).
24. The activity typically proceeds by:
 - a) Reading the ST to understand the meaning of each SFR in terms of what it does, what assets it protects and/or uses, what threats it counters (and hence what opportunities for bypass might exist). Even at this early point it is possible, and useful, to start creating a list of possible penetration tests (and perhaps even some of the scenarios that the developer would be expected to cover under ATE). This process of asking questions about the SFRs also directs the tracing of the SFRs into design deliverables. It is useful

⁹ An early 'group review' activity could be to compare and discuss the results for each SFR. In this way the whole evaluation team gains a shared understanding of the SFRs, and there may be opportunities to cross-check where the same asset has been examined in the contexts of different SFRs by different evaluators.

to ask questions such as ‘how would I expect to see this SFR or its effects’ and ‘what actions would I expect this SFR to prevent’?

- b) Tracing each SFR into the design representations and (if available at this stage) test documentation. This will clearly make use of the developer’s representation correspondence, but should also be treated as an independent checking of the correspondence. This generates results for the evaluator work units for ADV_FSP.*.2E, ADV_TDS.*.2E and ADV_IMP.1.1E. In general these results will be interim rather than final, especially if some of the evaluator inputs are not available at the beginning of the evaluation.
- c) Compiling questions to put to the developer – this is intended to be linked to the ‘Initial Development Site Visit’ activity. However, even if an initial development site visit is not carried out, it is likely to be beneficial to compile questions at this stage to put to the developer. This aspect is discussed further in the section on ‘Initial Development Site Visit’ (see below).

CEM activities covered by the ‘Familiarise with SFRs’ activity

- 25. Evaluator actions such as tracing SFRs into different design representations (e.g. for [CEM] work units ADV_TDS.3-4, 3-14 & 3-15) are carried out across this and subsequent activities (‘Initial Development Site Visit’, ‘Conduct Initial Tests’, ‘Analyse Test Results’, etc.). The vulnerability-centric approach does not remove these activities, but rather includes them as a means to the end of vulnerability analysis and penetration testing, instead of being an end in themselves. Ultimately it is for the evaluators to judge when best to start and complete these conventional activities, but this paper provides some suggestions in terms of work units that could be provisionally completed¹⁰ in each activity.
- 26. In summary therefore: the aim of this first activity is to familiarise with the SFRs in as many representations and tests as possible at the beginning of the evaluation. Recognising that not all deliverables may be available at the start of an evaluation, it is noted that the minimum set of deliverables expected to enable a worthwhile activity¹¹ are the ST, the FSP, the ARC and the TDS (to the level of the subsystems).

¹⁰ Here and elsewhere, this document refers to ‘completion’ of an activity. This reflects completion in the sense that an evaluator believes that they have enough understanding (and, where relevant, information to meet the ‘Reference/Analysis/Report’ evidence requirements of [UK007]) to assign a verdict to the relevant evaluator action. It is assumed that this verdict may always be changed by later work. In the vulnerability-centric approach it is more likely that verdicts will be revised because of the incremental nature of the work. Although this may make provisional verdicts less stable, this is not seen as a disadvantage (e.g. it makes provisional verdicts more visible and open to early challenge by reviewers and the Certification Body).

¹¹ Determining a minimum set of deliverables is always a matter for evaluator judgement in each specific case. However, it is difficult to see how a meaningful analysis can be done during the familiarisation if this minimum set is not present. It is also noted that assessment of the ST alone (ASE) may be done as a separate activity if required by the sponsor or the evaluation or certification process. This would then be part of the ‘Familiarise with SFRs’ activity, but the separation from analysis of other representations would be likely reduce the efficiency gains at this stage.

27. At the end of this activity it should be possible to complete at least the ASE work units. Depending on the deliverables and their suitability, it may also be possible to provisionally complete many ADV_FSP and some ADV_ARC and ADV_TDS work units (e.g. ADV_FSP.4-1, 4-4, and 4-10 to 4-12)¹².

Initial Development Site Visit

Purpose of the 'Initial Development Site Visit' activity

28. The Initial Development Site Visit is an optional part of the process, but is strongly recommended. The idea is that the 'Familiarise with SFRs' stage will inevitably generate questions and even confusion about the SFRs and their representation in the deliverables (including ATE deliverables). The evaluators therefore aim to produce question about the SFRs from the familiarisation stage, and to resolve these during the visit.
29. In some evaluations visits to the relevant site(s) may be difficult to arrange and/or expensive, for example where the sites are overseas. Although this obviously requires a pragmatic approach, it is believed that the costs of an early visit are compensated by the efficiency and assurance gains from the greater knowledge of the TOE gained at an early stage. This can lead to savings not only in evaluator time (because the site visit enables understanding of the TOE and the implementation of the SFRs to be more quickly and thoroughly acquired) but also in developer time (since it is usually more efficient to respond to evaluator questions face-to-face than via e-mail or telephone). Combining the initial visit with other activities such as site audits for ALC_DVS may also reduce any additional cost impact of an early site visit.
30. It is also likely that face-to-face meetings between developers and evaluators during an early site visit will make it easier and quicker to resolve other questions arising during the evaluation. The developers and evaluators will have some shared context which should enable easier use of terminologies and shared points of reference in the development environment.

Actions involved in the 'Initial Development Site Visit' activity

31. A typical set of areas to be addressed in the initial visit are:
 - a) confirming understanding of the SFRs and their implementation
 - b) ADV_IMP activities using deliverables that can only be examined on the developer's site¹³
 - c) familiarisation with the procedure for setting up a working TOE (for use in 'Establish Initial Working TOE' below).

¹² In the case of ADV_ARC, the evaluators will be in a position to come up with an initial view of the evidence. However, completion of the ADV_ARC work units will require access to other evidence, in particular the lower levels of design and the implementation representation in the case of an EAL4 evaluation.

¹³ For example: in smart card hardware evaluations it is not unusual to examine the implementation representation on the developer's site, partly because of confidentiality concerns and partly because a meaningful examination will often rely on use of a design toolset which cannot be reasonably reproduced in an ITSEF.

32. It may also be useful to combine this visit with aspects of:
- a) establishing the initial working TOE (see ‘Establish Initial Working TOE’ below)
 - b) familiarising with and assessing the development lifecycle, development environment security, configuration management system, and the TOE delivery process
 - c) repeating, observing or extending developer tests (if test deliverables are supplied and tests are automated then it is often possible to identify useful tests of this sort even at a very early stage).

CEM activities covered by the ‘Initial Development Site Visit’ activity

33. At the end of the activity (in combination with the earlier results from ‘Familiarise with SFRs’) it should be possible to reach provisional conclusions for many of the work unit requirements for ADV (and possibly ATE). Possible examples of provisionally completed items (even if they are not formally written-up at this stage) would be:
- a) All ADV_FSP work units
 - b) All ADV_ARC work units (to the extent that this can be achieved with the FSP and subsystem-level design as supporting evidence)
 - c) All ADV_TDS work units dealing with TOE subsystems (ADV_TDS.3-1, 3-3 to 3-5, and part of 3-14 and 3-15)
 - d) ADV_TDS.3-2 and to 3-6 to 3-8 (this leaves work units concerned with a more comprehensive view of the TOE modules to be completed at the end of ‘Analyse Test Results’ below)
 - e) ADV_IMP.1-1 and 1-2 (appropriateness of the deliverable).
34. Of course other aspects traditionally assessed in a site visit, such as requirements from ALC may also be covered, and possibly completed, during these visits.

Establish Initial Working TOE

Purpose of the ‘Establish Initial Working TOE’ activity

35. This activity is designed to give the evaluators hands-on experience of the TOE as early as possible. As a result of this, and following from the activities described above, the evaluators are expected to have achieved good understanding and knowledge of the TOE in both theoretical and practical terms. This highlights the difference from the linear process where *practical* understanding and experience may be deferred to a late stage. As discussed above, one of the main benefits of the vulnerability-centric approach is the acquisition and use of this practical knowledge to focus and lead the evaluation work.

Actions involved in the ‘Establish Initial Working TOE’ activity

36. The essence of this activity is to set up and familiarise with the working TOE. ‘Familiarisation’ here typically means making ad hoc tests of the security functions to see

that they operate in ways expected from the earlier activities. Other tests may also be suggested by deliverables for AGD (depending on availability at this stage). The basic objective is clearly for the evaluators to achieve a working TOE with as much independence from the developer as possible (this is designed to assist evaluator understanding, not because of any perceived conflict of interest). Specific targets to define completion of this activity are left to the evaluators' judgement and are strongly determined by the nature of the TOE and the availability of deliverables¹⁴.

CEM activities addressed by the 'Establish Initial Working TOE' activity

37. Depending on the deliverables available it may be possible to provisionally complete some of the work units for ADV, ATE, and AGD.
38. Depending on the amount of preparation possible (in terms of time and available deliverables), it may also be possible to repeat some developer tests (ATE_IND), carry out additional evaluator functional tests (ATE_IND) and even carry out some penetration tests (AVA_VAN).

Conduct Initial Tests

Purpose of the 'Conduct Initial Tests' activity

39. Conducting initial tests represents moving from the theoretical understanding of the TOE (from familiarising) and the practical familiarisation from establishing the initial TOE, to conducting tests of the TOE security. Conducting the tests includes both defining and carrying out tests (and as usual with penetration testing there may be some iteration and development of new tests on the basis of results obtained during this test period).

Actions involved in the 'Conduct Initial Tests' activity

40. Defining the tests is carried out by pursuing the same work as would normally be done for a vulnerability analysis. The activity combines strong familiarity with the SFRs, their meaning (e.g. in terms of assets manipulated), and their representation in the TOE (as established using the developer and independent evaluator tracing of the SFRs (ADV_FSP, ADV_TDS)). The evaluators use the security architecture description (ADV_ARC), relevant developer test evidence (ATE_FUN), external vulnerability information, and their own hypotheses and speculations to identify potential vulnerabilities (systematically) and to turn these into tests that will demonstrate actual vulnerabilities (or that the potential vulnerability is countered by some means), and clarify TOE behaviour (e.g. the ability to induce hardware faults at a certain point, or to discriminate operations in power traces).
41. In the course of defining and planning the tests it is therefore expected that the evaluator will largely complete the ADV and ATE work units, since this will be a requirement for understanding the TOE sufficiently to define the tests. The evaluator is therefore expected to incrementally complete results for the ADV and ATE work units (taking into account the evidence necessary for the ETR, such as that discussed in [UK007]) as the tests are

¹⁴ In some cases, where the TOE cannot be established at this stage, it may nonetheless be useful to carry out some limited work on items such as prototypes, simulators or emulators, if available. For example, analysing the TOE design for vulnerabilities at the beginning may indicate potential weaknesses at a protocol level which can be confirmed or refuted by reference to a simulator or prototype (e.g. by posing questions about undefined command sequences, or the impact of certain types of message alteration).

planned. The notion of ‘initial’ testing should not be taken to mean that the evaluator is expected to have an incomplete knowledge of the SFRs, rather that the evaluator acquires this knowledge in order to identify potential vulnerabilities in the TOE and its operation, and hence to define the tests.

42. In practice, this activity may overlap with, or at least immediately follow, ‘Establish Initial Working TOE’. The tests may combine repetition or observation of developer tests (for ATE_IND) and penetration tests (for AVA_VAN), and the activity may of course also provide evidence to support AGD work units.
43. The activity may also merge seamlessly with the later activities ‘Analyse Test Results’ and ‘Conduct Final Tests’ – indeed this is the intention. The reason for separating the activities in the vulnerability-centric description is to emphasise that testing is expected to start early, and in particular before the completion of ADV and ATE activities.
44. In some cases of course a TOE is not completely available at an early stage (whether this refers to ‘horizontal completeness’ where some parts of the TOE may not be available, or ‘vertical completeness’ where definitive versions of documents are not available, or ‘diagonal completeness’ where both situations are combined). This situation requires a degree of bespoke planning in either the conventional or vulnerability-centric approaches. The vulnerability-centric view is nonetheless to conduct tests as soon as possible, even if this is on draft or incomplete items. The incompleteness will necessitate analysis of the impact of the expected future changes to complete the TOE, which may even assist the testing. In some cases, incompleteness may even allow for white box tests that would not be possible on the final TOE (e.g. when using stub modules that can be adapted for test actions or logging).
45. If the state of the TOE is such that it is not possible to create a meaningful test system at an early stage then the *pure* vulnerability-centric approach may not be applicable, at least not if the evaluation has to remain a parallel activity with development. In this case, a ‘reduced¹⁵ vulnerability-centric’ approach should be taken with the aim of using vulnerability analysis as a focus, but without the ability to gain hands-on experience. This would mean starting the vulnerability analysis activity and designing penetration tests as early as possible (also including the ‘Initial Development Site Visit’). The ADV and ATE activities would still be expected to be completed as a side effect of the vulnerability analysis, rather than as a separate activity with penetration test ideas as a side effect.

CEM activities addressed by the ‘Conduct Initial Tests’ activity

46. Depending on the deliverables available it may be possible to complete some of the work units for ADV, ATE, and AGD.
47. Depending on the amount of preparation possible (in terms of time and available deliverables), it may also be possible to repeat some developer tests (ATE_IND), carry out additional evaluator functional tests (ATE_IND) and even carry out some penetration tests (AVA_VAN).

¹⁵ ‘Reduced’ in this sense means ‘constrained by the lack of early hands-on experience of the TOE’. It does not (of course) suggest that the focus on vulnerability analysis is diminished in any way.

Analyse Test Results

Purpose of the 'Analyse Test Results' activity

48. Separating analysis of initial test results (and hence the definition of new tests) into a separate activity in Figure 2 serves to make explicit the iterative nature of the penetration (and possibly repetition of developer testing) expected in the vulnerability-centric approach. It is generally expected that analysis of initial test results will either determine the next set of tests (e.g. by resolving questions about the feasibility of a hypothesis, or identifying an optimum way to test the hypothesis) or that they may yield partial results that suggest new tests (e.g. hardware fault induction tests may suggest new power analysis tests or additional code analysis in order to find exploitation methods for discovered fault types).

Actions involved in the 'Analyse Test Results' activity

49. The analysis and definition of new tests will inevitably involve analysis of module design and implementation representations, and hence to the completion of ADV_TDS and ADV_IMP work units.
50. Although this is presented as a separate activity, it is highly likely that there will be multiple iterations of the 'test-analyse-define new tests' cycle in any vulnerability-centric evaluation. In an extreme case, as may take place in smart card hardware evaluation for example, most of the evaluation period may comprise such cycles.

CEM activities addressed by the 'Analyse Test Results' activity

51. By the end of this stage it should usually be possible to complete all work units for ADV, ATE, and AGD. (Note that it may not be possible to complete the additional functional tests for ATE_IND until 'Conduct Final Tests'.)

Conduct Final Tests

52. As noted above, the 'test-analyse-define new tests' cycle may be repeated many times, but 'Conduct Final Tests' represents a stage in which final tests are carried out and vulnerability analysis is finally completed.

Examples of Applying the Activities

53. The following examples aim to show how the sequence of activities could be applied to different evaluation situations. The examples are inevitably abbreviated, leaving out many details. But they aim to show different ways in which the same fundamental vulnerability-centric approach is applied.

Smart card TOE combining hardware & software from the same developer

54. In this case all of the deliverables (including samples of the TOE) are available at the beginning of the evaluation. Because the TOE is relatively easy to set-up and operate in the ITSEF, 'Familiarise with SFRs' is carried out in parallel with 'Establish initial working TOE' and these comprise the first part of the evaluation. Both of these activities are used as preparation for an initial development site visit at which the objectives include: an audit of ALC_CMC and ALC_DVS requirements for the main development site, an analysis of

the implementation representations using the developer's tools, repeating a selection of developer tests. In addition, the evaluators start 'Conduct initial tests' during the development site visit by running evaluator-defined simulation tests using the developer's tools. As a result of carrying out 'Familiarise with SFRs' and 'Establish initial working TOE' before the visit, the evaluators have an extensive list of questions about tracing the SFRs into the various design and implementation representations, and about the mapping of SFRs to developer tests. Many of these questions are sent to the developer in advance, to enable the site visit to be more efficient; but some questions are reserved for the face-to-face discussions because they depend on other answers or the availability of developer tools.

55. Apart from the tests carried out during the initial development site visit, the remainder of the testing takes place in the ITSEF. In this case 'Conduct initial tests', 'Analyse test results' and 'Conduct final tests' are almost indistinguishable since testing is carried out almost continuously, with the results from one test leading to the definition of the next. During the course of the testing the ADV, ATE, and AGD are all completed as 'side effects'. (ALC is completed in a separate, parallel activity.)

System evaluation carried out in parallel with development

56. In this case the deliverables and the TOE itself are not available until late in the evaluation. The first activity 'Familiarise with SFRs' is therefore carried out once the ST, FSP, ARC and TDS (to subsystem design level) are available in a form that includes all the SFRs (although the evaluators recognise that later revisions to these deliverables are likely). The initial development site visit is carried out in order to discuss the SFR implementation and to carry out 'Establish initial working TOE' on the development site using initial prototype versions of the TOE. This enables the evaluators to understand the system environment and to create early hypotheses and scenarios for attacks and penetration tests. Other activities at this time are focused on tracing and understanding the SFRs, and creating hypotheses about potential vulnerabilities or about aspects of the design that are unclear and that therefore should be tested (either by the developer or the evaluator). The evaluators continue the evaluation, with progress being driven by the availability of new deliverables (lower levels of design, IMP, etc.) linked to system development milestones (e.g. releases). Progress and status are reviewed within the evaluation team by compiling and discussing tables of vulnerability hypotheses and penetration tests for each SFR.
57. When the final design deliverables and initial test deliverables are available then the evaluators construct their own version of the TOE and begin 'Conduct initial tests' using a test environment on the developer site. This leads to completion of the ADV and ATE activities (subject to any changes by the developer). This initial test period is limited, and linked to an initial phase of developer testing¹⁶. This also enables completion of significant amounts of the AGD activity.
58. When the initial tests are completed, the results are analysed and interpreted by reference to the design ('Analyse test results'). The developer's final ATE deliverables arrive and the evaluators use these to identify additional tests and their final penetration tests. The final version of the TOE is installed and the evaluators carry out 'Conduct final tests' as well as

¹⁶ For example, the activity might be linked to system and acceptance testing for an 'initial operating capability' for the system.

confirming that the results for AGD are still applicable for the final TOE.

Operating System or DBMS in parallel with ongoing evaluation consultancy

59. In this case many (but by no means all) of the deliverables are available at the start of the evaluation. A separate team of evaluation consultants is working on the more 'specialised' evaluation documentation that developers often do not produce themselves (for example, test coverage and depth analyses, SFR tracing information, and so on). Furthermore, although a copy of the working TOE is made available at an early stage, all the evaluated configuration details may not have been finalised (for example, there may be outstanding decisions to be made on the precise platforms to be covered, or there may be ongoing development work on a critical patch). The development team is based overseas, but it is expected that the consultancy team should be able to deal with many evaluator questions.
60. This scenario is therefore similar to the system evaluation case described above. The 'Familiarise with SFRs' activity can be carried out with at least the ST, FSP, ARC and TDS (to subsystem level) as input, although ATE deliverables may not be available. This activity may generate questions that could be posed to the developers in an 'Initial Development Site Visit', but it is possible that many of the evaluator's questions at this point could be answered at an early stage. In these circumstances the evaluators might choose to defer the initial site visit, focusing rather on the 'Establish Initial Working TOE' and 'Conduct Initial Tests' activities. These may be carried out in close collaboration with the consultancy team. Whilst the definitive evaluated configuration will not be known at this point, the early hands-on experience these activities bring will nonetheless provide the evaluators with a useful working understanding, sufficient to at least to produce test specifications and carry out 'dry runs' of those tests. This knowledge informs further analysis and testing.
61. The 'Initial Development Site Visit' is then carried out. This will cover the ALC activity, and (possibly) provide an opportunity to pose questions directly to the developers in aspects with which the consultants are not familiar, and witness or repeat the developer's tests (it is assumed at this point that all test-related issues have been resolved by the developer). The evaluation then follows the 'test-analyse-define new tests' cycle as the remaining deliverables become available, and culminates in the 'Conduct Final Tests' activity.

Relationship to CC Assurance Requirements

62. The table below shows the relationship of each of the CC assurance requirements (again taking EAL4 as the example assurance level), and how the requirements are addressed during the evaluator activities for the vulnerability-centric approach.

Assurance class/family	Vulnerability-centric activities
ASE	Covered by the usual CEM work units. ST evaluation is unaltered by the approach.
ADV_FSP	The evaluator accumulates evidence for all of the ADV family work units progressively during the vulnerability-centric activities. For example: 'Familiarise with SFRs' involves tracing SFRs into the design deliverables using representation correspondence information and the evaluator's independent analysis. In this process the evaluator will form and refine judgements about attributes of each of the design deliverables, including completeness, consistency, accuracy, sufficiency of informal explanatory text, etc. However, no separate activity is envisaged to judge completeness, and consistency in isolation.
ADV_ARC	
ADV_TDS	
ADV_IMP	
AGD_PRE	Covered by the usual CEM work units. Although the evaluation activities are generally unaltered, the vulnerability-centric approach gives an earlier opportunity to test the deliverables for installation, generation and start-up when establishing the initial TOE. This early exposure to installing the TOE may also aid the identification of potential vulnerabilities in the delivery or installation processes.
AGD_OPE	This is covered by the usual CEM work units, but could be started (and perhaps completed) at an earlier stage when the initial TOE is established.
ALC	
ALC_CMC	Covered by the usual CEM work units.
ALC_CMS	The site visits for these families may be combined with other visits related to establishing the initial TOE.
ALC_DEL	Covered by the usual CEM work units. Evaluation of delivery is unaltered, except where site visits to assess the delivery process may be linked to other site visits. If appropriate, the delivery process may be used (hence providing direct evidence for evaluation) when establishing the initial TOE.
ALC_DVS	Covered by the usual CEM work units. The site visits for this family may be combined with other visits related to establishing the initial TOE.
ALC_LCD	Covered by the usual CEM work units. Lifecycle evaluation is generally unaltered by the vulnerability-centric approach, although the site visits and establishment of an initial TOE may give more opportunities to observe details of the lifecycle.

Assurance class/family	Vulnerability-centric activities
ALC_TAT	Covered by the usual CEM work units. Evaluation of tools and techniques is unaltered by the vulnerability-centric approach.
ATE_FUN	The evaluator accumulates evidence for all of the ATE family work units progressively during the vulnerability-centric activities, mainly through the creation of the initial tests, the analysis of test results, and the creation of the final tests. In this process the evaluator will typically use the test deliverables to check his/her understanding of the TOE and the SFRs, and also as an input to the design of tests (as implied by, for example, [CEM, para 1388] and ATE_IND.2-4). The evaluator forms and refines judgements about attributes of each of the test deliverables, including completeness, consistency, accuracy, sufficiency of informal explanatory text, etc. The test coverage and depth can be assessed at an early stage, such as when familiarising with the SFRs, and judgements on the adequacy of developer test documentation for ATE_FUN are generally assisted by the early establishment of a TOE and perhaps the initial development site visit (if the opportunity to observe or to use the developer's test system is available as part of the visit).
ATE_COV	
ATE_DPT	
ATE_IND	
AVA_VAN	The essential point of the vulnerability-centric approach is that (evaluator) vulnerability analysis starts at a very early stage, and its requirements as used to drive other evaluator actions and to generate the results for other CEM work units.

ETR Structure

63. The vulnerability-centric approach is also related to a new structure suggested for ETRs. The conventional ETR structure tends to be based around [CEM, Figure 5], which shows the information content required ([CEM] does not mandate the use of this structure). However, an approach driven by vulnerability analysis and penetration testing suggests an alternative structure that reflects the way in which evaluators develop detailed vulnerability-oriented understanding of the TOE and then complete other aspects such as ADV and ATE as a consequence of this (rather than as prerequisite objectives in their own right).
64. The suggested structure is not intended to be mandatory: the evaluators should, as ever, adapt the reporting to the nature of the evaluation and the TOE (provided the [CEM] requirements are met, of course). However, the structure represents another view that may help to convey a detailed understanding of the vulnerability-centric approach. It is also intended to ‘draw out’ the new approach, since the structure for reporting will inevitably tend to define the structure of the evaluators’ work.
65. The structure is as follows¹⁷:

1. Introduction

This section is a typical introduction but in particular contains a subsection (not necessarily the first):

- 1.1. Approach to Evaluation

This section describes the way the evaluation was carried out in practice. For example, it identifies:

- *what site visits were made*
- *when, where and how the TOE was set up and tested (and hence how many different test phases were carried out)*
- *any specific approaches adopted to overcome constraints such as where final deliverables (or final versions of the TOE) were not available for some test phases.*

2. Architectural Description of the TOE

This section is unaltered from the conventional structure, and reflects the importance of understanding the TOE architecture in order to understand the evaluation conclusions. Additional comments about the security architecture might be made here if they would be useful as references from the detailed results in section 4 (see below).

¹⁷ This structure includes only the main reporting sections, intending to draw attention to the link to the vulnerability-centric approach. Other sections to cover [CEM] requirements, such as Glossary, References/List of Evaluation Evidence, and Observation Reports are, of course, assumed to be included as well.

3. Work Unit Results

This section describes how the evaluators have completed the individual work items required by [CEM] (or equivalent, where assurance requirements not included in [CEM] are used).

It is suggested that it is appropriate to use a simple table-based report, rather than a more discursive textual description, for the work units, providing comments and description as required by [UK007]. Where appropriate, further comments can be made in a set of notes to the tables, and references can be made to detailed evaluation work in section 4 (see below) to back up conclusions given in the table.

Part of the point about using this table-based reporting is that it again reflects that many of the work unit conclusions arise out of more general tasks (such as vulnerability analysis, independent tracing of security functions, or planning penetration tests), and that the work unit conclusions are simply deduced from these tasks.

Some examples of table-based reporting are shown below.

4. Detailed Evaluation Results

The purpose of section 4 is to describe the evaluation conclusions as they relate to vulnerability and exploitation themes, without being bound by CC terminology and concepts that may not fit the 'hacker viewpoint' that needs to be taken for vulnerability analysis and penetration testing. For example, it may be useful to consider all internet-based penetration attacks together (regardless of varying Security Functions involved in each attack scenario), or all key generation aspects, or all power analysis aspects.

This section is a free-format part of the ETR (it could equally be presented as one or more annexes or appendices) that describes relevant detail for the evaluator actions. The following is illustrative, with sections 4.3-4.5 in this example being specific to the type of TOE being evaluated):

4.1. Analysis and Test Strategies

This section describes the evaluators' overall test strategies and the sampling strategy for the ADV_IMP activity (where the evaluator considers these too detailed for the table-based reporting presented in section 3). It may also include the reports required by CEM on developer or evaluator test effort as well as test result summaries.

4.2. SFR Tracing and Analysis

This section describes the evaluators' tracing of SFRs into the design and test deliverables (and possibly other relevant deliverables for AGD and ADO).

4.3. Access Control Bypass – Vulnerability Analysis

For the purposes of example we assume that the evaluators established that access controls represented a strategic area in which vulnerabilities should be investigated and that this section describes the vulnerability analysis (including any references to sources for vulnerabilities, descriptions of systematic analysis, and penetration tests arising from the developer and evaluator vulnerability analyses).

4.4. Side-Channel Analysis

For the purposes of example we assume that the TOE included hardware that was susceptible to side channel analysis (e.g. power analysis and emanations analysis). In this case much of the analysis will be based on direct measurement by the evaluators rather than features present in the design and test deliverables. This section is therefore used to describe the details of the vulnerability analysis that leads to particular measurements and tests, as well as description of the test methods and results.

4.5. Random Number Generation – Vulnerability Analysis

For the purposes of this example we assume that the TOE provides random number generation (whether as a service to users, or as support to other security functions) and required evaluation. In this case the section would contain vulnerability analysis of the design and suitable test descriptions and results (e.g. reports from running the NIST test suite, or results from running tests in [AIS20]).

4.6. Development Environment Assessment

This section gives details of site audits, interviews and other material used to support the conclusions for the ALC_CMC, ALC_CMS and ALC_DVS activities.

5. Conclusions and Recommendations

This section contains the verdicts for evaluator action elements, notes and conclusions on any residual vulnerabilities, and other evaluator recommendations arising from the evaluation. (This could be combined with another section, such as section 1 or section 3, but it is important to report these aspects in clearly identified locations in the ETR.)

Examples of Table-Based Reporting

CEM Requirement (EAL4)	Comment
ASE_SPD.1	
ASE_SPD.1-1 The evaluator shall check that the security problem definition describes the threats.	Met in [ST, 3]. See note 1 in section XXX.
ASE_SPD.1-2 The evaluator shall examine the security problem definition to determine that all threats are described in terms of a threat agent, an asset, and an adverse action.	Met in [ST, 3]. See note 1 in section XXX.
ASE_SPD.1-3 The evaluator shall check that the security problem	Met in [ST, 3].

CEM Requirement (EAL4)	Comment
definition describes the OSPs.	See note 2 in section XXX
ASE_SPD.1-4 The evaluator <i>shall examine</i> the security problem definition to determine that it describes the assumptions about the operational environment of the TOE.	Met in [ST, 2 & 3].

66. Notes to the Table:

- 1 *This note might, for example, comment on the applicability of some of the threats – e.g. noting where threats may overlap, or to clarify how they are interpreted in tests.*
- 2 *This note might, for example, comment on the relevance and applicability of the Organisational Security Policies.*

CEM Requirement (EAL4)	Comment
ALC_DEL.1	
ADO_DEL.1-1 The evaluator <i>shall examine</i> the delivery documentation to determine that it describes all procedures that are necessary to maintain security when distributing versions of the TOE or parts of it to the consumer.	<p>[DEL] describes the delivery process and gives guidance to customers on how to maintain TOE security during the delivery process (and after receipt of the TOE).</p> <p>[DEL] and [OP] provide basic information about the assets, threats and countermeasures involved in the process of delivery (and subsequent secure operation). This includes both checks of documents ([DEL, 2]) and an integrity check function using the TOE itself.</p> <p>Keys necessary for issuance are delivered securely according to ([DEL, 3.4]) to protect their confidentiality and to allow detection of modification.</p> <p>The procedures in [DEL] alert users to the possibility of attempted masquerading and include procedures to confirm that a delivery conforms to items ordered by the user.</p>

CEM Requirement (EAL4)	Comment
<p>ADO_DEL.1-2 The evaluator <i>shall examine</i> aspects of the delivery process to determine that the delivery procedures are used.</p>	<p>The release process from the developer site was examined during the site visit in <i><date of visit></i>, and is described in more detail in section <i><subsection within 'Detailed Evaluation Results' section></i>.</p>

Reporting Vulnerability Analysis in the ETR

Purpose

67. The aim of the vulnerability analysis part of the ETR is to present the evaluators' analysis in a way that is freed from the structure imposed by the CEM work units. The following points should be noted in particular:
- a) The intent is that the evaluators write up this part of the ETR in 'note' form as the evaluation proceed, and amend the notes as necessary as their understanding of the TOE improves, and hypotheses about the TOE behaviour are confirmed or disproved. At the same time, the report is expected to demonstrate that the analysis has been sufficiently comprehensive, by conforming to certain minimum requirements (described here) regarding its structure and content.
 - b) By 'vulnerability analysis' we mean an analysis that is not strictly confined to the AVA_VAN activity, but rather one that explicitly draws on all other evaluation activities that contribute to it. This reflects the focus of the vulnerability-centric evaluation process. This part of the ETR will therefore provide an essential contribution to the reports for the ASE, ADV, AGD and ATE activities, as well as the AVA activity. (By contrast the development environment activity (ALC) will normally be separate, and as such reported in a separate section.)
68. This section focuses on the outputs from the vulnerability analysis. This includes guidance on flaw hypotheses that may feature in the evaluators' notes.

Structure of the Analysis Report

Top-level structure

69. The vulnerability analysis part of the ETR should be structured along the following lines:
- a) Firstly, there will be a number of sections that deal with SFR-specific aspects of the vulnerability analysis. These correspond to section 4.2 of the ETR structure described above, and will (collectively) cover all of the SFRs defined in the ST. A given section may cover a single SFR, or it may address several closely related SFRs should the evaluators judge that there is sufficient commonality between their respective analyses. (For example, evaluators might choose to group SFRs together by security policy.)
 - b) Secondly, there may be one or more sections dealing with particular types of vulnerabilities (possibly specific to the TOE type, or the technology on which it is based), where these issues transcend several (and perhaps all) SFR boundaries. (These correspond to sections 4.3 to 4.5 in the ETR structure illustrated above.) No further requirements are specified on the content of these sections, as this will depend on the type of vulnerability being addressed.

Structure within SFR-focused sections

70. Within each SFR-focused section¹⁸, there will be the following subsections, as appropriate to the SARs defined in the ST:
- a) Understanding of the SFR
 - b) Functional Specification
 - c) TOE Design Specification - subsystem level
 - d) TOE Design Specification - module level
 - e) Implementation Representation
 - f) Guidance and Misuse¹⁹
 - g) Functional/Independent Tests
 - h) Penetration Tests.

Detail of the Analysis

71. We now expand in greater detail what notes are expected in each subsection²⁰ of an SFR-focused analysis. This section inevitably takes a 'linear' approach in describing reporting against each level of activity (functional specification, subsystem level design, etc). It is therefore important to note that this 'linearity' relates only to the reporting structure, and not to the evaluation activities themselves.
72. It will be observed that the SFR-focused analysis reports are intended to cover all the traceability analysis reporting requirements of [UK007]. Evaluators are, nonetheless, free to choose to separate out any detailed traceability analyses to a separate annex or annexes. Such an approach might, for example, be appropriate if the traceability analysis proves to be unduly complex, such that the details of the report would distract attention from the rest of the vulnerability analysis report. Evaluators are reminded, however, that the traceability analysis is itself only a means to the end of finding vulnerabilities, and not an end in itself. If the traceability analysis report includes an excessive amount of detail, this is likely to detract from that end-goal.

¹⁸ The descriptions which follow are written as if the SFR-focused section covers only a single SFR. The reader is reminded that in practice a section may cover several SFRs.

¹⁹ Whilst CCv3.1 does not (in contrast with CCv2.3) require a developer-produced Misuse Analysis, Misuse is nonetheless recognised as an issue to be addressed by the evaluator's independent vulnerability analysis.

²⁰ In some cases it may be appropriate to report on each SFR in a table, in which case the 'subsections' could be rows. This would be applicable where the SFRs are relatively simple and clearly defined. In other cases, the need to map from the CC language of SFRs to the language of the TOE design documents may require separate subsections.

The flaw hypothesis approach

73. At the heart of the vulnerability analysis is the *flaw hypothesis approach*. This is described in the following terms in CEM for AVA_VAN.2:

A search of the evidence should be completed whereby specifications and documentation for the TOE are analysed and then potential vulnerabilities in the TOE are hypothesised, or speculated. The list of hypothesised potential vulnerabilities is then prioritised on the basis of the estimated probability that a potential vulnerability exists and, assuming an exploitable vulnerability does exist the attack potential required to exploit it, and on the extent of control or compromise it would provide. The prioritised list of potential vulnerabilities is used to direct penetration testing against the TOE. [CEM, 1455]

74. This follows closely the commonly accepted definition of a flaw hypothesis methodology. CEM, however, reserves explicit usage of the term *flaw hypothesis* for AVA_VAN.3 (and AVA_VAN.4), describing it in these terms:

A flaw hypothesis methodology needs to be used whereby specifications and development and guidance evidence are analysed and then potential vulnerabilities in the TOE are hypothesised, or speculated.

The evaluator uses the knowledge of the TOE design and operation gained from the TOE deliverables to conduct a flaw hypothesis to identify potential flaws in the development of the TOE and potential errors in the specified method of operation of the TOE. [CEM, 1504-5]

75. In the discussion which follows, we use the term ‘flaw hypothesis’ in a more general sense, referring to the method to be adopted regardless of which AVA_VAN level is the target.

76. The need for prioritisation at AVA_VAN.2, and the notion of a focused search at AVA_VAN.3, is an explicit recognition in CEM that an independent vulnerability analysis does *not* involve an exhaustive search for vulnerabilities. Rather, it recognises that evaluation resources are not unlimited, and that evaluators must therefore make a considered judgement as to where to those resources may be best deployed.

77. It is worth considering in a little more detail the factors involved in the process of prioritisation that applies at AVA_VAN.2:

- a) *Estimated probability*. This is not (of course) a call for evaluators to assign absolute probability values to every potential vulnerability, but rather a requirement to make a judgement based on the grounds for suspicion that they have. The judgement of probability may change as the evaluators trace SFRs into more levels of design (or as tracing of more individual SFRs is carried out). In some cases, initial experiments may be necessary before the probability can be estimated: for example, where a smart card TOE is to be assessed for power analysis vulnerabilities, it is likely that initial experiments to characterise the behaviour of the TOE will be used to indicate whether power analysis vulnerabilities are likely to exist.
- b) *Attack potential required*. This should be read as an instruction to (where possible) test those vulnerabilities first that will be most easily exploitable. It is also a recognition that it is not necessary to test every vulnerability, if it can be clearly shown

by analysis that the attack potential required to exploit it would be beyond the scope of interest of the AVA_VAN level.

- c) *Extent of control or compromise*. As with attack potential considerations, this should be read as an instruction (where possible) to test those vulnerabilities first that provide the greatest scope for compromise of assets. However, it would be expected that the evaluators would still test every suspected vulnerability revealed by the search where it is practical to do so (and the vulnerability has not been ruled out on the grounds of either low probability or excessive attack potential).
78. This notion of prioritisation is not an explicit feature of AVA_VAN.3 (other than in the case where the likely attack potential required to exploit the vulnerability is a factor [CEM, 1531]). However, the *focused* search approach to vulnerability analysis, which CEM states may be used for AVA_VAN.3, does itself involve a degree of *implicit* prioritisation. A focused search is driven by an identification of ‘areas of concern’ [CEM, 1924], directing evaluator effort towards those areas that the evaluators consider (based on the available evidence) to have the highest probability of revealing vulnerabilities. In this approach, the evaluators are not so much prioritising potential vulnerabilities that have been identified as prioritising the parts of the evidence to examine in order to find them.
 79. Evaluator judgement over the deployment of limited resources is also required once potential vulnerabilities have been identified. As a general principle, evaluators should always seek the most cost effective means of determining whether identified potential vulnerabilities actually exist in practice. In some cases testing will prove to be the most appropriate; in other cases, analysis of (for example) the implementation representation may provide the most efficient approach.
 80. The discussion which follows draws on the independent vulnerability analysis guidance that is presented in CEM, and which is expanded upon in [GVSA]. This provides a framework for the flaw hypothesis approach, helping to guide the evaluators’ thought processes, but not acting as a restraint on their ingenuity.

Minimum content for a vulnerability analysis report

81. We now discuss what should be covered (as a minimum) under each subsection heading. The minimum content directs what an evaluator is expected to do as part of their vulnerability analysis. In addition to this, the evaluator can record significant notes in a ‘free-format’ form; these may include:
 - a) points of understanding relating to the SFR (as appropriate to the section)
 - b) issues to be investigated or followed up in subsequent activities, for example to highlight concerns about design complexity, or a suspicion of a potential vulnerability.
82. Evaluators are not expected to record every single flaw hypothesis in the ETR. Rather, they should use their judgement as to whether a particular hypothesis is of sufficient interest to merit discussion in the ETR. For example, discussion would generally be appropriate in the case of those hypotheses that would be obvious to the reader of the ETR (where absence of discussion could otherwise be misinterpreted by the reader), and also where it would provide useful insights into the TOE’s behaviour, design or

implementation. By contrast, the report would not (in general) be expected to discuss a flaw hypothesis that has been discarded on the grounds that the likelihood of it existing is exceedingly remote.

83. In general, an evaluator could expect to write the notes as a first draft of the ETR content – this is useful both in terms of making ETR editing and production more efficient, and in making visible the final requirement for evidence (so that the evaluator remains aware of what remains to be done for each SFR and/or deliverable).
84. Note that some of the expected content described below may be moved to separate sections that deal with specific types of vulnerability (cf. sections 4.3-4.5 in the example ETR structure above).

Understanding of the SFR

85. This part of the analysis focuses on the ST content, the intent being to provide a context for the vulnerability analysis which follows. Based on the understanding gained from the evaluation of the ST, the evaluators should include notes on:
 - a) The consequences of a failure of the SFR (whether by incorrect implementation, incorrect use, or successful bypass or tampering attack, etc). Generally, this will be described in terms of the threats that would or might be realised, and the assets that would be compromised, in this eventuality.
 - b) Any other noteworthy aspects relevant to the SFR, for example if the SFR will be susceptible to direct attack, or if there are closely related SFRs (e.g. linked through dependencies), or if understanding of the SFR is helped by discussing it in terms of the underlying CC functional paradigm.
86. These notes may be compiled at the time of ST evaluation, but should in any case be completed during the ‘Familiarise with SFRs’ activity.

Functional Specification

87. The aim of the FSP analysis is to hypothesise or identify vulnerabilities based on information provided in the functional specification. In practice, given the nature of the evidence presented in the FSP, evaluators are more likely to *hypothesise* the existence of vulnerabilities than *identify* actual vulnerabilities at this stage. The FSP analysis will therefore provide a focus for the subsequent analysis and testing activities (during which the hypotheses can be confirmed or disproved). Early testing of the TOE is thus likely to prove particularly valuable in helping to inform the FSP flaw hypothesis process.
88. The FSP analysis is likely to involve the following stages:
 - a) **Familiarisation** with the various types of interface through which external entities interact with the TOE, and how the individual TSF interfaces are specified.
 - b) **SFR-focused analysis** where, for each SFR, the evaluators use the developer’s mappings to identify the TSF interfaces that are relevant to the SFR, i.e. those TSF interfaces that will be used to stimulate the SFR, and hence test it. The evaluator then examines the TSF interfaces to gain an understanding of each in terms of purpose,

inputs, and any SFR-relevant effects. The evaluators then hypothesise potential flaws in those interfaces that could be exploited to subvert the SFR.

- c) **Analysis completion** where the evaluators complete their review by covering the remaining TSF interfaces (i.e., those that are not mapped to any SFR). To ensure an appropriate VAN-focus, the evaluators should first determine a strategy for identifying potentially vulnerable TSF interfaces that should be targeted in subsequent analysis activities and/or testing. This strategy then guides the review of those interfaces, for example by generating a checklist of TSF interface characteristics (such as purpose, interface options, effects or error messages) that should raise evaluator suspicions (or at least merit further investigation). Such a strategy will be founded on a flaw hypothesis approach, taking into consideration known generic vulnerabilities relevant to the TOE type²¹.

89. It is likely that the bulk of this analysis (and hence this part of the report) will be done during the ‘Familiarise with SFRs’ activity (particularly the first two parts), and that it will be essentially complete by the end of the ‘Establish Initial Working TOE’ activity.
90. The review of TSF interfaces performed at each of the stages above provides input to the content and presentation of evidence checks (ADV_FSP.*.1E). Whilst performing the vulnerability analysis, the evaluators note any instances where the evidence fails to provide the level of detail required by the ADV_FSP criteria, or which contradicts other evidence provided.
91. The ‘SFR-focused analysis’ (Stage 2) generates the traceability analysis results (ADV_FSP.*.2E) as one of its outputs. In this part of the vulnerability analysis report, the evaluators’ notes should identify (in appropriate form) which TSF interfaces are relevant to the SFR. The notes should then go on to identify or hypothesise the existence of possible vulnerabilities, based on this information (completing the output from Stages 2 and 3).
92. The types of issue the evaluator should consider when constructing the flaw hypothesis include (but are not limited to) whether the SFR may be susceptible to:
- a) *Sequence manipulation* attacks. Such attacks will be suggested during the ‘SFR-focused analysis’ (Stage 2) if the functional specification describes SFR-relevant operations that follow a defined sequence of events, where the sequence itself may be subject to the influence of an attacker²². Sequence manipulation includes changing the expected order, missing out a step in the sequence, or injecting some other step (e.g. another TSF interface call) or entity (e.g. as in a ‘man-in-the-middle’ attack).
 - b) *Unexpected input* attacks. This includes provision of malformed input to a TSF interface. As a general principle, any TSF interface could be vulnerable to such attacks, and the functional specification may provide few pointers as to which

²¹ For example, evaluators may have established that the obvious covert channels are a concern to the evaluation. A strategy might thus be devised to flag interfaces whose effect is the modulation or observation of resources that are not subject to the information flow control policy, as well as those that generate error messages that could act as inference channels.

²² In this context a ‘sequence’ is simply a series of events involving some form of exchange of data between two or more entities. These will involve users (where the sequence may comprise a series of TSF interface calls and responses) or external systems/components (where the sequence follows a particular protocol).

interfaces should be targeted (analysis of the design or implementation is more likely to highlight candidates for testing). Nonetheless, the evaluators should (during the ‘SFR-focused analysis’, Stage 2), consider whether there are TSF data input parameters that have explicit or implicit ranges of permitted values. Indirect inputs (e.g. security relevant configuration or initialisation files) should also be identified for further investigation.

- c) *Unexpected purpose* attacks. As with the previous type of attack, the functional specification may not provide any significant pointers. Nonetheless, evaluators should consider the developer provided mappings for the SFR and, during the ‘analysis completion’ (Stage 3), consider whether there are any obvious TSF interfaces that appear to have been missed by the developer (and which may therefore offer simple bypass routes).
- d) *Privilege Inheritance* attacks. This form of attack will be relevant if the TOE includes privileged applications or commands that have access rights to bypass or override the SFR. Attacks against the application interface may lead to an uncontrolled exit to an operational state where the attacker has inherited (or can exploit to inherit) the privileges of the application. The potential for such flaws should be investigated during the final stage of the analysis (Stage 3).

TOE Design Specification – Subsystem Level

- 93. The aim of the subsystem-level design analysis is to hypothesise or identify vulnerabilities based on information provided in respect of the TSF subsystems, their functionality and interactions (including dependencies). In practice, the emphasis will again be on *flaw hypothesis* rather than *identification*, owing to the relatively abstract nature of the information being presented to the evaluators²³. As such it will guide subsequent analysis (e.g. of the module-level design at EAL4) and/or testing.
- 94. The subsystem level design analysis is likely to involve the following stages:
 - a) **Familiarisation** with the TOE architecture, identifying the subsystems that comprise the TOE, and their general responsibility in the enforcement of security. (As a minimum, each evaluator should acquire a level of understanding that is commensurate with the architectural description required by CEM.)
 - b) **SFR-focused analysis**, where, for each SFR, the evaluators use the developer’s mappings to identify the TSF subsystems that are involved in providing the SFR, and their responsibility in ensuring that the SFR is enforced. The evaluators determine how the assets protected by the SFR are represented in the design, and note the high-level dependencies of the SFR (for example the data or shared resources that it relies on, and how it is protected). The evaluators hypothesise potential vulnerabilities, such as tampering attacks based on manipulation of the identified SFR dependencies.
 - c) **Analysis completion**, where the evaluators complete their review of the TSF subsystems (i.e. covering those that are not mapped to any SFR). The focus of this review is to identify any areas of concern for further investigation. This will be

²³ Any bypass or tampering attacks suggested by the subsystem design level evidence should be obvious to the developer, and (as such) one would expect these to have been addressed at an early stage of development.

achieved by implementation of a strategy based on the hypothesis of flaws that might be evident in the subsystem level design, e.g. arising from interference between subsystems via shared resources, or from potential bypass routes offered by subsystem interfaces. The evaluators should also make appropriate use of information in the security architecture description (ADV_ARC), which will describe how the TSF protects itself from bypassing and tampering attacks.

95. It is likely that the bulk of this analysis (and hence this part of the report) will be done during the ‘Familiarise with SFRs’ activity (particularly the first two parts), and that it will be essentially complete by the end of the ‘Establish Initial Working TOE’ activity.
96. The review of the high-level design performed at each of the stages above provides input to the content and presentation of evidence checks (ADV_TDS.*.1E). The evaluators should note any instances where the evidence fails to provide the level of detail required by the ADV_TDS criteria, or which contradicts other evidence provided.
97. The SFR-focused analysis (Stage 2) provides the traceability analysis results (ADV_TDS.*.2E). In this part of the vulnerability analysis report, the evaluators’ notes should identify (in appropriate form) which TSF subsystems are relevant to the SFR, and their role in providing it. The notes should then go on to highlight how the assets are represented in the design and the identified dependencies of the SFR (e.g. on data files and other shared resources), and should identify or hypothesise the existence of possible vulnerabilities, based on this information (completing the output from Stages 2 and 3).
98. The types of issue the evaluator should consider when constructing the flaw hypothesis include (but are not limited to) whether the SFR may be susceptible to:
 - a) *Sequence manipulation* attacks. Such attacks will be suggested at Stage 2 (‘SFR-focused analysis’) if the TDS (viewed in conjunction with ARC) indicates that any TSF subsystem involved in providing the SFR is susceptible to external interference. This would typically be the case where the TSF subsystem is a process that could be killed, set to a lower priority, or otherwise disrupted by another entity locking a critical resource.
 - b) *TSF data* attacks. Such attacks will be suggested at Stage 2 (‘SFR-focused analysis’) through consideration of how TSF data (on which each SFR relies) is stored and protected.
 - c) *SFR dependency* attacks. Such attacks may be identified through the application of lateral thinking at Stage 2 (‘SFR-focused analysis’), identifying (through consideration of the design or knowledge of similar TOEs) dependencies the SFRs might have on (for example) such things as the availability of system resources, the values taken by system-wide configuration parameters, or external influences in the environment.
 - d) *Shared resource* attacks. Such attacks will be suggested at Stage 2 (‘SFR-focused analysis’) and Stage 3 (‘analysis completion’) where the subsystem level design identifies resources that are used by different subsystems. Such resources should be investigated for their potential for providing illicit access to sensitive user or TSF data (e.g. cleartext passwords or keys), or providing the means for illicit data flows.

TOE Design Specification – Module Level

99. The aim of the module level design analysis is to hypothesise or identify vulnerabilities based on information provided regarding the TSF modules, their functionality, interfaces (including use of shared resources such as global data structures) and their interrelationships. In practice, the analysis will be closely linked to the analysis of the implementation representation required by ADV_IMP.
100. The module level design analysis is likely to involve the following stages:
- a) **Familiarisation** with the module-level design, identifying what are the modules, and how they are specified. The evaluators also gain an understanding of how the module specifications relate back to the information in the subsystem-level design (what modules are associated with a given subsystem) and the functional specification (where module interfaces are externally visible).
 - b) **SFR-focused analysis** where, for each SFR, the evaluators use the developer's mappings to identify the modules that are involved in providing the SFR, and their responsibility in ensuring that the SFR is enforced. At this point the evaluators will identify the modules whose correct implementation is most critical to the SFRs, and also any SFR-enforcing modules whose design is especially complex. Such modules should be identified for inclusion in the ADV_IMP activity; to direct that analysis, the evaluators may hypothesise flaws that could arise from that complexity.
 - c) **Analysis completion** where the evaluators complete their review of the module specifications, covering those modules that are not mapped to any SFR. As with the FSP and subsystem-level design analyses, this review is to be guided by a flaw-hypothesis based approach, aimed at identifying potential vulnerabilities or significant concerns to be followed up in the ADV_IMP activity. The evaluators hypothesise the types of vulnerability that are most likely to become visible in this TSF representation (guidance on this is provided below), and determine how these might be identified from a review of the remaining parts of the module-level design.
101. Ideally this analysis (and hence this part of the report) will be started during the 'Familiarise with SFRs' activity, with a focus on aspects of particular interest. The analysis will continue through subsequent activities, in an incremental fashion, but should be essentially complete by the end of the 'Analyse Test Results' activity.
102. The review of the TSF modules performed at each of the stages above provides input to the content and presentation of evidence checks (ADV_TDS.*.1E). The evaluators note any instances where the evidence fails to provide the level of detail required by the ADV_TDS criteria of the module specifications (purpose, interface, interrelationships, and so on), or which contradicts other evidence provided.
103. The 'SFR-focused analysis' (Stage 2) provides the traceability analysis results (ADV_TDS.*.2E). In this part of the vulnerability analysis report, the evaluators' notes should identify (in appropriate form) which modules are relevant to the SFR, and what their role is in providing it. Modules of particular concern (owing to their criticality in enforcing SFRs, or the complexity of their design, etc.) should also be noted for further investigation in the ADV_IMP activity. The notes should then go on to highlight the

identified dependencies of the SFR (e.g. global data structures which store TSF data), and identify or hypothesise the existence of possible vulnerabilities, based on this information (completing the output from Stages 2 and 3).

104. The types of issue the evaluator should consider when constructing the flaw hypothesis include (but are not limited to) whether the SFR may be susceptible to:
- a) *TSF data* attacks. Such attacks will be suggested at Stage 2 ('SFR-focused analysis') through consideration of how temporary copies of TSF data on which each SFR relies is stored and processed. Modules that process TSF data that is expected to be in a particular format, or have a value that falls in a given range, should be targeted in the ADV_IMP activity to ensure that the implementation correctly handles scenarios where those assumptions are invalidated.
 - b) *Shared resource* attacks. Such attacks will be suggested at Stage 2 ('SFR-focused analysis') and Stage 3 ('analysis completion') where the module-level design identifies resources that are used by different modules (e.g. global data structures, temporary files). Such resources should be investigated for their potential for providing illicit access to sensitive user or TSF data (e.g. cleartext passwords or keys), or for providing the means for illicit data flows.
 - c) *Interface input* attacks. Such attacks will be suggested at Stage 2 ('SFR-focused analysis') where particular modules are identified as providing externally visible interfaces, and which should therefore validate SFR-relevant data inputs e.g. input parameters to modify the values of TSF data items. Such attacks may also be suggested at Stage 3 ('analysis completion') to follow-up any evaluator suspicions raised during analysis of the functional specification.
 - d) *Unexpected purpose* attacks. Such attacks may be suggested at Stage 3 ('analysis completion') through consideration of module interrelationships. It may be possible to discover simple bypass routes from an externally visible interface to modules that provide functionality that should be protected by a SFR, but which do not themselves enforce any SFR.

Implementation Representation

105. This section of the analysis should describe the application of the overall analysis strategy (described elsewhere in the ETR) as it relates to this SFR. The evaluators' notes should include an identification of those parts of the implementation representation that were examined, the reason for choosing that module in the sample, and the results of the investigations.
106. As with the module-level design analysis, ideally this analysis (and hence this part of the report) will be started during the 'Familiarise with SFRs' activity, with a focus on aspects of particular interest. The analysis will continue through subsequent activities, in an incremental fashion, but should be essentially complete by the end of the 'Analyse Test Results' activity.

Code Analysis

107. Evaluator notes should hypothesise the existence of code level vulnerabilities (using

information in the design, coupled with a knowledge of relevant generic vulnerabilities), and describe how the evaluators searched the code for such vulnerabilities. This again may form part of an overall strategy described elsewhere in the ETR (section 4.1 in the example ETR structure above), in which case the evaluators' notes should simply describe how the strategy was applied to this SFR.

108. The evaluators should target their analysis at those parts of the code that are:
 - a) Determined to be critical to the correct implementation of the SFR.
 - b) Responsible for the validation of inputs to targeted TSF interfaces.
 - c) Suspected to contain actual vulnerabilities (based on previous analysis activities).
 - d) Judged to have the greatest likelihood for containing vulnerabilities, e.g. based on consideration of design complexity.
 - e) Identified as worthy of further investigation, based on evaluator hypothesis of specific types of vulnerability that may be present in the code (e.g. buffer overflows) and which may be readily searched for (e.g. calls to 'unsafe' functions).
109. The first four cases of modules to be targeted should be identified during the low-level design analysis. The evaluator focus in these cases is on finding actual coding errors that are likely to undermine a SFR, or on constructing test cases to exercise code paths that might contain implementation errors in the SFR.
110. The fifth case relies less on input from the design analysis, and more on a knowledge of generic vulnerabilities that might be readily identified in the code. Using the flaw hypothesis approach, the evaluators search for code statements across the implementation representation that indicate that a particular module or function should be included in the sample to be analysed. Such statements might (for example) include calls to 'unsafe' functions, or updates to security critical global data. Any modules or functions thus identified for further analysis should then be inspected to determine whether the suspected vulnerability exists.

Hardware Analysis

111. The implementation representation for a hardware TOE will typically consist of a combination of a hardware design language (such as Verilog HDL for digital blocks) and layout information (for both analogue and digital blocks). However, the same general principles apply as for software module analysis: tracing and interpreting the security functions (assisted by the representation correspondence deliverables) leads the evaluator to the parts of the HDL and layout to be targeted. Particular concerns from the hardware point of view are likely to be driven by specific types of potential hardware vulnerability such as: probing to read or modify critical data, structures giving access to privileged modes of operation, structures that may disable or inhibit other security functions, or structures that may provide targets for specific attacks such as power analysis of internal signals instead of interface signals.
112. As noted earlier, in some cases it is likely that all or part of the hardware implementation representation will be available only at the developer's site. This may be for confidentiality

reasons, but also reflects the fact that exploring the implementation is likely to need the use of development tools (e.g. to identify critical registers or signal lines in HDL and then locate these in the layout).

Guidance and Misuse

113. This section should include notes on those parts of the guidance that are relevant to the operation of the SFR. It should then discuss the evaluator's hypothesis of possible misuse vulnerabilities (i.e. potential errors in the operation of the TOE, leading to insecure states arising from insecure use or configuration). This should draw on the results of other analyses where appropriate. For example, code analysis may determine that the TOE fails to properly handle TSF data if it is outside an expected range of values, but that this can only be set by an administrator. Evaluators should then hypothesise misuse scenarios in which an administrator might inadvertently set the TSF data to an 'unexpected' value.
114. This analysis (and hence this part of the report) will be started during the 'Establish Initial Working TOE', and be essentially complete by the end of the 'Analyse Test Results' activity.

Functional/Independent Testing

115. Notes in this section should describe how the overall test strategy applies to the testing of the SFR. This may include an identification of any developer tests that were repeated, or any independent tests relating to the SFR.
116. Testing (and hence this part of the report) could be started as early as the 'Familiarise with SFRs' activity (when specific tests might be identified). It is nonetheless envisaged that the majority of the work will be performed from 'Establish Initial Working TOE' onwards, and that it will be complete by the end of the 'Conduct Final Tests' activity.
117. The 'SFR-focused analyses' from the ADV sub-activities should influence the evaluators' independent testing, for example to:
 - a) Exercise TSF interface options that were not fully tested by the developer.
 - b) Focus testing on those SFRs whose design is found to be especially complex (and hence a potential source of implementation errors).
 - c) Exercise code paths identified in critical modules that may not have been fully covered by the developers.

Penetration Testing

118. Notes in this section describe the process of bringing together the vulnerability analysis described in the preceding sections and formulating them first into penetration test ideas, and then into scripts. Scripts relevant to the SFR should be referenced, their objectives identified, and results and conclusions discussed. The notes should also cover any new hypotheses or tests that arose from this activity.
119. Evaluators may choose to group all penetration testing under a single section rather than disperse it amongst the SFR-focused analyses. Or they may choose to report the testing at the level of defined SFR groups (such as Security Audit, Identification and

Authentication). This is acceptable provided there is clear traceability (where relevant) back to those parts of the analysis that prompted the test.

120. Penetration Testing (and hence this part of the report) could be started as early as the 'Familiarise with SFRs' activity (when specific tests might be identified). It is nonetheless envisaged that the majority of the work will be performed from 'Establish Initial Working TOE' onwards, and that it will be complete by the end of the 'Conduct Final Tests' activity.

Conclusions

121. The vulnerability-centric approach is based on a belief that finding vulnerabilities is the fundamental objective of a Common Criteria evaluation. Recognising this, and building on the experience of different types of evaluation (and different evaluation schemes), the core of the approach is therefore the vulnerability analysis and penetration testing of the TOE. Although the traditional 'linear' approach to CC tends to present this as a late activity in an evaluation, the objective of finding vulnerabilities is better served by focusing on vulnerability analysis and practical use of the TOE from as early a stage as possible. The traditional CC conclusions (in ADV, ATE, etc) can be drawn as a consequence of the analysis carried out in order to support vulnerability analysis and penetration testing.
122. The vulnerability-centric approach suggests a different approach to recording results in the ETR. This approach is also believed to give a more useful document for certifiers and other readers, since it is structured around reporting vulnerability-based concerns. Where early drafts of an ETR are delivered to the certifier as part of a 'high visibility' certification strategy, the early emphasis on documenting vulnerability analysis and provisional (incremental) conclusions of other work units can make the progress of an evaluation more visible. This visibility is considered important in minimising any delays in certification time (after final ETR delivery) and also in minimising the danger of any requirement for rework of the ETR. In essence: the certifier can see the important conclusions (and comment on or question the work yet to be done) at an early stage. Reporting using 'themes' based around security functions and vulnerability groups addresses a frequent criticism of ETRs, namely that it is difficult to find the important results and conclusions for the TOE in a report that is structured around reporting against a set of abstract evaluation activities. At the same time, the conventional rigour required in evaluation reporting is maintained by incorporating tables to justify that the traditional requirements for levels of detail in reporting CEM work units.

Appendix – Compliance

Compliance with CC and CEM

123. The proposed evaluation process has been designed to comply with CC and CEM by ensuring that all CEM work units for the SARs are covered and reported to an adequate degree.
124. Completeness of coverage is (of course) readily demonstrated by the table-based reporting of each of the work units, as described above.
125. It might, however, be argued that not all work units (especially the content and presentation of evidence checks) will be necessarily performed to the required depth and rigour. The authors recognise this as a legitimate concern, but believe it to be one that is without foundation. This is because the evidence the evaluators present to the certifiers, demonstrating that they have performed all work units to the required depth and rigour, is no different to that which they would present in a ‘traditional’ linear evaluation. Certifiers will therefore have the same level of confidence that all aspects of the evaluation have been performed to the required standard.
126. Moreover, if we accept the argument that a focus on vulnerability analysis runs the risk of the evaluators failing to identify instances of non-compliance with CC content and presentation of evidence requirements, then we must also conclude that the current approach (with its focus on the CC requirements) runs a similar risk of actually failing to identify vulnerabilities – a rather more serious concern, in the view of the authors.
127. Under the vulnerability-centric approach, the CC content and presentation of evidence requirements are seen as supportive of the vulnerability analysis, but are not an end in themselves. That is, the evaluators require a minimum level of input in order to perform the vulnerability analysis, and this broadly coincides with the CC requirements in respect of the content and presentation of evidence. Hence when performing the vulnerability analysis, the evaluators will also establish confidence that (at least) the *important* aspects of the CC documentation requirements have been met.
128. This is not to say that the vulnerability analysis will cover all CC requirements: there will inevitably be some element of a ‘top-up’ activity at the end of the evaluation to complete the CC documentation checks. (By definition, this ‘top-up’ activity serves only pick up documentation errors that have no bearing on the confidence that no vulnerabilities remain in the TOE²⁴.) The extent of this activity will largely depend on the experience of the evaluators who performed the analysis: experienced evaluators (who have complete familiarity with the CC requirements) will be able to complete more of the CC documentation checks concurrently with the vulnerability analysis.

²⁴ In other words, any Level 2 or Level 3 ORs relating to the CC content and presentation of evidence checks should be raised at the time of performing the vulnerability analysis. By contrast, the ‘top-up’ activity should, at worst, only lead to the raising of Level 4 ORs, since the problems it reveals should have no impact on the vulnerability analysis.

Compliance with UK interpretation 007

129. UK interpretation 007 specifies the minimum information that evaluators should include in the ETR to justify the verdicts they assign. The vulnerability-centric evaluation process is entirely compatible with the interpretation since verdict justifications are still expected to the granularity of individual work units. The following general approach should be adopted to ensure compliance:
- a) The *Affirmation* or *Reference* type of work unit report should be presented in the work unit result tables.
 - b) Elaboration of *Affirmation* or *Reference* type reports can, in most cases, be accommodated either in the work unit result tables themselves, or in notes referenced from the tables. Where a more detailed elaboration is called for, these can be included in the Analysis section(s) of the ETR, and referenced from the work unit result tables.
 - c) The *Report* and *Analysis* type of work unit report should be reported in the Analysis section(s) of the ETR, and referenced from the work unit result tables.
130. The following table illustrates in more detail how the [UK007] requirements may be satisfied under the vulnerability-centric approach. As such, it may be consulted by evaluators for guidance on compliance with [UK007]. This table highlights, for each CEM activity, where the UK interpretation calls for the reporting of details that may not be readily accommodated within the table-based reports in section 3 of the ETR.

Assurance class/family	[UK007] Reporting Requirements
ASE	Fully reported in the work unit result tables, except as follows: <ul style="list-style-type: none"> • Both ASE_OBJ and ASE_REQ include the need for ‘elaborated affirmations’ to independently validate and comment on aspects of the ST rationale. Evaluators may choose to report this separately, for example in an ‘ST analysis’ part within Section 4 of the ETR.
ADV_ARC	Fully reported in the work unit result tables, except as follows: <ul style="list-style-type: none"> • The traceability analysis aspects will be reported in the ‘SFR Tracing and Analysis’ part of Section 4 of the ETR (4.2 in the example ETR structure).
ADV_FSP	
ADV_TDS	
ADV_IMP	Fully reported in the work unit result tables, except as follows: <ul style="list-style-type: none"> • The strategy for sampling (IMP.1-3) will be described and justified in a separate part of Section 4 of the ETR (4.1 in the example structure above). • The traceability analysis aspects (IMP.1-3) will be reported in the ‘SFR Tracing and Analysis’ part of Section 4 of the ETR.

Assurance class/family	[UK007] Reporting Requirements
ADO_PRE	<p>Fully reported in the work unit result tables.</p> <ul style="list-style-type: none"> • PRE.1-2 requires a brief summary of the delivery procedures from the user’s perspective. This could be reported as a note referenced from the table.
AGD_OPE	<p>Fully reported in the work unit result tables, except as follows:</p> <ul style="list-style-type: none"> • The traceability analysis aspects will be reported in the ‘SFR Tracing and Analysis’ part of Section 4 of the ETR.
ALC_CMC ALC_CMS	<p>Fully reported in the work unit result tables, except as follows:</p> <ul style="list-style-type: none"> • ACM_CMC includes ‘Analysis’ type reports that would be best reported in the separate ‘Development Environment Assessment’ report (4.6 in the example ETR structure). This may also be a convenient place to report some other aspects, such as describing how the TOE and CIs are uniquely identified (e.g. CMC.4-1, CMC.4-3, CMC.4-9).
ALC_DEL	<p>Fully reported in the work unit result tables, except as follows:</p> <ul style="list-style-type: none"> • DEL.1-1 requires a brief summary of the delivery procedures, whilst DEL.1-2 is an ‘Analysis’ type report describing validation of the application of the procedures. These could be reported as notes referenced from the table.
ALC_DVS	<p>Fully reported in the work unit result tables, except as follows:</p> <ul style="list-style-type: none"> • DVS.1-4 is an ‘Analysis’ type report that would be best reported in the separate ‘Development Environment Assessment’ report (4.6 in the example ETR structure). This may also be a convenient place to report DVS.1-3 (identifying the types of evidence generated by application of the procedures).
ALC_LCD	<p>Fully reported in the work unit result tables, except as follows:</p> <ul style="list-style-type: none"> • LCD.1-1 requires a brief summary of the life-cycle model. This might be reported in the separate ‘Development Environment Assessment’ report (4.6 in the example ETR structure).
ALC_TAT	<p>Fully reported in the work unit result tables.</p>

Assurance class/family	[UK007] Reporting Requirements
ATE_FUN	Fully reported in the work unit result tables, except as follows:
ATE_COV	<ul style="list-style-type: none"> The tabular/matrix summary called for by COV.2-1, DPT.2-1 and DPT.2-4 would (if compiled by the evaluators from the evidence supplied) probably be best reported in a separate ‘Test Analysis’ part within Section 4 of the ETR (e.g. 4.1 of the example ETR structure).
ATE_DPT	
ATE_IND	<ul style="list-style-type: none"> Any sampling strategy adopted for ATE_FUN might be described in the form of notes referenced from the table, or could be reported in Section 4 of the ETR (e.g. 4.1 in the example ETR structure). The reports of developer test effort (FUN.1-7) and evaluator test effort (IND.2-11) would be best reported in a separate part of Section 4 of the ETR (4.1 in the example ETR structure). Descriptions of the sampling strategy (IND.2-4) and the overall evaluator test strategy (IND.2-9) would probably be best reported in a separate part of Section 4 of the ETR (4.1 in the example ETR structure). The results of testing would be reported in the ‘SFR Tracing and Analysis’ part of Section 4 of the ETR (4.2 in the example ETR structure). A separate part within Section 4 could be used to provide an overall summary of results if needed.
AVA_VAN	<p>Fully reported in the work unit result tables, except as follows:</p> <ul style="list-style-type: none"> All ‘Analysis’ and ‘Report’ type work units (such as VAN.3-3, VAN.3-4 and VAN.3-6) are reported throughout Section 4 of the ETR (4.2 through 4.5 in the example ETR structure). Similarly, any summaries of the results of penetration testing (as may be required for VAN.3-9) can be included in this part of the ETR.

Compliance with UKAS requirements

131. There is nothing inherent in the vulnerability-centric evaluation process that either contravenes UKAS requirements, or that makes compliance with the requirements more difficult. As explained above, CEM work units are still to be performed with, and reported to, an appropriate degree of objectivity and rigour. The work will still be performed as directed by the CLEF’s Quality Manual, with the evaluators keeping appropriate records of their findings as each activity is performed, in accordance with UKAS requirements.