5-2012

Security Verification and Validation by Software SMEs: Theory versus Practice

Matthew Nicolas Kreeger
Royal Holloway, University of London, matthew.kreeger@thales-esecurity.com

G. Harindranath
Royal Holloway, University of London, g.harindranath@rhul.ac.uk

Security Verification and Validation by Software SMEs: Theory versus Practice

Matthew Nicolas Kreeger
Royal Holloway, University of London, U.K.
and
Thales Information Technology Security, U.K.
matthew.kreeger@thales-esecurity.com

G. Harindranath
Royal Holloway, University of London, U.K.
g.harindranath@rhul.ac.uk

Abstract
To improve software engineering practice it is essential to observe the socio-technical realities that surround software development within an industrial context. There is a lack of empirical knowledge of security verification and validation practice within an SME context. When coupled with the recognised importance, and inherent complexities, of such practice, it appears fundamentally sound to understand the faced socio-technical realities to ensure continued process improvement and improved technology adoption and research guidance. Within this research-in-progress paper we highlight the importance of obtaining such an understanding.

Keywords
Software, Security, Verification, Validation, SMEs, Theory, Practice

1. Introduction
The software industry spends more money on locating and addressing defects than any other activity [Jones, 2009]. Further, it is reported that greater than 50% of all released software contains defects that affect its execution in some form [Shull et al., 2002]. NIST has estimated that the US economy suffers some $60 billion in costs per year due to software defects, but point out that this could be reduced to around $20 billion with improvements in software testing infrastructure [NIST, 2002]. Nevertheless, the annual costs due to software defects are reported to be in the billions of dollars [Vieira et al., 2006].

However, we are interested in security vulnerabilities - a specific type of defect with security related implications. Specifically, we are keen to understand the security verification and validation (V&V) practices (i.e. those practices aimed at locating security vulnerabilities) as adopted by Small to Medium Sized Enterprises (SMEs), as well as the surrounding socio-technical realities of such practices. Although an increasing amount of time is being spent on software testing [Sung and Paynter, 2006], and software V&V is known to comprise a substantial share of a project's budget [Kasurinen et al., 2010]), we observe that security vulnerabilities continue to be reported [Eschelbeck, 2005]. But we also note that there is a significant amount of academic research (predominately technical

in nature) on the topic [e.g., Parnas and Lawford, 2003, Lethbridge et al., 2007]. Therefore, that there is a distinction between theory and practice appears evident, however, the cause must be more than just technical in nature (since there exists many documented techniques for the practitioner to adopt). What then are the socio-technical barriers and influences surrounding security V&V practice? It is the addressing of this unknown, within an SME context, that forms the focus of our research.

2. Research Focus and Justification

SMEs are of fundamental economic importance to the majority of the world's economies [Mac an Bhaird, 2010] and the majority of software producing organisations are small in size [Fayad et al., 2000]. This study, therefore, focuses on software producing SMEs. However, despite the growth and contribution of the software industry to national economies, there are many examples of "software quality lapses that are shaking the public's confidence that software can be used to build safe, secure systems" [Parnas and Lawford, 2003, p. 20]. It is essential that software performs as expected, even when subjected to malicious treatment, as the consequences are known to be severe [Kreeger, 2009].

Software V&V has been a topic of focus since the early days of software product development - with the process of software testing being considered the de facto industry standard [Vieira et al., 2006]. There are a number of general objectives of software V&V: to validate that the system satisfies the requirements; to enable earlier defect detection and resolution; to gauge quality attributes etc. From a security perspective, software V&V can be seen as encompassing two objectives: demonstrating that the security properties and the behaviour of the software remain satisfied, predictable and secure, even when in the presence of a malicious party and the uncovering of security vulnerabilities.

The observation, that the costs incurred to address defects increase as a project nears completion, have been extensively published. However, it can be difficult to quantify some of the costs of security vulnerabilities, which go beyond the immediate cost of defect remediation - for example: damage to corporate reputation (and the cost to rebuild the reputation), loss of consumer trust, loss in market value etc.

3. Distinction Between Theory and Practice

In 1988 (formal test research is then approximately 15 years old and with modest results), Hamlet observed that "what is known is not finding its way into practice very well" [Hamlet, 1988, p. 663]. Unfortunately, the gap between industry practice and academic research is still reportedly in existence [Bertolino, 2003], with the transfer of research results to industry being viewed at the heart of the gap [Ivarsson and Gorschek, 2011]. Hartman indicates that industry needs help in adopting and implementing this work and that it is insufficient for academics to write papers on software testing [Hartman, 2002]. Certainly, it has been stated that very little of such research has had an impact on practitioners, and that the number of industrialists attending related conferences has reduced - thus "reflecting [the] different interests in the two communities" [Lethbridge et al., 2007, p. 22]. Practitioners "habitually neglect" research on the sometimes correct assumption that it is "irrelevant to them" - however, researchers "tend to write for each other" and "lose contact with the realities that practitioners must face" [Parnas and Lawford, 2003, p. 17].

Osterweil (1996) argues that technology transfer is particularly problematical in software quality - with what he terms the 'yawning chasm' between research and practice being both "wide and increasingly inculturated, to the detriment of both communities" (p. 746).

According to [Briand, 2010], a number of areas of software V&V research have failed to transfer to practice e.g. fault-based testing, which although an area that has undergone substantial research over the last two decades, there has been limited industrial application reported. In addition, regardless of the acknowledged theoretical importance of software inspections, they are noted as not being widely applied in practice [Kollanus and Koskinen, 2006]. Another example concerns test case selection, which is considered a dominating topic in software testing research, but "[p]aradoxically, … the least interesting problem for test practitioners" [Bertolino, 2003, p. 6]. Bertolino also details other problems evident to practitioners but which are underrepresented in research and, rather worryingly, when "practitioners lack knowledge of what researchers have already discovered, they are in no position to absorb new findings" [Lethbridge et al., 2007, p. 12].

To further compound this, there is evidence to suggest that the university education system does not provide significant focus and instruction in V&V [e.g., Kreeger, 2009]. There exists a clear relationship between the success, in terms of adoption and application of specific V&V techniques, and an employee's knowledge, competency and experience [e.g., Sung and Paynter, 2006]. This relationship, between an employee's education and their ability to perform specific V&V tasks, has an impact on the quality of a product being developed by an organisation (as an example, we note that meeting the challenges highlighted by Lethbridge et al will: "improve not only the quality of software engineering education, but also the quality of the workforce, and, consequently, the quality of software developed" [Lethbridge et al., 2007, p. 19]).

This distinction, between theory and practice, is somewhat surprising, since we noted earlier the costs of the V&V process itself, as well as the costs incurred when failing to perform the activity adequately. However, organisations are known to target V&V activities when needing to reduce, or remove, scheduled work to cope with project time constraints [Torkar and Mankefors, 2003]. This attitude is also reflected within academia [Marrero and Settle, 2005].

4. An Empirical Unknown

We observe, in the context of the research proposed, that existing studies typically fall into one of the following categories:

- Generic software engineering studies with somewhat limited treatment of software V&V.
- Software V&V studies which provide illumination on industry practice, however, predominately avoid mention of any security focused V&V (with it not being clear as to why) [e.g., Itkonen et al., 2009, Engström and Runeson, 2010].
- Studies which focus on larger organisations and / or are predominately technical in nature (i.e. focus on specific techniques). Fayad et al indicate that software engineering, within small companies, is not only overlooked by the literature, but also by software engineering societies and computing institutes [Fayad et al., 2000].

Therefore, to our knowledge, there are no dedicated studies examining security V&V practice, within an SME organisational context, from a socio-technical perspective. It is our intention to address this empirical unknown - since it is through understanding current practice that we can understand how to improve upon such practice for the benefit of both industry and the research community. Specifically, we aim to understand, within an SME organisational context, how security verification and validation is performed, and how, as a process, it is influenced by the socio-technical characteristics of such organisations.

5. Conclusions

We have identified a lack of knowledge concerning software security V&V practices within an SME organisational context. Existing studies have typically overlooked this as an area of focus, or just adopt a technical perspective; however, we state that it appears fundamentally sound to understand the faced socio-technical realities to ensure continued process improvement and improved technology adoption and research guidance.

References

[Bertolino, 2003] Bertolino, A. (2003). Software Testing Research and Practice. In Proceedings of the Abstract State Machines 10th International Conference on Advances in Theory and Practice, pages 1–21. Springer.

[Briand, 2010] Briand, L. C. (2010). Software Verification - A Scalable, Model-Driven, Empirically Grounded Approach. In Simula Research Laboratory, pages 415–442. Springer.

[Engström and Runeson, 2010] Engström, E. and Runeson, P. (2010). A Qualitative Survey of Regression Testing Practices. In Product-Focused Software Process Improvement, volume 6156, pages 3–16. Springer.

[Eschelbeck, 2005] Eschelbeck, G. (2005). The Laws of Vulnerabilities: Which security vulnerabilities really matter? Information Security Technical Report, 10(4):213–219.

[Fayad et al., 2000] Fayad, M. E., Laitinen, M., and Ward, R. P. (2000). Software Engineering in the Small. Communications of the ACM, 43(3):115–118.

[Hamlet, 1988] Hamlet, R. (1988). Special Section on Software Testing. Communications of the ACM, 31(6):662–667.

[Hartman, 2002] Hartman, A. (2002). Is ISSTA Research Relevant to Industry? ACM SIGSOFT Software Engineering Notes, 27:205–206.

[Itkonen et al., 2009] Itkonen, J., Mäntylä, M. V., and Lassenius, C. (2009). How Do Testers Do It? An Exploratory Study on Manual Testing Practices. In Proceedings of the 3rd International Symposium on Empirical Software Engineering and Measurement, pages 494–497. IEEE.

[Ivarsson and Gorschek, 2011] Ivarsson, M. and Gorschek, T. (2011). A Method for Evaluating Rigor and Industrial Relevance of Technology Evaluations. Empirical Software Engineering, 16(3):365–395.

[Jones, 2009] Jones, C. (2009). A Short History of the Cost Per Defect Metric. http://www.semat.org/pub/Main/PubsandRefs/a_short_history_of_the_cost_per_defect_metric.doc.

[Kasurinen et al., 2010] Kasurinen, J., Taipale, O., and Smolander, K. (2010). Software Test Automation in Practice: Empirical Observations. Advances in Software Engineering, volume 2010. Hindawi.

[Kollanus and Koskinen, 2006] Kollanus, S. and Koskinen, J. (2006). Software Inspections in Practice: Six Case Studies. In Product-Focused Software Process Improvement, volume 4034, pages 377–382. Springer.

[Kreeger, 2009] Kreeger, M. N. (2009). Security Testing: Mind the Knowledge Gap. ACM SIGCSE Bulletin, 41(2):99–102.

[Lethbridge et al., 2007] Lethbridge, T. C., Díaz-Herrera, J., LeBlanc, Jr., R. J., and Thompson, J. B. (2007). Improving Software Practice Through Education: Challenges and Future Trends. In 2007 Future of Software Engineering, pages 12–28. IEEE.

[Mac an Bhaird, 2010] Mac an Bhaird, C. (2010). Resourcing Small and Medium Sized Enterprises: A Financial Growth Life Cycle Approach. Springer.

[Marrero and Settle, 2005] Marrero, W. and Settle, A. (2005). Testing First: Emphasizing Testing in Early Programming Courses. ACM SIGCSE Bulletin, 37(3):4–8.

[NIST, 2002] NIST (2002). The Economic Impacts of Inadequate Infrastructure for Software Testing. http://www.nist.gov/director/planning/upload/report02-3.pdf.

[Osterweil, 1996] Osterweil, L. (1996). Strategic Directions in Software Quality. ACM Computing Surveys, 28(4):738–750.

[Parnas and Lawford, 2003] Parnas, D. L. and Lawford, M. (2003). Inspection's Role in Software Quality Assurance. IEEE Software, 20(4):16–20.

[Shull et al., 2002] Shull, F., Basili, V., Boehm, B., Brown, A. W., Costa, P., Lindvall, M., Port, D., Rus, I., Tesoriero, R., and Zelkowitz, M. (2002). What We Have Learned About Fighting Defects. In Proceedings of the 8th International Symposium on Software Metrics, pages 249–258. IEEE.

[Sung and Paynter, 2006] Sung, P. W.-B. and Paynter, J. (2006). Software Testing Practices in New Zealand. In Proceedings of the 19th Annual Conference of the National Advisory Committee on Computing Qualifications, pages 273–282.

[Torkar and Mankefors, 2003] Torkar, R. and Mankefors, S. (2003). A Survey on Testing and Reuse. In Proceedings of the IEEE International Conference on Software-Science, Technology and Engineering, pages 164–173. IEEE.

[Vieira et al., 2006] Vieira, F. E., Martins, F., Silva, R., Menezes, R., and Braga, M. (2006). On the Idea of Using Nature-Inspired Metaphors to Improve Software Testing. In Artificial Intelligence Applications and Innovations, volume 204, pages 541–548. Springer.