October 2018

THE CPA BUILD STANDARD

Version 1.4

National Cyber Security Centre
a part of GCHQ

1. FOREWORD

This document describes the Build Standard for NCSC's Commercial Product Assurance evaluation scheme. Comments should be sent to NCSC.

Document History

Version	Date	Description
1.0	April 2011	First release describing Foundation Grade requirements
1.1	May 2011	Includes Augmented Grade and CMMI links
1.2	August 2011	Minor changes based on comments received
1.3	September 2014	Removal of Augmented Grade Removal of Appendix A (Build Standard Validation Report Template)
1.4	October 2018	Amended to reflect formation of NCSC

Document ID: NCSC-1844117881-312

This document is authorised by:

Technical Director (Assurance), NCSC

This document is issued by NCSC

For additional copies of this document or for general enquiries please contact:

IES Service Management Team
NCSC
A2i
Hubble Road
Cheltenham
Gloucestershire
GL51 0EX

Email: cpa@ncsc.gov.uk

The CPA Authority may review, amend, update, replace or issue new Scheme Documents as may be required from time to time.

CONTENTS

INTRODUCTION **4**

CPA ROLES **5**

CPA Authority 5

PREREQUISITES **7**

PERFORMING BUILD STANDARD VALIDATIONS **8**

Renewing Build Standard Validation Reports 8

BUILD STANDARD VALIDATION REQUIREMENTS **10**

REFERENCES **25**

INTRODUCTION

1. The CPA Build Standard describes the engineering principles and practices that are expected from a Product Developer creating a good quality, secure product. It covers both development processes and the general security approach taken by the Product Developer. Validating these aspects of the development approach will give confidence of a secure and well-understood product throughout the product's lifecycle.

2. The Security Characteristics describe building the right features into the product. The Build Standard describes building the product right. Adherence with the Build Standard alone will not result in a secure product. However, the absence of key elements of the Build Standard makes assurance impracticable.

3. This document describes the Build Standard requirements for a Foundation Grade evaluation.

CPA ROLES

CPA Authority

4. This is the authority responsible for the CPA Evaluation scheme. At the time of writing, the CPA Authority is NCSC.

Evaluation Team

5. The Evaluation Team should discuss the applicable parts of each requirement in the Build Standard with the Product Developer, and then seek evidence from the Product Developer to justify the assertion that a requirement is successfully met. This evidence may simply be in the form of documentation or notes from conversations with members of the product development team. Evidence may also be in the form of training material or witnessed activities. The Evaluation Team may ask the Developer to acknowledge notes from conversations that are to be used as evidence to confirm that they have been correctly interpreted. Importantly, the Evaluation Team must ensure that in addition to having a well-documented process, that it is followed in practice.

6. It is recommended that members of the Evaluation Team visit the Product Developer's engineering premises to gather evidence that the Build Standard requirements have been successfully met.

Product Developer

7. The Product Developer is primarily responsible for developing the product being evaluated. Some Developers may use multiple development teams. Each development team may have different approaches to development security. This may be for a number of reasons e.g. geographical or organisational. The Evaluation Team must discuss the development approach and the extent to which it is followed across the different development teams. If the Evaluation Team has reason to believe that the evidence they have observed is only applicable to certain, but not all engineering teams, they should clearly document this in the Build Standard report. This may result in the Developer failing the Build Standard validation process. The CPA Authority may also revoke any existing Build standard validation report.

8. The Product Developer and any 3rd party suppliers is expected employ Industry related good working practices, such as ISO/IEC 27000:2005 series, ISO/IEC 18028: 2006 series, and should have local policy frameworks in place to address the following areas:

a. Leadership and Governance

b. Information Risk Management (IRM)

c. Through-Life Information Assurance and IT security measures

d. Assured Information sharing

e. Quality Assurance and Testing

f. Training, Education and Awareness

g. Audit and Compliance

9. Although these areas will not be formally audited by the Evaluation Team, it is unlikely a Product Developer would be able to provide satisfactory evidence for meeting the Build Standard requirements without having such policies and working practices in place.

PREREQUISITES

10. The Product Developer must meet the four key requirements listed below before it can be considered for a CPA evaluation. The Developer must provide evidence for each requirement during the evaluation of their product.

- The Product Developer must employ a configuration management system.
- A software-based issue tracker must be used as part of a defined flaw remediation process.
- The Product Developer must perform extensive testing of their products.
- The Product Developer must employ a flaw-reporting process that enables flaws that are discovered outside the Product Developer‘s organisation to be reported.

11. These requirements exist as a simple metric for the Developer’s approach to secure product development – it is unlikely that a Product Developer routinely creates and maintains a good-quality security-enforcing product if these four key requirements are not met. Therefore, the evaluation must not start unless the Developer can attest that they meet these requirements.

PERFORMING BUILD STANDARD VALIDATIONS

12. During the CPA Foundation Grade evaluation, the Evaluation Team will gather evidence that each of the requirements described below has been met. If the Developer believes that a requirement is not relevant to their product or engineering approach they may submit a reasoned argument describing why it does not apply in their circumstances to the Evaluation Team.

13. The Evaluation Team must capture and record evidence for each requirement listed. Paper claims by a Developer must be verified by witnessed activities / processes The Evaluation Team should look to logs, build script outputs or review session logs, i.e. actual engineering or process artefacts – to get a true picture of whether claimed processes are adhered to in practice. It is also permissible to gather evidence from interviewing members of the development team – again, claims must be verified wherever possible.

14. Where a Developer uses commercial tools to satisfy a requirement, these should be listed in the Build Standard validation report.

15. If the Evaluation Team is satisfied that each of the requirements has been fulfilled, they submit their findings to the CPA Authority, who will then release a Build Standard validation report.

16. A Build Standard validation report will be valid for two years, but may be revoked if evidence of noncompliance is discovered.

Renewing Build Standard Validation Reports

17. A Build Standard validation is an assessment of a Developer's engineering processes and does not necessarily apply only to a single product i.e., a Build standard validation may apply to more than one product. The Evaluation Team should note where there is evidence to suggest that the scope of the Build Standard validation is unlikely to apply to more than one product. A Developer may have multiple product evaluations covered by a single Build Standard validation report. However, a Build Standard validation report is only valid for two years. If more than two years have elapsed, a Build Standard revalidation will be required. It is up to the Developer to maintain the validity of the Build Standard.

18. Where less than two years have elapsed, the Evaluation Team must determine whether the product being assessed is still relevant to the existing Build Standard validation report, and whether the evidence in the report is still valid.

19. The Evaluation Team must consider the scope of the existing Build Standard validation report, and confirm that the new product has been developed using the same process with the Developer. If the original Build Standard is still valid, the Evaluation Team should continue with the evaluation as normal and a new validation of the Build Standard is not required.

20. At the conclusion of the product evaluation, the Evaluation Team should consider whether the Developer's processes are still as described in the existing Build

Standard validation report and whether they still achieve the requirements listed in the Build Standard.

21. If the Evaluation Team believes that the Build Standard is still valid, they should document this opinion in an update to the Build Standard validation report, which will be valid for a subsequent two years.

22. If the Evaluation Team does not believe that the existing Build Standard is still valid, this opinion and the reasons should be documented and the Evaluation Team will need to perform a new validation against the Build Standard as per the normal Build Standard Validation process.

23. Where the Evaluation Team believes that the original Build Standard validation is not valid anymore, they must inform the CPA Authority immediately. It is likely that a reassessment of the Developer's engineering approach will be required. This could cover the existing set of products that are already certified as well as their new product.

BUILD STANDARD VALIDATION REQUIREMENTS

24. The tables below describe the Build Standard requirements. The Evaluation Team should gather evidence as to whether each requirement has been met.

25. A number of these requirements are similar to those found in Common Criteria (CC). Product Developers that have been assessed with CC may be able to re-use some of the evidence used in the CC assessment for some of the requirements in the Build Standard. In these cases, the Developer or the Evaluation Team will need to show that the evidence used in the Common criteria evaluation meets the requirements of the Build Standard. The relevant CC assessment must have been completed within the previous two years. The references given in the CC Relationship sections have been given for guidance only.

26. Product Developers who have undertaken a Capability Maturity Model Integration (CMMI) assessment [3] may also have relevant evidence which can be employed to show adherence to requirements in the Build Standard. As a convenience, references to relevant CMMI requirements have been listed in this document.

27. The assurance activities listed are recommendations, and are provided as guidance to the amount of assessment that might reasonably be performed. The Evaluation Team may modify the assurance activities as they wish, as long as they are still able to make a case for why each requirement has been successfully met by the Developer.

Requirement 1: *Released versions of the Developer's products must be uniquely identified. Released versions must be able to be completely recreated at a later date, with traceability provided by the identifier(s).*	
Description:	Customers and the Product Developer should be able to uniquely identify released versions of products. It must be possible to recreate a released version of a security product (i.e. one that a UK Government customer might use). If a product has changed since release, the Developer should be able to identify all the changes that have been made as well as the Developer that made the change.
Assurance Activity:	The Evaluation Team must ensure that a unique identifier is assigned to released versions of products. The Developer should be able to associate these identifiers to specific builds of their products. The Evaluation Team must be confident that the Developer reliably follows a process that achieves this. The Evaluation Team must seek evidence that the development team can later reproduce new instances of released builds.
CC Relationship:	[1]: ALC_CMC.1.1C, ALC_CMC.3.7C, ALC_CMC.5.11C.
CMMI Relationship:	[CM]: SP 1.1.2, SP 1.2, SP1.3, SP 2.1, SP 2.2.1, SP 3.1.1

Requirement 2: *Updates that fix security flaws must be actively advertised to supported customers and categorised according to the severity of the flaw.*	
Description:	It should be straightforward for customers to establish whether a particular version of a product has known issues, and to establish if they have the most up-to-date version. Remediation procedures should be available to customers where security flaws are discovered.
Assurance Activity:	The Evaluation Team must examine the Product Developer's processes for informing customers of security issues discovered in their product, and ensure that these will ensure an effective and timely distribution of information. (Suggested methods could be using metrics i.e. 75% of customers made aware of issue within 2 working days. 95% within 7 days. Another approach may be to alert customers of a flaw within 1 working day via a website, regularly update customers and provide information on solution).
CC Relationship:	[1]: ALC_FLR.3.1C to ALC_FLR.3.11C.
CMMI Relationship:	[PPQA]: SP 2.1

Requirement 3: *All configuration items used in the Developer's products must be uniquely identified.*	
Description:	All constituent components that are used to create the finished product must be uniquely identified. Any change made to these components must be attributed to the Developer making the change. This ensures that configuration management can properly occur, and that changes are properly managed. Without this, it would be extremely challenging for a Developer to state with certainty precisely which components made up a given build of their product. The development and build environment must also be considered as configuration items.
Assurance Activity:	The Evaluation Team must ensure that all configuration items are uniquely identified. The Evaluation Team must also seek evidence by taking a sample of 10 to 20 configuration items and checking that they can be uniquely identified as defined by their process.
CC Relationship:	[1]: ALC_CMC.2.3C.
CMMI Relationship:	[CM]: SP 1.1.2

Requirement 4: *The Developer must use an audit mechanism that identifies the author of any change to a configuration item.*	
Description:	Any change to a component of the product must be attributed to an individual Developer. This audit mechanism must use time stamping with a reliable time source to enable root-cause analysis to be performed easily should problems arise. Authorised users should be aware of this audit mechanism.
Assurance Activity:	The Evaluation Team must ensure that all changes to configuration items are uniquely identified. The audit mechanism must also identify the Developer making the change. Changes must only be made by authorised Developers. All authors of changes must be uniquely identified within all systems used to hold and process configuration items. The Evaluation Team must seek evidence by selecting a random sample of changes to configuration items and check that they have been uniquely identified as required from a defined process. The Evaluation Team must seek evidence that the Developer's configuration management processes enable them to answer simple questions about their product. The granularity of the audit logs must enable the Evaluation Team to identify who originally authored a function and who made the last change to this file. It must also enable them to identify the purpose of any given change as well as the files to which the change applied. It must also identify the approver of a change.
CC Relationship:	[1]: ALC_CMC.3.4C.
CMMI Relationship:	[CM] SP 1.2, SP 1.2.1

Requirement 5: *The Developer must use defined processes for flaw remediation, and show that Developers are trained in these processes. The Developer must also show that mechanisms are in place to ensure that this process cannot be bypassed.*	
Description:	All but the very simplest products will contain incorrect behaviour or functionality. The Developer is expected to have a well-defined process for triaging flaws. This process should identify a resolution (if any) and make necessary changes to the product where appropriate. It is important that all staff that work in product engineering are trained in the relevant process, and understand how to apply it to the products they work with, to ensure that the quality of the product improves as flaws are discovered and remediated over time. Likewise, it is also important that Developers are not able to "game" any flaw remediation system, and that discovered issues really are resolved, rather than being hidden.
Assurance Activity:	The Evaluation Team must investigate the Developer's flaw remediation processes, and ensure that they are applicable to any realistic problems which the Evaluation Team can hypothesise may occur in the future. The flaw remediation processes must not be limited to security flaws only. The Evaluation Team should perform a paper-based test of the process with the Developer – walking through some hypothetical problems, and understanding how the Developer believes these would be captured and resolved. The Evaluation Team must investigate internal training that is provided to product engineering staff on these processes, and determine whether it appears likely that the process will be understood and applied in practice. The Evaluation Team must then discuss the flaw remediation processes with engineering staff and determine whether the processes appear to be understood and followed by staff.
CC Relationship:	[1]: ALC_CMC.4.7C, ALC_CMC.4.8C, ALC_CMC.4.9C, ALC_CMC.5.3C, ALC_CMC.5.9C.
CMMI Relationship:	[RD]: SP 3.2, [OPP]: SP 1.4

Requirement 6: *The Developer's configuration management system(s) must be protected from malfeasance from both internal and external sources. All configuration items must be protected at all times.*	
Description:	The configuration management system by necessity contains the sensitive information comprising of the Developer's products. It is therefore vital that the confidentiality and integrity of this system is robust against malfeasance and improper use at all times. The protection of the configuration management system is not limited to the security of the system itself. This protection must also extend to the systems where development effort is regularly performed, and should include physical, personnel and procedural security measures enacted around the development and configuration management environments. Developers are expected to be able to demonstrate that their site, network, and full development environment is given the protections deserved by the product. *At Foundation Grade* these protections must follow commercial good practice for network security.
Assurance Activity:	The Evaluation Team must investigate the Developer's physical and logical protection of their configuration management system. The Evaluation Team must verify that the physical protection prevents unauthorised access to development systems, both during working hours and out-of-hours. The Evaluation Team should investigate the logical protection of the development environment and configuration management system, to ensure that the integrity of the product is maintained. They should consider the architecture of the systems and ensure they follow commercial good practice and relevant HMG guidance. The Evaluators must review the incident reporting and disaster recovery procedures and interview developers to gain assurance that the procedures are implemented in practice.
CC Relationship:	[1]: ALC_DVS.1.1C, ALC_DVS.2.2C.
CMMI Relationship:	N/A

Requirement 7: *The way that products are assembled must be consistent and repeatable i.e., the build process should be automated.*	
Description:	Automation of regularly repeated activities helps to ensure that mistakes are limited. Assembly of the product from its constituent components as described in the Developer's configuration management system is one such automatable process. Automation in this context can help ensure that new instances of the same build can easily be recreated, and that changes to the build environment are controlled and understood. This helps to ensure that changes that may have a poorly understood security impact are minimised, and that the Developer has control over the product creation process.
Assurance Activity:	The Evaluation Team must ensure that there is an automated process for assembling the product from items held under configuration control. The Evaluation Team must ensure that this process is routinely followed in practice. The Evaluation Team must ensure that changes to the assembly process are controlled and audited by the Developer.
CC Relationship:	[1]: ALC_CMC.4.5C.
CMMI Relationship:	[VER]: SP 1.3

Requirement 8: All *Changes to the Developer's products must be purposeful, necessary and have very little impact on the overall quality of the product.*	
Description:	Developers should be able to demonstrate testing and acceptance procedures for all new versions of the product. This applies particularly to the maintenance of security requirements. The Developer should also be able to demonstrate how security flaws are resolved and prevented from being reintroduced. The Developer should also be able to demonstrate how regressions in security components are detected. This requirement is necessary to ensure that the quality of the product does not diminish over time.
Assurance Activity:	The Evaluation Team must ensure that the Developer has a process for managing changes to the product. The Evaluation Team must ensure that an acceptance procedure exists for new versions of the product. The acceptance criteria should be determined using a documented quality standard. The Evaluation Team must ensure that this procedure is routinely followed in practice. The Evaluation Team must select a security flaw that existed in the product[1] and determine how the Developer identified the cause of the problem and the measures that have been put in place that prevent it from being reintroduced. The Evaluation Team should investigate whether the measures are successful in practice.
CC Relationship:	[1]: ALC_FLR.1.1C, ALC_FLR.1.2C, ALC_FLR.1.3C, ALC_FLR.3.8C.
CMMI Relationship:	[CM]: SP 2.2.4

[1] If this is a genuinely new product from a Developer who has never previously produced a product, then it is acceptable to review of their intended processes and procedures. The Evaluation Team must ensure there is reason to be confident that the procedures will be followed in practice.

Requirement 9: *Flaw remediation is performed in practice.*	
Description:	The existence of plans for flaw remediation alone is insufficient to meet this requirement. The Developer must be able to show how problems in the product are discovered, analysed, corrected, tested, regression tested and managed in the configuration system in practise. This is to ensure that the product's overall quality improves over time as problems are discovered and rectified. The Developer should also be able to show how their development teams have learnt from problems discovered in the product. The Developer should have performed root cause analysis, and learnt from each issue. Flaws are not limited to coding errors and implementation mistakes. They also include problems in the specification and design, documentation.
Assurance Activity:	The Evaluation Team must seek evidence that the Developer's flaw remediation procedures are routinely followed in practice. They should pay particular attention to points in the development cycle where engineers are likely to be under significant pressures (e.g. in the run-up to the release of a new version of a product), to ensure that processes are not discarded at such times. The Evaluation Team should seek evidence from the Developer as to how they actively seek out problems in products throughout the product's life cycle. The Developer must also be able show to the Evaluation Team how a random sample of issues discovered (from minor to major) at all points in the product's lifecycle have been managed from discovery, through analysis, correction, testing and ultimate resolution.
CC Relationship:	[1]: ALC_FLR.3.6C.
CMMI Relationship:	N/A

Requirement 10: *The Developer's products must behave as designed. i.e. they should have reason to believe that they are correct. Using non-trivial test cases, Developers should be able to demonstrate evidence of security and functional testing. This testing should cover the whole scope of the product.*	
Description:	The Developer's engineering processes must ensure that the products they supply do what they intended them to do – from the initial concept, all the way through to the implementation, it is vital that whichever engineering approach is taken, it ensures a quality product. Products with good overall quality do not directly imply high security, but low overall product quality is a certain indicator of potential problems. Regardless of the development style that the development team employs, it is important that in practice the Developer has a good understanding of what it is their product actually does. Evaluated CPA products will necessarily be security enforcing, and thus the Developer must have reason to believe that they are secure; this confidence must be founded on the activities they have undertaken to develop it.
Assurance Activity:	The Evaluation Team must ensure that there is a good testing regime for products, and that tests are non-trivial, and are likely to exercise the product in practice. The Developer must be able to explain to the Evaluation Team how they ensure good test coverage (i.e. that a sizeable percentage of each product's functionality and implementation is routinely tested). The Evaluation Team must seek evidence that the Developer engages in security testing of their products. This must include positive (the product does what it is designed to), negative (the product doesn't do certain things it ought not to) and penetration (edge cases in the product's design or implementation do not cause unexpected behaviour, and perturbation of the product's environment does not interfere with its secure functioning). The Evaluation Team must also ensure that issues, which are discovered through such testing, are treated via the Developer's flaw remediation processes.
CC Relationship:	[1]: ATE_COV.1.1D, ATE _COV.2.2C, ATE _DPT.1.1D, ATE_FUN.1.3C, ATE_FUN.1.4C.
CMMI Relationship:	N/A

Requirement 11: *Security enhancing features of the underlying platform, implementation language and tool chain should have been considered and used unless evidence is presented as to why this is not necessary or possible.*	
Description:	Modern platforms, development languages, libraries and development tools regularly offer security enhancing

	technologies to both minimise the occurrence of security defects, and to minimise their impact should they occur by mistake. The Developer is expected to understand the security enhancing technologies offered by their chosen platform, language, libraries and tools and to have made an informed decision about whether to employ them. Security enhancing features are expected to be used unless there is a strong rationale not to[2].
Assurance Activity:	The Evaluation Team should discuss how the Product Developer uses security-enhancing technologies, and determine their approach to these. The Evaluation Team should also assess whether the Developer understands the security enhancing technologies available to them. The Evaluation Team should assess the development team's selected tool chain and development environment to determine whether the whole set of available security enhancing technologies have been used. Where this is not the case, the Evaluation Team must discuss this with the Developer and understand the rationale for why specific technologies have not been selected. The Evaluation Team must then determine whether these represent consistent and well-reasoned arguments for the omission of such technologies. The Evaluation Team must clearly record these reasons in the report. The Evaluation Team must ensure that this requirement is not solely applied to the Developer's own software, but also to third-party components they may employ.
CC Relationship:	[1]: ATE_FUN.1.4C.
CMMI Relationship:	[TS]: SP 1.1

[2] An example of this *might* be a new feature on the platform that has been widely proven as unreliable in practice.

Requirement 12: *Externally reported flaws in the developer's products must be handed appropriately.*	
Description:	Flaws in security products that are found externally must be treated with appropriate severity. These flaws must be investigated and resolved as appropriate. This is particularly important in a UK Government context where numerous penetration tests and vulnerability assessments may be conducted on networks where these products may feature. The Product Developer must be proactive in responding to externally reported issues in their products.
Assurance Activity:	The Evaluation Team must seek evidence that the Developer has a process for receiving externally discovered flaws that are reported. This process must be routinely followed in practice. The Evaluation Team should investigate whether externally reported flaws are passed into the Developer's standard flaw remediation processes. Vendors should demonstrate that they monitor publicly announced security flaws for their products and that they respond to issues identified in their products as appropriate.
CC Relationship:	[1]: ALC_FLR.3.11C.
CMMI Relationship:	N/A

Requirement 13: *The Developer should use a coding standard that is applied to all code in their finished products (including third-party components). The coding standard and associated processes must ensure that common software defects and easily avoided security weaknesses are not introduced into the product's code.*	
Description:	Regardless of the development language selected, there are both good and bad practices for secure coding. Coding standards can cover many aspects of development – from "house styles" through to lists of prohibited library APIs. This requirement is intended to help ensure that all code contributing to the product is developed with security in mind, and that development processes (such as code review and check-in gates) help to keep the software quality as high as possible from a security perspective. This requirement intends to ensure that simple mistakes are not accepted into code that is used in the final product, and that development staff are made aware of secure development practices. It is also vital that the Developer's use of third-party components does not introduce weaknesses in their products. The Developer should have a process to determine code quality before a third-party component is used. The third-party component must be kept up-to-date as security issues are discovered and resolved.
Assurance Activity:	The Evaluation Team must ensure the Developer has a coding standard[3], and that this coding standard addresses both software quality and security issues. The Evaluation Team should ensure that the coding standard covers all languages that the Developer uses, and that it covers good practice for common security errors made in these languages. These should also be up-to-date and based on industry-standard good practice – as an example, the SANS Institute and the Mitre Corporation publish a list [2] of what they consider the "most dangerous" coding errors, so a good coding standard should seek to minimise the occurrence of these errors. The Evaluation Team must ensure that the Build standard is maintained and kept up-to-date. The Evaluation Team should determine what secure development training is provided to engineers to ensure that Developers are informed about the importance of adhering to the coding standard. Engineers should be provided with sufficient information about the contents of the coding standard. The Evaluation Team may test the Developer's understanding of the coding standards at an interview. The Evaluation Team must also ensure that there is a

[3] It is acceptable for the Developer to employ a third-party coding standard, as long as it applies to their environment and their engineers make use of it in practice.

	process in place for individual developers to be notified of changes to the Developer's Coding Standard. The Evaluation Team must seek evidence that the Developer's engineering process uses the coding standard, and ensure that code that does not meet the coding standard cannot reach the product. They should seek evidence that the process is routinely followed. The Evaluation Team must ensure that the Developer has an on-boarding process for third party components (including open-source or the public-domain components). This process should ensure that these components follow the in-house coding standard, or an equivalent level of quality. The Evaluation Team must ensure that there is a maintenance process for all third party code and components, ensuring that they remain current and secure, and that this process is routinely followed.
CC Relationship:	[1]: ALC_FLR.3.11C.
CMMI Relationship:	N/A

REFERENCES

[1] Common Criteria for Information Technology Security Evaluation – Part 3: Security Assurance Components, Version 3.1 revision 4. CCMB-2012-09-003, September 2012.

[2] M. Howard Improving Software Security by Eliminating the CWE Top 25 Vulnerabilities. IEEE Security and Privacy, vol. 7, no.3, pp. 68-71, May/June 2009.

[3] Capability Maturity Model Integration, Version 1.3 Nov 2010, Carnegie Mellon University Software Engineering Institute.