A Systems Engineering Approach for Assured Cyber Systems

Major Logan O. Mailloux, Dr. Brent T. Langhals, and Dr. Michael R. Grimaila
Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, United States

Systems Engineering (SE) has gained favor as a means to tame the complexity of modern systems, specifically the design, analysis, and development of complex systems. This paper describes a SE approach for system assurance of modern Information Technology (IT) centric "cyber systems". In this paper, we discuss recent trends in information security towards the establishment of security patterns and identify key security patterns for the development of cyber systems. Specifically, this paper provides a cursory review of security patterns and highlights the utilization of key cyber patterns during the SE development process for a given cyber system. SE functional decomposition and system integration activities are described as they pertain to meeting formal system assurance claims resulting in secure and assured cyber systems.

Keywords—Systems Engineering; Security Patterns; Cyber Assurance

I. INTRODUCTION

Systems Engineering (SE) has gained favor as a means to tame the complexity of modern systems, specifically the design, analysis, and development of complex systems. The authors propose a SE approach for system assurance of modern Information Technology (IT) centric "cyber systems" that addresses cyber system complexity and security through established information security practices and fundamental SE processes. Specifically, this paper provides a cursory review of security patterns and highlights the utilization of key cyber patterns during the SE development process for a given cyber system. SE functional decomposition and system integration activities are further described as means to provide formal system assurance justification for cyber systems.

Section II describes the cyber security problem as it pertains to system complexity, while Section III provides a background of essential SE processes and principles used to tame system complexity. Section IV describes findings related to key security patterns, applicability within SE development processes, and justification of system assurance claims.

II. PROBLEM STATEMENT

The problem of IT security is so pervasive that "cyber security" has turned into a multi-billion dollar industry over the past decade. Recent public announcements from the highest levels in the U.S. government are quite telling on the subject. On February 12, 2013 President Obama issued an Executive Order which cited improving critical infrastructure cyber security as one of the most serious national security challenges the country faces [1]. Furthermore, while facing the most ominous Department of Defense (DoD) budget cuts since the cold-war, U.S. Cyber Command announced a colossal expansion plan from 900 to 4,900 personnel [2]. Perhaps even more telling, the private security firm Mandaint® released a comprehensive report *APT1* which details an extensive cyber espionage campaign conducted by the Chinese military against U.S. private firms [3].

Current cyber security approaches are failing in almost every area of interest from common operating system vulnerabilities to meticulously planned attacks against security software vendors. The cyber security problem is especially difficult because IT systems have deeply rooted design flaws, software bugs, weak assumptions, configuration issues, and various other deficiencies which result in vulnerabilities and weaknesses. The challenge is further intensified since these deficiencies can be introduced anytime in the system lifecycle from initial design and development to system fielding, configuration, and day-to-day operation. Modern cyber systems also fail because of unknown software issues, unexpected hardware failures, and unrealized operational and support system dependencies.

Principally, the cyber security problem is due to rising complexity—"the measure of how difficult a *system* is to understand, and thus to analyze, test, and maintain" [4]. Today's cyber systems are so complex that effectively designing and developing secure systems is exceedingly difficult bordering on nearly impossible. The beloved security engineer's Orange Book i.e., the DoD's 1985 Trusted Computing Evaluation Criteria correctly states the dilemma precisely: "the [cyber system] must be of sufficiently simple organization and complexity to be subjected to analysis and tests, the completeness of which can be assured" [5]. The elusive problem of providing secure and assured cyber systems is a significant cause of concern in the U.S. as a whole, and the DoD in particular.

III. BACKGROUND

This section provides a baseline of SE processes and principles to facilitate shared understanding of complex systems. The Defense Acquisition Guidebook (DAG) definition of systems engineering is provided for inspection:

> Systems Engineering. An interdisciplinary approach and process encompassing the entire technical effort to evolve, verify and sustain an integrated and total life cycle balanced set of system, people, and process solutions that satisfy customer needs [6].

The DAG definition identifies a three part interdisciplinary approach which addresses the entire solution space across systems, people, and processes. The holistic SE approach is particularly important when considering the cyber security

This work was supported by a research grant from the Air Force Research Laboratory (F4FBFV1297J001).

problem in operational environments where users and administrators are responsible for the operation, configuration, and maintenance of critical cyber systems. History has shown that people and processes are much more vulnerable than specific technologies. Consider for example, the devastating results of the Stuxnet worm against the Iranian Nuclear facility, Natanz. The facility was arguably one of the most protected facilities in the world, yet a single well planned cyber attack was able to cause untold damage through processes and personnel vulnerabilities [7].

The challenge for systems engineers is not only to support the entire technical effort to evolve, verify, and sustain systems but to completely understand the complex system under development. The International Council on Systems Engineering (INCOSE) handbook elaborates this concept:

> The SE process has an iterative nature that supports learning and continuous improvement. As the processes unfold, systems engineers uncover the real requirements and the emergent properties of the system. Complexity can lead to unexpected and unpredictable behavior of systems; hence, one of the objectives is to minimize undesirable consequences [8].

Systems engineers are therefore responsible to discover and facilitate shared understanding regardless of system complexity. To this end, the SE developmental process, commonly known as the V-model, is captured in Fig. 1 [9].[1] Note: The SE V-model will be referred to as the SE development process throughout this paper to more accurately capture its intended purpose.

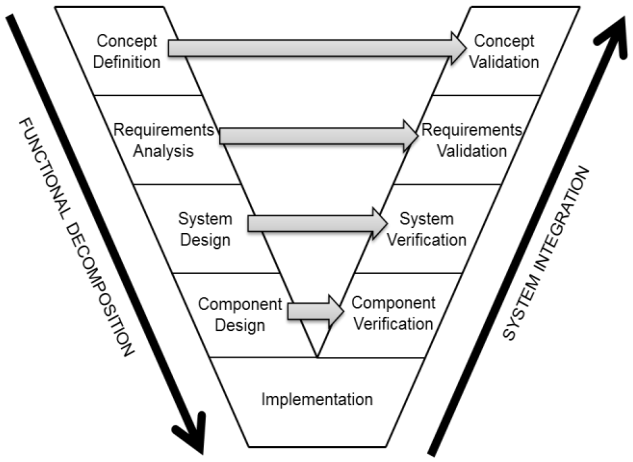

Fig. 1: The Systems Engineering Development Process

The goal of the SE development process is to make complex systems more readily understandable for users, developers and decision makers. The process demonstrates SE functional decomposition activities down the left side, system implementation at the bottom, and system integration activities up the right side. Although the process shows a linear progression, there are numerous iterations both vertically and horizontally within and across the various SE activities.

Functional decomposition is an essential SE tool to define, analyze, and understand complex systems. Functional decomposition products based on core mission requirements can provide great benefit for system users, developers, and decision makers. However, functional decomposition is also a challenging task; one that is often misunderstood, underutilized, and readily dismissed. As a general guideline, functional decompositions should be accomplished at the level of detail necessary for a given project, while addressing:

- Functional Requirements
- Inputs, Outputs, and Controls
- System Boundaries
- System Interfaces
- System Dependencies

Basic functional decompositions can be accomplished quickly using informal block diagrams or common work breakdown structures, while detailed decompositions can be accomplished through formal system architectures. Those associated with system development in the DoD should be familiar with the DoD Architecture Framework (DoDAF) which provides a detailed set of products for this specific task. Although the DoDAF can be somewhat cumbersome, it can yield great benefits when properly and proportionally utilized.

System implementation corresponds to the realization of the system under development by many specialized domain engineers. Components are built to specification in order to meet design criteria and higher level requirements. Implementation is typically accomplished over long periods of time across multiple assembly initiatives, which result in components ready for individual verification. Supply chain integrity is a critical part of the implementation, and a current focus item for DoD acquisition efforts.

System integration activities provide Verification and Validation (V&V) of specific components, system design, and system requirements against their respective functional decomposition activities. The overall purpose of V&V is to provide confidence to the user that the developed system will meet user requirements. Verification activities demonstrate that the implemented components, subsystems, and systems function as specified, while validation activities confirm the system operates as intended by the user.

SE V&V activities consist of assessment, test, and evaluation efforts which span the entire system development process in a continuous fashion. For example, requirements validation starts early on in concept definition, continues down into verification of individual components in the form of derived requirements, and ends with the validation of a fully integrated system against the original user system definition. Lastly, as with much of the SE development process, V&V activities are scalable from brief efforts to meticulously detailed efforts which can take months or years in some cases.

1. Although other common development models and processes exist, such as evolutionary acquisition and spiral development, the SE V-model was chosen because of its general acceptance in the SE community and clear linkages between decomposition and integration activities. Furthermore, distinct verification and validation activities readily lend themselves to the formalized ‘justified confidence’ requirements of system assurance claims as described in Section IV, parts C and D.

IV. RESEARCH

This section details the SE development process in conjunction with key security patterns to provide a baseline for secure and assured cyber systems.

A. Assessment of Security Patterns

For the software engineering community, design patterns are widely accepted to communicate object-oriented concepts and architectural structures. Because of their demonstrated benefits, other communities, such as the IT security community have attempted to develop equally effective security patterns. The seminal text for security patterns is "Security Patterns—Integrating security and systems engineering" by Schumacher et al. [10], while other publications of note are "Secure Design Patterns" by the Carnegie-Mellon Software Engineering Institute [11], a survey of security patterns by Yoshioka et al. [12], the online security pattern repository [13], and the open group security pattern technical guide [14]. Additionally, the First International Workshop on Cyberpatterns was hosted in 2012, which focused on attack-oriented security patterns [15]. Formally, a security pattern is defined as:

> Security Pattern. A security pattern describes a particular reoccurring security problem that arises in specific contexts, and presents a well-proven generic solution [10].

A strict interpretation of security patterns yields a growing, yet limited, set of results. However, given the broad nature of a pattern's stated purpose i.e., a well-proven generic solution to a reoccurring problem, there are many such security patterns available for review. Less formal security patterns exist in many formats captured as policy, standards, best practices, guides, processes, instruction manuals, checklists and many more, which ultimately result in a very large body of knowledge with significant depth.

From the practitioners' point of view, the key problem then becomes knowing when and where to apply the appropriate patterns. Therefore, the proposed SE approach for secure and assured systems highlights the utilization of key cyber patterns during specific points in the system development process. The key security patterns are introduced in Section B and fully described in Sections C and D.

B. Key Cyber Patterns for Cyber System Development

Key cyber security patterns were derived from the reviewed literature and categorized into the formalized SE developmental process as shown in Fig. 1. Because of overall similarity, concept definition and requirements analysis will be addressed together. System design will be addressed singularly, which supports the use of conventional IT enterprise architectures used in the design and operation of many IT centric systems. Component design and implementation activities will likewise be addressed together due to similarity of purpose.

1) Concept Definition and Requirements Analysis: Security patterns supporting system concept definition and requirements analysis are mostly related to high-level policy and risk management. These activities should not be discounted as they form the backbone of all cyber system security decisions. There are a number of critial issues at this level of system development which security patterns can aid. Key security patterns for consideration:

- Risk Management – conduct threat and vulnerability assessments, predict likeliness factors, calculate expected loss, prioritize results, plan/implement mitigation actions, and re-evaluate expected loss. Risk management is perhaps the single greatest concern for a cyber system under development.
- Asset Identification and Assessment – determine critical information and technologies, mission threads, core business processes, intellectual property, essential knowledge, personnel, and other crucial resources. This security pattern may be considered part of risk management proper; however, because of its importance we have specifically identified the security pattern as a uniquely important task. It also serves to prioritize security and assurance measures during the entire development process.
- Formalize Security Requirements – define and document the extent to which cyber security attributes are desired and/or required. Consider confidentiality, integrity, and availability, along with other security attributes specific to the system under development.

2) System Design: Security patterns supporting system design have grown out of enterprise level security, software, and network architectures used extensively by software and IT professionals for design and operation. While the practioners' architectures are focused more on building and operating cyber systems, the systems engineer's design architecture is focused more on understanding the system under development as discussed in Section III. Key security design patterns for consideration:

- Determine Appropriate Security Approach – select a scheme to achieve the desired security state i.e., approaches for deterrence, prevention, detection, and recovery. These decisions will heavily influence the system design and specific components.
- Security-Oriented Functional Decomposition – conduct decomposition using established security principles i.e., separation of duties, least privilege, and defense-in-depth.
- Separate Security Functionality – separate security functionality from other system functionality.

3) Component Design and Implementation: This grouping of security patterns is by far the largest, with many software and application specific security patterns available for consideration. These technology specific security patterns are generally applicable to very specific problems, although there are a number of very helpful objective-based security patterns such as "The Twenty Critical Security Controls" [16] and "Raising the Bar for Cybersecurity" [17]. Because of the abundance of security patterns available at this level, three types of security patterns are considered:

• Design Pattern Extensions for Security – security oriented extensions of longstanding object oriented software design patterns. These patterns provide great utility and ease of use for cyber system software implementers.

• Objective-Based – robust solutions to the broader cyber security problem. Generally, cover multiple security attributes and referred to as best practices.

• Application Specific – focused on the configuration and operation of cyber systems or security devices. Examples include firewalls, audit/logging, input validation, white listing, secure web applications, and many others.

Next, Section C briefly describes system assurance with respect to the SE development process, before entering a detailed discussion of the confluence of key cyber security patterns, the SE development process, and their contribution towards system assurance in Section D.

C. Systems Engineering Approach for System Assurance

Given the complex nature of modern cyber implementations and the current threat environment, system assurance is sometimes considered an unobtainable goal. Despite this bleak view, the National Defense Industrial Association (NDIA) recently published a comprehensive text for system assurance titled *Engineering for System Assurance*, which makes progress towards solidifying the practice. The definition of system assurance is provided for inspection:

> System Assurance. The justified confidence that the system functions as intended and is free of exploitable vulnerabilities, either intentionally or unintentionally designed or inserted as part of the system at any time during the life cycle [18].

Notably, NDIA describes 'justified confidence' with formal assurance claims including: context, assumptions, justification, evidence, and criteria which are used to formally answer the desired level of assurance for critical functional requirements.

The SE development process provides the flexibility to account for assurance claim context and assumptions early on while addressing justification, evidence, and criteria during system integration. SE functional decomposition and system integration activities can be used at each point in the development process to provide justified confidence through a formal and defined process. Each SE activity from concept definition to validation is scalable, which provides selectively robust support for critical system functions and their assurance claims. Specifically, SE system integration activities readily lend themselves to meeting assurance claims through defined V&V activities.

D. Detailed SE Process Activities for System Assurance

This section addresses each SE functional decomposition and system integration activities as they contribute towards cyber system assurance. A description of the relationship between the SE activity, key cyber security patterns, and formalized assurance claims i.e., context, assumptions, justification, evidence, and criteria is described. Because of inherent dependencies concept definition and requirements analysis will be addressed together.

1) Concept Definition & Requirements Analysis

As the starting point in the SE development process, concept definition and requirements analysis form the basis of all future decomposition and integration activities. Modern cyber systems face many common requirements problems, however, these problems are compounded by the additional challenges associated with cyber system complexity and security requirements. Further, all future assurance claim justification efforts will be accomplished against these formalized requirements.

The assurance claim is intended to define system assurance requirements, which take the form of confidence levels during concept definition and requirements analysis. With respect to assurance of cyber systems, concept definition and requirements analysis activities should result in both formalized security requirements and desired levels of assurance for functional requirements.

In cyber system development risk management is a critically important, yet a rather challenging task due to system complexity. For the cyber system developer, there is an overwhelming amount of information dedicated to IT risk management as it is a completely unique field of study. Fortunately, much of the conceptual material is similar in nature. Of note, the National Institute of Standards and Technology (NIST) produced an IT security framework which is rather verbose, but clearly articulated and detailed. NIST Special Publication 800-30 is an excellent place for the cyber systems engineer to start learning this vital security pattern [19]. A general rule of thumb is the level of security should be commensurate with an item's value as described in the asset identification and assessment security pattern.

Cyber system risk management must consider many difficult software issues, significant operational and support dependencies, and a hostile cyber threat environment that can change in a moment's notice. Because of the unbounded nature of risk management assessment and mitigation, the systems engineer should ensure baseline level of risk management is conducted, while providing additional detailed analysis where necessary to meet the user's needs. SE concept definition and requirements analysis activities coupled with risk management aid in the development of secure and assured cyber systems by appropriately addressing critical functions for cyber systems.

There are many very effective security patterns available for performing asset identification and assessment, especially as they pertain to risk. There is an expected overlap with risk management as asset identification and assessment are often cited as the first and second steps in the risk management process. However, in the cyber community asset identification and assessment is typically focused on physical asset inventory and does not adequately address the entire range of mission critical functions, information, and assets. This common shortfall supports why we have chosen to included this task as a uniquely important security pattern for cyber systems.

Within the DoD, Critical Program Information (CPI) has been a mainstay in program protection planning for identifying information and technology assets [20]. Criticality Analysis, and to a lesser extent Critical Mission Threads, are also

methods widely employed in the DoD [6]. The purpose of these activities is simply to gain an accurate understanding of the resources—systems, people, processes—a system utilizes and prioritize their significance towards mission accomplishment. These prioritized critical functions and resources should then serve as the focus of system security and assurance efforts throughout the SE developmental process.

Formalized security requirements have recently been recognized as a necessary functional requirement for cyber system and not merely a support requirement. The distinction is significant and puts more attention on securing and assuring a system rather than falling into a security checklist mentality. The goal of the formalized security requirements is to consider and define what security attributes should be for critical functions in a cyber system under development. System assurance levels should be considered at this point in like manner. The direct link between formalized security requirements and desired assurance levels is a natural fit.

Security requirements are usually described in the context of the industry established security attributes: Confidentiality, Integrity, and Availability. For each core function of a cyber system, cyber security attributes should be enumerated. Typically, enumeration is done categorically in the form of "high, medium, low" or "extreme, urgent, high, medium, low" in a comparative fashion. The scale is subject to user preference and specific application. Two simple examples follow: 1. A command and control system may have an "urgent" data integrity requirement along with "high" availability and confidentiality requirements; and 2. An intelligence signal processing cyber system may have "extreme" confidentiality and integrity requirements, along with "medium" availability requirements.

Security professions often extend these core attributes with various other information security principles. For example, the core security attributes will often be extended with Identification, Authentication, and Non-Repudiation for secure communication functions. Furthermore, formalized security requirements specify the desired security principles and assurance levels for a system under development, which are also used during validation activities. Early SE functional decomposition activities are very helpful when paired with formalized security requirements to address the whole cyber system—systems, people and processes—security problem.

2) System Design

Once system requirements are established, design is the next step in providing an effective system. System design is where SE faces the challenge of realizing system-wide requirements in a functional design. In order to accomplish this task, systems engineers must first have a detailed operational and technical understanding of the subject domain(s). Second, systems engineers must study system functionality, inputs, outputs, controls, boundaries, interfaces, and dependencies. Additionally, there are many known and unknown complexity issues that may surface during system design activities. These issues will need to be addressed as they arise to assure the functionality of the system.

In general, SE should analyze and document system complexity through functional decomposition activities which typically result in a set of architectural products called views. These architectural views or products are scalable to a desired level of specificity, from conceptual block diagrams to detailed design architectures for the most complex systems. SE design activities fall right in line with providing assured systems through justifiable decision making to "build-in" smarter and more effective security solutions right from the start at less cost. System design decisions also need to be considered in a cost-benefit manner, with a clear understanding of the programmatic nature of large developmental efforts.

Appropriate security approaches should be selected to meet the overall security requirements. Often there is a mutually supportive overlap between security approaches as discussed in the security pattern literature. The security objective(s) will drive the security approach resulting in a mix of deterrence, prevention, detection, and response solutions. These approaches are tempered by the cost-benefit nature of risk management. Clearly articulating and applying security approaches for critical system functions directly supports assurance claims, contributing to a more stable system.

Recent trends are moving towards more cost-effective approaches of detection and response. It is often much simpler and quicker to rebuild a compromised system than attempt to prevent future compromises, which can be seen as impossible. There has also been a significant movement towards resilient and agile systems, which can self-recover from failures. System design is the optimal time to consider which approaches will be implemented for a given cyber system to meet the defined security and assurance requirements.

Separation of duties, least privilege, and defense-in-depth constitute the core of modern information security principles and should be considered throughout functional decomposition and specifically system design activities. As with selecting appropriate security approaches, applying core security principles builds towards a stable cyber system. In principle, this security pattern requires only a slight modification to existing SE functional decomposition practices. Considering key security principles during the design phase of a system is perhaps more of a basic practice and less of a documented security pattern, however, its importance cannot be overstated to solving the assured cyber system problem.

Separate security functionality should be considered and designed in whenever possible. This security pattern is often viewed expressly for software development efforts to separate security checks from object creation, however, this security pattern should be more broadly applied. For example, each security test or check should be considered separately from any system functional requirement. Separating security functionality has the benefit of clearly identifying security checks and reducing implementation complexity. Separate security checks should be enforced throughout system design and implementation wherever feasible given the appropriate risk management and cost-benefit considerations.

3) Component Design & Implementation

Although it is not the aim of this paper to discuss security patterns as they pertain to component design and implementation, there are a couple of interesting comments which should be made. First, because of the wide popularity of

object oriented design patterns a rash of security-oriented extensions quickly arose. These design pattern extensions for security are very helpful for software engineers attempting to "build-in" security and should be applied wherever possible.

Second, there is essentially no limit to existing and potential application-specific security patterns, because they are tightly coupled to technological solutions for specific problems i.e., filtering on a firewall. There are literally thousands of application-specific security patterns available for review across many cyber security related problem sets. These security patterns should be investigated for a given application as many excellent ideas exist for securing cyber devices.

Third, because of the numerous application-specific patterns at the component level, consolidated security patterns have appeared. We've termed these objective-based security patterns which attempt to answer the broader cyber system security problem. Popular examples of objective-based security patterns are the SANS Top Twenty critical controls and the Australian DoD's top 4 mitigation strategies.

4) Component and System Verification

As system integration begins, there is a natural fit between component and system verification activities and system assurance goals. Verification activities can range from documentation reviews to detailed line by-line code reviews spanning days, weeks, or months. Formally, verification is described as "the purpose of the Verification Process is to confirm that the specified design requirements are fulfilled by the system" [21]. The prioritization of the cyber system verification activities should be driven by function criticality as described during functional decomposition activities. Fig. 2 shows the key cyber security patterns mapped to the SE development process for concept/requirements validation, system design verification, and component design/implementation verification.

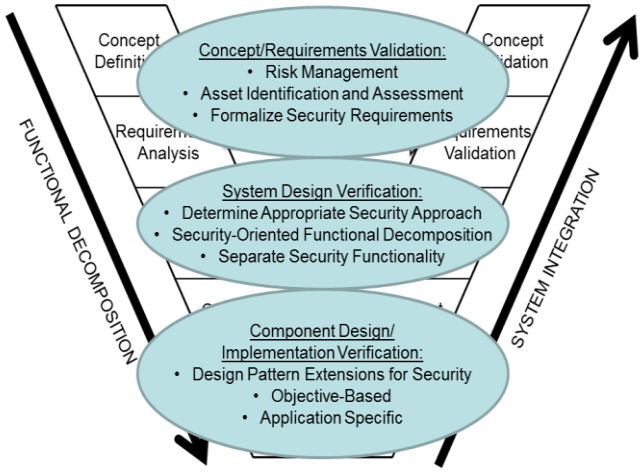

Fig. 2: Key Cyber Security Patterns

Revisiting the definition of verification, the systems engineer's goal is to "confirm that the specified design requirements are fulfilled" and should use whatever tools necessary to accomplish the task. A simple list of verification techniques includes: basic function testing, result comparisons, input/output sensitivity, parameter checking, structural code review, detailed code walkthrough, review of math/logic proofs, and detailed end-to-end input/output traces. This list is by no means exhaustive and meant merely as a starting point. Due to specific nature of verification activities and desired level of justification evidence, it is difficult to recommend specific verification activities for a given cyber system implementation. The author recommends reviewing cyber system verification activities and processes found in ISO/IEC 15288 [21], ISO/IEC 26702 [22], the DAG [6], and the INCOSE SE handbook [8].

As a pertinent aside, security measurement and evaluation of cyber systems is a highly debated subject. The effectiveness of current evaluation criteria and resulting formal measurement i.e., security audits are being called into question. It seems that despite significant effort put towards the development of thorough IT security criteria, they have not provided a sufficiently suitable solution to the cyber system assurance problem. This is evidenced by the weekly announcements of cyber security breaches, vulnerabilities, and the non-stop release of critical patches by major software developers.

5) Requirements and Concept Validation

During validation activities the entire technical effort described as systems, people, and processes should be evaluated to determine if it can meet the specified users' requirements. Formally, validation is described as: "the purpose of the Validation Process is to provide objective evidence that the services provided by a system when in use comply with stakeholders' requirements, achieving its intended use in its intended operational environment" [21]. The output of the validation activities should be a determination to what extent the systems meet's the desired capability.

SE validation activities tie closely to system assurance case evaluation methodology as described by NDIA's Engineering for Systems Assurance: "the purpose of an assurance case is to provide convincing justification to stakeholders that critical system assurance requirements are met in the system's expected environment(s)" [18]. SE validation activities significantly contribute to demonstrable proof that a particular system meets its documented purpose(s). There are some very helpful security patterns to assist in system validation as shown in Fig. 2 as previously discussed.

Some additional issues for consideration are system re-purposing and unexpected operational environments. System evaluation is typically only considered for the planned operational environment, despite a high likelihood of other possible implementations. Without a broader consideration, systems become immediately vulnerable when re-purposed or deployed in less than ideal environments. The system engineer must also consider that the system itself is always changing due to regular patching cycles, scheduled upgrades, operator rotations, and process improvement initiatives.

A basic list of SE validation activities consists of addressing: Key Performance Parameters (KPPs), initial assumptions, requirements traceability, concept review and requirements assessment (i.e., a document review addressing the stated purpose, requirements, and key functions), external validity (i.e., black box testing), and internal validity (i.e.,

white box testing). This list is by no means exhaustive and meant merely as a starting point for validation activities. It is difficult to recommend specific validation activities for a given cyber system implementation and the author recommends reviewing cyber system validation activities and processes found in ISO/IEC 15288 [21], ISO/IEC 26702 [22], the DAG [6], and INCOSE SE handbook [8].

Once the system is fully integrated, SE validation activities can justifiably determine if the system can meet the desired levels of assurance. The SE development process, particularly system integration, culminates in validation activities which are designed to provide objective evidence that user requirements are being meet. These same results can be leveraged to provide justified confidence in the desired system functionality for system assurance claims.

V. CONCLUSION

This paper presented a description of key cyber security patterns categorized to the SE development process. This paper further examined SE decomposition and integration activities, detailing their contribution to achieving cyber system security and assurance. SE V&V activities were examined and determined to provide sufficient justification to meet formal systems assurance claims. Specifically, verification can be used to directly support the justification, evidence, and criteria associated with formal assurance claims, while validation defines and supports the context, assumptions, justification, and criteria associated with these claims for cyber systems. In conclusion, this paper builds upon cyber security patterns and established SE process to provide assured cyber systems.

ACKNOWLEDGMENTS

Thank you to Mr. Rick Dove, INCOSE system security engineering working group chair, who provided helpful guidance towards the study of security patterns.

This work was supported by a research grant from the Air Force Research Laboratory (F4FBFV1297J001).

REFERENCES

[1] President Barack Obama. *Improving Critical Infrastructure Cybersecurity*. Executive Order, Office of the Press Secretary. 12 February 2013.

[2] Nakashima, Ellen. "Pentagon to boost cybersecurity force," The Washington Post, 27 January 2013. [Online]. Accessed: 23 February 2013. Available: http://www.washingtonpost.com/world/national-security/pentagon-to-boost-cybersecurity-force/2013/01/19/d87d9dc2-5fec-11e2-b05a-605528f6b712_story.html.

[3] *APT1 - Exposing One of China's Cyber Espionage Units,* Mandiant® [Online]. Accessed: 23 February 2013. Available: http://intelreport.mandiant.com/Mandiant_APT1_Report.pdf

[4] Definition of complexity is adapted from Information Technology—Security Techniques—Evaluation Criteria for IT Security, Part 1: Introduction and General Model, Third Edition. ISO/IEC 15408-1:2009(E). Switzerland: International Organization for Standardization, 15 December 2009.

[5] *Trusted Computer System Evaluation Criteria*. DoD 5200.28-STD. Washington: Department of Defense, 26 December 1985.

[6] Defense Acquisition University. *Defense Acquisition Guidebook, Chapter 4, Systems Engineering.* [Online]. Accessed: 11 October 2012. Available: https://acc.dau.mil/CommunityBrowser.aspx?id=490091.

[7] Langner's Stuxnet Deep Dive S4 Video. [Online]. Accessed: 3 March 2013. Available: http://www.digitalbond.com/blog/2012/01/31/langners-stuxnet-deep-dive-s4-video/.

[8] INCOSE System Engineering Handbook v. 3.2.2. INCOSE-TP-2003-002-03.2.2. San Diego, CA: International Council on Systems Engineering (INCOSE), October 2011.

[9] See [8]. Adapted from Forsberg, K., H. Mooz, H. Cotterman, Visualizing Project Management, 3rd Ed., J.Wiley & Sons, 2005.

[10] Schumacher, Markus., Eduardo Fernandez-Buglioni, Duane Hybertson, Frank Buschmann, and Peter Sommerlad. *Security Patterns: Integrating Security and Systems Engineering*. West Sussex, England: John Wiley & Sons, Ltd. 2006.

[11] Secure Design Patterns, Software Engineering Institute, Chad Dougherty et al., CMU/SEI-2009-TR-010, updated oct 2009.

[12] Yoshioka, Nobukazu., Hironori Washizaki, and Katsuhisa Maruyama. "A Survey On Security Patterns," *Progress in Informatics, National* Institute of Informatics, no .5, pp.35-47, 2008.

[13] *Security Patterns Repository Version 1.0*. [Online]. Accessed 22 January 2013. Available: http://www.securitypatterns.org/.

[14] Blakley, B. and Heath, C. et al.. Security Design Patterns. Technical Guide, G031. The Open Group, April 2004.

[15] Proceedings of Cyberpatterns 2012. Abingdon, UK. 9-10 July 2012.

[16] *Critical Controls for Effective Cyber Defense, Version 4.0*. SANS Institute InfoSec Reading Room. [Online]. Accessed 22 February 2013. Available: http://www.sans.org/critical-security-controls/cag4.pdf.

[17] Lewis, James A. "Raising the Bar for Cybersecurity." Center for Strategic & International Studies: Technology & Public Policy. 12 February 2013.

[18] National Defense Industrial Association (NDIA) Assurance Committee. 2008. *Engineering for Systems Assurance*. Arlington, VA: NDIA.

[19] National Institute of Standards and Technology. *Risk Management Guide for Information Technology Systems.* SP 800-30. Gaithersburg, MD: Information Technology Laboratory, July 2002.

[20] Department of Defense. Critical Program Information (CPI) Protection Within the Department of Defense, Change 1. DoD Instruction 5200.39. Washington: Under Secretary of Defense for Intelligence, 28 December 2012.

[21] *Systems and software engineering – System life cycle processes.* ISO/IEC 15288:2008(E). Switzerland: International Organization for Standardization, 31 January 2008.

[22] *Systems engineering – Application and management of the systems engineering process.* ISO/IEC 26702:2007(E). Switzerland: International Organization for Standardization, 15 July 2007.