Security Engineering

Technical Report SERC-2012-TR-028-1

January 31, 2012

Principal Investigator: Dr. Barry Horowitz, University of Virginia

Team Members:

Stevens Institute of Technology: Dr. Jennifer Bayuk,

University of Virginia: Dr. Peter Beling, Dr. Alfredo Garcia,

Contract Number: H98230-08-D-0171 Task Order: 0002, DO 002 RT 028

Report No. SERC-2012-TR-028-1

The Systems Engineering Research Center (SERC) is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology.

This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Office of the Assistant Secretary of Defense for Research and Engineering (ASD(R&E)) under Contract H98230-08-D-0171 (Task Order 0002, DO 002 RT 028).

ABSTRACT

This report presents the major results for the Security Engineering research effort funded through the Systems Engineering Research Center. A major result of the efforts to-date are the introduction of a point defense approach to cyber security that provides defense solutions that are embedded in the systems to be protected (as opposed to the access perimeter to those systems and the networks that support those systems). These solutions are referred to as System Aware security because their designs depend upon intimate knowledge of the designs of the systems being protected. In addition to introducing a new design concept for cyber security, a scoring system has been introduced that provides a basis for comparing alternative security architectures that employ point defense solutions. The results to-date have been documented in a paper that has been accepted for publication in the peer reviewed journal, Systems Engineering, Vol 15, No 2 in 2012.

This page intentionally left blank

TABLE OF CONTENTS

FIGURES AND TABLES

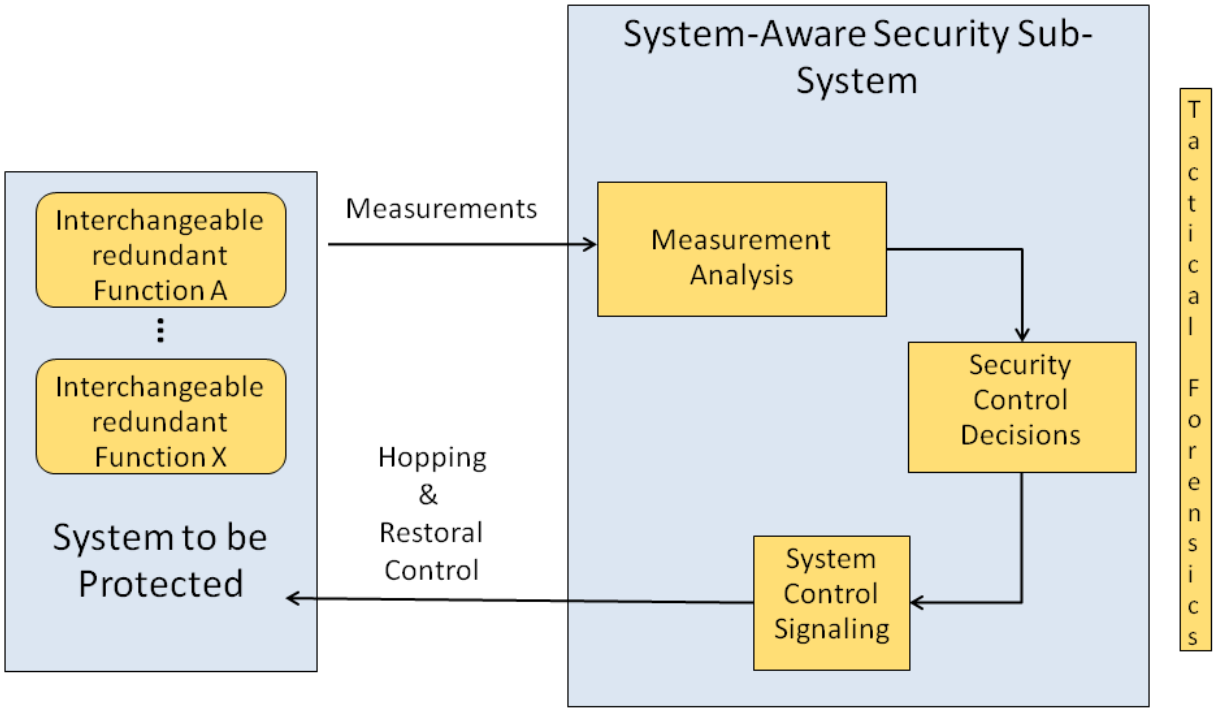

Figure 1: A high-level block diagram of the System-Aware Cyber Security Architecture, including measurement features for attack detection and tactical forensics, decision functions for system control, and signaling functions for managing system restoration and configuration hopping.

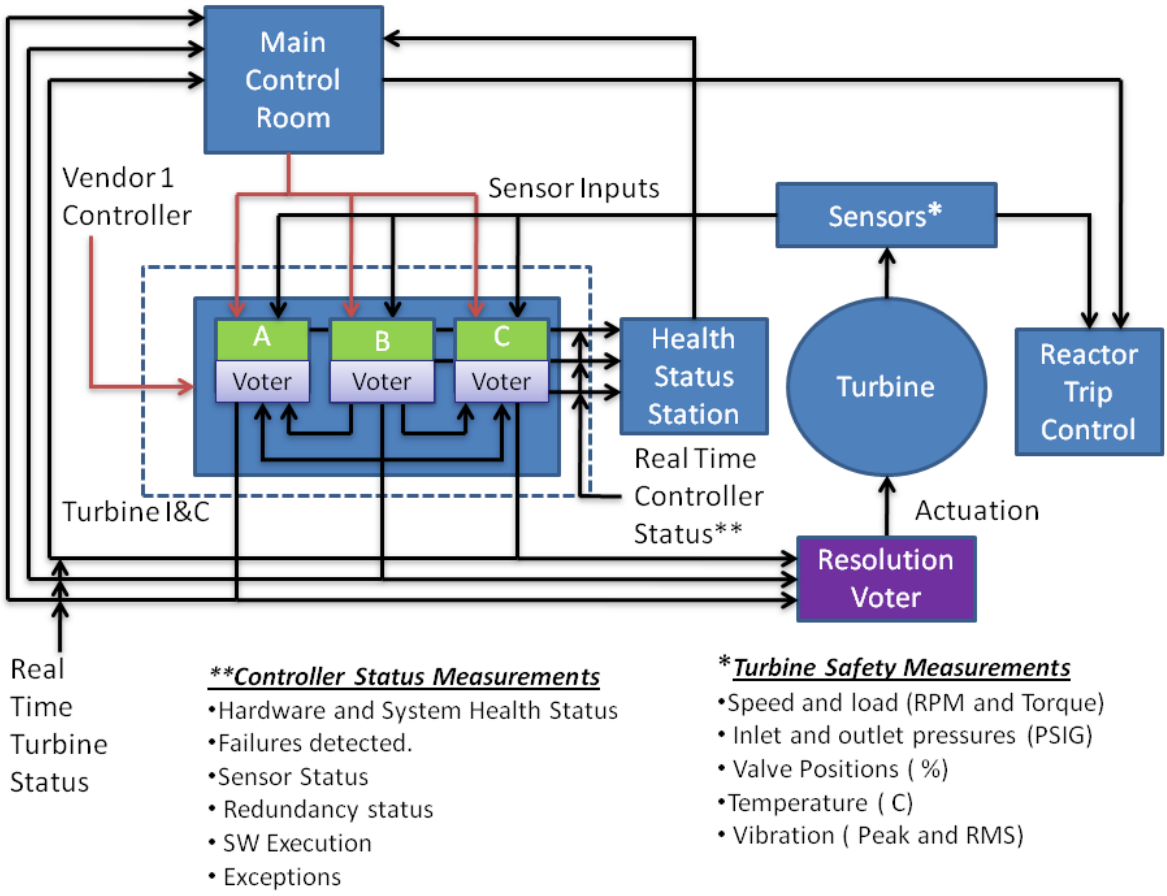

Figure 2: A high-level system diagram for a typical steam fed nuclear reactor powered turbine control system. The turbine controller is designed to meet high reliability and safety standards by employing redundancy and a resolution voter.

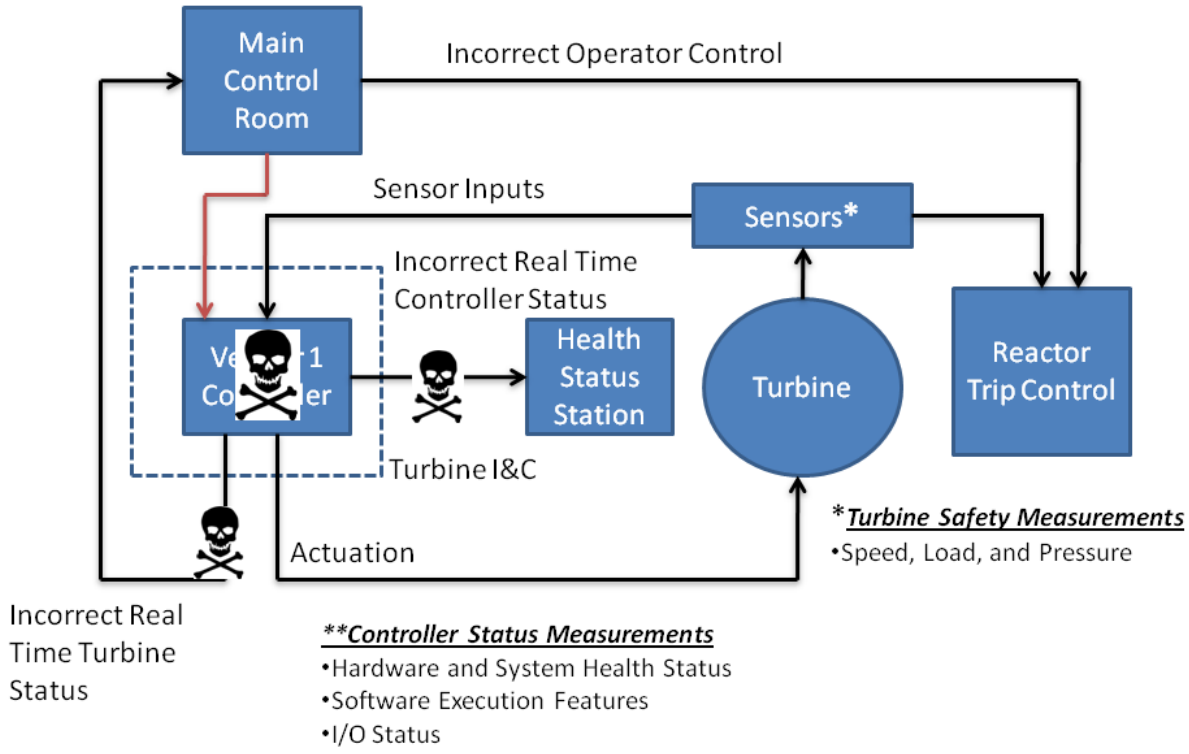

Figure 3: Illustration of a hypothetical cyber attack on a turbine control system. The controller is embedded with a Trojan horse that can modify, nullify, or replace information sent to the operators in the Main Control Room and the automatic monitoring station. The intended consequence of the attack would be to have the operator(s) believe that the turbine is operating sufficiently outside of specification so as to warrant a procedural action (e.g., shutdown of the turbine), which would induce a plant trip.

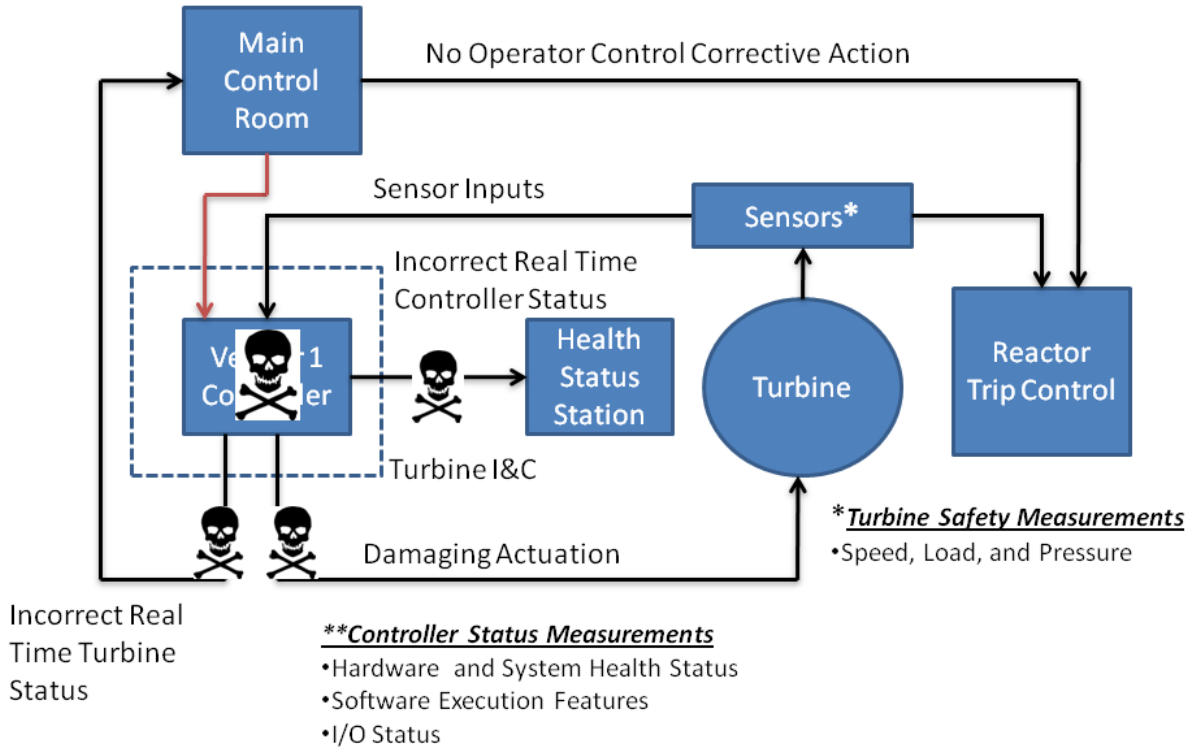

Figure 4: Illustration of a hypothetical cyber attack on a turbine control system. The controller is embedded with a Trojan horse that can modify, nullify, or replace information sent to the operators in the Main Control Room, Health Status Station, and the actuation commands sent to the Turbine. The consequence of a successful attack would be to create severe damage to the turbine, as well as to trip the reactor by sending disruptive control signals to damage the turbine and manipulating control information sent to the Main Control Room.

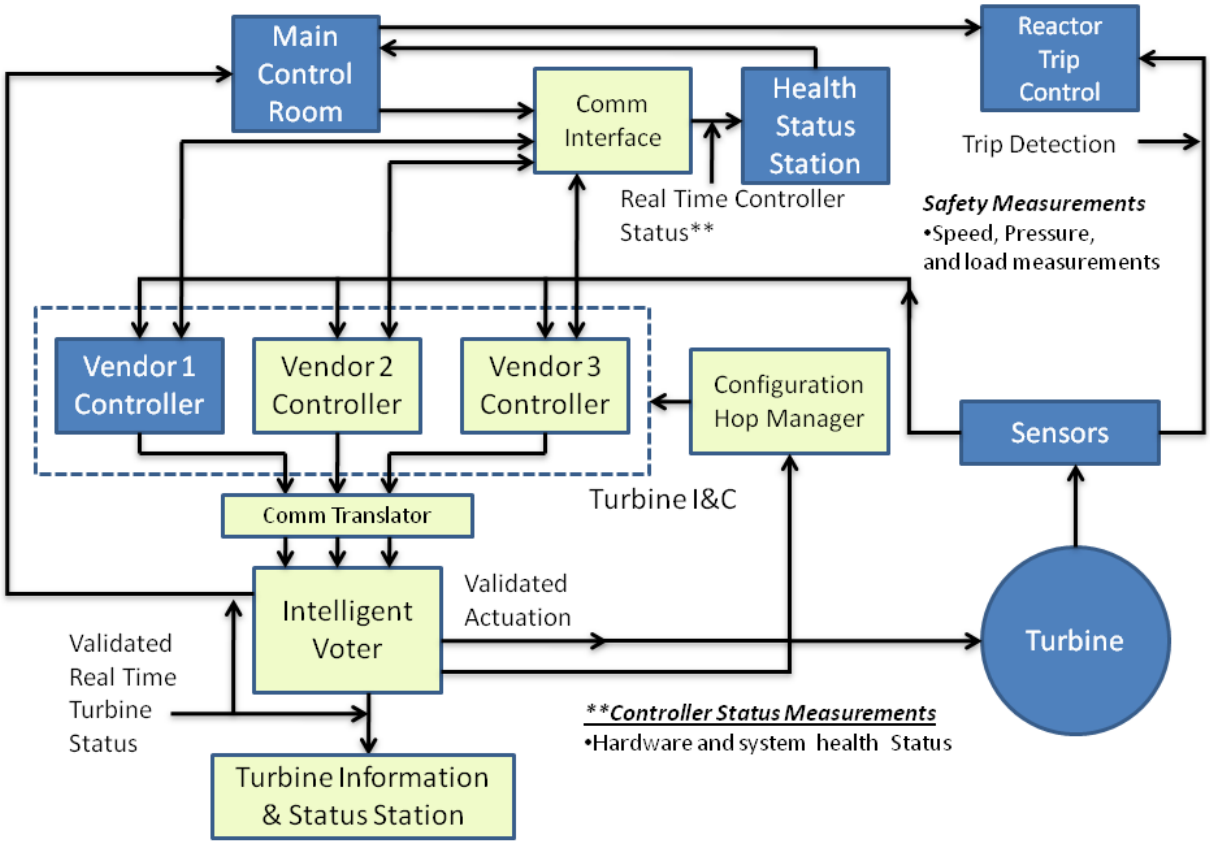

Figure 5: Possible System-Aware security architecture to address the threats illustrated in Figure 3 and Figure 4. This includes three controllers in the system architecture and an intelligent voting process to rapidly disable a differentiated controller from carrying out its turbine actuation functions.

Value Factors → / Security Services ↓	Deterrence	Real Time Defense	Restor-ation	Collateral System Impacts	Implemen-tation Cost	Life Cycle Cost	Other
Diversity (s_1)	s_{11}	s_{12}					s_{1j}
Hopping (s_2)	s_{21}	s_{22}					s_{2j}
Data Consistency Checking (s_3)	s_{31}	s_{32}					s_{3j}
Tactical Forensics (s_4)	s_{41}	s_{42}					s_{4j}
Other (s_i)	s_{i1}	s_{i2}					s_{ij}

s_{ij} = Assurance Level of the ith service as related to the jth value factor

s_{ij} = Quantized Assurance Level = 0...M

$$\text{Security Score} = \sum_{j=1}^{p}\sum_{i=1}^{n} s_{ij}$$

Max Possible Score = p x n x M

Figure 6: A possible representation of the scoring elements outlined in section 5.3. Table composed of System-Aware security services and value factors. Each service-value factor pair is given an assurance level, *s*, based upon how the service level effects the given value factor.

Relative Value Weights	k_1	k_2	k_3	k_4	k_5	k_6	k_j
Value Factors → / Security Services ↓	Deterrence	Real Time Def.	Restor-ation	Collateral System Impacts	Implemen-tation Cost	Life Cycle Cost	Other
Diversity (s_1)	s_{11}	s_{12}					s_{1j}
Hopping (s_2)	s_{21}	s_{22}					s_{2j}
Data Consistency Checking (s_3)	s_{31}	s_{32}					s_{3j}
Tactical Forensics (s_4)	s_{41}	s_{42}					s_{4j}
Other (s_i)	s_{i1}	s_{i2}					s_{ij}

$$\sum_{j=1}^{p} k_j = 1$$

s_{ij} = Assurance Level of the ith service as related to the jth value factor

s_{ij} = Quantized Assurance Level = 0...M

$$\text{Security Score} = \sum_{j=1}^{p} \sum_{i=1}^{n} k_j s_{ij}$$

Max Possible Score = n x M

Figure 7: An extension of the scoring representation shown in Figure 6. This extended table is composed of System-Aware security services and weighted value factors. Each service-value factor pair is given an assurance level, s, based upon how the service level effects the given value factor. Furthermore, each value factor is assigned a weight, k, based upon its importance to a given system owner and operator.

SUMMARY

1 INTRODUCTION

Increasingly systems are being digitized in a drive toward lower costs, improved efficiency, and reduced time to market. However, this drive to digitization also renders these systems susceptible to an increasingly sophisticated and debilitating array of cyber attacks. For example, as shown by Karl Koscher et al. [2010], it is possible for an attacker to embed an infection that is capable of completely disabling an automobile's braking system. Further, in the 2010 Stuxnet attack [Falliere, Murchu, and Chien, 2011], an embedded infection was used to successfully damage up to 1000 centrifuges in Iran (10 percent of the available capacity) [Albright, Brannan and Walrond, 2010]. Such attacks cannot be completely addressed by traditional perimeter security solutions [Wulf and Jones, 2009], as they have been in the past. A new systems engineering focused approach is introduced, integrating fault tolerant system design concepts with advanced cyber security concepts to address these expanding threats. This involves the development of a security architectural formulation [Bayuk and Horowitz, 2011] based on reusable system security services to create a defense that is referred to as System-Aware Cyber Security [Jones and Horowitz, 2011].

Security services are defined to be security elements that are integrated and embedded as a solution into a system, providing unique security functionality designed and tailored for the specific application. The architecture includes services that (1) collect and assess real-time security relevant measurements from the system being protected, (2) perform security analysis on those measurements, and (3) execute system security control actions as required. These services include (1) significantly increasing the difficulty for adversaries by avoiding a monoculture environment through the integration of a diverse set of redundant subsystems involving hardware and software components provided by multiple vendors, (2) the development of subsystems that are capable of rapidly changing their attack surface through hardware and software reconfiguration (configuration hopping) in response to perceived threats, (3) data consistency checking services for isolating faults and permitting moving surface control actions to avoid continuing operations in a compromised configuration, and (4) forensic analysis techniques for rapid post-attack categorization of whether a given fault is more likely the result of a cyber attack than other causes (i.e. natural failure).

Integrated solutions would be determined through awareness of what the system applications do, how they are designed, what they communicate with as well as how they communicate, what their performance requirements are, what missions they are tied to,

what risks are posed through potential attacks, etc. In addition, System-Aware Cyber Security provides the means to hypothesize specific threats in relation to specific application functions. When combined with an understanding of the damage that could occur, this provokes the application of risk sensitive mitigation solutions (i.e. appropriate system security services). This enables systems owners, operators, and regulators to directly link the risk mitigation benefits of specific security services to the cost associated with designing, implementing, and utilizing them. When designing and implementing security solutions, several high level design issues emerge, including (1) selection of the subsystems for redundant diversification; (2) use of moving target solutions for attack avoidance, attack detection, and system restoral functions; (3) selection of which HW/SW components to protect; (4) selection of virtual and/or physical configuration hopping solutions; (5) selection of regimes for physical hopping (local and/or remote); (6) selection of data used to ensure consistency; (7) selection of forensic analysis techniques for rapid categorization of faults; (8) avoidance of interference with normal functioning of the applications; (9) assurance of appropriate isolation of the security solutions; (10) exploitation of opportunities for reuse of existing security solution services; and (11) establishment of administration requirements for control of the security solutions. To illustrate the benefits as well as the design issues that emerge when applying System-Aware Cyber Security, a detailed example for securing a nuclear power plant turbine control system is presented.

2 CURRENT STATE OF SECURITY SOLUTIONS

It is recognized that perimeter security is the mainstay of the current cyber security solution space [Wulf and Jones, 2009]. This has enabled the system engineering and security communities to respond to perceived risks and threats through the addition of new perimeter security solutions on a responsive basis. The most recognizable of these solutions being the firewall, a device utilized in network security to control access to resources. Another example of perimeter security is utilized to ensure the integrity of the supply chain against insider attacks. In this case interviews, background checks, and constrained purchasing from selected suppliers are used to help prevent individuals with malicious intent from gaining access to resources, facilities, information, etc. In all of these cases the goal of perimeter security is to strictly control access to and from key components.

However, while the focus on perimeter security has provided some advantages, it has also brought with it several disadvantages; disadvantages that have become more significant as the cyber threat has evolved. In particular, there has not been widespread development and application of reusable solutions embedded into the system to be protected. The rising threat of successful attacks warrants the consideration of a top-down systems engineering approach that develops solution strategies regarding cyber security that go beyond the perimeter model. Furthermore, the systems engineering

community could be considering tightly coupled system security solutions, starting from the time that a new system architecture is developed and continuing through its entire life cycle. However, while significant attention has been paid to best practices for dealing with the bottom-up approach for engineering security solutions, including Webber et al. "Applications That Participate in Their Own Defense" [2003] and Cai et al. "Honeygames" [2006], the systems engineering community has yet to develop a corresponding architectural framework for a top-down approach for addressing cyber security. For example, in 1980 the United State's intercontinental ballistic missile warning system falsely indicated to its operators that a full-scale nuclear attack had been initiated against the U.S. [US Comptroller General, 1981]. After-the-fact analysis revealed that the system was not designed to recognize a condition where there was no missile-related data being received by the system's sensors but, at the same time, there was data indicating an attack on the screens being observed by operators. Although this event was the result of malfunctioning hardware, it could have just as well been a Trojan horse inserted through the supply chain [Defense Science Board, 2005; DoD, 2009]. As illustrated by this example, there exists an opportunity for the systems engineering community to integrate information assurance and cyber security as smart reusable security services in the broad scope of their overall efforts, and in an architectural manner as is done for other system attributes, such as reliability, maintainability, availability, and supportability.

3 System-Aware Cyber Security Architecture

Figure 1 presents a high-level block diagram that serves to illustrate the System-Aware Cyber Security Architecture described in this paper, including measurement features for attack detection and tactical forensic analysis, decision functions for system control, and signaling functions for managing system restoration and configuration hopping. These capabilities can be integrated in a variety of ways so as to provide several features for enhancing the security of a system. These features can be designed to deter attackers from exploiting a system; avoid attempts to compromise a system; identify when a system has been compromised, prevent the system from being damaged, isolate the compromised components, and enable restoration of the system to a non-compromised state; and enable operators and administrators to confirm that an event has been caused by a cyber attack and to take the appropriate action(s). Several such security features are described below, and an example integrating these features into a nuclear power plant turbine control system is presented in section 4.

3.1 DIVERSITY

Diverse hardware/software system implementations enhance security by creating distinct, dynamically interchangeable redundant functions in a system (thereby avoiding a monoculture environment). While added redundancy potentially provides additional opportunities for attackers to exploit specific implementations, together, the combination of diversely redundant subsystems potentially deters attackers by increasing the necessary effort that is required to compromise all of the implementations. There are many variations of diversity, each providing a different form of security. For example, having a subsystem assembled by two different vendors makes it more difficult for an attacker to inject a hardware or software Trojan horse during assembly, by forcing the attacker to have insiders at two different companies. As another example, utilizing multiple operating systems can complicate an attacker's activities by requiring the utilization of multiple exploits: one for each operating system. The amount of security gained is directly dependent on the amount and form of diversity integrated into the system. However, while diversity reduces the risk of an attacker being able to compromise a system, as a bi-product, it complicates the design and maintenance for the system. This requires the systems engineer to conduct the necessary tradeoff assessments regarding how and where to best apply diversity. Diversity can be used in conjunction with configuration hopping (section 3.2) and data consistency checking (section 3.3) to facilitate in the restoration of a compromised system. Specifically, by having multiple diverse redundant components, a system can be restored to a different configuration than the one that it was in during the time it was compromised. This prevents the attacker from repeating the same exploit and increases the difficulty of completely bringing down the system.

3.2 CONFIGURATION HOPPING

Configuration hopping is a security service that, on a randomized basis within scheduled intervals, enables the dynamic modification of an overall system configuration. This is accomplished through interchanging the modes of operation among diversely implemented redundant components while executing their specified system functions. Interchanges can be accomplished virtually across multiple operating systems, as well as physically across machines that can be either co-located or located over a geographic region (e.g. such as can be employed in cloud computing [Vaquero et al., 2009]). This dynamic interchange provides defense by forcing an attacker to operate within time interval constraints while using a family of coordinated exploits that addresses the complications introduced through diversity. In order to provide this security service, an interchange capability must be developed that does not unacceptably degrade desired system operation. For example, the rate of hopping may have to be constrained due to specific system characteristics that may demand stability. The selection of the specific subsystems to interchange would be determined based upon security and economic considerations.

To offer configuration hopping as a security service to users, two capabilities are needed: tools to configure selected application functions for interchange and tools for controlling interchange subject to specified design criteria. Solutions for a specific system will be derived from (1) the unique system design attributes and estimates of the significance of time constraints on complicating attacker exploits, (2) mission objectives, (3) resource requirements related to making interchanges, (4) system performance costs, and (5) the security risks surrounding the system.

3.3 Data Consistency Checking

Data consistency checking is a service that, for the purposes of data integrity, compares data at different points in a system for logical consistency. Consistency violations can be employed in a variety of ways, ranging from informing system operators of a potential problem to stimulating the automatic reconfiguration of a system so as to avoid operating in a compromised state. For example, in the case of a command and control decision support system, inconsistency of internal system measurements, as determined by security-aware decision support applications, can be used as a basis for recognizing a cyber attack, and potentially isolating which subsystems are most likely to have been affected. Isolating faulty components can also be accomplished through the use of voting [Clarkson, Chong, and Myers, 2008; Fujioka and Okamoto, 1992] across a diverse set of redundant components. For example, an automobile brake control system can be diversely and redundantly replicated and the outputs automatically compared as the basis for determining which of the configurations is at fault. This fault isolation also permits restoration control actions to avoid operations in a compromised system configuration. The data consistency checking service should include evaluations of those data elements embedded in operations that impact system functions that are deemed to be related to the critical operation of those applications being secured. It is envisioned that this function would be designed as an agent that interacts with system application functions to query, collect, and analyze required information, and should interact with other System-Aware security services, such as configuration hopping, to provide capabilities designed to avoid operating in a compromised state.

3.4 Tactical Forensics

Traditionally, in the context of cyber security, forensics has often been associated with attacks with significant consequences and obtaining legal evidence to present in a court of law [Noblett, Pollitt, and Presely, 2000]. However, in the context of System-Aware Cyber Security, forensics are intended to provide tactical analysis which enables system operators and administrators to rapidly distinguish between those faults caused by a compromised component (i.e. cyber attack) and those resulting from other causes (i.e. natural failure). This distinction is critical, as it can have a significant influence on how system administrators and operators should proceed post-attack. For example, assume that a system utilizes both diversity and data consistency checking to secure a vehicle's braking system. Also assume that, in a particular instance, the System-Aware security

system successfully detects and averts a fault that would have disabled the brakes from working. The fault is reported to the owner of the vehicle who now needs to know whether the fault was a result of a failing component or the result of an embedded Trojan horse. If it was a failing component, the owner can have the faulty part replaced. However, if it is the result of a Trojan horse, the owner must report the problem to the relevant parties, for the possibility that many vehicles with the same component are potentially at risk. In the latter case, a forensic investigation is required in order to determine the restoration solution and possibly aid in identifying the culprit.
To separate cyber attacks from other causes of faults, System-Aware Cyber Security architects can utilizes a variety of tools, including decision aids based upon proactive analysis methods, such as Fovino, Masera, and Cian's work [2009] on integrating cyber attacks into fault trees; software and/or hardware embedded in the system specifically designed to identify malicious actions (e.g., a radio frequency spectrum analyzer embedded in a subsystem's hardware chassis, and listening for a wireless triggering command at the time of an actual attack); and application of decision theory to relate evidence to alternative causes.

Finally, it is emphasized that tactical forensics is focused upon rapid attribution and restoration, and would serve to complement traditional forensic analysis techniques, not replace or replicate them.

4 Application of System Aware Cyber Security to Nuclear Power Plants

This section provides examples of potential cyber attacks on a nuclear power plant, and illustrates how a System-Aware Cyber Security Architecture would potentially deter and/or defend against such attacks. While the integration of a set of security services is a general solution approach, the examples serve to illustrate the genesis for potential architectures to establish specific solutions. For the examples, the turbine control subsystem for the power plant is selected as the target for cyber attacks. This selection is based on the significant economic consequences of serious damage to the turbine, and the need to shut down (trip) the nuclear reactor in the event of a turbine shut-down. It is assumed that the attacking mechanism to be defended against is embedded in the equipment that is part of the turbine control subsystem. Furthermore, it is assumed that the actual attack may either be triggered through a pre-established protocol that is built into the infected equipment and deployed through the maintenance process, or through a power plant insider communicating the attack initiation in real time via a built-in communications channel, which is part of the infected equipment. The projected solutions have impacts in the pre-attack stage (deterrence), trans-attack stage (defense and temporary restoral), and the post-attack phase (longer-term restoral and threat

reduction for power plants with similar equipment to the attacked plant) of a cyber attack. Each of these phases of attack is discussed for each of the attack scenarios.

4.1 Representative Model of a Turbine Control Subsystem

This section of the paper describes a model of a turbine control system that is used as the basis for postulating specific supply chain related cyber attacks on such a system, and to address both the potential impacts of attacks and possible System-Aware Cyber Security Architectures to either reduce the consequences or possibly eliminate the attacks.

Figure 2 presents a high-level system diagram for a typical steam fed nuclear reactor powered turbine control system. As indicated in Figure 2, the turbine receives actuation commands from a controller, currently available from a variety of vendors (e.g., the GE Mark VI, and Triconex Tricon). Operators located in the main control room of the power plant are responsible for controlling the turbine. These individuals receive status information from the controller that influences their operational actions, which can include stopping the turbine and correspondingly tripping the reactor to stop steam flow into the turbine. In addition to operator actions, the controller receives sensor information (listed in Figure 2) that together influences its automatic control actions. In situations where the turbine operation is such that it is of immediate importance to stop steam flow, the reactor is automatically stopped (i.e. scrammed), with a reactor shutdown process that is supported by the sensor information related to turbine operation.

Figure 2 also highlights the fact that nuclear power plant turbine controllers are designed to meet high operational reliability and safety standards, and accordingly often employ various types of redundancy. Controller replication is a prevalent application for redundancy, and is depicted in Figure 2 as channels A, B, and C. In this example, the employment of a distributed voting scheme among control elements provides fault tolerance against randomly occurring hardware faults in the redundant controllers. The distributed voters typically exchange and vote on how to utilize the individually derived input sensor information, internal controller state information, and output commands. The results of the distributed voting process are forwarded to a master voter that resolves the output controller commands to the actuation system in the turbine. The master voter function is typically integrated as part of the vendor provided controller platform design. While the system diagram in Figure 2 is representative of a typical fault tolerant controller, different vendors may employ different architectural principles and implementation strategies to manage redundancy and realize the desired level of fault tolerance.

4.2 Potential Supply Chain Related Cyber Attacks

This section describes two hypothetical cyber attacks that relate to current turbine control systems. Both attacks could result in the turbine operation being halted and the reactor being tripped. The attacks are structured along similar lines as the Stuxnet attack referred to earlier in the paper.

Figure 3 provides a diagrammatic representation of an attack where the turbine controller is infected with a Trojan horse. The Trojan horse implementation can involve a mixture of hardware and software manipulations. The Trojan horse is designed such that it can modify, replace, or nullify information that is forwarded to control room operator(s). The intended consequence of the attack would be to have the operator(s) believe that the turbine is operating sufficiently outside of specification so as to warrant a procedural action (e.g., shutdown of the turbine), which would induce a plant trip. The success of this attack depends on the attacker having knowledge of those operator procedures that demand rapid shutdown decisions. In general, this information is known to experienced nuclear power plant designers, integrators, and operators, and is thus likely to be readily available to attack designers. Furthermore, the attack demands that the Trojan horse be designed to circumvent the controller's security processes and internal voting process for fault tolerance. This could occur if the redundant elements all contain the same technology infection across all channels, which could occur if the Trojan horse were to be embedded by a common supplier of the hardware and/or software for the turbine controller.

Figure 4 provides a diagrammatic representation of a more aggressive attack, based on a more sophisticated Trojan horse than the one required for the attack represented earlier. The intended consequence of the attack would be to create severe damage to the turbine, as well as to trip the reactor. For this case, the Trojan horse sends disruptive control signals to damage the turbine (e.g., misguided oil or temperature control commands), as well as manipulating attack-revealing information being sent to the control room operators. This would need to include sensor and feedback information regarding turbine status. The success of the attack depends on the assumption that the control room operator(s) would not be able to recognize, in a short period (1's of seconds), through independent means of the normal data paths that the turbine must be brought to a fail safe stop.

4.3 System-Aware Cyber Security Architecture

To address attacks of the nature described in section 4.2, a System-Aware security solution is embedded within the infected turbine control system. Figure 5 represents the turbine control system integrated with the four System-Aware Cyber Security protection services outlined in section 3.

4.3.1 Diversely Implemented Redundancy of Subsystems and Components

For each of the examples in section 4.2, an attacker must be able to embed a Trojan horse in the turbine control system. As outlined in section 3.1, this threat can be addressed by the turbine system integrator selecting diverse vendors to supply multiple controllers; the underlying principle being that attackers would find it increasingly difficult to design and embed coordinated Trojan horses into diverse controllers. In addition, the overall control system can be designed to integrate the outputs from these diverse controllers through a cyber security sensitive intelligent voter.

Figure 5 presents one possible implementation. This includes three controllers in the system architecture and an intelligent voting process to rapidly disable a differentiated controller from carrying out its turbine actuation functions. During turbine operation, should a specific controller be discovered as a potential source of a cyber attack, the remaining controllers could continue to operate utilizing a new logic to assure that the two controllers' outputs are compatible. If the outputs are not compatible, because of the ambiguity regarding which is the flawed controller, the turbine would need to be shut down, and the reactor would need to be tripped. Figure 5 also includes two communication translator subsystems providing communications between elements of the secured turbine control system. The purpose of these subsystems is to perform the necessary protocol translations that enable communications from the diverse controllers to be integrated for voting or other system related purposes. Currently, a variety of products exist to perform integration across different communication buses; products of this sort would be required as part of the overall turbine control system as a response to introducing diversity for cyber security. Finally, the voting system itself can be diversely and redundantly implemented in order to protect the voting system from attack. Returning to Figure 2, it is useful to note that an alternate security architecture from the architecture in Figure 5, but one that also builds on diversity, would incorporate diversity within a particular vendor's controller. While it is likely that organizing for a single vendor with diverse components would provide less difficulty to an attacker, an assessment of the cost and risk reduction differences would be required in order to draw a conclusion regarding architecture selection.

4.3.2 Configuration Hopping

As shown in Figure 5, configuration hopping would be instantiated within the configuration hopping manager subsystem. For this security service the turbine control system dynamically switches among controllers, during normal operations, so as to time-vary which controller would actually be sending its actuation outputs to the turbine at a given time. The hopping would occur in a randomized manner so as to create uncertainty for attackers about when the infected controller might actually be designated to control the turbine. As a result, during a given time interval, the remaining two controllers would only be used for voting purposes until called upon by the configuration hopping manager to take physical control of the turbine. The hopping

process would need to be designed to assure that none of the controllers had physical control of the turbine for a sufficient time so as to be able to cause significant damage (i.e., 1's of seconds). When combined with diversity, configuration hopping would serve to complicate matters for the attacker regarding the best time to initiate a desired attack, because the infected controller might not ever have the opportunity to actually control the turbine post-attack initiation. This complication occurs because the infected controller is likely not to be in control of the turbine at the time of attack initiation; enabling the diversity voting process, which is continuously searching for anomalies, to enable the security system to take action(s)—after attack initiation—to prevent the infected controller from taking control of the turbine. Note that this approach treats an infected controller in much the same way as a failed controller. This means that the additional costs for this form of protection are mitigated.

4.3.3 Data Consistency Checking

The application of data consistency checking, as suggested in section 4.2, is that while an in-progress cyber attack can be successfully masked by manipulating the most prominent data, it would be extraordinarily difficult for an attacker to adjust all system data that might give indication of the attack in progress. The use of voting, as described in section 4.3.1, is the most direct application of data consistency checking. However, there could be checks across a broader set of system components that would also be revealing, and could become part of the dynamic management of system configuration. For example, in a nuclear power plant there are numerous measurements that occur for safety and operational reasons. It is possible to combine these measurements with turbine system measurements to reveal discontinuities, contributing to a more robust basis for signaling a cyber attack. The application of System-Aware security to a nuclear power plant would need to include an exploration of this opportunity as part of the system design process.

4.3.4 Tactical and Strategic Forensics

Should a subsystem in the turbine control system fail (i.e. be voted out of operation), the question would remain for the system owner as to whether the differentiation from its diversified peers was purposefully caused as part of a cyber attack, or was the result of a natural fault. This question must be addressed for cases where the consequences are limited (e.g., a component being voted out of service with no impact on turbine operation) as well as for cases where the consequences are much more significant (e.g., the turbine being damaged). There are three time constants associated with receiving answers to this question. One time constant is relatively short, and relates to the processes for immediate restoration of the failed component(s) or subsystem. This short period is referred to as tactical. Another time constant relates to management strategies for sustaining operations at other nuclear power plants that are using the same equipment as the potentially compromised system. A third time constant relates to issues surrounding the supplier and the use of equipment containing components from the supplier in question. These latter two cases are referred to as strategic.

This paper focuses on the tactical case addressing a turbine control system. For this case, one can implement specific hardware and software forensic capabilities that can be brought to bear quickly for restoral decisions. Normally, in current operations, a technician would find the part of the system that failed and, when needed, would take immediate action(s) to return the system to its normal operational mode. However, in the event that the failure was actually caused by a cyber attack, such action(s) might not be desirable. Instead, one might choose to restore the system by using one of the other vendors' controllers. This would reduce diversity, but would increase security during a period required for deeper analysis. In any event, one can develop a logical decision process for forensic analysis that could work as follows: (1) if a subsystem is voted out of operation, and (2) if at the time of the vote that component was producing signals that were within the normal specifications for equipment performance (i.e., the component was not "broken"), and (3) if the voted out signals had been applied to the turbine at the time of the vote they would have caused significant consequences, then a forensic analysis supporting the hypothesis of a cyber attack would be considered as conceivable. Specific forensic tools could provide additional information confirming that the software in operation at the time of the failure was identical to the software that was believed to be in operation. Another forensic tool could look for external signaling from an attacker to initiate the attack. This could potentially be accomplished by embedding a radio spectrum frequency analyzer into the hardware for the controller for the purpose of observing wireless communications signals. It may also be possible to discover data insertions into the protected hardware through the various data entry ports that are part of the hardware (e.g., serial ports, USB ports). Regardless of the mechanism, forensic tools can provide a useful contribution by adding confirming evidence in a situation that has the potential for being a cyber attack.

4.4 Solution Assessment

This section summarizes the benefits and cost of the overall System-Aware security solution for turbine control during pre-attack, trans-attack, and post-attack phases.

4.4.1 Pre-Attack

The diverse redundancy portion of the System-Aware security architecture forces an attacker to create a network involving a larger number of suppliers than otherwise would be called for. It also requires the attacker to learn about more subsystem designs than otherwise would be necessary in order to design a successful attack. The configuration hopping portion of the solution requires the attacker to consider the impact of timing on the attack. This requires the attacker to possess greater knowledge about the influence of system dynamics on the technique employed for attack. It also points to the need for managing the attack initiation with greater precision than otherwise called for. The data consistency checking across broad segments of the entire system forces the attacker to consider more sophisticated data manipulation designs.

Finally, the tactical forensic analysis for restoration support raises concerns about being detected even when the cyber attack fails. Together these elements could serve as a significant deterrent to attacks being directed at, in this case, the turbine control system. In addition to the cyber security benefits, the power system would be more reliable due to the added redundancy and diversity.

Negative consequences include an increase in system costs (one-time and life cycle), added complexity in system validation, and the possible need for more skilled technical support staff at the plant. In addition, design and evaluation for achieving "bumpless" performance for a turbine control system that incorporates diverse controllers will require design and evaluation activities before incorporating this approach.

4.4.2 Trans-Attack

As described in section 3, the various elements of the System-Aware security architecture together increase the likelihood of identifying and disabling an attack. The suggested architecture can also provide the basis for post-attack forensics to play a role, by time stamping voting results and storing information that can help identify when a cyber attack has occurred.

Negative consequences could include the complications of keeping plant operators sufficiently informed about the system configuration and security outputs both in normal operation and during a possible attack.

4.4.3 Post-Attack

The application of tactical forensic information would guide system restoration actions so as to avoid a follow up attack. It would also influence strategic decision-making regarding the supplier and other on-going use of the problematic equipment. Negative consequences include the development of confident methods for tactical forensic analysis that can be used by technicians at the plant, and could require significant training time and effort.

5 Metrics

Section 3 discussed how System-Aware Cyber Security can address the threats of infections embedded in mission critical systems. This was illustrated in section 4 through the use of an example showing how System-Aware security services could be used to mitigate two specific threats in a nuclear power plant turbine control system. This included a discussion of how these security services would increase the difficulty to an adversary and provided a high-level assessment of the benefits and cost. However, as detailed in section 3.1, and illustrated in section 4, multiple implementations of the

System-Aware security services could be candidates for addressing these threats. Thus, there is a call for security analysis methodologies that are able to compare alternative system security architectures accounting for the selection and integration of security services, as well as the details of specific service designs. The desired methodology must include an array of methodological elements for assessing the level of security afforded by a given System-Aware security architecture (e.g., how much more difficult has this solution made an adversary's task and how rapidly can system restoration be accomplished), as well as accounting for the collateral impacts on the system as a whole (e.g., performance and cost); identifying those system functions that warrant the most protection from a risk perspective; evaluating, from a security perspective, the hardware/software implementation of the System-Aware security architecture (e.g. avoidance of buffer overflow opportunities in the implementing software); and evaluating the life cycle management plan for the System-Aware security architecture, including approaches for responding to discovered security solution design flaws and to overall system design enhancements that occur over the life cycle. With regard to risk based security protection, standards exist for conducting risk based security analysis [ISO/IEC, 2009; Stoneburner, Goguen, and Feringa, 2002]. Similarly, industrial methodologies exist for software implementation and software patching over the life-cycle of a security solution [Howard, 2002; Stackpole and Hanrion, 2007]. However, to completely address the four elements outlined above, the System-Aware security approach requires a supporting methodology for the assessment of the level of security potentially afforded by a given security solution. This section presents an initial outline for a scoring methodology to assess and compare System-Aware security architectures, building upon existing work in the field of safety.

5.1 BACKGROUND

Just as digitization has created new challenges in the field of cyber security, it has also created new challenges in the field of safety systems. One such challenge is causally related failures of redundant or separate equipment. When systems were analog, these common-cause failures (CCFs) were typically caused by slow moving processes such as corrosion. However, as systems increasingly became digital, software design flaws and bugs arose as a new source for CCFs. The nuclear safety community mitigated the risk posed by this increasing threat by employing multiple types of diverse redundancy, including design diversity, human diversity, and software diversity. This new solution created a call for a methodology for comparing alternative designs and assessing the level of CCF risk mitigation provided by a given design.

One methodological solution is employed by the Nuclear Regulatory Commission (NRC), as documented in NUREG/CR-6303. This methodology provides system designers and implementers with a guide to achieve resilience to CCFs through use of a procedure related to mitigation of consequences. This is achieved by assessing the amount of diversity offered by a particular system's design—based on the principal that more diversity in a system results in less susceptibility to CCFs. A system's diversity is

evaluated through the application of a weighted rank ordering process that utilizes a wide range of criteria, such as differing technologies, similar technologies within different architectures, and different manufacturers of fundamentally different designs. A weighting scheme has been determined based upon assumptions and principals derived from designers' experiences with avoiding CCFs. While the method is not supported by an underlying mathematical theory, the results permit the NRC and system designers to engage in a constructive dialogue regarding the attention paid in a specific design regarding the avoidance of CCFs.

5.2 Unique Challenge Posed by Cyber Security

As shown in section 3.1, security solutions must address cyber attacks that can concurrently exploit common redundant components, leading to diverse redundancy as one of the security services employed by System-Aware Cyber Security. Recognizing the employment of diverse redundancy utilized by safety engineers for addressing CCFs, one can look to the significant efforts of the safety community as a source for an initial scoring methodology that can be enriched for cyber security application. However, there are several critical challenges unique to cyber security that requires a CCF-based foundation for scoring to be enriched before it can be effective. First, while CCFs can be addressed solely through diverse redundancy, as indicated earlier in this paper, security solutions must include additional solution components, that go beyond the application of diversity, in order to fulfill its functions. Second, unlike CCF solutions, cyber security solutions attempt to deter, defend, and restore a system against an intelligent adversary exploiting available vulnerabilities, including the capability to assess the cause of failure indeed being a cyber attack. Finally, a variety of security services, including diverse redundancy, can be integrated into solutions, thereby requiring a scoring methodology that establishes criteria for assessing and comparing the value contributed by the individual elements of the broader solution space.

5.3 Outline for a Possible Scoring Model

Using NUREG/CR-6303 (outlined in section 5.1) as a starting point, and given the unique challenges outlined in section 5.2, this section provides an outline of a methodology for developing scores for comparing and assessing System-Aware security architectures. This scoring methodology involves addressing three key factors: (1) for each of the individual security services within a System-Aware security architecture, identifying the potential contribution and its importance to the overall security being offered, (2) determining the potential effectiveness of each security service within a particular System-Aware security architecture, and (3) evaluating the cost and collateral impacts of the solution services on the system's normal operations. Multiple methods can be utilized to evaluate the resulting value of an integrated set of security services combined into a System-Aware architecture. For the purposes of this paper, a linear model is introduced for combining the values of individual security services into a resulting score. One would anticipate that future work would develop alternative

methods for evaluating combined value, such as emphasizing the deficiencies of the weakest contributors to security.

5.3.1 Identifying the Security Contribution of Individual System-Aware Security Services

Every System-Aware security architecture is composed of an integrated set of security services. Each of these services enhances the security of the system being protected by (1) deterring the attacker (pre-attack); (2) identifying, isolating, and preventing malicious attacks (trans-attack); and/or (3) aiding in the restoration of the system to a non-compromised state (post-attack). For example, in the turbine control example discussed in section 4, tactical forensics is a post-attack security service that contributes to restoration and also deters an attacker by making it more likely the attacker will be caught. Diversity is a pre-attack and post-attack service that deters an attacker by making the attack more difficult to design and execute, and also aids in restoration through the employment of diverse elements that have not been compromised. The desired methodology should include the classification of every security service in a given System-Aware security implementation in terms of its contribution to security and the stage(s) of attack for which it is designed to operate. This enables system owners and operators to analyze the tradeoff between the type of security offered by a given System-Aware security architecture, and the impacts this has on the system. For example, a system owner concerned about interfering with the normal performance of their system may select a given System-Aware security architecture that provides a significant amount of low interfering post-attack capabilities and no high interfering pre-attack and trans-attack capabilities. In addition to providing restoration, this solution provides a higher likelihood that an attacker will be caught and potentially a significant amount of deterrence, while maintaining a minimum impact on the system's performance. In contrast, the architects and operators of a more mission critical system may choose an implementation that offers a significant number of pre-attack, trans-attack, and restoration services to ensure maximum availability, even if this somewhat degrades or increases the cost of the normal operations of the system.

5.3.2 Assessing the Potential Effectiveness of Individual Security Services

As shown above, it is possible to evaluate a given System-Aware security architecture in terms of the contribution to security offered by a particular security service, as well as which phases of the attack it is effective. However, there still remains the issue of assessing the efficacy of a particular architecture. One possible means for achieving this objective proceeds as follows. First, it would be necessary to determine the mission critical functions to be protected in a particular system. For the purposes of this paper, it is assumed that an existing security risk analysis method would be employed. These analyses would be conducted by parties accountable for achieving system mission requirements and would be supported by red teams providing visibility into possible attack methods. The System-Aware security architects would utilize results regarding

missions to be protected combined with system design information to allocate and integrate security services into desired solution alternatives. This stage of the evaluation provides the basis for assessing how well a particular System-Aware security architecture addresses the parts of the system identified as critical.

However, it is not enough to ensure that a particular System-Aware security architecture would provide some amount of defense; it is also necessary to evaluate the potential efficacy of particular solutions. For example, as described in section 3.2, configuration hopping forces an attacker to operate within a given time interval. This could add value to security whether or not other security services are employed by a particular System-Aware security architecture. However, the efficacy of configuration hopping in the absence of other security services depends upon the predicted attack execution difficulties imposed on an adversary. If it is believed that a principal difficulty to an adversary is the exact timing of the attack, then implementation of configuration hopping as an isolated security service might prove to be highly effective. However, if triggering the potential attack(s) is predicted to be simple, and does not include stringent timing requirements, then configuration hopping will be less effective and contribute most to system restoration functions.

5.3.3 IMPACTS

It is generally expected that security solutions will increase the implementation and life cycle cost of a system. It is also expected that security solutions will require additional resources—CPU, memory, bandwidth, etc.—and in some cases could potentially degrade the overall performance of the system to be protected. In addition, since System-Aware security architectures provide solutions embedded into the system to be protected, they may also introduce intricate collateral impacts on the system. For example, as discussed in section 4.3.2, a potential solution to protect a nuclear power plant turbine control system involved dynamically switching control between a diverse set of vendor controllers. This solution improves the security of the system by making it harder for an adversary to gain control of the system, but also introduces the need to make the handoff between controllers bumpless in order to ensure the proper functioning of the turbine. Another example of the possible collateral impacts introduced by a System-Aware security architecture can be illustrated through the usage of configuration hopping between communication switches. As in the case of the turbine controller, hopping across diverse communication switches reduces the possibility that an attacker will gain control of the system; however, it can also result in information loss if the switches being hopped are not synchronized. This loss of information is a collateral impact that requires additional analysis—how much information is expected to be lost and how important is that information—and a solution—should the system owners and operators just accept the loss of information or implement algorithms to ensure switch synchronization.

Collateral damage need not be purely technical. In the turbine control example tactical forensics were utilized to help technicians distinguish faults from malicious attacks (see section 4.3.4). This not only requires new tools and techniques, but also requires the introduction of new policies and procedures. This in turn, introduces the need for additional training for technicians on how to properly utilize these new tools, techniques, policies, and procedures to distinguish faults from malicious attacks.

5.3.4 Architectural Scoring Framework

Figure 6 is a possible representation of what an architectural scoring framework containing the elements outlined in section 5.3 would produce: a table where each row contains a score regarding the value of a security service and each column a specific value factor. As outlined in section 3, each of these security services enhances the overall security of the system, while also affecting the system's performance and cost to a varying degree. These effects are represented in the table as value factors. For example, **Error! Reference source not found.**Figure 6 utilizes four System-Aware security services: diversity, configuration hopping, data consistency checking, and tactical forensics. Each of these services can enhance the system's security through increased deterrence, greater real-time defense, and/or improved restoration capabilities. In contrast, each security service may negatively affect a system through collateral impacts, increased implementation cost, and/or increased life cycle cost. Finally, each service may provide positive collateral impacts, such as reduced CCFs and improved reliability.

To assess how security services affect value factors, a security assurance level, s, is assigned to every security service, i, based upon its contribution to a given value factor, j, yielding specific service assurance scores, s_{ij}. Recognizing that System-Aware security architectures offer security values and unavoidable disvalues (e.g. increased cost), scores can be provided related to system impacts as well as security. Larger scores can be used to represent greater security value added or less collateral disvalue added, depending on the value factor being scored. It is noted that alternative scoring means could be utilized. For example, these scores could be represented as negative values. The assurance score is a discrete value selected from a range (0 to M inclusive) determined by a desired level of granularity. For example, an assurance level s could take on the value 0 or 1. This has the benefit of making a given solution easy to evaluate, but it would only be possible to compare alternative solutions based upon whether or not a given benefit or cost was offered by a given solution. However, if s could be a varying value, for example between 0 and 5, then it would be possible to compare the level of security offered by alternative solutions. This would require a more complete analysis to assign a given assurance score. For example, in the turbine control example illustrated in section 4, there were two possible ways to provide diversity; one through controllers purchased from separate vendors and one through diverse components within a single vendor's controller. If s could only take on a value of 0 or 1, then both solutions would result in the same score, since the diversity affects the same value factors. However, if s

could be any value from 0 to 5, the two solutions could score differently as the level of assurance provided by the differing diversity methods could be different.

Assuming a method for deriving individual scores, s_{ij}, several pieces of information can be derived from this scoring methodology. First, it is possible to derive a single security score for a given solution, $\sum\sum s_{ij}$, and compare it to a theoretical maximum score ($\sum\sum s_{ij} \le i * j * M$). Second, it is possible to evaluate the strengths and shortfalls of a given security solution. For example, it is possible to evaluate whether a given solution is more effective in addressing real-time defense or restoration. Finally, recognizing that different combinations of value and disvalue scores require a multi-objective solution selection approach, it is possible to compare scores across alternative security solutions addressing a specific security need.

In addition to providing a method for scoring and evaluating multiple System-Aware architectural designs for securing a given system, the methodology can be augmented to also recognize that every system owner and operator may have a different perspective on the importance of different factors when securing their system. For example, some owners and operators may desire solutions that minimize the collateral impacts to the system, while others may seek solutions that maximize the security. To take these differences into account, system owners and operators can adjust scored System-Aware architectures to account for their individual assessments of which factors are most important by providing a set of relative value weights. A relative value weight, k, can be assigned to each value factor, j, such that $\sum k_j = 1$. Figure 7 **Error! Reference source not found.**provides a representation of a scored architecture filtered by a set of relative weights. As illustrated in the figure, it is still possible to derive a single security score for a given solution, $\sum\sum k_j s_{ij}$, and compare it to a theoretical maximum score ($\sum\sum k_j s_{ij} \le i * M$). Furthermore, it is still possible to evaluate the strengths and shortfalls of a given security solution.

To illustrate how such a scoring methodology could be used to evaluate a given System-Aware security solution, the methodology is utilized to evaluate the example illustrated in section 4. For this example, several assumptions were made. First, s_{ij} can be assigned discrete scores between 0 and 5 inclusive (5 representing the best score and 0 representing the worst score). Second, system owners are concerned about six value factors: deterrence, real-time defense, restoration, collateral system impacts, implementation cost, and life-cycle cost. Third, four security services are available for utilization: diversity, configuration hopping, data consistency checking, and tactical forensics. Finally, the scored system was filtered by a set of weighting factors, k_j, which, for this example, were selected by the authors, weighted to emphasize security—deterrence (0.30), real-time defense (0.20), and restoration (0.10)—over cost—collateral

system impacts (0.20), implementation cost (0.05), and life cycle cost (0.15). Finally, given these assumptions, the maximum possible score is 20 ($\sum\sum k_j s_{ij} \le i * M$).

In the absence of a needed methodology for deriving assurance scores, s_{ij} , the assurance scores were chosen by the authors based on rationale provided in section 4. For example, tactical forensics was noted to be a security service that provided significant capabilities to aid in the proper restoration of a given system. Assuming, that the specific suggested tactical forensic solution is judged as providing a significant contribution towards restoration, the maximum score, 5 is assigned. Furthermore, tactical forensics did provide some deterrence by increasing the likelihood that an attacker would be caught. For the purpose of this example, this leads to a score of a 3 out 5 related to deterrence. On the other hand, tactical forensics does require that every failure go through additional testing before a given component can be replaced. This would results in increased life cycle cost and hence tactical forensics is assigned a low value of 2 out 5 for the life cycle cost. This same process was repeated to provide a complete table of assurance levels. Finally, the authors created a set of relative weights emphasizing the security of the system over the cost. Overall, utilizing the judgments of the authors for scoring and the authors' preference for security over cost, the security architecture scored an 11.5 out of a possible 20. Furthermore, the architecture was evaluated as being strongest in the area of restoration and weakest in the area of life cycle cost.

5.3.5 Structured Arguments for Architectural Scoring

Section 5.3.4 outlined a possible scoring framework that could be utilized to evaluate System-Aware security architectures. Key to the framework was the ability to assign assurance scores to each security service based upon its contribution to a given value factor; however, no formal method for assurance score evaluation was provided. As part of introducing a formal method for assigning assurance scores for cyber security, one must recognize the need to rely on the judgments of experts in providing supporting rationale. This includes making judgments regarding the deterrence values for solutions, and the potential for adversaries to discover alternate attacks that could circumvent the defense. One possible approach for determining assurance scores could be derived from the use of Goal Structuring Notation (GSN) [Kelly, 2004] to communicate logically structured arguments in support of security claims in a clear and repeatable manner, utilizing rigorous evidence where it exists. Such an approach has been utilized by the UK Ministry of Defence [Menon, Hawkins, and McDermid, 2009; MoD, 2007] and the Food and Drug Administration (FDA) [FDA, 2009; FDA, 2010] for safety case evaluation. In the case of the FDA, safety cases (i.e. structured arguments) are starting to be used for the purpose of evaluating the safety of medical devices. A specific example is the case of approving new infusion pumps before they are certified for public use [FDA, 2010]. New infusion pumps may possess different technological characteristics—new implementations of software, hardware, and/or changes in

material, design, and/or performance—that are both different from the existing approved infusion pumps and existing tools and methods of development, but are intended to provide similar functionality. In addition, in order to demonstrate the safety of these pumps, a large number of claims must be substantiated by a significant body of evidence and numerous arguments. This can result in a complex web of claims, arguments, sub-claims, and evidence that can potentially obscure relationships and makes the overall safety of the medical device difficult to assess. To address these issues, the FDA has called for manufacturers of medical devices to present safety cases that demonstrate that their devices meet certain claims about safety in a clear, traceable, structured manner. Safety cases force manufacturers to provide their reasoning at a level of granularity that clearly separates claims, from arguments, and supporting evidence. Furthermore, properly structured safety arguments make the underlying context and relationships between claims, arguments, and evidence explicit. The UK Ministry of Defense has also utilized safety cases for submarine propulsion system and air traffic management systems safety justifications.

Utilization of the GSN for evaluating System-Aware security architectures is based on the principle that the security architects should have a rational conceptual basis for their architectures (i.e. structured argument), and furthermore, these architectures should address a vast set of possible future attacks involving situations that have yet to occur, thus negating experimental validation. Where possible and worthwhile, solution architects should be required to gather or develop existing evidence to support their claims. One would anticipate that architects would rely upon support from expert testimony, analytical assessments, experimental data, and historical information. The set of claims, logical arguments, and evidence, could provide a basis for determining each of the assurance scores, s_{ij} .

It is important to note that in order to develop security claims, formulate rigorous arguments, and gather available evidence would require a scoring team possessing a variety of skills. For example, the skills required to determine scores related to the level of deterrence are different from those required for scoring restoration. To score deterrence, one would need to employ the experience and skills of a cyber security red team, as they would be most capable of providing the knowledge and perspective necessary to assert that a given solution would make attackers more hesitant to attempt an attack. However, a red team would not necessarily provide the skills and knowledge related to restoring a system. Rather, restoration assessment would likely be best served by a team of system architects and forensic analysts. Forensic analysts would likely know what tools exist or could be developed for obtaining information that would reveal a cyber attack in a timely manner. System architects could apply their knowledge of system architecture toward the evaluation of security solutions that could restore the system to a non-compromised state, and the amount of time that this would likely take.

It is recognized that the application of the GSN would not necessarily provide repeatability of results from one scoring team to another. Differences would arise

regarding the claims, the corresponding arguments, and the supporting evidence. In addition, differences would arise regarding the translation of results into numeric scores. An approach for addressing these differences is to parallel what is occurring in the regulation of safety related systems. In the case of security scoring, the evaluation of a given system would rely on the owners and operators to supply an appropriate context, including an available risk analysis for the system being protected and scoring guidelines. Assuming that such an approach were accepted, one would expect the safety certification and security communities to share experiences and improve the methodologies surrounding the employment of GSNs for decision making.

6 CONCLUSION

This paper builds upon the authors' previous work that outlines a vision for System-Aware Cyber Security, provides an application example focusing on addressing the threat of cyber attacks against nuclear power plants, and outlines an initial vision for a security analysis framework to compare alternative security architectures. It describes four example security services and illustrates how each of these services can be utilized to enhance the security of a system. Through the use of a nuclear power plant turbine control system example, it is shown how System-Aware Cyber Security can enhance pre-attack, trans-attack, and post-attack security by deterring attackers, preventing damage, isolating compromised components and enabling restoration to a non-compromised state. In addition, a structured argument methodology utilizing GSN is suggested as a means for deriving assurance scores for specific architectures.

Future work to further develop the System-Aware Cyber Security methodology includes, (1) exploring a broader set of applications for System-Aware Cyber Security (2) expanding the set of security services and service integration patterns based upon the expanded set of applications, (3) further developing the suggested System-Aware Cyber Security scoring framework, and (4) further exploration of the utilization of structured arguments using GSN as a method for deriving specific assurance scores for suggested architectures.

APPENDICES

APPENDIX A: REFERENCES

D. Albright, P. Brannan, and C. Walrond, Did stuxnet take out 1,000 centrifuges, Institute of Science and International Security, 2010.

J. L. Bayuk and B. M. Horowitz, An architectural systems engineering methodology for addressing cyber security, Systems Engineering 14 (2011), 294-304.

J. Cai, V. Yegneswaran, C. Alfeld, and P. Barford, Honeygames: a game theoretic approach to defending network monitors, University of Wisconsin-Madison, Technical Report 1577, 2006.

M. R. Clarkson, S. Chong, A. C. Myers, Civitas: toward a secure voting system, IEEE Symposium on Security and Privacy, 2008, pp. 354-368.

Defense Science Board, High performance microchip supply, Department of Defense, Washington, DC, 2005.

DoD, The Under Secretary of Defense for Acquisition Technology and Logistics and The Assistant Secretary of Defense for Networks and Information Integration DoD Chief Information Officer, Report on trusted defense systems, Department of Defense, Washington, DC, 2009.

N. Falliere, L. O. Murchu, and E. Chien, W32.Stuxnet Dossier, Symantec, 2011.

FDA, Guidance for industry: evidence-based review system for scientific evaluation of health claims, U.S. Department of Health and Human Services, Silver Spring, MD, 2009.

FDA, Guidance for industry and FDA staff: total product life cycle: infusion pump – premarket notification [510(k)] submissions, U.S. Department of Health and Human Services, Silver Spring, MD, 2010.

I. N. Fovino, M. Masera, and A. D. Cian, Integrating cyber attacks within fault trees, Reliability Engineering & System Safety 94 (2009), 1394-1402.

A. Fujioka and T. Okamoto, A practical secret voting scheme for large scale elections, Advances in Cryptology – AUSCRYPT '92, 1992, pp. 244-251.

M. Howard, Writing secure code, 2nd edition, Microsoft Press, Seattle, WA, 2002.

ISO/IEC (International Organization for Standardization/International Electrotechnical Commission), Risk management - risk assessment techniques, ISO/IEC 31010, Geneva, Switzerland, 2009.

R. A. Jones and B. M. Horowitz, System-Aware cyber security, itng, 2011 Eighth International Conference on Information Technology: New Generations, 2011, pp. 914-917.

T. P. Kelly and R. A. Weaver. The goal structuring notation – a safety argument notation, Proceedings of the Dependable Systems and Networks Workshop on Assurance Cases, 2004.

K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, Experimental security analysis of a modern automobile, IEEE Symposium on Security and Privacy in Oakland, 2010, pp. 447-462.

C. Menon, R. Hawkins, and J. McDermid, Defence standard 00-56 issue 4: towards evidence-based safety standards, Seventeenth Safety-Critical Systems Symposium, 2009, pp. 223-243.

MoD, Defence standard 00-56 issue 4: safety management requirements for defence systems, UK Ministry of Defence, 2007.

M. G. Noblett, M. M. Pollitt, and L. A. Presely, Recovering and examining computer forensic evidence, Forensics Science Communications 2 (2000).

B. Stackpole and P. Hanrion, Software deployment, updating, and patching, CRC Press, Boca Raton, FL, 2007.

G. Stoneburner, A. Goguen, and A. Feringa, Risk management guide for information technology systems, NIST (National Institute of Standards and Technology), Gaithersburg, MD, Special Publication 800-03, 2002.

US Comptroller General, Report to Chairman, Committee on Government Operations, House of Representatives of the United States, Washington, DC, 1981.

L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, A break in the clouds towards a cloud definition, SIGCOMM Computer Communication Review 39 (2009), 137-150.

F. Webber, P. P. Pal, M. Atighetchi, C. Jones, and P. Rubel, Applications that participate in their own defense (APOD), BBN Technologies, Cambridge, MA, 2003.

W. A. Wulf and A. K. Jones, Reflections on cyber security, Science Magazine, vol. 326, 2009, pp. 943-944.