

CROSSTALK

March / April 2014

The Journal of Defense Software Engineering

Vol. 27 No. 2



MITIGATING RISKS OF
COUNTERFEIT AND
TAINTED COMPONENTS

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE APR 2014		2. REPORT TYPE		3. DATES COVERED 00-03-2014 to 00-04-2014	
4. TITLE AND SUBTITLE CrossTalk. The Journal of Defense Software Engineering. Volume 27, Number 2. March/April 2014				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) 517 SMXS/MXDED,6022 Fir Ave,Hill AFB,UT,84056-5820				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 36	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

Cover Design by
Kent Bingham

Departments

3 From the Sponsor

34 Upcoming Events

35 BackTalk

Mitigating Risks of Counterfeit and Tainted Components

4 **Non-Malicious Taint: Bad Hygiene is as Dangerous to the Mission as Malicious Intent**

Until both malicious and non-malicious aspects of taint can be dealt with in ways that are visible and verifiable, there will be a continued lack of confidence and assurance in delivered capabilities throughout their lifecycle.

by **Robert A. Martin**

10 **Collaborating across the Supply Chain to Address Taint and Counterfeit**

The community of acquirers and providers of technology must reach a consensus on two basics questions: 1) Where is the mitigation focus?, and 2) Are we discussing issues that occur in technology development or just products that have been tampered with?

by **Dan Reddy**

15 **Software and Supply Chain Risk Management Assurance Framework**

The DoD, the defense industrial base, and the nation's critical infrastructure all face challenges in Supply Chain Risk Management Assurance. These diverse challenges span infrastructure, trust, competitiveness, and austerity.

by **Don O'Neill**

20 **Malware, "Weakware," and the Security of Software Supply Chains**

The need for security often exceeds the ability and will of software engineers to design secure software architectures, implement secure coding methods, perform functional security testing, and carefully manage the installation of software products on various platforms and in different environments.

by **C. Warren Axelrod, Ph.D.**

25 **Problems and Mitigation Strategies for Developing and Validating Statistical Cyber Defenses**

The development and validation of advanced cyber security technology frequently relies on data capturing normal and suspicious activities at various system layers. However, getting access to meaningful data continues to be a major hurdle for innovation in statistical cyber defense research.

by **Michael Atighetchi, Michael Jay Mayhew, Rachel Greenstadt, and Aaron Adler**

30 **Earned Schedule 10 Years Later: Analyzing Military Programs**

While progress has been made in understanding the utility of Earned Schedule (ES) in some small scale and limited studies, a significant analysis of ES in DoD acquisition programs is missing.

by **Kevin T. Crumrine, Jonathan D. Ritschel, Ph.D., and Edward White, Ph.D.**

CROSSTALK

NAVAIR Jeff Schwalb

DHS Joe Jarzombek

309 SMXG Karl Rogers

Publisher Justin T. Hill

Article Coordinator Heather Giacalone

Managing Director David Erickson

Technical Program Lead Thayne M. Hill

Managing Editor Brandon Ellis

Associate Editor Colin Kelly

Art Director Kevin Kiernan

Phone 801-777-9828

E-mail Crosstalk.Articles@hill.af.mil

Crosstalk Online www.crosstalkonline.org

CROSSTALK, The Journal of Defense Software Engineering is co-sponsored by the U.S. Navy (USN); U.S. Air Force (USAF); and the U.S. Department of Homeland Security (DHS). USN co-sponsor: Naval Air Systems Command. USAF co-sponsor: Ogden-ALC 309 SMXG. DHS co-sponsor: Office of Cybersecurity and Communications in the National Protection and Programs Directorate.

The USAF Software Technology Support Center (STSC) is the publisher of **CROSSTALK** providing both editorial oversight and technical review of the journal. **CROSSTALK's** mission is to encourage the engineering development of software to improve the reliability, sustainability, and responsiveness of our warfighting capability.

Subscriptions: Visit www.crosstalkonline.org/subscribe to receive an e-mail notification when each new issue is published online or to subscribe to an RSS notification feed.

Article Submissions: We welcome articles of interest to the defense software community. Articles must be approved by the **CROSSTALK** editorial board prior to publication. Please follow the Author Guidelines, available at www.crosstalkonline.org/submission-guidelines. **CROSSTALK** does not pay for submissions. Published articles remain the property of the authors and may be submitted to other publications. Security agency releases, clearances, and public affairs office approvals are the sole responsibility of the authors and their organizations.

Reprints: Permission to reprint or post articles must be requested from the author or the copyright holder and coordinated with **CROSSTALK**.

Trademarks and Endorsements: **CROSSTALK** is an authorized publication for members of the DoD. Contents of **CROSSTALK** are not necessarily the official views of, or endorsed by, the U.S. government, the DoD, the co-sponsors, or the STSC. All product names referenced in this issue are trademarks of their companies.

CROSSTALK Online Services:

For questions or concerns about crosstalkonline.org web content or functionality contact the **CROSSTALK** webmaster at 801-417-3000 or webmaster@luminpublishing.com.

Back Issues Available: Please phone or e-mail us to see if back issues are available free of charge.

CROSSTALK is published six times a year by the U.S. Air Force STSC in concert with Lumin Publishing luminpublishing.com. ISSN 2160-1577 (print); ISSN 2160-1593 (online)

CROSSTALK would like to thank DHS for sponsoring this issue.



Our information and communications technology (ICT) assets are under constant attack. Thwarting the active attacker is not something most designers, engineers, developers, or project managers normally consider or have been trained to address. Yet encouraging resilience as a criteria in every stage of development and supply of ICT must continue to be the forward-leaning focus of the Software and Supply Chain Assurance efforts within government and industry. Attacks against our supply chains unite acquirers and suppliers in the search of scalable means for sharing information about ICT risks that arise through malice or negligence. Suppliers and acquirers need standardized means for conveying information about common issues related to both the hardware and software aspects of ICT, especially regarding non-conforming products that contain counterfeit, tainted, or defective components that can cause subsequent harm.

How can we collaboratively orchestrate industry and government response to these attacks? One way is through the Common Vulnerabilities and Exposures (CVE) List, which is an extensive listing of publicly known vulnerabilities found after ICT components have been deployed. Sponsored by the Department of Homeland Security (DHS), the ubiquitous adoption of CVE has enabled the public and private sectors to communicate domestically and internationally in a consistent manner the vulnerabilities in commercial and open source software. CVE has enabled our operations groups to prioritize, patch, and remediate nearly 60,000 openly reported vulnerabilities.

Unfortunately, vulnerabilities are proliferating rapidly thus stretching our capabilities and resources. As we seek to discover and mitigate the root causes of these vulnerabilities, sharing the knowledge we have of them helps to mitigate their impact. In order to keep pace with the threat, we must facilitate the automated exchange of information. To achieve that, DHS sponsors “free for use” standards, such as:

- **Common Weakness Enumeration (CWE)**, which provides for the discussion and mitigation of architectural, design, and coding flaws introduced during development and prior to use;
- **Common Attack Pattern Enumeration and Classification (CAPEC)**, which enables developers and defenders to discern the attacks and build software resistant to them;
- **Malware Attribute Enumeration and Characterization (MAEC)**, which encodes and communicates high-fidelity information about malware based upon behaviors, artifacts, and attack patterns;
- **Structured Threat Information eXpression (STIX)**, which conveys the full range of potential cyber threat information using the Trusted Automated eXchange of Indicator Information (TAXII) to define the technical mechanisms for exchanging actionable cyber threat indicators.

These open specifications for interoperable security automation enable secure, machine-to-machine communication of actionable indicators between organizations that want to share this information. The components have been developed collaboratively between Federal Government and industry partners working toward information sharing mechanisms and solutions to reduce the risk of counterfeit and tainted ICT components. These standardized means for sharing information are already being used, and they contribute to our efforts to enable all stakeholders to secure their part of cyberspace.

Though not an exhaustive list, the articles in this issue of **CROSSTALK** demonstrate the breadth of anti-counterfeiting and supply chain risk management efforts taking place as well as the depth of the need to share data and lessons learned. We hope you find this issue to be a useful resource to address the very real challenges we face in software assurance, supply chain risk management, and operations.

Roberta “Bobbie” Stempfley
Acting Assistant Secretary
Office of Cybersecurity and Communications
Department of Homeland Security

Non-Malicious Taint

Bad Hygiene is as Dangerous to the Mission as Malicious Intent

Robert A. Martin, MITRE Corporation

Abstract. Success of the mission should be the focus of software and supply chain assurance activities regardless of what activity produces the risk. It does not matter if a malicious saboteur is the cause. It does not matter if it is malicious logic inserted at the factory or inserted through an update after fielding. It does not matter if it comes from an error in judgment or from a failure to understand how an attacker could exploit a software feature. Issues from bad software hygiene, like inadvertent coding flaws or weak architectural constructs are as dangerous to the mission as malicious acts. Enormous energies are put into hygiene and quality in the medical and food industries to address any source of taint. Similar energies need to be applied to software and hardware. Until both malicious and non-malicious aspects of taint can be dealt with in ways that are visible and verifiable, there will be a continued lack of confidence and assurance in delivered capabilities throughout their lifecycle.

Background

Every piece of information and communications technology (ICT) hardware—this includes computers as well as any device that stores, processes, or transmits data—has an initially embedded software component that requires follow-on support and sustainment throughout the equipment's lifecycle.

The concept of supply chain risk management (SCRM) must be applied to both the software and hardware components within the ICT. Because of the way ICT hardware items are maintained, the supply chain for ongoing sustainment support of the software is often disconnected from the support for the hardware (e.g., continued software maintenance contracts with third parties other than the original manufacturer). As a result, supply chain assurance regarding software requires a slightly unique approach within the larger world of SCRM.

Some may want to focus on just “low hanging fruit” like banning suspect products by the country they come from or the ownership of the producer due to their focused nature and ignore more critical issues surrounding the software aspect of ICT like the exploitable vulnerabilities outlined in this article. It is a misconception that “adding” software assurance to the mix of supply chain concerns and activities will add too much complexity, thereby making SCRM even harder to perform. Some organizations and sectors are already developing standards of care and due-diligence that directly address these unintended and bad hygiene types of issues. That said, such practices for avoiding the bad hygiene issues that make software unfit for its intended purpose are not the norm across most of the industries involved in creating and supporting software-based products. Mitigating risk to the mission is a critical objective and including software assurance as a fundamental aspect of SCRM for ICT equipment is a critical component of delivering mission assurance.

During the past several decades, software-based ICT capabilities have become the basis of almost every aspect of today's cyber commerce, governance, national security, and recreation. Software-based devices are in our homes, vehicles, communications, and toys. Unfortunately software, the basis of these cyber capabilities, can be unpredictable since there are now underlying rules software has to follow as opposed to the rest of our material world which is constrained by the laws of gravity, chemistry, and physics with core factors like Plank's Constant. This is even more true given the variety and level of skills and training of those who create and evolve cyber capabilities. The result is that for the foreseeable future there will remain a need to address the types of quality and integrity problems that leave software unreliable, attackable, and brittle directly. This includes addressing the problems that allow malware and exploitable vulnerabilities to be accidentally inserted into products during development, packaging, or updates due to poor software hygiene practices.

Computer language specifications are historically vague and loosely written. (Note: ISO/IEC JTC1 SC22 issued a Technical Report [1] with guidance for selecting languages and using languages more secure and reliably.) There is often a lack of concern for resilience, robustness, and security in the variety of development tools used to build and deploy software. And there are gaps in the skills and education of those that manage, specify, create, test, and field these software-based products.

Additionally, software-based products are available to attackers who study them and then make these products do things their creators never intended. Traditionally this has led to calls for improved security functionality and more rigorous review, testing, and management. However, that approach fails to account for the core differences between the engineering of software-based products and other engineering disciplines. Those differences are detailed later in this article.

The need to address these differences has accelerated as more of the nation's critical industrial, financial, and military capabilities rely on cyber-space and the software-based products that comprise this expanding cyber world. ICT systems must be designed to withstand attacks and offer resilience through better integrity, avoidance of known weaknesses in code, architecture, and design. Additionally, ICT systems should be created with designed-in protection capabilities to address unforeseen attacks by making them intrinsically more rugged and resilient so that there are fewer ways to impact the system. This same concern has been expressed by Congress with the inclusion of a definition of “Software Assurance” in Public Law 112-239 Section 933 [2] where they directed DoD to specifically address software assurance of its systems.

Defining “Taint” and Software Assurance

While there is no concrete definition of what “taint” specifically means within the cyber realm, we would be remiss not to look to the general use of the term, as well as synonyms and antonyms. Merriam Webster [3] provides a useful point-of-departure, as shown in Table 1 below.

Taint Synonyms	blemish, darken, mar, poison, spoil, stain, tarnish, touch, vitiate
Taint Antonyms	decontaminate, purify

Table 1: Merriam Webster Dictionary Taint Information

Note: Taint is also defined in the Universal Dictionary of the English Language as:

“Taint - n. trace of physical corruption or decay; degradation, imperfection; contamination, pollution. Vb. To infect with physical or moral deterioration and corruption; to render unwholesome or noxious. To become infected, corrupted, by something noxious, by decay. Taintless - adj. Without taint; uncorrupted, pure.”

It is important to note that ‘taint’ is a state, consequently independent of intent. Taint for ICT components is expressed in terms of “stateful” properties associated with programmable logic in the components that could have malware, exploitable weaknesses, or vulnerabilities – independent of how the components might have become tainted (e.g., through negligence, sloppy manufacturing hygiene, or malicious intent) – in development or supply chain management.]

So a “tainted” ICT product could be described as one that is blemished, marred, spoiled, or in need of being purified and/or decontaminated. Within the DoD community one must also make use of the definition of “software assurance” provided by the United States Congress in Public Law 113-239 Section 933. Therein, software assurance is defined to mean “the level of confidence that software functions as intended and is free of vulnerabilities, either intentionally or unintentionally designed or inserted as part of the software, throughout the lifecycle.” (Note – this is the same definition used by the Committee on National Security Systems (4001 Glossary.) Taken together, software assurance would also ensure that the software was not tainted, since tainted software would offer an attacker the opportunity to make the software function in an unintended manner.

Similarly, within the software vulnerability community, vulnerability has been defined [4] as an occurrence of a weakness (or multiple weaknesses) within software, in which the weakness can be used by a party to cause the software to modify or access unintended data, interrupt proper execution, or perform incorrect actions that were not specifically granted to the party who uses the weakness.

This paper proposes that for any product making use of software, “taint” would generally be considered any weakness/issue that impacts its ability to function as intended or that otherwise introduces vulnerabilities or malware.

Cyber Hygiene – an Evolution of Systems Engineering Needed

Some offer that we could address our collective cyber software hygiene and assurance problems if only information assurance, cybersecurity, and supply chain assurance could “fit” into the general systems engineering practice. However, as described in the following sections, there are three things that are different about software-based systems when engineering them for today’s cyber world that go beyond what most consider good hygiene and systems engineering. Consequently, past norms of

“good practice” related to software that remain in use in government and industry may be inadequate given today’s pervasive use of software-based products in our new cyber world.

Fragile in Unexpected Ways and Deployed in an Unknown Manner

Software based systems often have additional features, interfaces, and functionality due to the use of 3rd party libraries, general purpose commercial and open source applications, and the multiplicity of features in system libraries and system calls. Moreover, developers often make risk decisions for which they are not held accountable (such as disabling compiler warnings about security flaws in code). As witnessed by the myriad of patches we are regularly called to address due to the exploitable vulnerabilities they address, today’s software-based systems are inherently frail, and susceptible to attack and manipulation in varying ways. To address the differences between what is conceived and what is delivered we need to think about how software-based systems are actually integrated and deployed, as opposed to how they were designed and envisioned. Generally speaking, the real “deployed” system is what attackers study and attack, not the idealized engineering plans often used to manage the systems.

In other words, it is the software actually used in the field that has to be the focus of assurance efforts. So, if libraries are incorporated and deployed by a compiler, or configuration choices undermine design choices, or someone otherwise exposes a weakness, we need to detect this before our adversaries do.

The various human disciplines that relate to engineering (of any sort) have been shaped by our collective historical learning from working in and with the physical world. Within that world, things are essentially repeatable and sufficient description simply comes down to an adequate understanding of the basic physical science’s behaviors and properties. Our training from birth is in this physical world, and our expectations are shaped by that predictable world. Consequently, it is no wonder that we subconsciously and consciously expect that same level of consistency to hold in the cyber domains, though without the evidentiary basis for that belief.

The software-based code and logic underpinning cyber capabilities lack the predictability that the physical world follows. The cyber action and interactions due to the mistakes and flaws in our computer languages, software tools, the training or accountability of software developers, and the ingenuity of attackers does not follow a nice scientific formulation. Until we can get those who create, build, and support the cyber systems that people depend on to understand that cyber is a man-made and man-defined environment and it will not follow any rules other than those we impose and enforce upon them, almost any aspect of a system can become an avenue of attack that puts the mission and our people at risk.

Non-Benign Environments With Attackers

We operate in a non-benign cyber world with attackers and attack techniques that need to be considered as “motivated forces of nature.” In traditional engineering one is often dealing with known hazards and threat agents (e.g., hurricanes, fires,

tornadoes), and generally, with the exception of corrosion, these are one time or short-term issues, from which one can recover.

In contrast, attackers in cyber space are persistent, target-specific entities that will work to identify weaknesses in their specific targets. They will evolve their tradecraft to better leverage the weaknesses of their ICT targets and users. Attackers are a given fact of life for the cyber environment that software-based systems will have to work in and survive to support their intended missions. Mission resilience within our cyber-assisted world is very difficult without ICT resilience.

Adversary Evolution

The third thing that differs from what most engineering approaches expect is the speed and adaptability of the attacks. This is especially true with regards to the rapid evolution of the attack techniques and ability of attackers to adapt to change and new elements of the offensive and defensive cyber environment. While attacker and defender have always had a race, the speed of that race and the breadth and scope of changes are orders of magnitude quicker and broader in the software-based cyber realm.

As an example, if one builds a tank it is understood that eventually the adversary will come up with a better weapon that will force the development of a new and better tank or way of using tanks. Both the development of the tank and subsequent countermeasures will likely take years, fitting nicely into the traditional acquisition lifecycle.

In contrast, the development and evolution of cyber adversary attacks can change in hours or days. Admittedly, some of these are tactical changes but they can still impact the mission. Even if one were to argue it takes weeks to develop a new strategic cyber-attack technique, this is still far faster than the normal acquisition and systems engineering process.

Integrated System Engineering for Hygiene Assurance

While tainted products can be a concern in their own right, the three differences surrounding the engineering of software-based systems from above and the implications that these differences represent in ensuring the systems meet their mission in spite of the intentions of others needs to be threaded throughout the systems engineering activities when the system in question has any cyber components. So rather than fitting into today's systems engineering process, that engineering activity itself needs to be adjusted to better fit the challenges that software brings to our systems so that the systems deployed, with all their fuzzy edges and unplanned features are what we validate, verify, and gain assurance about just as those systems are what the adversaries reconnoiter and attack.

A deep, broad, strategic approach to evolving systems engineering to better address cybersecurity issues in software-based systems that covers education, research, legal liabilities and expectations, business understanding, and systems development methodologies is needed. Only this reworked approach can make software-based systems as reliable and resilient as they need to be given the role they play in the myriad of governance, business, security, and personal endeavors they support.

Assurance For The Most Dangerous

Non-Malicious Issues

There are a wide variety of ways software can become exploitable to an attacker, allowing them to make use of the products in ways that the software's developers and/or those running the software never intended. With this comes the question as to which of these non-malicious issues should an enterprise focus on eliminating or mitigating? Unfortunately, there is no simple answer to that question. Because different organizations can use the same type of software in vastly different ways, the same flaw could be critical to one and a trivial nuisance to the other. Two questions that each enterprise needs to be ready to answer are the questions of what any particular piece of software is doing for them, and how dangerous would different failure modes of that software be to the enterprise. This is the "fitness for use" consideration that each mission must address before accepting software into its operational environment.

While there may be "over 1,000 different categories of security mistakes that developers can make" [5] in the Common Weakness Enumeration (CWE) [6], the community-developed dictionary of software weakness types such as constructs in code, design, architecture and deployment of software that can lead to exploitation by attackers, there appear to be only eight different consequences or technical impacts [7], as shown in Table 1, to which these failures lead. In other words, if a weakness manifests itself in a product in an exploitable manner and an attacker successfully exploits it, then there will be one of eight technical impacts or consequences from "Threat and Vulnerability Assessments" within that weakness. With each CWE entry the "common consequences" field lists the "technical impacts" that can result from each weakness in CWE. The technical impact and its translation into an impact to the mission are important criteria within the DoD's approach to program protection planning and can be equally useful to any organization that needs to have reasonable assurance that their software-based capabilities do what they are intended and nothing more [8, 9].

The collapsing of the hundreds of types of errors into a small set of technical impacts offers a simplification to the question of what an organization should focus on to gain assurance in their software-based products. Instead of trying to remove all weaknesses, they could decide which of the eight impacts are either more or less dangerous to them, given what the software product is doing for their organization. For example, a public web site utilizing Akamai to provide information may not worry about weaknesses that lead to resource consumption denial-of-service exploits but could be extremely concerned about weaknesses that can lead to someone modifying the data. Using this approach they could focus their assurance activities on those weaknesses that could lead to this unacceptable failure mode.

The eight technical impacts are:

- **Modify data**
- **Read data**
- **Denial-of-Service: unreliable execution**
- **Denial-of-Service: resource consumption**
- **Execute unauthorized code or commands**
- **Gain privileges/assume identity**
- **Bypass protection mechanism**
- **Hide activities**

Similarly, there is a “Detection Methods” field within many CWE entries that conveys information about what types of assessment activities that weakness can be found by. More and more CWE entries have this field filled in over time. The recent Institute of Defense Analysis (IDA) State of the Art Research report conducted for DoD provides additional information for use across CWE in this area. Labels for the Detection Methods being used within CWE at present are: Automated Analysis, Automated Dynamic Analysis, Automated Static Analysis, Black Box, Fuzzing, Manual Analysis, Manual Dynamic Analysis, Manual Static Analysis, Other, and White Box.

This offers a second simplification where stakeholders can now match weaknesses against type of assessment activities, and will thereby gain insights into whether that weakness is still an issue, or whether it has been mitigated or removed. Continuing the example above, using the information in Figure 1, the specific CWEs that can lead to that type of impact can be reviewed and the ones that dynamic analysis, static analysis, and fuzzing can gather evidence about and which ones they can not.

Understanding the relationship between various assessment/detection methods and the artifacts available over the lifecycle, better enables decision-makers to plan for: specific issue(s) to review; at what point(s) in the effort; using what method(s); and through the use of the coverage claims representations [10] of the various tools and services, which capability(s) could be leveraged, etc. This is depicted in Figure 2.

This information can assist project staff in planning their assurance activities; it will better enable them to combine the groupings of weaknesses that lead to specific technical impacts with the listing of specific detection methods. This provides information about the presence of specific weaknesses, enabling them to make sure the dangerous ones are addressed.

Figure 1 conveys information associated with the “Software Assurance On-Ramp” portion of the CWE web site. This area of the site is focused on providing help to projects on how they can make use of the information about weaknesses to manage their software security efforts.

Finally, the same type of information in this table could be used to produce an assurance tag for an executable code bundle, leveraging ISO/IEC 19770-2:2009 [11] as implemented for Software Identification (SWID) Tags [12]. SWID Tags can contain assurance information to convey which types of assurance activities and efforts were undertaken against what types of failure modes. The receiving enterprise could then review this tag and match that information against their plan for how they will use the software and what failure modes they are most concerned about. This would be invaluable in determining if sufficient efforts were taken in those areas. [Note: This also supports ISO/IEC 15026 assurance cases.]

Managing Risks Attributable To Taint In Software And Hardware

Hardware follows the physical laws applicable to their composition, electrical characteristics, and construction. Statistical process variations, logical errors of design, or mechanical instabilities may not be originally understood, but can be studied

Technical Impact	Automated Analysis	Automated Dynamic Analysis	Automated Static Analysis	Black Box	Fuzzing	Manual Analysis	Manual Dynamic Analysis	Manual Static Analysis	Other	White Box
Execute unauthorized code or commands		78, 120, 129, 131, 476, 805	78, 79, 98, 120, 129, 131, 134, 190, 798, 805	79, 129, 134, 190, 494, 698, 798		98, 120, 131, 190, 494, 805	476, 798	78, 798		
Gain privileges / assume identity			798	259, 798		259	798	798, 807	628	
Read data	209, 311, 327	78, 89, 129, 131, 209, 404, 665	78, 79, 89, 129, 131, 134, 798	14, 79, 129, 134, 319, 798		89, 131, 209, 311, 327	209, 404, 665, 798	78, 798		14
Modify data	311, 327	78, 89, 129, 131	78, 89, 129, 131, 190	129, 190, 319		89, 131, 190, 311, 327		78		
DoS: unreliable execution		78, 120, 129, 131, 400, 476, 665, 805	78, 120, 129, 131, 190, 400, 805	129, 190	400	120, 131, 190, 805	476, 665	78		
DoS: resource consumption		120, 400, 404, 770, 805	120, 190, 400, 770, 805	190	400, 770	120, 190, 805	404	770		412
Bypass protection mechanism		89, 400, 665	79, 89, 190, 400, 798	14, 79, 184, 190, 733, 798	400	89, 190	665, 798	798, 807		14, 733
Hide activities	327	78	78			327		78		
Other		400, 404	400, 798	198, 484, 494, 698, 733, 798	400	494	404, 798	595, 798, 807	628	484, 733

Figure 1: Weakness Technical Impacts by Detection Methods

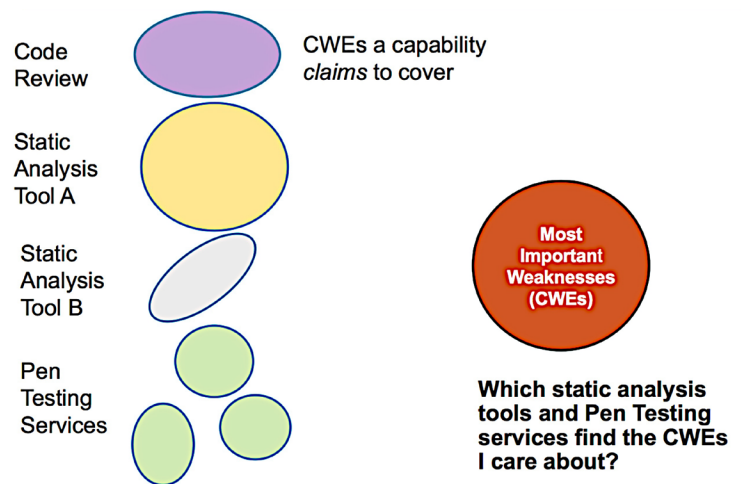


Figure 2: Matching Coverage Claims to Your Organization's Needs

and addressed using general engineering and process improvement methodologies. However, it is clear that software fails from things other than these causes. As discussed above, software follows no laws unless their creators impose them and can fail due to individual implementation mistakes or through the introduction of weaknesses or malicious logic.

Few software developers or systems engineering practitioners have the training and experience to recognize, consider, and avoid these weaknesses. Few (if any) tools or procedures are available to review and test for all weaknesses in a systematic manner. Developers are rarely provided with criteria about what types of problems are possible, and what their presence could mean to the fielded software system and its users.

To manage these risks we cannot just expect to come up with the “right security requirements.” We also need to provide a methodology that assists in gaining assurance through the gathering of evidence and showing how that information provides assurance and confidence that the system development process addressed the removal or mitigation of weaknesses that could lead to exploitable vulnerabilities. The changes in revision 4 of National Institute of Standards and Technology (NIST) Special Publication 800-53 [13] directly bring assurance into the security posture equation.

Making Change Through Business Value

Key to a successful SCRM strategy (beyond good intelligence about threats) is an approach to engage industry hardware and software developers, manufacturers, and resellers, and not just the “contractors” and “integrators.” If necessary appreciation of the problems and requisite risk mitigating behaviors are going to become a core part of the marketplace, then these foundational and ultimate sources of products are where the discipline has to reside. As the software security industry’s norms of behavior evolve, those who sell to governments, the critical infrastructures, and the larger global ICT-consuming economy will leverage and adopt these norms in their own operations. All the various facets and aspects of the marketplace have their own business incentives and cost considerations that can be influenced. Given the right set of motivations, we can all benefit from assured software-based products through a sanitized ICT supply chain. That will contribute to the assurance and confidence that products are fit for use in the respective mission and business environments.

Through their own upstream efforts to their parts and component suppliers, and downstream to their customers, the vendor communities must be motivated to control and manage the quality issues for the supply chain going to government as well as to the civilian critical infrastructure and broad consumer markets. This can be in the interest of both the producer industry and the consumers if the right business value proposition can be found. Under that type of market, the government customer can effectively become (almost) a “free rider beneficiary” of these broader supply chain hygiene and assurance changes.

Not to be confused with classical “motivations” for business, aligning the self interests of the business community with the interests of government and industry on concerns such as ‘taint’ can transform the way everyone conducts their activities. While most community interests cannot drive industry, it is possible to lead industry to a different way of “doing business.” Collectively we can show sustained business value propositions to the various participants through either cost avoidance or market dynamics, which reward their behavior in alignment with the collective interests of all participants in our software-based economy and critical infrastructures.

For an example of a behavior change in an industry motivated by a new perceived business value, consider that many of the vendors currently doing public disclosures are doing so because they wanted to include CVE [14] Identifiers in their advisories to their customers. However, they could not have CVE Identifiers assigned to a vulnerability issue until there was publicly available information on the issue for CVE to correlate. The vendors were motivated to include CVE Identifiers due to requests from their large enterprise customers who wanted that information so they could track their vulnerability patch/remediation efforts using commercially available tools. CVE Identifiers were the way they planned to integrate those tools. Basically the community created an ecosystem of value propositions that influenced the software product vendors (as well as the vulnerability management vendors) to do things that helped the community, as a whole, work more efficiently and effectively.

Similarly, large enterprises are leveraging CWE Identifiers to coordinate and correlate their internal software quality/security reviews and other assurance efforts. From that starting point, they have been asking the Pen Testing Services and Tools community to include CWE identifiers in their findings. While CWE Identifiers in findings was something that others had cited as good practice, it was not until the business value to Pen Testing industry players made sense that they started adopting them and pushing the state-of-the-art to better utilize them.

While motivating business interest usually comes down to incentives and perceptions about market share possibilities, aligning self interests can be an alternate approach to changing things that are both sustainable and win-win for suppliers and customers. These types of symbiotic situations are most certainly available in the various parts of the SCRM challenge space and they present a topic that we collectively need to explore for opportunities and common benefits. Over the past 15 years the community has explored many different ways to influence industry using a wide variety of standards. Community repositories, languages, acceptable usage, or process standards are being considered in terms of the best fit for the variety of different situations faced within the community; and to-date, these have substantively changed the global IT industry in positive and effective ways.

Conclusion

The software and hardware fields need to holistically approach the questions around the hygiene and quality activities that provide assurances that products are fit for their intended use. Negative impacts to the mission can be just as deadly and unmanageable in the field from tainted ICT software-based products regardless of intent (from malice or negligence). Within the military, taint considerations can be addressed as part of the ‘fitness for use’ criteria in program protection planning as can the risk based remediation strategies for addressing software vulnerabilities. Use of CWE and the consideration of the technical impacts that software weaknesses can lead to as a guide to reviewing an organization’s hygiene practices, along with the information about which detection methods are best suited for gathering assurance about the presence or absence of exploitable vulnerabilities, can help when managing a project’s assurance activities in a manner that others will understand and can verify. Until both malicious and non-malicious aspects of taint are dealt with in ways that are visible and verifiable there will be a continued lack of confidence and assurance in the delivered capabilities and the supply chain that sourced and services them.

Acknowledgements

The summary work contained in this article is based on the discussions with a number of individuals at MITRE and throughout the industry; a special thanks is given for the contributions of the CWE Team members and those working Supply Chain Risk Management, including Joe Jarzombek, DHS Director for Software & Supply Chain Assurance, who provided input to this article. The DHS is acknowledged as the sponsor of this work. The MITRE Corporation operates the Homeland Security System Engineering and Development Institute under contract HSHQDC-09-D-0001 for the Department of Homeland Security. ♦

ABOUT THE AUTHOR



Robert A. Martin is a senior principal engineer in MITRE's Cyber Security division. For the past 14 years, his efforts have been focused on the interplay of enterprise risk management, cybersecurity standardization, critical infrastructure protection, and the use of software-based technologies and services. Martin is an ISC2 Certified Secure Software Lifecycle Professional and a member of the ACM, AFCEA, NDIA, INCOSE, IEEE, and the IEEE Computer Society. He has both bachelor's and master's degrees in electrical engineering from Rensselaer Polytechnic Institute, and an MBA from Babson College.

Phone: 781-271-3001

Email: ramartin@mitre.org

REFERENCES

1. "ISO/IEC TR 24772:2013 Information technology -- Information technology -- Programming languages -- Guidance to avoiding vulnerabilities in programming languages through language selection and use", International Organization for Standardization, (http://standards.iso.org/ittf/PubliclyAvailableStandards/c061457_ISO_IEC_TR_24772_2013.zip)
2. "National Defense Authorization Act for Fiscal 2013, Public Law 112-239, Section 933", Jan. 2013 (<http://www.gpo.gov/fdsys/pkg/PLAW-112publ239/pdf/PLAW-112publ239.pdf>)
3. "Definition of Taint", Merriam-Webster Dictionary, Dec. 2013 (<http://www.merriam-webster.com/dictionary/taint/>)
4. "Threat-Classification-Glossary", The Web Application Security Consortium (<http://projects.webappsec.org/w/page/13246980/Threat-Classification-Glossary>)
5. "Secure Code Starts With Measuring What Developers Know", Information Week, Dec. 19, 2013 (<http://www.informationweek.com/security/application-security/secure-code-starts-with-measuring-what-developers-know/d/d-id/1113154>)
6. "The Common Weakness Enumeration (CWE™) Initiative", MITRE Corporation, (<https://cwe.mitre.org/>)
7. "Common Weakness Enumeration - Enumeration of Technical Impacts", MITRE Corporation, (https://cwe.mitre.org/cwraf/enum_of_ti.html)
8. "Requirements Challenges in Addressing Malicious Supply-Chain Threats", Paul R. Popick, and Melinda Reed, INCOSE Insight, July 2013 (<http://www.acq.osd.mil/se/docs/ReqChallengesSCThreats-Reed-INCISE-Vol16-Is2.pdf>)
9. "Program Protection and System Security Engineering", Office of the Deputy Assistant Secretary of Defense (ODASD) Systems Engineering, January 2014 (http://www.acq.osd.mil/se/initiatives/init_ppsse.html)
10. (<https://cwe.mitre.org/compatible/ccr.html>)
11. "ISO/IEC 19770-2:2009 Information technology -- Software asset management -- Part 2: Software identification tag", International Organization for Standardization, (http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=53670)
12. "Software Identification (SWID) Tags", TagVault.org, (<http://tagvault.org/swid-tags/>)
13. "NIST Special Publication 800-53 Revision 4 - Security and Privacy Controls for Federal Information Systems and Organizations", National Institute of Standards and Technology, (<http://dx.doi.org/10.6028/NIST.SP.800-53r4>)
14. "The Common Vulnerabilities and Exposures (CVE®) Initiative", MITRE Corporation, (<https://cve.mitre.org/>)



CALL FOR ARTICLES

If your experience or research has produced information that could be useful to others, **CROSSTALK** can get the word out. We are specifically looking for articles on software-related topics to supplement upcoming theme issues. Below is the submittal schedule for three areas of emphasis we are looking for:

Acquisition of Software-Reliant Capabilities

Sep/Oct 2014 Issue

Submission Deadline: April 10, 2014

Software Engineering Tools and the Processes They Support

Nov/Dec 2014 Issue

Submission Deadline: June 10, 2014

Please follow the Author Guidelines for **CROSSTALK**, available on the Internet at www.crosstalkonline.org/submission-guidelines. We accept article submissions on software-related topics at any time, along with Letters to the Editor and BackTalk. To see a list of themes for upcoming issues or to learn more about the types of articles we're looking for visit www.crosstalkonline.org/theme-calendar.

Collaborating Across the Supply Chain to Address Taint and Counterfeit

Dan Reddy, EMC Corporation

Abstract. Before the community of acquirers and providers of technology can get to the heart of supply chain risk management regarding taint and counterfeit they must reach some consensus on two basics questions: 1) Where is the mitigation focus when we discuss supply chain?, and 2) Are we discussing both quality issues that occur in technology development or just products that have been tampered with along the supply chain?

Introduction

When one speaks of risks inherent in building Information and Communication Technology (ICT) products, is it really a supply chain discussion from soup to nuts? Should there be an equal discussion about the quality of the technology produced and whether the product has been deliberately tampered with during the product lifecycle process? These two questions can quickly derail conversations about secure software development and supply chain risk, two topics that are already complex enough. Framing these debates properly might allow the discussion to productively proceed to how the risks can be mitigated by applying best practices by the right party at the right juncture. The answers to these framing questions can help focus the discussion on where to effectively apply the controls to offset threats. Should the provider take the lead in applying controls only within its own organization? How should the provider ensure that best practices are applied throughout the rest of the supply chain?

What are we guarding against when we consider supply chain risk management? The first major threat is an attack that tampers with a product as it is being sourced, built, or delivered and potentially introduces capability or maliciously inserted code that the original provider of the product never designed or planned to deliver. This tampering related threat also extends to hardware where the attacker's planned substitution of faulty counterfeit components could undermine the manufacturer's planned capability, its performance or introduce new malicious capability. The second threat area is related to the quality of the product. Poor quality practices during the development of software or firmware could lead to bugs or errors that can be exploited by attackers before, during, or after installation, thus undermining another dimension of software assurance. There can be poor quality counterfeit components in the supply chain because someone wants to make money through a lower cost substitute. Therefore not all counterfeits are due to the introduction of malicious capability.

Yes, the customer who ultimately acquires the information technology should reasonably expect quality products without exploitable vulnerabilities stemming from known weaknesses

or malware. This customer should expect that the operational environment in which the product is deployed is uncompromised. The customer should reasonably assume that it is the authentic product from the original provider and it is the high quality product that functions as the technology provider intended, no more and no less. Technology providers stand behind the product that they make and sell. They convey that it is the real deal and the product's integrity has been preserved along the complex creation and delivery journey to the customer. None of these expectations should be in doubt. Every component supplier along the way is inherently a provider in its own right and must stand behind its product in the same way, offering authenticity, integrity and security. These are the three elements of assurance as described by SAFECode, an industry led group formed in 2007 to focus on software assurance [1].



Figure 1 SAFECode Three Elements of Assurance [1]

Is it all Part of the Supply Chain? A Provider-centric View

To address this first pivotal question on the scope of supply chain, one could view the entire process from the concept of a product through its delivery to the customer as a series of complex supply chain interactions with a multitude of players and lose sight of where the primary ICT provider's role, activities and oversight come into play. The provider's role is strongest in what it directly controls in its own shop and more indirect when it relies on others to build and deliver hardware and software components. When they rely on other suppliers, providers can have strong expectations, tests, contracts, acceptance procedures and audits but they do not directly oversee and control many aspects of any one supplier's component in the same way they cover their own practices; they always rely on other parties to some extent.

When SAFECode first published an article on software supply chain integrity [2] in July of 2009, they framed the software supply chain as being comprised of Supplier Sourcing, Product Development and Testing, and Product Delivery. This view is notably provider centric. It is the providers who should develop a program based on best practices as they engage with outside suppliers to source people and components for their products. They then may have their own in-house development and testing activities that govern the software as it is being built and tested to prepare for the final delivery phase which may either be under their own control or may be another opportunity to engage within the supply chain. The framework laid out in the SAFECode whitepaper envisioned a staircase effect comprised of n tiers of suppliers whereby each supplier in the chain would concentrate on applying the best practices for each of these three phases.

This model distributes the operational responsibility to make the practices work most effectively at each tier. It does not change the overall acquirer's or customer's expectation of the provider of the product. Providers can enforce controls in their own organizations while they focus on indirect verification when they engage suppliers. Just as the customer cannot effectively enforce the controls inside the provider's shop, the provider must turn to verifying that controls are in place within their suppliers. If the right players apply the right controls at the right spot, the industry will achieve overall scalability and accountability across the supply chain. It is better than trying to have each customer, each agency, each branch of service or each procurement officer create its own approach. That simply would not scale and is likely to become derailed in the pivotal discussions outlined here.

SAFECode in a later publication [4] outlined the specific set of controls that is applicable at each lifecycle process phase. For all dealings with suppliers and delivery partners who source people, services and components, one might describe the interactions to protect the supply chain as "engagement controls" such as writing contracts to set expectations or looking for measures of authenticity and integrity such as digitally signed code or verifying cryptographic checksums to validate a binary deliverable. Engagement controls begin in the provider's enterprise but extend out into the supply chain at various touch points. These controls include how the provider's enterprise brings on contractors for in-house work, how they accept delivery and test software components from a supplier, and how they determine who is an authorized service partner. Such controls are shaped first in the provider's organization and then come into play in preparation for engagement with outside suppliers. Since they are applied between organizations, they differ in their reach from the direct enforcement controls that a software provider should

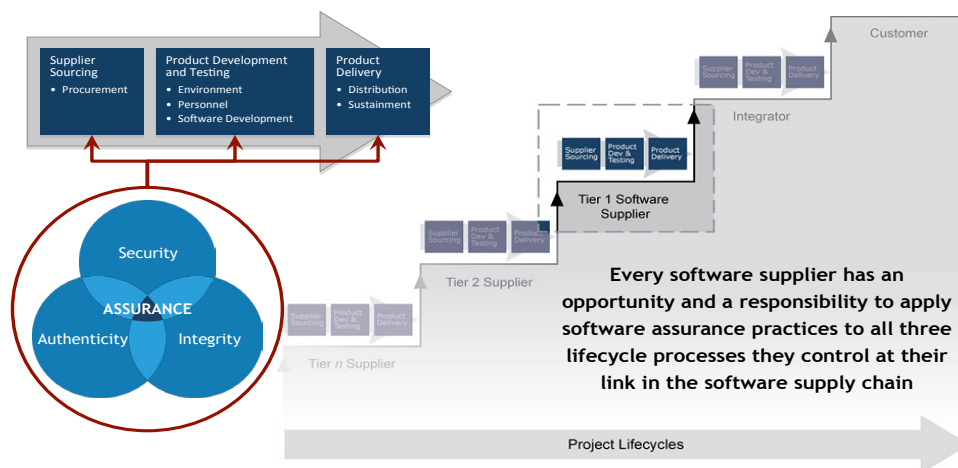


Figure 2 SAFECode Supply Chain Framework [3]

apply in-house with their own resources as they develop and test good software. The application, monitoring and governance of controls naturally will also require ongoing sustainment and recalibration.

The same approach can apply to avoiding counterfeits throughout the supply chain. One must first acknowledge that an attacker could create a perfect replica of a component that functions and performs well from a quality perspective—but it could have intentional malicious capability added. Testing hardware for meeting functional specifications alone is not sufficient for this abuse case. Knowing the strength of the chain of custody of the component and being alert for the potential impact of an inauthentic component in the architecture may inform proper negative testing or additional hardware tests for traceability and other characteristics that may still be required. However, testing hardware for quality can identify faulty counterfeit components that have entered the supply chain purely for someone's monetary gain. These different approaches can be applied both in-house where hardware may be involved and also by engaging suppliers in the chain when some of the hardware work is done through sourcing.

The best way to mitigate risk throughout the supply chain is for all ICT providers to adopt common industry best practices while delivering their own quality products and mitigating the risk of having a product tampered with or including counterfeit components along the way. If each supplier whose components, code, or assemblies along the supply chain subscribes to best practices that could be measured based on commonly defined outcomes, the customer could develop a deeper sense of trust down multiple levels into the supply chain. No single provider can reach down deep enough into tiers of suppliers to highlight where best practices do occur unless an overall ecosystem that includes provider, supplier and acquirer evolves to expect and measure compliance as each supplier comes into view. An ICT provider may have dozens or hundreds or more suppliers. Even if there is rigor and consistency in how the expectations are set, monitored and verified by the primary provider as component items are

specified, built, sourced and delivered across the supply chain, it will not convey the same confidence as it would if each supplier along the chain could also adopt global standards. Each supplier could pass along assurance as to their compliance so that the provider could summarize the results in the aggregate for their customers. Then the confidence would be evident and visible at each tier in the supply chain, at least for some basic assurance. The lens then needs to focus on each supplier to make sure its own house is in order. This is not to beg for a pass for the distant tiers of suppliers in the chain because they are remote, but to recognize that the law of physics works against having the same level of deep control throughout the chain as providers can when they supervise their own organizations.

The ideal in the fully evolved ecosystem is to have best practices occur both within the sphere of the provider's own shop and also at each tier in the supply chain. In order to make such an ecosystem scalable and viable, there must be practical methods to achieve and measure common outcomes. The most effective method is to have each supplier along the complex supply chain be evaluated against a global standard by a qualified assessor who can perform such an assessment in a reliably consistent manner. Then the provider does not have to sustain a unique conversation with each supplier as to expected good software development practices, good anti-counterfeiting practices along with good practices to prevent acquiring products that have been tampered with at any point along the lifecycle. The global Open Group's Open Trusted Technology Provider Standard (O-TTPS) [5] to mitigate maliciously tainted and counterfeit products is designed to enable all ICT providers to be evaluated by recognized third-party assessors. O-TTPS includes more than 50 requirements relating to how products are developed, how secure engineering is applied, and how supply chain security risks for maliciously tainted and counterfeit are addressed. Ideally then in the future state of the ecosystem each provider should be able to expect that their own preferred suppliers would have gone through their own process of becoming accredited and be listed in the Open Group's Trusted Technology Provider registry of accredited organizations. Each customer or acquirer would then be able to identify the associated set of products from each accredited organization that conforms to the best practices.

The O-TTPS outlines a distinction between those requirements that are specifically related to the provider's own shop (considered as part of Technology Development) and those requirements that involve an engagement with suppliers (considered as part of Supply Chain). In fact Figure 3 shows an example of mottled shading over the blocks depicting the stages of the lifecycle in relation to the technology development and

supply chain. This reflects the reality that the number of touch points may vary between the provider and various suppliers that are engaged in any particular product's development lifecycle. Some products have a high internal development profile and others may have more touch points with external supplier organizations that contribute to the product along the lifecycle. All of the requirements must be met by the provider, but the O-TTPS does reference a best practice whereby providers seek qualified suppliers that follow the same set of practices as those embodied in O-TTPS. This recursive requirement should help facilitate the ecosystem in reaching its potential.

Addressing Software Assurance, Quality, and Tampering

Quality begins at home in the provider's own shop, regardless of whether they sell to an end-user or act as a supplier to another ICT provider. Good software development with security in mind should follow an array of good practices to avoid common mistakes so that the ultimate software produced is less subject to bugs or weaknesses that can be exploited during an attack. Following a software development lifecycle (SDLC) with security in mind is a discipline unto itself. A product development lifecycle imbued with secure engineering starts (like the supply chain discussion above) with making sure that developers in the provider's own shop are well trained, focused on activities like secure design, threat modeling, secure coding, proper iterative testing, ensuring a hardened state of all components and good documentation for the ultimate customer concerning the correct usage of the security related configurations. These are a few of the everyday practices that are considered quality related on the part of the developer in the provider's shop. As an industry-led organization for sharing best practices, SAFECode has outlined how to securely develop software in its whitepaper [6]. If an organization wants to model its own secure software development practices on those of the industry, this whitepaper would be a great place to start for some detailed recommendations. The software development organization's first obligation is to do the right thing from the beginning. It is no longer acceptable for a developer to say, "I did not know about the most dangerous errors to avoid while building software; I will be better next time."

Once these practices become routine for the provider and institutional knowledge is strengthened through ongoing measurement, adjustment and oversight, the bar is raised and development teams need to tackle the next challenges in building quality products that are resilient. For the moment let us assume that coding errors that are found are due to insufficient software design and hygiene being applied while building a technology

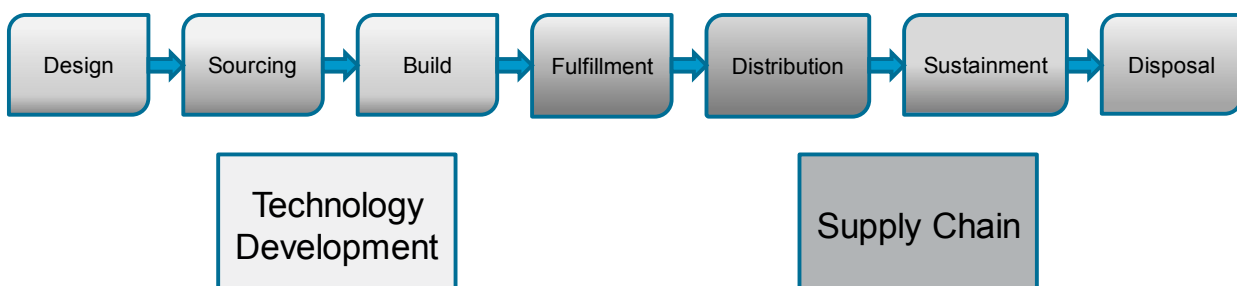


Figure 3. Sample view of the relationship between Technology Development and Supply Chain in O-TTPS [4]

WANTED

Electrical Engineers and Computer Scientists *Be on the Cutting Edge of Software Development*

The Software Maintenance Group at Hill Air Force Base is recruiting **civilians** (*U.S. Citizenship Required*). Benefits include paid vacation, health care plans, matching retirement fund, tuition assistance, and time paid for fitness activities. **Become part of the best and brightest!**

Hill Air Force Base is located close to the Wasatch and Uinta mountains with many recreational opportunities available.



facebook

www.facebook.com/309SoftwareMaintenanceGroup

Send resumes to:
309SMXG.SODO@hill.af.mil
or call (801) 775-5555



product. These errors may lead to vulnerabilities that can in turn be exploited by attackers. Such errors, bugs and vulnerabilities are a fact of life in the world of software development; they can be reduced, but they are not going away any time soon. The immediate customer priority is a requirement for the product organization to have a mature process to respond to known problems and address them effectively, maintaining close linkages to both the customer community and engineering teams. The next challenge is building a sustainable means of avoiding errors or bugs in the future to the extent possible. Good software development and support from the provider's internal governance process can reduce obvious gaps through the diligent application of such best practices by each provider across the supply chain.

The question is not whether the provider needs to be concerned with the quality issues. It is a matter of how the provider focuses to engage and verify what they receive from each supplier. Software is often delivered from its original supplier to a provider who in turn embeds the software as a component (perhaps as firmware) in an overall product. The supplier can be a commercial entity or an open source community. The provider cannot effectively go in and manage the SDLC process for the supplier or run all of the same tests that the original developer can run. For example, assume that the software development

team uses threat modeling during design and again for later testing and verification. Let us also assume that their static source code is analyzed, triaged and fixed on an iterative basis during ongoing development and updates. The provider can reasonably expect to determine if the original development organization follows such practices and conducts them with growing competence and repeatability. It is not reasonable or scalable to assume that the provider will literally inspect or oversee such activities in someone else's original development organization. Instead, if each development organization could be accredited for having and following good product development and secure engineering along with supply chain practices, then the unique conversations between each tier of technology provider and the acquirer of the technology can be reduced. The provider's contracts can then require demonstrated adherence to measureable global standards such as the Open Group's recently announced Open Trusted Technology Provider accreditation process.

With such a foundation of development practices to guard against exploitable software quality weaknesses, each organization can then focus on guarding against tampering with a product. Tampering could occur during lapses in custodial care in the original development organization or elsewhere throughout the rest of the cycle among supply chain players. Each supplier must make sure that the integrity and authenticity of the end product

are strong and evident during the entire development cycle and afterwards throughout the cycle of being installed at a customer site and updated over its lifetime. Supplier and provider alike can require that proof of authenticity and integrity are evident as they exchange packages. In addition to these checks, the provider can test to make sure that no known malware resides in the received package. If such malware is found, is it likely to have been maliciously inserted along the way either in the original development shop or somewhere in the rest of the supply chain. Could it have spread to the environment by malicious design or could the contamination have been somewhat inadvertent? All of these best practices can be tied to the achievement of accreditation against a global standard and thereby define an important foundational stratum of capability within a development organization.

The public expects ICT providers to produce quality results delivered with provable and intact authenticity and integrity along the way. Each provider and supplier in the ecosystem must do its share to deliver these results. The improvement cycle starts with the global definition of industry practices that can be shared by providers to achieve security, integrity and authenticity of the software and hardware components they supply. Then the baseline of industry practice needs to be complemented by an accreditation regime that can measure and report how well these controls are being applied to each provider involved in the ICT supply chain. With those elements now in place the industry can move forward and leverage these defined practices and measure basic adherence to them instead of spending energy debating whether the customer and provider are referring to the same risks. ♦



Homeland Security

The Department of Homeland Security, Office of Cybersecurity and Communications (CS&C) is responsible for enhancing the security, resiliency, and reliability of the Nation's cyber and communications infrastructure and actively engages the public and private sectors as well as international partners to prepare for, prevent, and respond to catastrophic incidents that could degrade or overwhelm these strategic assets. CS&C seeks dynamic individuals to fill critical positions in:

- Cyber Incident Response
- Cyber Risk and Strategic Analysis
- Networks and Systems Engineering
- Computer & Electronic Engineering
- Digital Forensics
- Telecommunications Assurance
- Program Management and Analysis
- Vulnerability Detection and Assessment

To learn more about the DHS, Office of Cybersecurity and Communications, go to www.dhs.gov/cybercareers. To apply for a vacant position please go to www.usajobs.gov or visit us at www.DHS.gov.

ABOUT THE AUTHOR



Dan Reddy leads Supply Chain Assurance in EMC's Product Security Office. He was co-chair for SAFECode's whitepaper on Supply Chain Integrity Controls. He's co-chair of the Open Group's Trusted Technology Forum's Acquisition workstream. Dan spent 15 years at New England Electric, a major utility with critical infrastructure. He teaches CIS at Quinsigamond Community College. Dan graduated from Tufts and holds M. Ed. degrees from Worcester State University in education and computer science.

EMC Corporation
171 South Street
Hopkinton, MA 01748
Phone: 508-435-1000
E-mail: dan.reddy@emc.com
E-mail: dan.reddy@gmail.com

REFERENCES

1. "Overview of Software Integrity Controls An Assurance-Based Approach to Minimizing Risks in the Software Supply Chain" June 2010: 1. pag. SAFECode. SAFECode, June 2010. Web. <www.safecode.org>
2. "Framework for Software Supply Chain Integrity July 2009: n. pag. SAFECode. The Software Assurance Forum for Excellence in Code, July 2009. Web. <www.safecode.org>
3. Copyrighted combined graphic from SAFECode Supply Chain Integrity presentation - The Software Assurance Forum for Excellence in Code, July 2010)
4. "Overview of Software Integrity Controls An Assurance-Based Approach to Minimizing Risks in the Software Supply Chain June 2010: n. pag. SAFECode. SAFECode, June 2010. Web. <www.safecode.org>
5. "Open Trusted Technology Provider Standard (O-TTPS)™"Version 1.0, Mitigating Maliciously Tainted and Counterfeit Products" 15. pag. Open Group. The Open Group, April 2013. Web. <www.opengroup.org>.
6. "Fundamental Practices for Secure Software Development 2nd Edition." SAFECode. The Software Assurance Forum for Excellence in Code, Feb. 2011. Web. <www.safecode.org>.

Software and Supply Chain Risk Management Assurance Framework

Don O'Neill, Independent Consultant

Abstract. The DoD, the defense industrial base, and the nation's critical infrastructure all face challenges in Supply Chain Risk Management Assurance. These diverse challenges span infrastructure, trust, competitiveness, and austerity. Beginning with acquisition where Supply Chain foundations are laid, Supply Chain Risk Management Assurance extends into operations and sustainment.

Terms of Reference

Supply Chains are essential to global competitiveness and national security. Consequently, a Supply Chain Risk Management Assurance Framework will be produced, refined, and sustained. In addition its rollout and maturity of adoption will be measured. To encourage adoption and motivate maturity progress, appropriate public policy measures will be sought.

Supply Chains in the wild are intrinsically risky, vulnerable to Cybercrime and Cloud Computing risks as well as organizational neglect and unmet needs. The practice of risk management using smart and trusted tactics is necessary because software-based supply chains are inherently insecure, the risks and uncertainties are prolific, and vulnerabilities abound. The combination of unmet needs, industry neglect, and austerity coupled with the immature state of software, Cyber Security, and Cloud Computing infrastructure yield a rich environment of uncertainty and risk in establishing and maintaining infrastructure, being trusted, being competitive, and being austere.

The objective of rolling out the Supply Chain Risk Management Assurance Framework is to advance the assurance of smart and trusted risk management principles and practices useful and essential for military, government, critical industries, and commercial industry.

The benefit of the Supply Chain Risk Management Assurance Framework will be to elevate the expert application of risk management principles and practices in order to reduce uncertainty in the assurance of software-intensive Supply Chains through the use of smart and trusted tactics designed to deliver consequential outcomes in assuring Supply Chain trustworthiness, security, resilience, product integrity, coordination, control, and flexibility.

Specifically, adopters of the Supply Chain Risk Management Assurance Framework will be better able:

1. To systematically pinpoint the factors and sources of risk involved in Supply Chain Risk Management on software-intensive projects.
2. To identify concrete Supply Chain Risk Management objectives involved and consequential outcomes sought in meeting each of the goals associated with establishing and maintaining infrastructure, being trusted, being competitive, and being austere.

3. To visualize the operations of the Software and Supply Chain Risk Management Assurance and achievable Service Level Agreements through the use of assurance assertions based on goals and objectives and appropriate indicators, measures, metrics, and analytics.

4. To calculate Supply Chain assurance risks based on the factors evaluated for each goal, a count of factors that might serve as sources of problems in goal achievement, and a count of factors that represent objectives that are in a failed state. For each goal, the calculated risk is the number of problems divided by factors evaluated expressed as a percent.

5. To calibrate Supply Chain Risk Management Assurance rollout and maturity progress with military, government, critical infrastructure, and commercial industry peers useful in boosting brand value.

The criteria for success of the Supply Chain Risk Management Assurance Framework lies in the rate of adoption by Supply Chain owners and operators followed by their progressive advancement in Supply Chain Risk Management Assurance maturity. Five levels of maturity are envisioned spanning management, commitment, engineered flow control, process risk calculation, and global challenge strategies.

Framework Foundations

Software and Supply Chain Risk Management Assurance is the application of risk management principles and practices to reduce uncertainty in the assurance of software-intensive Supply Chains through the use of software-based smart and trusted tactics designed to deliver consequential outcomes in assuring Supply Chain trustworthiness, security, resilience, product integrity, coordination, control, and flexibility.

The purpose of the Supply Chain Risk Management Assurance Framework is not to prescribe or impose supply chain practice or methods. Instead the purpose of the Supply Chain Risk Management Assurance Framework is to encourage, stimulate, and incentivize supply chain owners and operators to positively and proactively assure the identification, mitigation, transfer, or acceptance of the sources of risk, problems, and factors that may impede the achievement of supply chain goals and objectives.

Descriptive not prescriptive, it is important to understand that the Supply Chain Risk Management Assurance Framework, by necessity, operates under three levels of indirection. First, it is a framework, a basic underlying structure. Second, it promises assurance, a positive declaration intended to inspire confidence. Third, it manages risk through the identification, elimination, mitigation, transfer, or acceptance of factors, sources of risk, and problems that may impede the achievement of supply chain goals and objectives.

1. As a framework, the Supply Chain Risk Management Assurance Framework focuses on the infrastructure of management, engineering, process, technology, and skills needed to acquire, field, and operate trusted, competitive, and austere software-based supply chains with intelligence and confidence.

2. As an assurance mechanism, the Supply Chain Risk Management Assurance Framework validates positive declarations intended to inspire confidence using assurance assertions and levels of confidence.

3. As a risk mechanism, the Supply Chain Risk Management Assurance Framework seeks to prudently assign the disposition of risks associated with factors, sources of risk, and problems and to calculate the risk associated with supply chain goals and objectives.

Setting goals is the first step in managing risk. Goals are attributes to be assured. Goals associated with smart and trusted Software and Supply Chain Risk Management Assurance include maintaining infrastructure, being trusted, being competitive, and being austere. Objectives associated with goals are factors with consequential outcomes, for example, performance, product integrity, mission, resilience, innovation, flexibility, efficiency, control, leadership, coordination, and risk. See Figure 1.

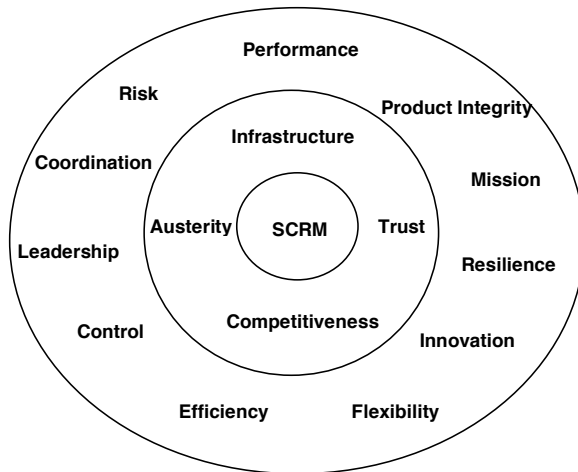


Figure 1. Strategic Goals and Tactical Objectives

Software and Supply Chain Risk Management Assurance involves the assessment of trust in the chain of custody of components, the detection of counterfeit and tainted components, and much more. The Supply Chain Risk Management Assurance Framework spans a wide range of goals, objectives, elements, issues, and factors associated with systems and software management, process, and engineering methods, practices, technologies, tools, and skills underlying acquisition and operations. See Table 1.

Risk Management

Risk is uncertainty and the prospect for loss or gain depending on the outcome of an event. An industrial strength software risk management practice is one that treats risk

as uncertainty, carefully distinguishes risks from the sources of risk and problems, and doesn't use risk management as an off ramp to avoid actually solving problems. Since risk is uncertainty, the challenge is to calculate the uncertainty of a risk and accept only those risks whose joint probability of occurrence and prospect for loss or gain are prudent choices. These are considered calculated risks.

Setting goals is the first step in managing risk. Goals are attributes to be assured. Goals associated with smart and trusted Software and Supply Chain Risk Management Assurance include maintaining infrastructure, being trusted, being competitive, and being austere. Objectives associated with goals are directed at consequential outcomes. Sources of risk are objective outcomes whose achievement is at risk and uncertain. Problems are objective outcomes that have failed. See Table 2 for a work in progress illustration.

Supply Chain assurance risk is calculated based on the factors evaluated for each goal, a count of factors that might serve as sources of problems in goal achievement, and a count of factors that represent objectives that are in a failed state. See Table 3 and Figure 2. Determining the level of confidence in assigning a failed state is assisted by evidence-based assurance assessment questions such as those in Table 4. For each goal, the calculated risk is the number of problems divided by factors evaluated expressed as a percent.

Overall Supply Chain assurance risk is determined by rule.

- R1- If infrastructure or Trust or Competitiveness or Austerity = High Risk, Then Supply Chain := High Risk

Software and Supply Chain Risk Management (SCRM) Factors	Acquisition/Operations	Goal: Infrastructure, Trust, Competitiveness, Austerity
Supply Chain Risk Management (SCRM)	Acquisition, Operations	Infrastructure
Risk Management (RM)	Acquisition, Operations	Infrastructure
Assurance Assertion Management (AAM)	Acquisition, Operations	Infrastructure
Capability Maturity Model Integration (CMMI)	Acquisition, Operations	Infrastructure
State of Austerity	Acquisition, Operations	Infrastructure, Austerity
State of Software	Acquisition, Operations	Infrastructure, Trust, Competitiveness
State of Security	Acquisition, Operations	Infrastructure, Trust
State of Cloud Security	Acquisition, Operations	Infrastructure, Trust
NIST Cloud Computing Reference Architecture (CCRA)	Acquisition, Operations	Infrastructure, Trust, Austerity
NIST Cloud Service Level Agreements (SLA)	Acquisition, Operations	Infrastructure, Trust
NIST Cyber Framework (CF)	Acquisition, Operations	Infrastructure, Trust
Cyber Tactics (CT)	Operations	Trust
Software Assurance (SA)	Acquisition, Operations	Trust
Trusted Chain of Custody (TCC)	Acquisition, Operations	Trust
Counterfeit and Tainted Component Detection (CTCD)	Acquisition, Operations	Trust
Software Product Engineering (SPE)	Acquisition	Trust
Trustworthy Software Engineering (TSE)	Acquisition	Trust
Technical Debt (TD)	Acquisition, Operations	Trust, Austerity
Software Project Management (SPM)	Acquisition	Austerity
Earned Value Management (EVM)	Acquisition	Austerity
Fixed Price Contracting (FPC)	Acquisition	Austerity
Offshore Outsourcing (OO)	Acquisition	Competitiveness, Austerity
Cost Return Ratio (CRR)	Acquisition	Competitiveness, Austerity
Next Generation Software Engineering (NGSE)	Acquisition	Competitiveness, Austerity
Frequency of Release (FR)	Acquisition, Operations	Competitiveness, Trust
Global Software Competitiveness (GSC)	Acquisition, Operations	Competitiveness
Science, Technology, Engineering, Mathematics (STEM)	Acquisition, Operations	Competitiveness
Team Innovation Management (TIM)	Acquisition	Competitiveness

Table 1. Supply Chain Risk Management Factors

- R2- If infrastructure or Trust or Competitiveness or Austerity = Moderate Risk and not High Risk, Then Supply Chain := Moderate Risk
- R3- If infrastructure and Trust and Competitiveness and Austerity = Low Risk, Then Supply Chain := Low Risk

Goal	Objective	Source of Risk	Problem	Indicator
Maintain infrastructure	SSCRM Framework Risk Management Assertion Mgt. Austerity risk Software risk Internet risk Cloud risk	SSCRM Framework Risk Management Assertion Mgt.	Austerity risk Software risk Internet risk Cloud risk	Adoption rate Assertions met Assertions made Unmet needs Neglect, defects Incidents SLA mix
Be trusted	Reputation Build Security In Completeness Correctness Consistency Software Product Eng. Quality Technical Debt Cyber Security Cyber Tactics Trustworthiness Security Resilience Chain of Custody Counterfeit High Assurance Survivability Mission compliant	Reputation Completeness Correctness Consistency Software Product Eng. Quality Cyber Tactics Trustworthiness Chain of Custody Counterfeit High Assurance Survivability Mission compliant	Build Security In Technical Debt Cyber Security Security Resilience	Customer loyalty Vulnerabilities Defects, traceability Defects Defects Defects, Complexity Defects, complexity Neglect Incidents Defects Unmet need, defects Incidents Unmet need, Incidents Traceability Incidents Unmet need, Failures Unmet need, Failures Mission failure
Be competitive	Control work force STEM Control customers Control competition Control event threats Sustainability of wages	Control work force Control customers Control competition Sustainability of wages	STEM Control events	Open requisitions STEM Customer satisfaction Win/loss ratio Unmet need Wage metrics
Be austere	Economics NGSE Fixed Price Software Project Mgt. Earned Value Offshore Outsourcing CMMI Cloud Computing	Economics NGSE Software Project Mgt. Earned Value Offshore Outsourcing CMMI Cloud Computing	Fixed Price	Cost Adoption rate Adoption rate Schedule, cost var. EVM adoption rate Cost Return Ratio CMMI Maturity Level Adoption rate

Table 2. Supply Chain Risk Management, Assurance Goal, Objective, Source of Risk, Problem, and Indicator Table

Goals	Factors Evaluated	Sources of Problems	Problems	Calculated Risk	Risk by Rule
Infrastructure	7	3	4	57.7%	High
Trust	18	13	5	27.7%	Moderate
Competitiveness	6	4	2	33.3%	Moderate
Austerity	8	7	1	12.5%	Low
Supply Chain	39	27	12		High (R1)

Table 3. Calculated Risks

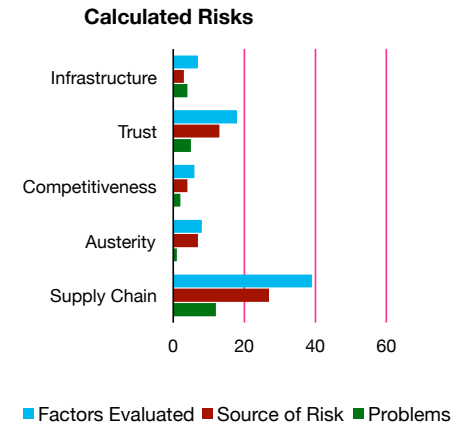


Figure 2. Calculated Risks

Specific findings:

1. Maintaining infrastructure is considered high risk due to the failed or immature state of austerity, software, Internet, and Cloud.
2. Being trusted is considered moderate risk due to the failed or immature state of Build Security In, Technical Debt, Cyber Security, Security, and Resilience.
3. Being competitive is considered moderate risk due to the failed or immature state of STEM and control of event threats.
4. Being austere is considered low risk due to the failed or immature state of fixed price contracting.

Recommendations:

1. Supply Chain success depends on maintaining an infrastructure of austerity, software, Internet, and Cloud. Very high priority attention and investment in management and engineering are recommended.
2. Supply Chain success depends on being trusted. High priority attention and investment in management, engineering, and process in the factors of Build Security In, Technical Debt, Cyber Security, Security, and Resilience are recommended.
3. Supply Chain success depends on being competitive. Priority attention and investment in STEM and control of event threats are recommended.
4. Supply Chain success depends on being austere. Attention and investment in management, engineering, and process in the factor of fixed price contracting is recommended.

Conclusion

With austerity as the context and contemporary software and security as unmet challenges, the Software and Supply Chain Risk Management Assurance Framework focuses on the infra-

Software and Supply Chain Risk Management (SCRM) Factors	Evidence-based Assurance Assertion Argument Assessment Questions
Cloud Computing (CC)	1. Has the Cloud vendor's reputation been assessed and found to be acceptable? 2. Does the Cloud vendor comply with the NIST Cloud Computing Reference Architecture (CCRA)? 3. Does the Cloud vendor accept a Service Level Agreement to protect and safeguard proprietary data and information? 4. Have the possibility and security risks of multi-tenancy been assessed and found to be acceptable?
Software Assurance (SA)	1. Are build security in practices followed? 2. Are known security vulnerabilities understood, monitored, and avoided? 3. Are known security weaknesses understood, monitored, and avoided?
Trusted Chain of Custody (TCC)	1. Is the chain of custody identified? 2. Is the accountability of permissible access maintained, updated, and reviewed? 3. Is the chain of custody unbroken? 4. Are access logs kept, maintained, and reviewed for evidence of tampering?
Counterfeit and Tainted Component Detection (CTCD)	1. Does the organization have the capability to apply Static Analysis of source code? 2. Does the organization have the capability to apply Function Extraction of object code? 3. Does the organization have the capability to apply rigorous software inspections process of source code?
Science, Technology, Engineering, Mathematics (STEM)	1. Are STEM workforce requirements known and understood? 2. Are outstanding personnel requisitions periodically reviewed by senior management? 3. Are alternate sourcing measures including offshore outsourcing identified, understood, monitored, and activated as necessary?
Offshore Outsourcing (OO)	1. Is a cost return ratio calculated during planning and recalculated periodically? 2. Are control points established for the global enterprise? 3. Are control points established for the outsource vendor? 4. Does a trusted pipe architecture feature an in-country control point connected by high speed, secure line to an out-country control point with defined capabilities and protocols. 5. Do intelligent middlemen possess necessary hard and soft skills? 6. Are outsourcing risks understood, monitored, and controlled?
Global Software Competitiveness (GSC)	1. Are suppliers understood, monitored, and controlled? 2. Are customers understood, monitored, and controlled? 3. Are competitors understood, monitored, and controlled? 4. Are event threats understood, monitored, and controlled?
Fixed Price Contracting (FPC)	1. Is the organization committed to Fixed Price Contracting? 2. Is the organization committed to systems engineering and software engineering collaboration? 3. Are software development plans structured as incremental development with well specified design levels and cost accounts? 4. Is there strict accountability of cost accounts based on a work breakdown structure? 5. Are fixed price doctrine tenets for project management, process management, and product engineering adhered to in practice?

Table 4. Examples of Evidence-based Assurance Assertion Argument Assessment Questions Useful in Determining Level of Confidence

structure of management, engineering, process, technology, and skills needed to acquire, field, and operate trusted, competitive, and austere software-based supply chains with intelligence and confidence.

Within the Software and Supply Chain Risk Management space uncertainties associated with vulnerabilities, threats, sources of risk, and problems abound. Beginning with infrastructure uncertainties including state of austerity, software, security, and Cloud Security, these uncertainties go on to include an uncertain industry state of readiness to develop and field trusted large scale software intensive systems with confidence.

Supply Chain security has emerged as a challenge calling for strategies to combat Cyber crime, economic espionage, military espionage, and Cyber warfare. Most notably the critical infrastructure needs to be trusted both with respect to economic security and public safety. What is the industry response to this challenge?

In short, Supply Chains must be trusted; Supply Chains must have integrity. Seeking trust, one approach in assuring integrity is a transparent certification regime based on a convincing demonstration and sufficient assurance assertion argument evidence. Rejecting trust, another approach in a different direction is indigenous innovation where no one trusts anyone and each

country avoids foreign dependency and limits itself to domestic sourcing.

Even where trust is sought, there is the realistic acknowledgement of residual risk even in the presence of transparent, audited lifecycle processes for design, sourcing, and sustainment and the inclusion of a credible response and recovery process for event threats that may occur despite best efforts.

In conclusion, industry needs to evolve global standards for Supply Chain Risk Management Assurance. There needs to be transparency on how products are produced and distributed. Finally, flexibility must be a foremost objective so as not to stifle innovation, which is best achieved by focusing on the what not the how. ♦

ABOUT THE AUTHOR



Don O'Neill served as the President of the Center for National Software Studies (CNSS) from 2005 to 2008. Following 27 years with IBM's Federal Systems Division (FSD), he completed a three-year

residency at Carnegie Mellon University's Software Engineering Institute (SEI) under IBM's Technical Academic Career Program and has served as an SEI Visiting Scientist. A seasoned software engineering manager, technologist, independent consultant, and expert witness, he has a Bachelor of Science degree in mathematics from Dickinson College in Carlisle, Pennsylvania. His current research is directed at public policy strategies for deploying resiliency in the nation's critical infrastructure; disruptive game changing fixed price contracting tactics to achieve DoD austerity; smart and trusted tactics and practices in Supply Chain Risk Management Assurance; and a defined Software Clean Room Method for transforming a proprietary system into a Clean System devoid of proprietary information, copyrighted material, and trade secrets as well as investigating, confirming, verifying, and validating the results.

E-mail: oneilldon@aol.com

REFERENCES

1. "Hacksurfer Cybercrime Info- As Everything Becomes Outsourced, The Dangers to Business Increase", Hacksurfer, 23 September 2013
2. Sheffi, Yossi, "Supply Chain Strategy: Building a Resilient Supply Chain", Harvard Business Review, Volume 1 Number 8, October 2005
3. Goldratt, Eliyahu and Jeff Cox, "The Goals: A Process of Ongoing Improvement", North River Press, ISBN-10 0884271951, Revised Edition, June 1, 2012, 408 pages
4. Wieland, A., Wallenburg, C.M., 2012. Dealing with supply chain risks: Linking risk management practices and strategies to performance, International Journal of Physical Distribution & Logistics Management, 42(10).
5. O'Neill, Don, "Extending the Value of the CMMI to a New Normal", CrossTalk, The Journal of Defense Software Engineering, January/February 2012 <<http://www.crosstalkonline.org/storage/issue-archives/2012/201201/201201-O'Neill.pdf>>
6. "Software 2015: A National Software Strategy to Ensure U.S. Security and Competitiveness", Center for National Software Studies, May 2005, <<http://www.cnsoftware.org/nss2report/NSS2FinalReport04-29-05PDF.pdf>>
7. Resilient Military Systems and the Advanced Cyber Threat, Department of Defense Defense Science Board Task Force Report, January 2013
8. "Professionalizing the Nation's Cybersecurity Workforce? Criteria for Decision Making", National Research Council of the National Academies, The National Academies Press, Washington DC, prepublication, 2013
9. Kramer, Franklin D., Stuart H. Starr, and Larry K. Wentz, Cyberpower and National Security, National Defense University, ISBN 978-1-59797-423-3, Potomac Books, Inc., 2009, page 26
10. Badger, Lee, Tim Grance, Robert Patt-Comer, and Jeff Voss, Cloud Computing Synopsis and Recommendations, NIST Computer Security Division Special Publication SP 800-146, May 29, 2012
11. "Supply Chain Risk Management Practices for Federal Information Systems Organizations", Initial Public Draft, August 2013
12. O'Neill, Don, "Cyber Strategy, Analytics, and Tradeoffs: A Cyber Tactics Study", CrossTalk, The Journal of Defense Software Engineering, September/October 2011 <<http://www.crosstalkonline.org/storage/issue-archives/2011/201109/201109-O'Neill.pdf>>
13. O'Neill, Don, "Peer Reviews", Encyclopedia of Software Engineering- Volume 2, Second Edition, Edited by John Marciniak, John Wiley & Sons, Inc., January 2002, pp. 929-945
14. Schulmeyer, G. Gordon, "Handbook of Software Quality Assurance", Artech House, Inc. 2008, ISBN-13: 978-1-59693-186-2, 464 pages
15. Hevner, Alan R., Richard C. Linger, Rosann W. Collins, Mark G. Pleszkoch, Stacy J. Prowell, and Gwendolyn H. Walton, "The Impact of Function Extraction Technology on Next-Generation Software Engineering", Carnegie Mellon University CERT, CMU/SEI-2005-TR-015, July 2005
16. Wheeler, David A. and Gregory N. Larsen, "Techniques for Cyber Attribution", Institute for Defense Analysis, IDA paper P-3792, October 2003
17. Michels, William L., "Managing the Chain of Custody: Minimizing Your Risk and Exposure!", ADR America, LLC, 94th Annual International Supply Chain Conference, May 2009
18. Linger, R.C., H.D. Mills, B.I. Witt, "Structured Programming: Theory and Practice", Addison-Wesley Publishing Company, Inc., 1979
19. O'Neill, Don, "Technical Debt in the Code: Cost to Software Planning", Defense AT&L Magazine, March-April 2013 <http://www.dau.mil/pubscats/ATL%20Docs/Mar_Apr_2013/0%27Neill.pdf>
20. "Restructuring FTE Debt: The Role of Activism in a Firm", Venturecapital, September 8, 2013
21. O'Neill, Don, "Preparing the Ground for Next Generation Software Engineering", IEEE Reliability Society, Annual Technology Report 2008, pp. 148-151, June 2009
22. O'Neill, Don, "Introduction to Global Software Competitiveness", CrossTalk, The Journal of Defense Software Engineering, Online Articles, October 2003 <http://www.crosstalkonline.org/storage/issue-archives/2003/200310/200310-O'Neill.pdf> [Defense AT&L 12] "A Disruptive Game Changer to Achieve DOD Austerity", Defense AT&L Magazine, May-June 2012 <http://www.dau.mil/pubscats/ATL%20Docs/May_Jun_2012/0%27Neill.pdf>
23. Larman, Craig, "Agile & Iterative Development: A Manager's Guide", Pearson Education, Inc., ISBN 0-13-111155-8, 2008, 342 pages
24. O'Neill, Don, "Inside Track to Offshore Outsourcing Using the Trusted Pipe™: What Global Enterprises Look For in Offshore Outsourcing", Making the Business Case for Software Assurance Workshop, Carnegie Mellon CyLab, Pittsburgh, PA, September, 2008, pages 59-75
25. O'Neill, Don, "Team Innovation Management: Research into Practice", International Process Research Conference, Software Engineering Institute, CMU/SEI-2006-SR-001, January 2006, pages 60-71
26. Chenok, Dan, "Six Trends Driving Change in Government", IBM Center for the Business of Government, October 28, 2013 <<http://www.businessofgovernment.org/blog/business-government/looking-ahead-key-challenges-and-opportunities-government>>
27. Charney, Scott, Corporate Vice President, Trustworthy Computing, Microsoft, Keynote Speech at the Second Worldwide Cybersecurity Summit, London, June 2, 2011, "Supply Chain Risk Management: Toward a Global Vision of Transparency and Trust", <http://www.youtube.com/watch?v=SEO_ieLXQ8o>
28. Goertzel, Karen Mercedes, "Integrated Circuit Security Threats and Hardware Assurance Countermeasures", CrossTalk, The Journal of Defense Software Engineering, November/December 2013 <<http://www.crosstalkonline.org/storage/issue-archives/2013/201311/201311-Goertzel.pdf>>
29. O'Neill, Don, "Meeting the Challenge of Assuring Resiliency Under Stress", CrossTalk, The Journal of Defense Software Engineering, September/October 2009 <<http://www.crosstalkonline.org/storage/issue-archives/2009/200909/200909-O'Neill.pdf>>



**CIVILIAN TALENT IS MISSION-CRITICAL.
LET'S GET TO WORK.**

Work for Naval Air Systems Command (NAVAIR) and you'll support our Sailors and Marines by delivering the technologies they need to complete their mission and return home safely. NAVAIR procures, develops, tests and supports Naval aircraft, weapons, and related systems. It's a brain trust comprised of scientists, engineers and business professionals working on the cutting edge of technology.

You don't have to join the military to protect our nation. Become a vital part of NAVAIR, and you'll have a career with endless opportunities. As a civilian employee you'll enjoy more freedom than you thought possible.

Discover more about NAVAIR. Go to www.navair.navy.mil.

Equal Opportunity Employer | U.S. Citizenship Required

**NAVAIR
CIVILIAN**

CHOICE IS YOURS.

Malware, “Weakware,” and the Security of Software Supply Chains

C. Warren Axelrod, Ph.D., Delta Risk

Abstract. Increasing effort is being made to build security into software—but with mixed results. The need for security apparently exceeds the ability and will of software engineers to design secure software architectures, implement secure coding methods, perform functional security testing, and carefully manage the installation of software products on various platforms and in different environments. COTS/GOTS software often harbors numerous vulnerabilities (we will call such software “weakware”) and such software occasionally contains “malware” (malicious software). The main difference between weakware and malware is that the weaknesses of the former are mostly unintentional or accidental, whereas the damaging characteristics of the latter are planned and intentional and usually require some measure of technical expertise to implement effectively. Nevertheless, from the security perspective the potential consequences are undesirable and damaging irrespective of how the weak or bad code got into the program in the first place. In this article, we examine how and where such damaging code or programs might be introduced throughout the software supply chain lifecycle and how such weakware and malware might be avoided, deterred, eliminated or mitigated.

Introduction

In general, software products are either purchased from software manufacturers or distributors, or they are built in-house and/or by third parties, such as contractors. Homegrown, off-the-shelf and open-source software modules are regularly combined to form overall working systems. Usually customers do not think about software as being produced via supply chains, although it often is since the manufacture of software involves a series of interrelated steps and is made up of components, many of which are acquired from subcontractors or other vendors. Custom software may be built in-house using contractors, farmed out to software consulting firms, which often further subcontract various phases of the software development lifecycle to other parties at home and abroad. System integrators incorporate off-the-shelf programs, particularly operating systems, system utilities, and the like, into working software systems. Software support and maintenance may be handled by software manufacturers, distributors or service companies. Support is now mostly provided over the Internet. Open-source software, developed and supported by communities and specialized service providers, incorporates its own version of supply chains, which is even more widely distributed over community members and geographies.

It is an enormous and complex task for customers to identify various channels through which software has passed and, for the most part, attempts to determine the structure and components of software supply chains have been thwarted by lack of knowledge and cooperation.¹ Yet this phase of the project

is critical if risks are to be identified, assessed, mitigated and managed. When such channels are known, then customers will be confronted with trying to measure security risks for each component—a field where suitable metrics are sorely lacking and cooperation from suppliers is wanting. Even if one is able to assess these risks, one’s ability to control them is hampered by the customer or end-user not having sufficient influence and control over supply chains that have been identified.

In this paper, we present some definitions to help us understand the various contexts in which software supply chain risks are experienced. We also develop a framework against which to identify and assess the risks. And finally, we point to ways in which the risk management process can be facilitated.

Supply-related Risks

One definition of supply chain risk as it relates to physical products is as follows:

Supply risk is ... the probability of an incident associated with inbound supply from individual supplier [or market] failures, ... in which its outcomes result in the inability of the purchasing firm to meet customer demand or cause threats to customer life and safety [1].

In addition to the above, we must consider specific software-related threats, which are far less tangible with respect to their impact. These include unauthorized and malicious access to intellectual property and sensitive data, and damage or destruction of applications or data. Also, piracy issues arise from phony software [2].

Supply Chain Risk Management generally addresses limiting the risk of disruptions, typically those that delay deliveries of an item to a manufacturer or consumer both for physical products, such as pharmaceuticals, luxury goods and entertainment media, and for software. However, Supply Chain Risk Management is also used to effect the reduction or elimination of counterfeit and/or malicious software during the supply chain lifecycle when insiders and others may have access to the software [3].

Software supply risk relates both to the impact of adverse events and the probability of those events occurring. Potential consequences include those that:

- Prevent the purchaser from meeting customer demand
- Prevent the supplier from providing contracted technical and operational support
- Compromise the supply chain management processes causing disruption and delays in meeting deadlines, diversion of products, theft of products, unauthorized copying and distribution, and the like

Types of Software

The following categories of software are of interest:

- COTS/GOTS software
- Open-source software
- Custom-built software (developed internally, externally or both)
- Hybrid—Combination of custom, open-source and off-the-shelf (OTS) software
- Embedded software—software or firmware built into physical products
- Supply chain management software—a specialized category of software that monitors supply chain processes and reports deviations from expected behavior

Software products are closed or open with respect to access to their source code by customers or other parties. Ordinarily, source code of off-the-shelf software is not available to customers. Users of both open-source and custom-built software usually have access to source code, which allows for code reviews and static testing. Hybrid software should generally be considered to be as weak as its weakest component, which is often thought to be a shrink-wrapped product, although studies have shown that open-source software can have as many and as severe security issues as COTS/GOTS software. Even though embedded software may not be top of mind for manufacturers and distributors of physical products, software-specific risk factors must be considered.

Software that is used to manage supply chains (whether the supply chains are for software, physical goods, or combinations of software and hardware) also needs to be considered since effective supply chain management can mitigate many risk factors normally encountered. However, such supply chain management software can also be a vector for cyber attackers. Few researchers appear to have considered the risk of compromise of supply chain management software, which might, for example, be made to report that everything is in order when it is not, and therefore could represent a significant risk.²

Another security category, occasionally considered in the literature, involves software used by computer chip foundries to produce complex integrated circuits. If hackers can gain access to such software, then they can change the circuit designs to perform nefarious functions with little chance of detection.

Risk Characteristics of Software

Software differs from manufactured products in several important ways, as follows:

- Software can be stolen without having to remove or otherwise change the original copy
- Physical transportation of software is not required since copies can be downloaded electronically
- Valid and lawful versions of software can be modified and still be made to appear valid even though they have been tampered with

For physical products, risks relating to manufacturing and distribution usually predominate. When it comes to software, more emphasis needs to be placed on the early phases of the product development process, such as the design and requirements phases, since manufacturing and distribution represent much smaller parts of the overall software supply chain than they do for physical products [4].

The following software supply chain attributes are at risk [5]:

- Confidentiality (intellectual property and personal and business data)
- Integrity (processes, products and data)
- Availability (flows, products and data)
- Authenticity (products and data)
- Trustworthiness (processes, products and people)

With respect to confidentiality, not only must one consider the potential risk of someone stealing intellectual property and trade secrets, but also one must be aware of the potential consequences of compromise of customer and employee personal data. When it comes to integrity, one can imagine the supply

chain processes themselves being exploited by criminals, as well as the modification of software products and related data.

One must also be able to demonstrate that the mitigation efforts have been effective. The following properties enable one to have greater confidence that the risks have been adequately mitigated: transparency, quality, and accountability.

A report by the DoD Information Assurance Technology Analysis Center suggests that constituents are subjected to various supply chain threats [6]. Table 1 assigns threats to constituents.

An SEI (Software Engineering Institute) report points to similarities and differences between product suppliers and system

Threats	Supply Chain Constituents					
	Products	Supply Chain Processes	Product Flows	Supply Chain Data Flows	Management Data	People
Sabotage	X	X		X	X	
Tampering	X			X	X	
Counterfeiting, piracy	X			X	X	X
Theft	X	X		X	X	
Destruction/deletion	X			X	X	
Disruption/delay		X	X	X	X	
Exfiltration—thrift			X	X	X	
Exfiltration—disruption		X		X	X	
Infiltration, subversion		X		X		
Diversion			X	X	X	
Export control violations		X	X			X
Undesirable physical items	X		X			X
Corruption		X		X	X	X
Social engineering		X	X	X	X	X
Insider threat	X	X	X	X	X	X
Pseudo-insider threat	X	X	X	X	X	X
Foreign ownership, influence	X	X	X	X	X	X

Table 1: Assignment of Threats to Supply Chain Constituents

development contractors [7]. The report notes that acquirers' assessments of software takes place after the product development is completed, whereas for custom-built systems, acquirers are able to "actively monitor both contractor and product supply chain risks during the development process."

The report suggests that risk analysis include the following three components:

- Attack analysis, i.e., analysis of threats and exploits leading to successful attacks
- Ability to limit product vulnerabilities by supplier
- Identification of "attack enablers" and business risks by acquirer

Table 2 illustrates how software systems are combined and the characteristics of the combined systems.

The management of risk will vary with the various phases of the supply chain or acquisition lifecycle. The SEI report enumerates those activities as they relate specifically to security risks. Table 3 assigns such activities to the lifecycle phases. A number of activities have been added to the original list in the SEI report.

Forms of Combination	Characterizations
Embedded software	Many software products and systems contain software within the product or system about which acquirers might not be aware. ³
Integrated systems	Software products are inserted into an existing environment and integrators ensure that the new software is compatible with the existing environment and validate that the combined functionality satisfies requirements.
Systems of systems	Disparate systems are combined to form systems of systems, which produce functionality that is greater than the sum of the individual systems.
Cyber-physical systems	Existing and/or new distributed information processing systems and networks and previously isolated industrial control systems are connected so that the control systems can be accessed over public and private networks and data from the control systems can be accessed over public and private networks [8].

Table 2: Combinations of Systems and Their Characterizations

SDLC Phase	Risk Management Activities
Requirements and design	• Perform a risk assessment. • Establish security requirements. • Develop auditing plans.
Manufacture (development)	• Monitor processes and product flows. • Inspect, test, verify and validate final products.
Distribution	• Monitor processes and product flows.
Warehousing	• Monitor processes and product flows. • Check that the product has not been removed, substituted or added.
Deployment	• Monitor processes and product flows. • Check that delivered products and systems are correct and authentic. • Provide user guidance to ensure that products and systems are not adulterated or otherwise compromised.
Operation	• Monitor operation for unusual behavior and damaging events. • Review operational readiness on a continuing basis. • Develop and implement a plan for responding to security incidents.
Maintenance and support	• Monitor suppliers of products and components for any adverse reports relating to the viability of supplier companies or any security or safety issues with products. • Develop contingency plans for potential disruptions in supply of parts or patches, for example, and support.
Disposal	• Monitor disposal of intellectual property and sensitive data, such as personal information and health data, and destruction of media containing such information.

Table 3: Supply Chain Security Risk Management by SDLC Phase

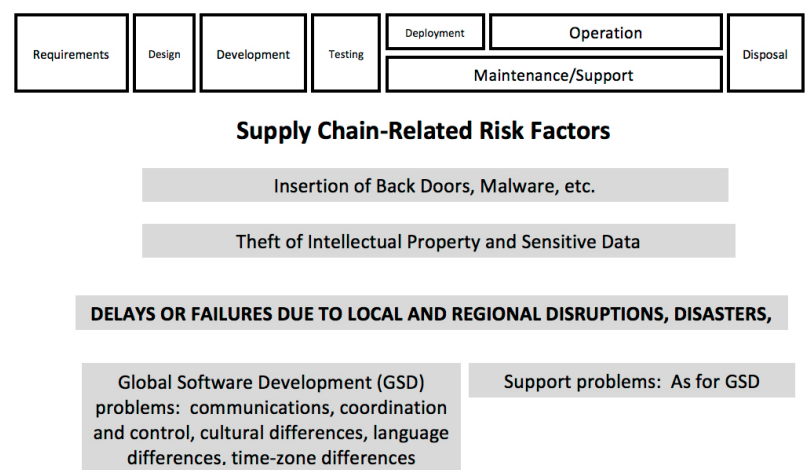


Figure 1: Supply Chain Risks by Phases of the SDLC

It is particularly difficult to get a full picture of all the complexities and nuances of global supply chains as they consist of so many constituents and components. Perhaps the only effective means of doing so is to build computer models of the processes, flows and controls, and use them in exercises to better understand the interaction of the components and the impact of various attacks and events [9].

Software Supply Chain Risks

A major differentiator, with respect to those risk factors related to software design and development, is the location of those efforts and the culture of those doing the work. Location can be defined in terms of whether the design and development is done internally or is outsourced, as well as by geographic location.

The level of risk varies greatly with factors such as loyalties and motivations of employees and contractors; legal, social and economic differences across countries and ethnic groups; and so on.

Figure 1, which is based on [10], illustrates factors affecting supply chain risk throughout the development lifecycle. As can be seen from the diagram, there are events, such as natural and man-made disasters, that can affect all supply chains including software supply chains. However, there are also a number of compromises, such as the insertion of malware, that are unique to software. Other incidents, such as the theft of intellectual property and personal data, are common across many products, but are facilitated for software by the ability to copy software and data without changing the original or having to be onsite to do the copying.

Simulation Models

There have been several research efforts relating to resolving issues relating to global software development and risk relating to software supply chains. Simulation is needed to optimize various characteristics of dispersed software development efforts due to the complexity and dynamic nature of such arrangements.

Researchers have developed models for the impact of cultural, communications and other factors on the distributed development of all types of software. For example, the U.S. financial services sector has worked on supply chain issues and surveyed industry members with respect to various aspects of supply chain risk mitigation. However, there does not appear to be much in the way of modeling the impact of adverse natural and human-invoked events on software supply chains. The need for such models is evident, but the effort to develop such models is substantial.

Software Development and Distribution

To make decisions with respect to any particular supply chain it is necessary to understand each phase as well as the interaction between phases. Figure 2 shows the lifecycles for three major aspects of software development, testing and deployment; namely, manufacturing, oversight, and assurance. Manufacturing is the “nuts and bolts” of developing and distributing software. Oversight consists of the independent oversight of the processes and products of the manufacturing lifecycle. Assurance includes separate evaluations of the quality, integrity and trustworthiness of the software being manufactured.

The shaded boxes in Figure 2 represent functions that are frequently given inadequate attention in the SDLC. Among these important areas are functional security testing, which is testing performed to ensure that the software does not do that which it is not supposed to do [11], and activities relating to the disposal of the software and any sensitive information that it might contain. Whereas verification and validation phases are common components in the development lifecycles followed by the DoD and other government agencies, they are often not fully developed in the private sector.

For the sake of comparison, one can consider highlights of the activities for the various processes and phases of the DoD's Integrated Defense Acquisition, Technology and Logistics Lifecycle Management System as presented by the Defense Acquisition University [12]. For the most part, many of the phases of the DoD model are similar to those shown in figure 2 and some of the processes are the same, although the scope of the DoD activities includes other important procedural areas, such as planning, contracting and financial management. The DoD model provides a more complete framework for complex processes, procedures and reporting.

The risks relating to the software supply chain largely depend on the nature and origin of the software. Table 4 shows the levels of risk that might be expected with respect to software from different sources and whether or not technical support is available.

Conclusions and Recommendations

The initial challenges addressed in this article are to identify and understand software-specific supply chain threats and vulnerabilities and protect against them. Risk was considered at each stage of the software development and software supply chain life cycles and activities suggested to mitigate the risk factors. It is also suggested that the only way to fully understand complex supply chains is to develop computer simulation models that represent those supply chains at the transaction level—i.e., from the process and product-flow perspectives.

Much has already been accomplished in various industries to gain a better understanding of software supply chains and their inherent risk. However, much remains to be done in the public and private sectors in order to achieve an acceptable level of understanding of related risks and their mitigation. ♦

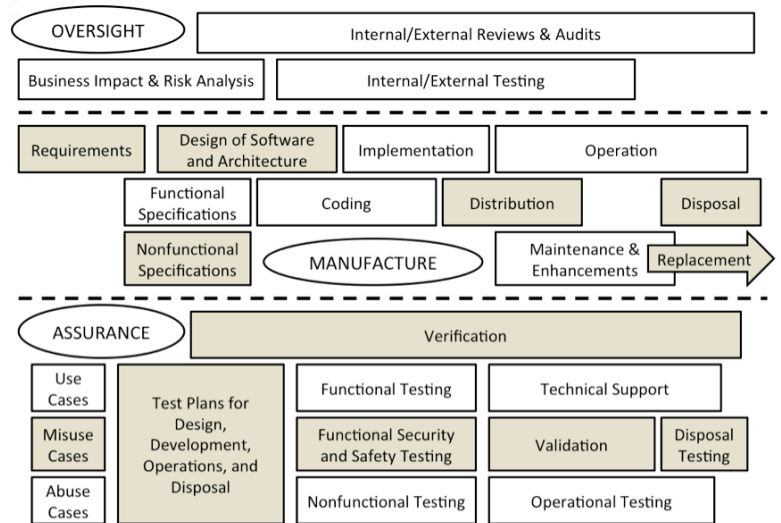


Figure 2: Oversight, Manufacture and Assurance in the SDLC

Source: C. Warren Axelrod; *Engineering Safe and Secure Software Systems*, Artech House, 2013. Reprinted with permission.

Risks	Sources			
	COTS/GOTS	Custom	Open Source	Unsupported
Risk that malware has been introduced during the development phase	Moderate	Low	Moderate	High
Risk that malware will be introduced during operation	Moderate to high	Low to moderate	Low	High
Risk that improper disposal will lead to compromise	Moderate to high	Moderate to high	Low	Low to moderate

Table 4: Supply Chain Risk by Origin of Software

ABOUT THE AUTHOR



C. Warren Axelrod, Ph.D., is a senior consultant with Delta Risk, a consultancy specializing in cyber defense, resiliency, and risk management. Previously, Axelrod was the chief privacy officer and business information security officer for US Trust.

He was a co-founder of the FS-ISAC (Financial Services Information Sharing and Analysis Center). He represented the financial services sector at the national command center over the Y2K weekend

and testified before Congress about cyber security in 2001. He has participated in a number of initiatives at sector and national levels.

Dr. Axelrod was honored with the prestigious ISE (Information Security Executive) Luminary Leadership Award in 2007 and, in 2003, he received the Computerworld Premier 100 IT Leaders Award and Best in Class Award. His article "Accounting for Value and Uncertainty in Security Metrics" won ISACA's Michael P. Cangemi Best Book/Best Article Award in 2009.

Dr. Axelrod has published five books on various IT risk, outsourcing, cyber se-

curity, privacy and safety topics. His most recent book is *Engineering Safe and Secure Software Systems*, released in 2012 by Artech House. He has published three prior articles in CrossTalk magazine.

He holds a Ph.D. (managerial economics) from Cornell University and MA (economics and statistics) and B.Sc. (electrical engineering) honors degrees from the University of Glasgow. He is certified as a CISSP and CISM.

Phone 917-670-1720

E-mail: waxelrod@delta-risk.net

REFERENCES

1. Zsidisin, George A.; "A Grounded Definition of Supply Risk," *Journal of Purchasing and Supply Management*, vol. 9, no. 5-6, September/November 2003, p. 217-224
2. Goertzel, Karen Mercedes; "Protecting Software Intellectual Property Against Counterfeiting and Piracy," *STSC CrossTalk: The Journal of Defense Software Engineering*, vol. 24, no. 5, September/October 2011, p. 6-9
3. Ellison, Robert J., et al; Evaluating and Mitigating Software Supply Chain Security Risks, Technical Note CMU/SEI-2010-TN-016, Software Engineering Institute, May 2010, <www.sei.cmu.edu/library/abstracts/reports/10tn016.cfm>
4. Dehmen, Josef; Mohamed Ben-Dayia; Omera Khan; "Integrating Supply VChain Risks in Product Development: A Conceptual Framework," in Samir Dani (ed.) *Proceedings of the Tenth International Research Seminar on Supply Chain Risk Management, ISCRiM*, September 2010, p. 56-61, <www.husdal.com/wp-content/uploads/2010/09/Proceedings-ISCRiM-2010.pdf>
5. Goertzel, Karen Mercedes; "Supply Chain Risk Management and the Software Supply Chain," *OWASP AppSec DC*, 2010, <https://www.owasp.org/images/7/77/BoozAllen-AppSecDC2010-sw_scrm.pdf>
6. Goertzel, Karen Mercedes, et al; State of the Art Report on Supply Chain Risk Management for the Off-the-Shelf (OTS) Information and Communications Technology (ICT) Supply Chain, Department of Defense Information Assurance Technology Analysis Center (IATAC), USA, 2010.
7. Ellison, Robert J., et al; Software Supply Chain Risk Management: From Products to Systems of Systems, Technical Note: CMU/SEI-2010-TN-026, Software Engineering Institute, December 2010, <www.sei.cmu.edu/library/abstracts/reports/10tn026.cfm>
8. Axelrod, C. Warren; "Mitigating the Risks of Cyber-Physical Systems," *Proceedings of the 2013 IEEE LISAT (Long Island Systems, Applications and Technology) Conference*, Farmingdale, NY, May 2013.
9. Axelrod, C. Warren; "Risks of Unrecognized Commonalities in the Information Technology Supply Chain," *Proceedings of the 2010 IEEE International Conference – Technologies for Homeland Security*, Waltham, MA, November 2010.
10. Raffo, David M.; Siri-on Setamanit; "A Simulation Model for Global Software Development Project," *The International Workshop on Software Process Simulation and Modeling*, USA, 2005, <www.sba.pdx.edu/faculty/davidr/draccess/WEB/publications/JOURNAL/ProSim'05-GSD.pdf>
11. Axelrod, C. Warren; "The Need for Functional Security Testing," *STSC CrossTalk: The Journal of Defense Software Engineering*, March/April 2011, p. 17-21.
12. Defense Acquisition University, Highlights of the Integrated Defense Acquisition, Technology, and Logistics Life Cycle Management System, Version 5.4, 2010, <www.wired.com/images_blogs/dangerroom/2010/09/atl_wall_chart.jpg>

NOTES

1. In 2009, the U.S. Banking and Finance Sector, under the auspices of the FSSCC (Financial Services Sector Coordinating Council), initiated a project to determine IT (information technology) products used by the industry and evaluate relevant threats and risks. The report from Phase 1 of the effort, dubbed "Protecting the Resiliency of the Supply Chain," comprised the results of surveys about leading security practices as they related to purchased software, internally-developed software, and custom software developed by third parties, as well as computer and network hardware, firmware and appliances. Phase 2 was an attempt to identify the full range of IT resources used by banks and securities firms. Unfortunately, it proved difficult to gather even rudimentary information, much less specific data that would allow for a full analysis.
2. One of the characteristics of the infamous Stuxnet worm, which caused centrifuges in Iranian nuclear materials processing plants to self-destruct, was that it reported to operators that all was well even while bad things were happening.
3. One example of an embedded product about which the acquirer might not have been aware is SQL Server. When the SQL Slammer worm hit in January 2003, it surprised IT management by affecting applications that had silently loaded SQL Server. For a description of the worm and a list of affected applications, see F-Secure Corp., <www.f-secure.com/v-descs/mssqlm.shtml>

Problems and Mitigation Strategies for Developing and Validating Statistical Cyber Defenses

Michael Atighetchi, Raytheon BBN Technologies
 Michael Jay Mayhew, Air Force Research Laboratory
 Rachel Greenstadt, Drexel University
 Aaron Adler, Raytheon BBN Technologies

Abstract. The development and validation of advanced cyber security technology frequently relies on data capturing normal and suspicious activities at various system layers. However, getting access to meaningful data continues to be a major hurdle for innovation in statistical cyber defense research. This paper describes the data challenges encountered during development of the machine learning approach called Behavior-Based Access Control (BBAC), together with mitigation strategies that were instrumental in allowing R&D to proceed. The paper also discusses results from applying a spiral-based agile development process focused on continuous experimental validation of the resulting prototype capabilities.

1. Introduction

Enterprise business processes are more connected than ever before, driven by the ability to share the right information with the right partners at the right time. While this interconnectedness and situational awareness is crucial to success, it also opens the possibility for misuse of the same capabilities by sophisticated adversaries to spread attacks and exfiltrate or corrupt critical sensitive information. This is particularly true for an insider threat scenario in which adversaries have legitimate access to some resources and unauthorized access to other resources that is not directly controlled by a fine-grained policy.

BBAC augments existing authorization frameworks, such as Firewalls, HTTP proxies, and application-level Attribute Based Access Control [1] to provide a layered defense in depth. The specific focus of BBAC is to analyze behaviors of actors and assess trustworthiness of information through machine learning. BBAC uses statistical anomaly detection techniques to make predictions about the intent of creating new TCP connections, issuing HTTP requests, sending emails, or making changes to documents. By focusing on behaviors that are nominally allowed by static access control policies but might look suspicious upon closer investigation, BBAC aims to detect targeted attacks that are currently going unnoticed for an extended amount of time, usually months before defenders are aware of cyber attacks.

Figure 1 shows a high-level diagram of the processing flow in BBAC together with the various data sets involved. As shown on the bottom, BBAC needs to ingest a large variety of data from real time feeds through a feature extraction process. During online use, this data will be used for classification purposes. However, for training purposes, BBAC needs to persist and manage training data sets. After parsing the raw observables, BBAC proceeds to go into a feature enrichment phase, where aggregate statistics are computed and information from multiple

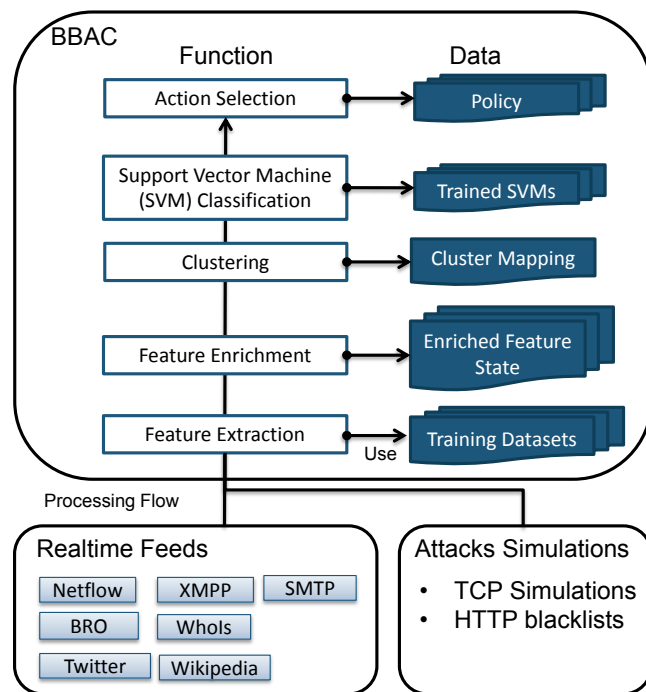


Figure 1. BBAC is a data-intensive system that turns real-time feeds into actionable information through a combination of unsupervised and supervised machine learning (clustering and SVMs).

feeds is merged into a consistent representation. At this stage, BBAC needs to manage intermediate state required for more complex enrichment functions, e.g., calculating periodicity of events. For unsupervised learning, BBAC uses KMeans++ [2] clustering to group actors into clusters, which leads to a mapping function between actors and clusters. Within a cluster, BBAC performs supervised learning through a Support Vector Machine (SVM) [3] that is dedicated per cluster. Finally, action selection determines how to react to classification results obtained from the SVMs. Specific choices such as blocking requests, notifying administrators, or accepting requests are controlled via threshold policies.

2. Cyber Security Data Sets: Problems and Mitigation Strategies

As shown in Figure 1, BBAC is a data-intensive system with successful execution hinging on (a) access to a large amount of external data and (b) efficient management of internal data. Specifically, meaningful data sets are needed to develop and validate the accuracy, precision, and latency overhead of the BBAC algorithms and prototypes.

During development of BBAC, the following problems associated with cyber security data occurred and a number of mitigating strategies were devised. While there is no proven claim about coverage or even success associated with the strategies at this point, these strategies enabled BBAC research to proceed, both in terms of development and continuous validation.

Granularity mismatch. BBAC's analysis techniques work best with data that has a rich context and feature space. What is needed is a large amount of granular data to do statistical inference. Existing repositories, e.g., PREDICT [4], and deployed Host Based Intrusion Detection Systems (HIDSs), frequently

have two different problems. First, in the case of PREDICT, the repository has examples of labeled attack instances, but they are very narrow in scope (packet captures), leading to few features, and only representative of a small number of specific attacks. Second, in the case of HIDSs, access is limited to data that has been preprocessed by correlator nodes. Getting access to more granular information, e.g., involving access patterns of processes on end-systems, generally means installing software on end-systems or even recompiling applications (to map memory regions etc.), both of which raise practical concerns.

To address granularity issues, BBAC focuses its analysis on data that is easily observable without new software or modifying end systems. BBAC uses the following data sets:

- Bro packet sniffer logs both for extracting features about TCP connections and HTTP requests.
- Netflow data for TCP connections
- E-mail data from SMTP logs
- Chat data from XMPP logs
- Microtext data (from Twitter message archives)
- Page edit sequences (from Wikipedia page archives)
- Domain age and country IP information, as determined by information from Whois databases.

Table 1 lists the specific data sets used during the development and validation of BBAC.

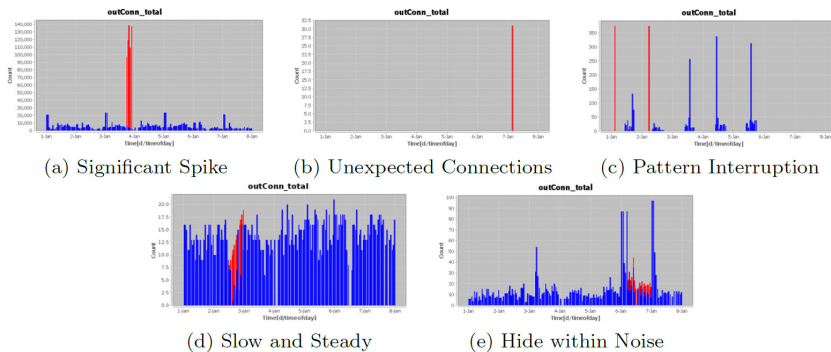


Figure 2. Simulated attack behaviors visible at the TCP layer. Blue lines indicate normal behavior and red lines indicate simulated attack behavior.

Lack of ground truth. To evaluate behaviors at the TCP level, the BBAC developers obtained a large data set of Bro [7] network traces from the BBN network. One immediate problem is that very little can be linked to actual confirmed attacks, leading to the situation in which a large part of the traffic needs to get labeled as unsuspicious.

To address the resulting lack of ground truth concerns, the BBAC developers simulated a number of different attack variants and observables based on how attackers are expected to exfiltrate data or spread attacks at the TCP level. This led to development of the following randomized attacks:

- **Category 1:** Significant Spike. Significant consistent increase in outbound connections.
- **Category 2:** Unexpected Connections. These attacks show outbound activity where there never was any, e.g., a server that has never made outbound requests suddenly making outbound requests.

- **Category 3:** Pattern Interruption. Many hosts follow a regular pattern (e.g., servers fetching updates at regular intervals). The attacks cause interruptions in those patterns.

- **Category 4:** Slow and Steady. Slight increase over normal values, should still be detectable, though with lower accuracy.

- **Category 5:** Hide Within Noise. A highly sophisticated adversary might craft attacks that stay below trained thresholds. In the most extreme case, these attacks form a control case, as BBAC should not be able to detect them. However, it is likely that even the most skilled adversary will trigger Category 1-4 behaviors in a significant part of the hundreds of features observed by BBAC, raising detection accuracy to be significantly higher than random.

Type	Data Set
Network Flows	Connection summary data from Bro and Netflow data captured on the BBN network over the period of 1 month, plus simulated attacks
WHOIS	Domain name record information gathering by BBN for a set of 27839 domains.
HTTP Requests	HTTP request and response headers extracted by Bro on the BBN network collected over the period of 2 weeks, totaling ~17 million requests. In addition, a set of thousands of known bad URLs from PhishTank and URLblacklist. • http://www.phishtank.com • http://urlblacklist.com/?sec=download
Document Changes	Full page history dump of Simple English Wikipedia from 2/27/2012, containing 237,000 pages, 176,000 users, and 3.1 million revisions. • http://dumps.wikimedia.org/simplewiki
Email	Publicly available email archives from Enron that were made public as part of US court proceedings [11] containing ~619k messages belonging to 158 users. • https://www.cs.cmu.edu/~enron/
Chat	Archives of message exchanges directly obtained from Twitter.

Table 1. Data Sets Used During BBAC Development and Evaluation

While the multi-category attack simulations help with attack realism, concerns remain that the BBACs defenses are limited by assumptions made about attacks. Going forward, it would be nice to get better ground truth, e.g., by tying the simulated attacks more directly to actual cyber events, such as the Stratfor Hack [8]. Another part of dealing with this problem is focusing on data that has proper ground truth, such as HTTP requests (for which blacklisted URLs exist) and Wikipedia edits (for which it is known whether the edits were reverted by the Wiki community or not).

Size of available data sets. Wikipedia [9] archives are one of the data sets BBAC uses for assessing document trustworthiness. Wikipedia has rich collaborative semantics attached to it and the archives not only contain the text of wiki pages but also maintain edit sequences over time and provide notions of ground truth, e.g., by including events where edits were reverted or users were banned. So, on first look, Wikipedia seems free of many of the problems described earlier: it is easy to access,

has good granularity, and observable ground truth. Aside from it only covering a very specific trust model and interaction style, the main problem is actually parsing and processing the vast amount of Wikipedia data. In addition, some important features such as the text that was changed between two edits, become computationally intractable quickly, as Wikipedia stores the full text across revisions, requiring extensive use of greatest common substring algorithms to find changes.

To avoid processing nightmares associated with standard Wikipedia, early development on BBAC switched to working with Simple English Wikipedia [10] instead. Simple English Wikipedia is a wiki comprised of a subset of pages from English Wikipedia written using simple vocabulary and grammar meant for those learning the English language. While maintaining content and structure that is similar to English Wikipedia, Simple English Wikipedia archives are significantly smaller in size, thereby increasing efficiency of early investigation of prediction algorithms.

Independence of data sets. Since BBAC performs analysis at multiple different system layers, it not only needs access to data from sensors at these layers but the data in each layer needs to be linked to the other layers to represent a consistent picture of observables.

To address the problem of independence between data sets, BBAC uses an approach for injecting malicious URLs into request streams of benign hosts. Known bad HTTP requests are retrieved from blacklists, and intelligently inserted into existing connections patterns. It is important to keep the ratio of normal vs. abnormal traffic roughly equal allowing the resulting classifier to make decisions both on known proper behavior as well as known improper behavior.

Furthermore, BBAC has a twofold mitigation strategy for establishing correlations across data sets: First, later versions of Bro already link data from TCP connections with data about HTTP requests through the use of session IDs. BBAC will reuse the session ID for establishing cross correlations in this case. Second, for cases in which multiple data streams are collected independently and no session ID exists, BBAC will create a session ID of its own by correlating information based on information about source and destination IP addresses and ports, together with size counts and timestamps.

Model overfitting. BBAC explores the space of machine learning parameters used by SVMs to find a combination of parameters that work best for a given data set. While this maximizes accuracy for the particular data set at hand, it also biases the model so that it might perform significantly worse on future data sets.

To prevent bias through model overfitting, it is advisable to keep a set of continuously updated data that has never been used for model training and parameter space exploration. These data sets can then be used to test the accuracy of the trained models while avoiding training bias.

3. Developing and Validating BBAC

To control the risks of developing new technology without a clear path to data access, the development approach to BBAC is based on the following key tenants that together aim to produce an innovative and applied prototype capability.

Spiral-based: The 3 year project was divided into 6 half-year long spirals, with technology demonstrations happening every 3 months. This rapid prototyping approach provided opportunity

for frequent feedback and enabled the project to stay on track through a number of course corrections. Figure 3 shows the evolution of functionality across the current set of spirals, with the following key milestones:

1. Initial capability showing feasibility of using SVMs for the purpose of detecting suspicious TCP connections.
2. First graphical demonstration of analysis of TCP connections and HTTP requests, using features that were extracted ahead of time from real traffic collected at BBN as well as simulated attacks.
3. Enhanced demonstration, shown at the NSA Information Assurance Symposium 2012, including analysis of Wikipedia page edits as well as inclusion of Whois records to boost analysis accuracy.
4. Inclusion of KMeans clustering (non-supervised learning) to complement SVMs (supervised learning) to boost accuracy.
5. First support for mapping un-anticipated new actors to existing clusters through the use of decision trees to implement "intelligent clustering." Expansion of data sets to include stylometric analysis on email traffic. First version of using a cloud-based compute platform for classification of observables.
6. Expansion of observables to include chat messages and first scalability results associated with training.
7. Inclusion of analysis of blogs and first integrated demonstration showing feature extraction, training, and classification all being performed using the cloud framework.

Looking through the capabilities developed in those seven milestones, a number of points stand out that speak for the utility of the spiral-based approach. First, the BBAC team was able to give demonstrations on limited data sets very early on during the development (steps 1-3), which fostered discussions with a number of stakeholders, including transition partners and other researchers. Next, the focus in steps 4-5 switched to tackling more sophisticated use cases, such as handling of never before seen actors and dealing with environments where little training data is available. Finally, the operational realism of the prototype increased in steps 5-7 by including additional types of data and porting the implementation over to a cloud platform to address scalability requirements of expected deployments.

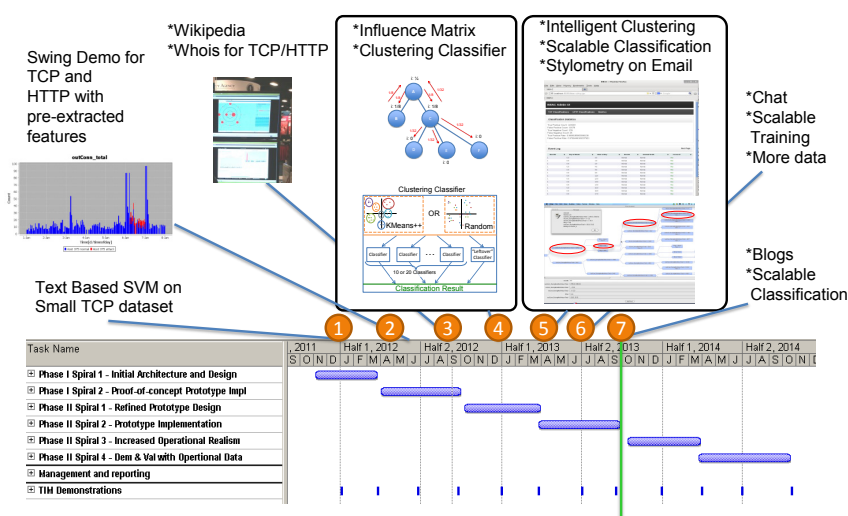


Figure 3. Spiral-based Agile Development Model for BBAC

Metric	Phase 1 Target	Phase 2 Target	Phase 1 Apr '12	Phase 1 Jun '12	Phase 1 Sep '12	Phase 2 Dec '12	Phase 2 Mar '13	Phase 2 Jun '13	Phase 2 Oct '13
Accuracy Characterization	TP > 70%	TP > 80%	76%	85%	91%	96%	96%	96%	96%
	FP < 1%	FP < 1%				0.45%	0.45%	0.45%	0.45%
Accuracy Attribution	TP > 70%	TP > 80%		89%	71%	76%	73%	73%	88%
	FP < 1%	FP < 1%							0.53%
Precision	> 80%	> 90%	76%	93%	97%	97%	98%	98%	98%
Latency	< 1s	< 100ms	623 ms	0.7 ms	0.7 ms	0.7 ms	0.7 ms	0.7 ms	0.7 ms
Overhead	< 200%	< 100%			70%	70%	70%	70%	70%
Scalability	~constant	~constant							
Security OWASP	> 2 nodes	> 10 nodes						40 nodes	40 nodes
Security 800-53	100%	100%		100%	100%	100%	100%	100%	100%
Flexibility	> 25%	> 60%		68%	68%	68%	68%	68%	68%
TRL	2	5	1 (TCP)	2 (+HTTP)	3 (+Wiki)	3	4 (+Email)	5 (+Chat)	5
	2	3		2	3		3	3	3

Table 2. Results from Continuous Assessment

Metric	Measured	Phase II Target	Status	Details
Accuracy	Correctness of characterization (CC) TP = % of attacks correctly identified FP = % of normal traffic incorrectly labeled as attack	TP ≥ 80% FP ≤ 1%	TP = 96% FP = 0.45%	TCP : [92.9%, 0.5%]* *reported as [TP, FP] HTTP : [99.6%, 0.9%]
	Correctness of attribution (CA)	TP ≥ 80% FP ≤ 1%	TP = 88% FP = 0.53%	Wiki: [76%, 1%] Twitter: [96%, 0.18%] Email: [93%, 0.4%]
Precision	Positive Predictive Value (PPV) for CC PPV = # TP / (# TP + # FP)	≥ 90%	97%	TCP: 94.2%, HTTP: 99.1%
	PPV for CA	≥ 75%	99.6%	Wikipedia Edits: 99%, Email: 99.99%, Twitter: 99.8%
Timeliness	Classification latency	< 100 ms	0.7 ms	TCP: m=0.74ms, s=0.52ms HTTP: m=19ms, s=5.4ms Wiki: m=1.3μs, s=30.2μs
	Overhead on inline access control decisions latencies	≤ 100%	70%	TCP(1ms): 70% HTTP(100ms): 19% Wiki(100ms): < 0.01%
Scalability	Training data set size / Number p of parallel processors Training data set size / Training time	~constant p > 10 ~constant	~constant, p = 40 ~constant	
Security	Coverage over OWASP Top 10 threat categories	100%	100%	Assessed current system at 100%
	Adherence to applicable NIST-800-53 security controls (over 150)	30-60%	100%	Assessed current system at 100% 68 controls directly impacted by BBAC
Flexibility	Number of supported document and service types	≥ 5	5	TCP, HTTP, Wikipedia, Email, Chat
TRL	Technology Readiness Level (TRL)	3-4	3	Running in BBN cluster and delivered on VM

Table 3. Project Metrics

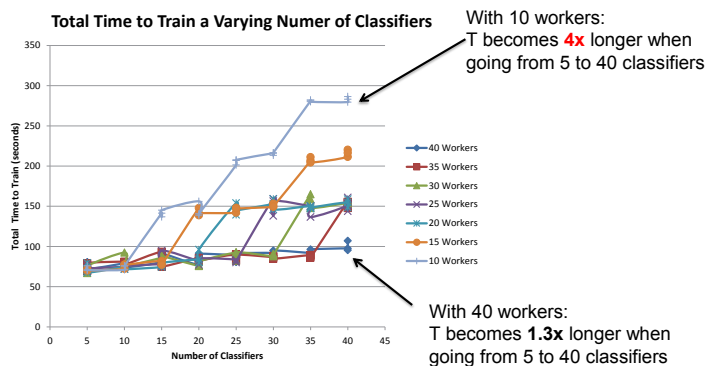


Figure 4. Scalability graph showing that the total training time for SVM classifiers (y axis) can be controlled by adding more worker nodes as the number of classifiers increases (x axis).

Metrics-driven: Progress is measured by tracking seven distinct quantitative metrics, as displayed in Table 3. Accuracy and Precision are determined via 10-fold cross validation. During n-fold cross validation, a data set is divided into n equal parts for n separate iterations. In each iteration, the model is trained on n-1 parts and tested on the remaining part. This guarantees that the data that is used for model construction is never used for model testing. Timeliness is determined by measuring classification and enforcement latencies. Scalability measures the ability of BBAC to just use more hardware to accommodate an increase in load. Security is measured by determining coverage over multiple threat and IA control models, while Flexibility measures the number of data sources BBAC can ingest. Finally, the Technology Readiness Level measures the maturity of the resulting software system.

Continuously Assessed: The set of metrics is assessed frequently throughout the development cycle, usually at the end of every spiral. Table 2 shows the progression of metric compliance across the project execution, with a number of interesting trends.

Attribution accuracy initially met the goal, but then stayed below the target threshold. This can be explained by (1) increasing the amount of Wikipedia data processed and (2) including stylometric analysis of email and chat as part of this metric.

Scalability was ignored during initial development, allowing development to focus on accuracy and precision metrics, but then addressed with refactoring of the functionality to a cloud framework. Figure 4 shows how BBAC's processing scales with increased load by distributing processing over a dynamically assignable set of processing nodes. The figure shows the total time to train a varying number of classifiers. Each colored line represents a different configuration with the number of worker nodes ranging between 10 and 40. Looking at the curve with 10 workers, training time of 10 classifiers takes about 70 seconds, while training of 40 classifiers takes about 280 seconds (~ 4 times as long). By adding more worker nodes, training time of 40 classifiers can be brought down to 100 seconds through parallelized processing across cluster nodes.

Conclusion

Development and validation of statistical cyber defenses needs a well-labeled, appropriately sized, and readily available amount of relevant data to make innovative progress, yet too little of such data sets are available today. This article describes an initial set of data challenges and solutions from the work on BBAC, and describes how agile project management techniques helped deliver innovative technology in a difficult to work in data-intensive environment. Going forward, we intend to provide a list of requirements that will help data providers and clearing-houses create and maintain data sets that are more effective in driving R&D development of the next generation of cyber security technologies.

Disclaimers:

This work was sponsored by the Air Force Research Laboratory (AFRL). Distribution A. Approved for public release; distribution unlimited (Case Number 88ABW-2013-4318).

Acknowledgments:

We acknowledge the support of various team members that helped address data problems on the BBAC effort, including John Benner of Booz Allen Hamilton; Andrew McDonald and Jeffrey Segall of Drexel University; and Jeffrey Cleveland of Raytheon BBN Technologies. ♦

REFERENCES

1. NIST SP 800-162: Guide to Attribute Based Access Control (ABAC) Definition and Considerations (Draft), http://csrc.nist.gov/publications/drafts/800-162/sp800_162_draft.pdf
2. Arthur, David, and Sergei Vassilvitskii. "k-means++: The advantages of careful seeding." Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, 2007.
3. Wang, Lipo, ed. Support Vector Machines: theory and applications. Vol. 177. Springer, 2005.
4. Protected Repository for the Defense of Infrastructure Against Cyber Threats (PREDICT), <https://www.predict.org/>
5. B.Claire, "Cisco Systems NetFlow Services Export Version 9", October 2004, RFC 3954, <http://tools.ietf.org/html/rfc3954>
6. L. Daigle, "WHOIS Protocol Specification", September 2004, RFC 3912, <http://tools.ietf.org/html/rfc3912>
7. The BRO Network Security Monitor, <http://bro-ids.org/>
8. Strafor Hack, <http://www.stratfor.com/weekly/hack-stratfor>
9. Wikipedia - The free encyclopedia, <http://www.wikipedia.org/>
10. Simple English Wikipedia, http://simple.wikipedia.org/wiki/Main_Page
11. Klimt, Bryan, and Yiming Yang. "Introducing the Enron Corpus." CEAS. 2004.

ABOUT THE AUTHORS



Michael Jay Mayhew, a Senior Computer Scientist, has been with AFRL Information Directorate for the past 17 years, 10 of those years as a Federal Civilian. As the Program Manager of the Cross-Domain Innovation & Science (CDIS) group, Mr. Mayhew leads a team of research engineers in finding and developing new cross-domain technologies and maturing those technologies for integration within existing cross-domain products. Mr. Mayhew is a frequent presenter at worldwide program and technical conferences each year and is recognized as a subject matter expert in the area of cross-domain technology.

Air Force Research Laboratory
525 Brooks Road
Rome, NY 13441
Phone: 315-330-2898
E-mail: michael.mayhew@rl.af.mil



Dr. Rachel Greenstadt is an Assistant Professor of Computer Science at Drexel University. Dr. Greenstadt's research centers on the privacy and security properties of multi-agent systems and the economics of electronic privacy and information security. Her lab -- the Privacy, Security, and Automation Laboratory (PSAL) -- focuses on designing more trustworthy intelligent systems that act autonomously and with integrity, so that they can be trusted with important data and decisions. The lab takes a highly interdisciplinary approach to this research, incorporating ideas from artificial intelligence, psychology, economics, data privacy, and system security. However, a common thread of this work has been studying information flow, trustworthiness, and control. Recently, much of the work has focused on using machine learning to better understand textual communication.

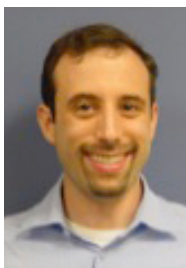
Drexel University
University Crossings 140
Philadelphia, PA 19104
E-mail: greenie@cs.drexel.edu

ABOUT THE AUTHORS



Michael Atighetchi is a Senior Scientist in the Information and Knowledge Technology business unit at BBN Technologies. Since Mr. Atighetchi joined BBN more than 13 years ago, he has contributed to research and technology development in the areas of cross domain information sharing, adaptive QoS middleware, cognitive reasoning in cyber defense, system survivability and security verification through red team testing. He has been a key technical contributor to several DARPA- and USAF-sponsored research projects. Mr. Atighetchi has published over 60 technical papers in peer-reviewed journals and conferences and is Senior Member of the IEEE.

Raytheon BBN Technologies
10 Moulton Street
Cambridge, MA 02138
Phone: 617-873-1679
E-mail: matighet@bbn.com



Dr. Aaron Adler is a Scientist in the Information and Knowledge Technologies business unit at BBN Technologies. Dr. Adler joined BBN in 2009 after finishing his Ph.D. thesis at MIT. He has contributed to research and technology development in several areas including: cyber defense, synthetic biology, advanced design tools, AI learning, and quantum compilers. He is currently a co-PI for a DARPA-funded project and has been a key technical contributor to several other DARPA- and USAF-funded research projects.

Raytheon BBN Technologies
10 Moulton Street
Cambridge, MA 02138
Phone: 617-873-3517
E-mail: aadler@bbn.com

Earned Schedule 10 Years Later

Analyzing Military Programs

Kevin T. Crumrine, Air Force Operations C2 Division
Jonathan D. Ritschel, Ph.D., Air Force Institute of Technology
Edward White, Ph.D., Air Force Institute of Technology

Abstract. It has been 10 years since Walt Lipke first introduced the concept of Earned Schedule (ES). While progress has been made in understanding the utility of ES in some small scale and limited studies, a significant analysis of ES in DoD acquisition programs is missing. This paper first analyzes whether ES and Earned Value Management (EVM) provide fundamentally different information for program managers. It then examines which technique, ES or EVM, provides more timely and accurate schedule predictors in a broad spectrum of military weapon system programs. We find ES to be more timely and accurate both in software intensive contracts and in the sample size as a whole.

Background

EVM has been the premier method of program management and program cost forecasting within the DoD since its inception in the 1960s. However, there are well-documented limitations to EVM particularly with respect to schedule analysis [1]. These limitations include: 1) reporting schedule variance in terms of dollars rather than time 2) the regression of EVM schedule efficiency metrics (SPI(\$)) to 1 as projects near completion, despite variable schedule performance and 3) the regression of EVM schedule variance metrics (SV(\$)) to zero as projects near completion. For practitioners in the field, these issues make traditional EVM schedule analysis unwieldy. To mitigate these limitations, Walt Lipke developed the concept of ES as an alternative to EVM [1]. Lipke's ES construct measures schedule performance with analogous earned value metrics dubbed Schedule Performance Index (SPI(t)) and Schedule Variance (SV(t)) where (t) indicates the metric is reported in time.

But the question remains: Should DoD managers utilize ES as a preferred schedule analysis technique? Program managers should only implement ES analysis as part of their tool kit if it provides additional benefit beyond the established EVM techniques. Thus, the answer to the question becomes an empirical matter. Previous studies (Henderson [2] [3], Lipke [4], Vanhoucke & Vandevoorde [5], Rujiranyong [6], Tzaveas, Katsavounis & Kalfakakou [7], Lipke [8]), have examined the efficacy of ES, but these studies were all limited by their extremely small sample size or lack of relevance to the DoD.

This paper overcomes the previous literature shortcomings by analyzing over 64 contracts in major Air Force aircraft acquisition programs to determine whether ES provides more timely and accurate information. These contracts include software intensive contracts such as avionics along with hardware intensive contracts such as engines, capturing the full spectrum of an aircraft acquisition effort. The large sample size and direct relationship to military programs makes the results of this analysis directly applicable to DoD software and hardware program managers.

Data Source

The data for this analysis is from the Defense Acquisition Management Information Retrieval (DAMIR) system. DAMIR is comprised of all Contractor Performance Report (CPR) data for major DoD acquisition programs. The CPR data contains the monthly and quarterly performance information derived from the contractors EVMS system for all Work Breakdown Structures (WBS) within each contract of a program. Thus, it provides the cost and schedule status for the contract [9].

This analysis focuses on 64 Acquisition Category (ACAT) 1 aircraft contracts at the summary level (WBS 1). The programs comprising the dataset have completed their acquisition phase, and are either in their operational phase, or have been retired from the Air Force fleet. The 64 contracts result in 1,087 data points in the full analysis. We specifically examine the software intensive avionics contracts as a group, in addition to an aggregated analysis of all 64 contracts.

Methodology and Results

Preliminary Analysis

The first question to answer is whether ES and EVM provide fundamentally different information to program managers. Once this is ascertained, the method that provides better information, measured in this paper by timeliness and accuracy, can be determined. We statistically test the difference between ES and EVM through a paired t-test of SPI(\$\$) and SPI(t). A paired t-test measures the mean difference between two sets of numbers. The null hypothesis is that there is no difference between the methods. Table 1 shows the results.

t-Test: Paired Two Sample for Means		
	Variable 1	Variable 2
Mean	0.939165476	0.95750293
Variance	0.008831643	0.006653895
Observations	1087	1087
Pearson Correlation	0.689419981	
Hypothesized Mean Difference	0	
df	1086	
t Stat	-8.623145392	
P(T<=t) one-tail	1.13734E-17	
t Critical one-tail	1.646257934	
P(T<=t) two-tail	2.27467E-17	
t Critical two-tail	1.962150792	

Table 1: Paired t-test SPI(\$\$) vs SPI(t)

As shown in Table 1, the p-value of the t-test is 2.27E-17, well below our significance level of 0.05. Therefore, the null hypothesis is rejected. This means there is a statistically significant likelihood that ES and EVM information are fundamentally different from each other. In practical terms, this indicates that utilizing the ES technique provides additional information to the program manager. The question then becomes whether the ES information is more valuable, as measured by its timeliness and accuracy.

Testing Timeliness

Metrics help managers determine when a problem is occurring so that corrective action may be taken. For this analysis, a problem was defined as a SPI(\$) or SPI(t) < 0.90. The intent of this test is to determine whether EVM or ES is an earlier detector of problems in meeting program schedule objectives.¹

The initial dataset examined is the subset of software intensive avionics contracts. Of these contracts that both ES and EVM identify as a problem, EVM identifies the problem at the 18.87% completion point, while ES identifies the problem at the 16.88% completion point. EVM, therefore, detects about 2% earlier than ES. However, drawing conclusions based on this is misleading. Rather the analysis necessitates that we look at all the avionics contract problems detected, even if only one of ES or EVM detects it. See Figure 1.

Figure 1 shows that ES strictly dominates EVM. ES identifies more problems at every completion point of the contract. More importantly, at the earlier stages of the program, ES detects more problems. For instance, at the 20% completion point, ES detects seven programs with problems while EVM only detects two. This early difference in detection is critical as it allows program managers to take corrective action early in the program. Figure 1 also demonstrates a second area where ES is more valuable than EVM. Note that around the 2/3 program completion point, EVM no longer detects any problems, while ES remains useful in problem detection through the end of program completion.

Next we analyzed the full 64-contract dataset. The total number of SPI(t) and SPI(\$) values below 0.90 were analyzed at each of the following program completion points: 20%, 40%, 50%, 60%, 80%, and 90%. See Table 2.

Avionics Comparison of Numbers of SPI Values Below .90

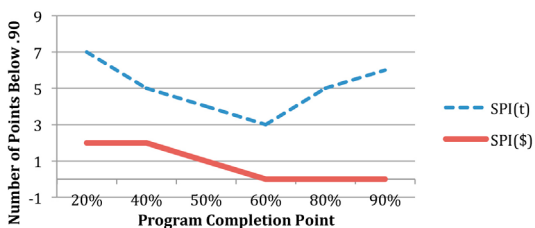


Figure 1: Avionics Comparison of Numbers of SPI Values Below 0.90

	20%	40%	50%	60%	80%	90%
SPI(t)	20	17	11	14	15	20
SPI(\$)	12	11	4	5	2	1

Table 2: Number of SPI Value Below 0.90 Over Time

Table 2 shows quite clearly that as early as the 20% program completion point, the ES metric was indicating a problem more frequently than the EVM metric. Additionally, this gulf in detection exacerbates over the life of the program, consistent

with previous literature: as a contract approaches its completion point, EVM yields an SPI(\$) value that approaches 1.0, indicating that the program is on schedule even if it is not. This is seen at the 90% completion point where SPI(t) correctly found 20 programs to be “in trouble,” while SPI(\$) found only 1.

Testing Accuracy

Two analyses are performed to compare the accuracy of ES and EVM. First, we measure the SPI(\$) and SPI(t) in relation to the final schedule result. Whichever method is closer to the final contract over/under run is deemed to be the more accurate technique. The results for the avionics subset of contracts are shown in Table 3.

	Number of Occurrences	Percentage of Overall Occurrences (%)
Earned Value Management	107	43.67
Earned Schedule	126	51.43
EVM = ES	12	4.90

Table 3: Accuracy of ES and EVM in Avionics Contracts

Table 3 shows that ES is more accurate than EVM in the avionics subset. There is approximately an 8% difference between the techniques for these software intensive contracts. While this finding is significant, the accuracy margin widens to 21% when the full 64-contract dataset is analyzed. Of the 1,087 data points, EVM is closer to the final schedule result 37% of the time, while ES is the more accurate technique 58% of the time. The EVM and ES values are equivalent 5% of the time. Thus, for both the avionics subset and the dataset as a whole, ES trumps EVM in accuracy.

The second analysis, shown in Figure 2, depicts the frequency of contracts having a particular percentage of their data points closer to the final schedule result. For instance, the B1B Offensive Avionics Lot 1 has 15 points where the SPI(t) is closer to the final schedule result than the SPI(\$). There are 20 data points for this program, so ES is closer to the final schedule result 75% of the time. As depicted in Figure 2, this contract is 1 of 9 contracts where the SPI(t) value is closest to the final schedule result between 70% and 75% of the time. There is a definite skew left to this histogram, demonstrating the greater accuracy of ES. In fact, there are only four programs that have less than 30% of their data points with SPI(t) values closer to the final schedule result.

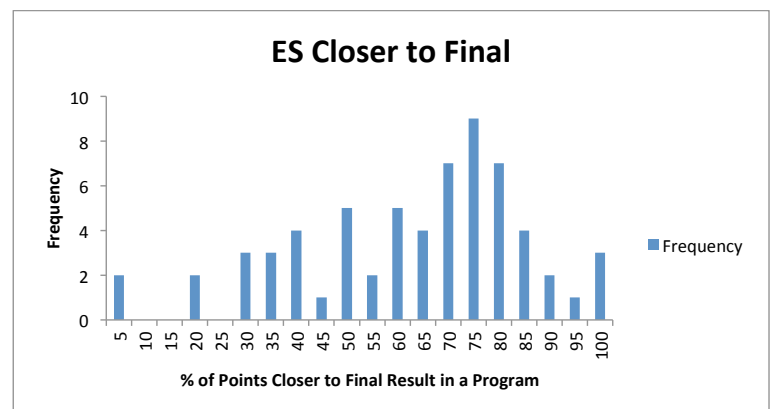


Figure 2: Distribution of Programs With ES Closer to Final Program Delivery

In addition to analyzing the contracts at an individual level, we also want to determine how the entire portfolio acts over a period of time. As shown in Figure 3, the ES metric dominates the EVM metric at all program completion percentage points. This result points to ES providing valuable information to the program manager.

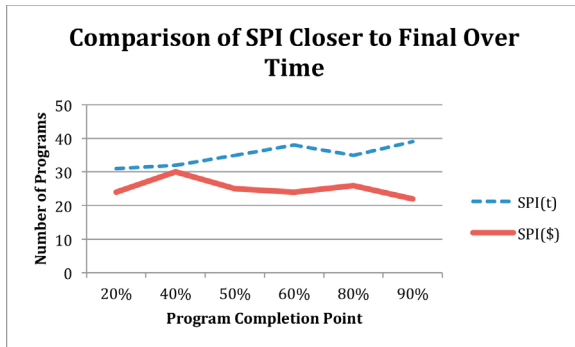


Figure 3: Comparison of SPI Closer to Final Over Time

Other Schedule Techniques: the Critical Path

EVM is not the only technique used by DoD program managers to analyze schedule. The most common methodology is the Critical Path Method. Lipke [4] argues that Earned Schedule is applicable to the critical path. We examine this finding in a small subset of our data. Our results show a fundamental disconnect between the level of Earned Value data collected and the level of Critical Path data utilized by the program offices. Specifically, we find that earned value data is collected at a much higher level than the level in which critical path analysis is being performed, rendering a comparison infeasible. This does not necessarily suggest that ES is inapplicable to the CPM in the DoD. Rather, it points to the necessity of making contractor EVMS reporting at a lower level as part of contract deliverables than is typical today. This, of course, would result in increased contract costs. More research is needed in this area to determine that cost/benefit ratio.

Conclusion

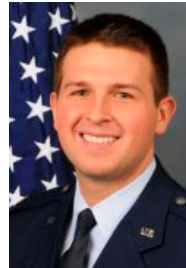
This paper has demonstrated with statistical significance that ES is fundamentally different from EVM. Our empirical analyses of 64 contracts show that not only is there a difference between the two techniques, but that difference is wide enough to warrant a reconsideration of the use of ES in DoD programs. Specifically, we find ES to be both timelier and more accurate than traditional EVM schedule analysis.

The practical implications of our research are straightforward. Due to our inability to thoroughly test ES against CPM, we stop short of recommending ES as its replacement. However, our analysis indicates ES warrants more intensive use for schedule analysis in DoD programs. Specifically, based on the findings of our research, we believe that DoD ACAT I programs should embrace ES as a complementary tool (i.e. the primary cross-check) to the CPM method that is predominately utilized. Traditional EVM schedule analysis techniques should not be abandoned completely, but should be secondary to the CPM and ES techniques.

Disclaimers:

The views expressed in this article are those of the authors and do not necessarily reflect the views of the DoD or any of its agencies. ♦

ABOUT THE AUTHORS



Kevin T. Crumrine is a Captain in the United States Air Force. He graduated from the U.S. Air Force Academy in 2008 with a B.S. in Management, and from the Air Force Institute of Technology in 2013 with a M.S. in Cost Analysis. Kevin currently serves the Deputy Cost Chief for the Operations C2 Division at Hanscom AFB, MA.

E-mail: kevin.crumrine@gmail.com



Jonathan D. Ritschel, Ph.D., is an Assistant Professor and Director Cost Analysis Program in the Department of Systems Engineering and Management at the Air Force Institute of Technology (AFIT). He received his BBA in Accountancy from the University of Notre Dame, his M.S. in Cost Analysis from AFIT, and his Ph.D. in Economics from George Mason University. Lt Col Ritschel's research interests include public choice, acquisition reform effects on cost growth, and economic institutional analysis.

E-mail: jonathan.ritschel@afit.edu



Edward "Tony" White, Ph.D. is an associate professor of Statistics at the Air Force Institute of Technology. His research interests include design of experiments, biostatistics, growth curves, linear and nonlinear regression, categorical data analysis, log-linear models, statistical simulation, and response surface modeling. White received his Ph.D. in statistics from Texas A&M University in 1998.

Phone: 937-255-3636 x4540

E-mail: edward.white@afit.edu

REFERENCES

1. Lipke, Walt. "Schedule is Different," *The Measurable News* 3 (2003): 10-15.
2. Henderson, Kym. "Earned Schedule: A Breakthrough Extension to Earned Value Theory? A Retrospective Analysis of Real Project Data," *The Measurable News* (2003): 13-23.
3. Henderson, Kym. "Earned Schedule in Action," *The Measurable News* 8 (2005): 23-30.
4. Lipke, Walt. "Applying Earned Schedule to the Critical Path and More," *The Measurable News* (2006): 26-30.
5. Vanhoucke, M., S. Vandevorde. "Measuring the Accuracy of Earned Value / Earned Schedule Forecasting Predictors," *The Measurable News*, Winter (2007a): 26-30.
6. Rujiranyong, Thammasak. "A Comparison of Three Completion Date Predicting Methods for Construction Projects," *Journal of Research in Engineering and Technology*, Volume 6, (2009).
7. Tzaveas, Theodoros, Stefanos Katsavounis, and Glikeria Kalfakakou. "Analysis of Project Performance of a Real Case Study and Assessment of Earned Value and Earned Schedule Techniques for the Prediction of Project Completion Date." *Proceedings of IPMA Conference*, May 2010 (Crete, Greece).
8. Lipke, Walt. "Earned Schedule Application to Small Projects," *The Measurable News* 2 (2011): 25-31.
9. Defense Contract Management Agency. Department of Defense Earned Value Management Implementation Guide. October 2006. Alexandria, VA: DCMA, 2006. 10 (2012) <<http://guidebook.dcm.mil/79/EVMIG.doc>>

NOTES

1. Preliminary data analysis demonstrated that there are frequent occurrences where a program's SPI value drops below 0.90 early in a program and quickly recovers. This led to the potential for false conclusions, necessitating a different analysis. Therefore, to be counted as "detecting" a problem in our analysis, the SPI metric must remain below 0.90 for multiple consecutive time periods.

Upcoming Events

Visit <http://www.crosstalkonline.org/events> for an up-to-date list of events.

Spring 2014 Software & Supply Chain Assurance Forum

11-13 March 2014
McLean, VA
<https://buildsecurityin.us-cert.gov/swa>

30th Annual National Test & Evaluation Conference

24-27 March 2014
Seattle, WA
<http://www.ndia.org/meetings/4910/Pages/default.aspx>

The 26th Annual IEEE Software Technology Conference

Long Beach, California, USA
29 March - 3 April 2014
<http://ieee-stc.org>

CGO '14 – 12th Annual IEEE/ACM International Symposium on Code Generation and Optimization

29 Mar-02 Apr 2014
Orlando, FL,
<http://www.cgo.org>

8th Annual IEEE International Systems Conference

March 31-April 3 2014
Ottawa Ontario, Canada
<http://www.ieeesyscon.org/syscon-home>

The 2nd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud '14)

Oxford, UK, 8-11 April 2014
<http://www.mobile-cloud.net>

The Blockbuster Conference on Software Testing and Analysis & Review

May 4-9, 2014
Orlando, FL
<http://stareast.techwell.com>

The CMMI Conference: SEPG North America 2014

6-7 May 2014
Washington, DC
<http://cmminstitute.com/thecmmiconference2014>

2014 IEEE 22nd Annual International Symposium on Field-Programmable Custom Computing Machines

11-13 May 2014
Boston, MA
<http://www.fccm.org>

Better Software Conference West

June 1-6, 2014
Las Vegas, NV
<http://bscwest.techwell.com>

Summer 2014 Software & Supply Chain Assurance (SSCA) Working Group Sessions

24-26 June 2014
McLean, VA
<https://buildsecurityin.us-cert.gov/swa>

American Society of Quality International Conference

February 25-26, 2014
Dallas, TX
<http://asq-icsq.org>

The Active Denial System, Risk Mitigation, and Pain Relief

Let me start out by saying, “Getting old(er) is not for wimps!” Legacy code and legacy coders can both cause trouble.

A few weeks ago, I decided to try and move a table up a flight of stairs—all by myself. Not a good idea.

Two weeks later I was still experiencing a bit of back pain. In fact, I was having trouble sleeping, so I asked my wife if she would mind running to the drugstore and grabbing some more ibuprofen for me. She was happy to make the trip for me (most likely, happy to get away from a whiney husband whose back hurt). She called from the store, and asked if I wanted some “pain relief patches.” I had seen them advertised on TV. Supposedly, they provide “gently, soothing heat for pain relief lasting up to 8 hours.” Sounded good!

The next morning, as I was leaving for work, my wife reminded me about the pain relief patches. I slapped one on and started my 40-minute commute to work.

For the first five to 10 minutes of driving (while I was in range of several gas stations with bathrooms), I felt nothing at all. No “gently, soothing heat.” Soon, however, I cleared the town, and was on the 15-mile stretch through the countryside—no gas stations, no rest areas—and only then did I start to feel the “gently, soothing heat.” And, in a more few minutes, I felt the “direct, REALLY hot heat.” And then, a few seconds later, somebody set off a small, localized nuclear meltdown on my back.

I drive a small car, and was unsuccessfully trying to lean forward, and remove the smoldering patch before it burned both my back and my leather seats. The patch, however, was pretty much permanently fused to my backside (and a permanent part of my anatomy as it first melted and then fused with my skin). Finally, I pulled over, stepped out of the car, and in a rather hurried rush, ripped the patch off. Unfortunately, the sticky material (that obviously contained the magic ingredients that provided soothing gentle relief) stayed tightly attached to the dermal layer of my tuchas (go Google it).

Let me digress a bit here, and point out that later in the day, when I returned home, I examined the box the patches came in. The active ingredient was capsaicin. At this point, let me point out what Wikipedia has to say about this particular chemical, “Capsaicin is the active component of chili peppers, which are plants belonging to the genus *Capsicum*. It is an irritant for mammals, including humans, and produces a sensation of burning in any tissue with which it comes into contact.”

When I was a consultant about seven years ago, the company I worked for helped provide testing on the Active Denial System (ADS). The DoD describes the ADS as, “A non-lethal, directed-energy weapon developed by the U.S. military, designed for area denial, perimeter security and crowd control.”

Informally, the weapon is also called the heat ray since it works by heating the surface of targets, such as the skin of targeted human subjects.

According to Wikipedia the original ADS contract was slightly over \$6 million. A spokesman for the Air Force Research Laboratory was quoted as saying, “For the first millisecond, it just felt like the skin was warming up. Then it got warmer and warmer and you felt like it was on fire.... As soon as you’re away from that beam your skin returns to normal and there is no pain.”

Few subjects are able to stand the ADS for more than a few seconds.

I got the same effects for less than \$5! The DoD could save millions by buying some of those guns they use at ball games to shoot T-shirts up into the stands. Load them with “gentle, soothing heat relief patches” then fire into the crowd, and watch them run! On the other hand, never mind. It’s probably illegal under the Geneva Convention.

Back to me standing by the side of the road: I had pulled over near the Angelina River, a small and muddy river on the way to work. While I eventually was able to rub off most of the sticky residue, only the thought of a headline reading, “Local University Professor Caught Skinny Dipping In River” prevented me from a quick rinse (but I seriously considered it for several LONG moments).

Back at home that night, I saw that in 2-point font on the back of the box, there is a warning that says, “Some users might be sensitive to capsaicin. Effects may vary, and you should test on a small patch of skin prior to applying the entire patch.” No kidding!

Reasonable people (unlike myself) might consider testing a new type of pain relief system prior to use. I might have even asked my friends or students if they had a similar experience. In fact, when I mentioned a brief version of this story to a class of seniors, several of the athletes laughed and said that those patches were like pouring gasoline on the painful area, and then lighting a match!

Do you really read warning labels on non-prescription drugs? Do you really read product warnings? Switching topics, do you thoroughly check software reviews and sources?

Ever download software from an unknown source? Get a virus unexpectedly? Just last week I downloaded an application for my Mac that causes the system to freeze every time I use it. Come to find out, it’s a known problem.

Who makes the decisions to update/download/integrate your system? Are the risks fully evaluated? Do the people making the decisions really have the knowledge to make trustworthy evaluation?

Or is YOUR tuchas going to be burning one day, too?

David A. Cook
Stephen F. Austin State University
cookda@sfasu.edu



Homeland
Security

Software and Supply Chain Assurance

Software and Supply Chain Assurance are essential to enabling the Nation's critical infrastructure.

To ensure the integrity of that infrastructure, the software and the IT supply chain must be secure and resilient.

The Software Assurance Community Resources and Information Clearinghouse provides corroboratively developed resources. Visit <https://buildsecurityin.us-cert.gov/swa/index.html> to learn more about relevant programs and how you can become involved.

Software and Supply Chain Assurance must be “built in” at every stage of development and supported throughout the lifecycle.

Visit <https://buildsecurityin.us-cert.gov/bsi/home.html> to learn about the practices for developing and delivering software to provide the requisite assurance. Sign up to become a free subscriber and receive notices of updates.

The Department of Homeland Security provides the public-private collaboration framework for shifting the paradigm to software and supply chain assurance.



<https://buildsecurityin.us-cert.gov/swa>



NAV AIR



CROSSTALK thanks the above organizations for providing their support.