

# Validating Cyber Security Requirements: A Case Study

Robert K. Abercrombie  
 Oak Ridge National Laboratory  
 abercrombier@ornl.gov

Frederick T. Sheldon  
 Oak Ridge National Laboratory  
 sheldonft@ornl.gov

Ali Mili  
 New Jersey Institute of Technology  
 ali.mili@njit.edu

## Abstract

*Vulnerabilities in a system may have widely varying impacts on system security. In practice, security should not be defined as the absence of vulnerabilities. In practice, security should not be quantified by the number of vulnerabilities. Security should be managed by pursuing a policy that leads us first to the highest impact vulnerabilities. In light of these observations, we argue in favor of shifting our focus from vulnerability avoidance/removal to measurable security attributes. To this effect, we recommend a logic be used for system security, which captures/presents security properties in quantifiable, verifiable, measurable terms – so that it is possible to reason about security in terms of its observable/perceptible effects rather than its hypothesized causes. This approach is orthogonal to existing techniques for vulnerability avoidance, removal, detection, and recovery, in the sense that it provides a means to assess, quantify, and combine these techniques.*

## 1. Introduction

Vulnerabilities in a system may have widely varying impacts on system security. In fairness, the variance may be wider for reliability than for security, because in the case of malicious security violations high impact vulnerabilities may be more attractive targets than lower impact vulnerabilities. In this way, wide variances are quite plausible, in light of our analysis of the analogies between reliability and security. To the extent that they exist, these variances have broad impacts on security management. In practice, security should not be defined as the absence of vulnerabilities, no more than reliability is defined as the absence of faults, because low impact vulnerabilities do not affect security in a meaningful way. In practice therefore, security should not be

quantified by the number of vulnerabilities because the number of vulnerabilities means very different things depending on whether the vulnerabilities are of low or high impact. Measuring reliability by the number of faults per KLOC (Kilo Lines of Code) has long since been discredited, for similar reasons. Security cannot be achieved by focusing on vulnerabilities, as we have no way to tell whether a given vulnerability has low (1) or high (50) impact on security [1]. Rather, security should be managed by pursuing a policy that leads us to the highest impact vulnerabilities first (a similar approach to usage pattern testing). In light of these observations, we argue in favor of shifting our focus from vulnerability avoidance/removal to measurable security attributes. To this effect, we recommend a logic used for system security, which captures/presents security properties in quantifiable, verifiable, measurable terms —so that it is possible to reason about security in terms of its observable/perceptible effects rather than its hypothesized causes [2]. This approach is orthogonal to existing techniques for vulnerability avoidance, removal, detection, and recovery, in the sense that it provides a means to assess, quantify, and combine these techniques.

To illustrate this, in Section 2, we concentrate on a real world system, the Weigh-in-Motion Reach Back Capability (WIM-RBC) through the viewpoint of two real-world use cases (the creator of the data and the user of the data) as a way to actually validate (partial not exhaustive) the conceptual framework of the Cyber Security Econometrics System (CSES). These concepts are discussed in Section 3, which leads us to a better understanding of general security issues via the context of the users of real world systems. Additionally, in Section 3, a brief analysis of CSES is provided against the Carnegie Mellon University Software Engineering Institute's (CMU/SEI) Mission Assurance Analysis Protocol (MAAP) [3], an accepted methodology in understanding security issues, which demonstrates where future work is needed. Finally, in Section 4, the importance of this subject matter is summarized.

---

The submitted manuscript has been authored, in part, by a contractor of the U.S. Government under contract DE-AC05-00OR22725. Accordingly, the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

## 2. Towards Security Certification: Validating Cyber Security Requirements

The Federal Information Processing Standards endorse *confidentiality*, *availability*, and *integrity* as the core information security categories (challenges). Many computer security experts include *reliability* and *authentication* to the list of canonical security properties. These standards have proven difficult to achieve.

- *Confidentiality* protects against unauthorized disclosure of information [4]. One of the most serious threats to confidentiality is infiltration. An insider could compromise an end node [5] and plant hostile code.

- *Integrity* protects against unauthorized modification or destruction of information [4]. The integrity of data relies in large part on the quality of the routing network. However, mobile code routing networks are not guaranteed to function. The network such as the one described in Section 2.1 can be in constant flux, could easily be faulty, garble, or drop messages.

- *Availability* protects against disruption of access to or use of information or an information system [4]. Nevertheless, mobile nodes cannot guarantee availability. The communications abilities of mobile nodes are limited by the physical capabilities of their communications devices and therefore nodes can expect to be cut off from the network occasionally.

- *Authentication* is the process of establishing confidence in electronically presented identities [6]. If an adversary can hijack mobile nodes, confidence in identity becomes meaningless. A formerly trustworthy node does not lose any information, knowledge, or capability after falling into adversarial hands yet a hijacked node can also provide any representation or reassurance that the trustworthy node could have provided.

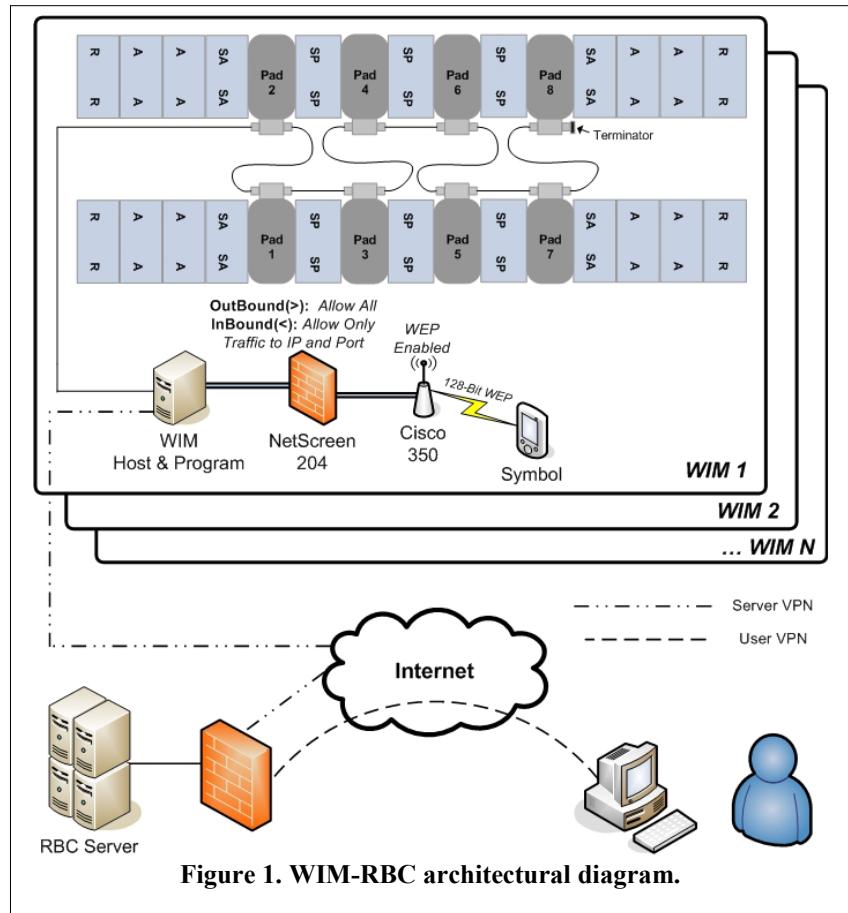
- *Reliability* measures a systems ability to maintain stated performance objectives under stated environmental conditions for a stated period of time. Mobile code runs on small platforms with

inferior batteries, small memories, limited computing power, and faulty software, which presents the essential dilemma for deploying reliable mobile code/platforms.

Important and challenging research hurdles remain for defining novel security primitives that can compellingly validate revolutionary approaches that provide sound, resilient, survivable, and dependable security assurances for real-world deployments. In the following subsections, we outline our approach implemented as the Weigh-In-Motion Reach Back Capability (WIM RBC) within the WIM system, which exemplifies many of the security properties discussed above.

### 2.1. WIM-RBC: Real-World Pervasive Computing Platform

Functionally, the military's portable Weigh-In-Motion (WIM) system provides a means for accurately weighing and measuring the volume vehicles and their cargo. As each vehicle crosses the weigh scales, the WIM determines axle weights and spacing for vehicles, total vehicle/cargo weight, and longitudinal



center of gravity<sup>1</sup>. Structurally, the WIM is a transportable network consisting of wired/wireless devices that are integrated into a distributed computing environment (e.g., civilian and military operations). The WIM RBC (Reach Back Capability) provides a secured, web-enabled, and central data and service repository for configuration and data management processing that is used to continuously analyze and improve the WIM system. The RBC stores and retrieves disparate types of system logistics information in the form of text, data, images, and video using relational SQL-based data sources, flat files, and external Web Services. All of this is readily accessible through a simple Web browser User Interface (UI). Figure 1 provides a high-level architecture of the WIM-RBC<sup>2</sup>.

- *First, WIM nodes collect, process, and stores weigh data,*
- *Second, the nodes transmit weigh and vehicle characteristics securely to the reach back server (RBC), and*
- *Finally, authorized user(s) access current and/or historical weigh and vehicle characteristics for analysis.*

The XML is validated, processed, and committed to the database through the Web Services interface. WIM scale data includes “raw” weight measurement information from each pad for further analysis to identify efficiencies, improvements, and performance correlations. By design, all information concerning WIM operations and equipment, including manufacturer, serial number, model, and specifications are stored in the repository.

In the context of validating cyber security requirements, those requirements can be addressed in terms of three methods (vulnerabilities avoidance, vulnerability neutralization, and exposure limitation) leading to threat assessment and risk analysis. Let us consider the essential steps for threat assessment and risk analysis.

### 2.1.1. Develop a security target<sup>3</sup>.

Much of the security analysis in the literature postulate codes that run on untrusted platforms, platforms that run

---

<sup>1</sup> Individual vehicles can be weighed continuously at low speeds (approximately 3-5 mph) and at intervals of less than one minute.

<sup>2</sup> WIM data accumulated via TAS-Net pushes information to the WIM-RBC over the Internet using eXtensible Markup Language (XML) that includes the schema (format) and contents for: (1) the Deployment Equipment List (DEL), (2) detailed WIM scale data, (3) images (such as jpeg) of vehicles, (4) additional conveyance information, (5) measurement and timing data, (6) weather data, and (7) video as required.

<sup>3</sup> A security target is a set of security requirements and specifications to be used as the basis for evaluation of an information system and its associated resources.

untrusted code, or both [7]. These assumptions, however, do not completely describe many mobile code implementations. In many real-world applications, a limited number of mobile platforms are deployed pre-loaded with a small number of known software packages. Additional software packages could potentially be loaded if needed, but many mobile platforms will never load any additional software. The platforms are of known provenance, and are in the physical custody and control of their owners until they are deployed. Fielded sensors, mobile units given to first responders and mobile battlefield units all fit this description. Accordingly, the security requirements and analysis are substantively different from the analysis in most of the literature. Our approach will formulate security targets using the context of real world deployments.

### 2.1.2. Develop specific approaches to help achieve the security targets.

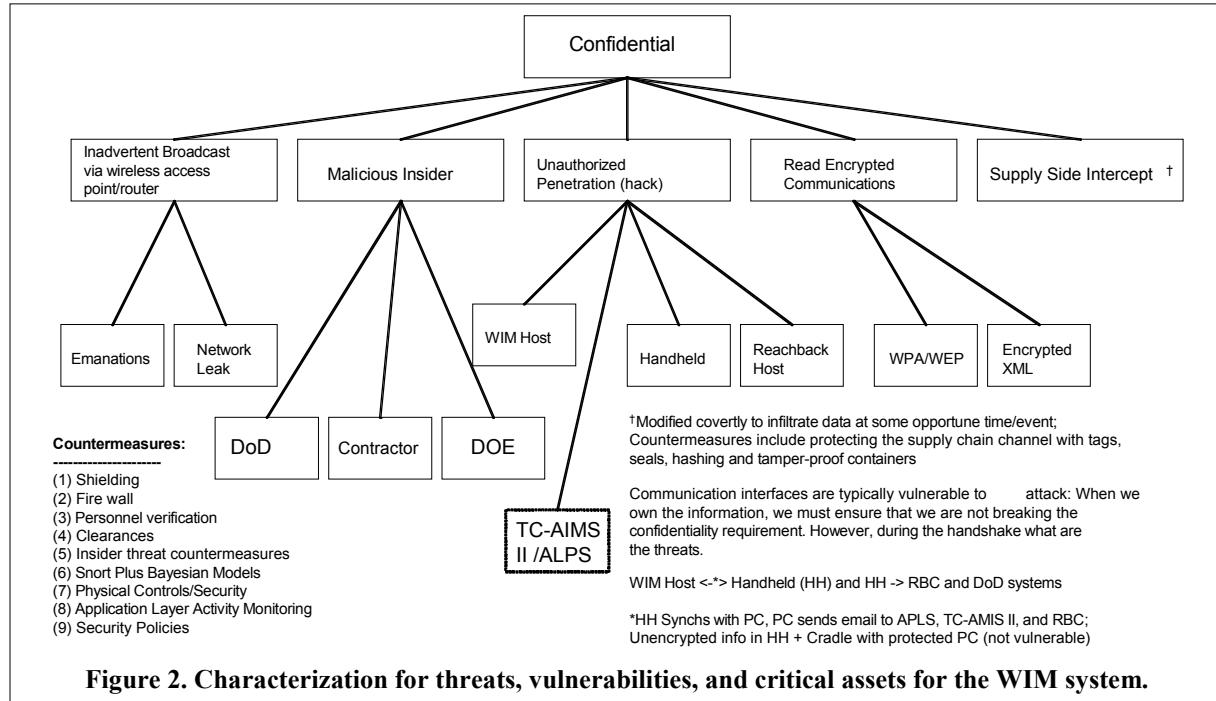
- *Trust gradients:* Consider a network with mobile, unattended sensors. The sensors can be accessed, cloned, or destroyed. The same network could incorporate fixed servers, constantly attended and surrounded by physical security. It is intuitively obvious that the servers are more trustworthy than the unattended sensors. Typical security models, however, find it difficult to distinguish multiple levels of intercommunicating trust<sup>4</sup>. This approach will formally define a gradient of trust, in which some units are trust worthier than others [8].

- *Route tracing.* Since the mobile code units were in the custody and control of friendly forces before their deployment, each unit can be pre-loaded with unique identifiers. These identifiers could be useful for a variety of reasons. For example, using recursive fixed size cryptographic hashes similar to Bloom filters; it is possible to probabilistically record the path taken through the network by each message. This information could reveal man-in-the-middle attacks, as well as identify units that are handling traffic in suspicious ways, such as asking to route traffic that should be routed in a different direction. These probabilistic records also allow calculations using a trust calculus such as the trust gradient noted above.

---

<sup>4</sup> The field of Multi-Level System (MLS) design is similar, as it deals with multiple security levels. However, MLS design is fundamentally different in that such systems are designed to incorporate a small number of discrete levels, with minimal information interchange among levels. Only recently have policy makers considered obtaining technology that would allow information to flow from highly classified networks to unclassified networks.

**2.1.3. Validation Testing.** Evaluation of these novel security mechanisms to understand better how new countermeasure technologies help to strengthen the overall integrity of the critical asset is a key objective. Figure 2 gives a security “hazard” diagram, which



**Figure 2. Characterization for threats, vulnerabilities, and critical assets for the WIM system.**

identifies the critical assets, vulnerabilities/threats, and countermeasures<sup>5</sup>. This diagram is the starting point for applying the CSES conceptual framework in Figure 3. The process involves refining facility requirements or policies into sub-blocks. Each sub-block can be assigned one or more distinct verification mechanisms (e.g., formal verification, expert opinion, test results, and attestation). Strict hierarchy is not enforceable (one program can address multiple needs). Using such an approach we can compose queries that address such questions as expected system performance, best investments, performance guarantees (limit false assurance) and lack of performance guarantees (limit false alarms).

Figure 3 [9] illustrates that when assessing cyber security, we must consider many dimensions. Primarily, these dimensions are focused on either the left or right side of boom (e.g., a significant even such as a break-in). On the left side are preemptive/protective measures including all steps prior to the “system’s” deployment as well as those that are designed to support measures on the right (post deployment). The right side includes damage

assessment and recovery measures. This framework is designed to enable comprehensive exploration of the “likely” consequences of the various trade-offs on both sides. For example, we may (i) identify vulnerabilities and provide options to mitigate those at their earliest

stages before they become more pernicious, (ii) codify the concomitant methodologies and processes that consider the full range of stakes (criticality/assets) and associated (operational) risks, and (iii) manage explicit investments (countermeasures, certification and accreditation (C&A) among the many feasible left side courses of action). Ultimately, as the system evolves the precision (and accuracy) of the assessments will help all aspects from C&A, intrusion avoidance to attribution including such measures as return on investment (ROI) and MFC.

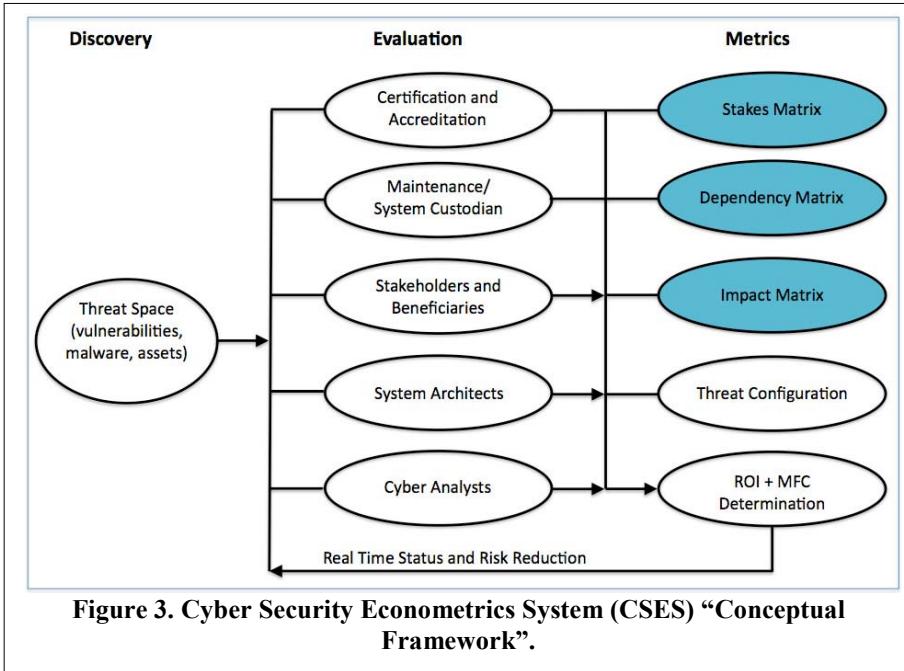
The Cyberspace Security Econometrics System (CSES) offers the following advantages over traditional measurement systems:

1. CSES reflects the variances that exist among different stakeholders of the same system. Different stakeholders, in the WIM example, the WIM operator (creator of vehicle weight and dimensional characteristics) and the WIM analyst (user of data), will typically attach different stakes to the same requirement or service (e.g., a service may be provided by an information technology system or process control system, etc.).
2. For a given stakeholder, CSES reflects the variance that may exist among the stakes one attaches to meeting each requirement. The same

<sup>5</sup> Paradoxically, this diagram exemplifies the very weakness (i.e., defensive premise) we address in Section 3.1 below.

stakeholder may attach different stakes to satisfying different requirements within the overall system specification. As illustrated in Table 1, in the authentic real world WIM use cases, the security requirements for the respective stakeholders consist of WIM data collection, processing, storage, transmission, and access for data analysis in the context of the canonical security properties of Confidentiality, Integrity, Availability, Authentication, and Reliability.

3. For a given compound specification (e.g., combination(s) of commercial off the shelf software and/or hardware), CSES reflects the



**Figure 3. Cyber Security Econometrics System (CSES) “Conceptual Framework”.**

malicious insider, unauthorized penetration, and accessed encrypted communications (a partial threat list).

**Table 1: Stakes Matrix – Step 1. Generation of Stakes Matrix**

Step 1. Stakes Matrix		WIM Security Requirements Canonical Security Properties of Confidentiality, Integrity, Availability, Authentication, and Reliability				
Stakeholders		Collection	Processing	Storage	Transmission	Access for Analysis
WIM Operators		√	√	√	√	
WIM Data Analysts						√

variance that may exist among the levels of verification and validation (i.e., certification) performed on components of the specification. As illustrated in Table 2, data taken from the WIM use cases (the above mentioned specification security requirements from Table 1) are assessed within the context of the following components, WIM Host PC, mobile platform handheld, RBC, downstream Information Systems, WPA, and encrypted XML. The certification activity may produce higher levels of assurance across different components of the specification than others.

4. For a given information system, account for, in a consistent and structured way, the various dimensions of information assurance (i.e., linear type classification/rating) yielding a priority ordering of vulnerabilities based on impact severity in terms of cost. As illustrated in Table 3, the Impact Matrix is generated from the components and their respective threat from

Traditionally, the verification and validation (V&V) effort is charged uniformly on all stakeholders, which may/may not include the cost of information assurance (IA) and/or certification and accreditation<sup>6</sup>. With the quantification infrastructure that has been previously introduced to compute Mean Failure Cost (MFC), we can employ a scheme where the cost of any V&V effort is charged on the stakeholders according to what they stand to lose or gain [10]. Thus, if a particular V&V effort is aimed at improving the level of confidence that refines a component (i.e., that implements a service and/or satisfies a requirement), then stakeholders are charged according to the stake

<sup>6</sup> Certification and Accreditation (C&A) pulls its authority from two major areas of guidance. The Federal Information Security Management Act (FISMA) of 2002, and the Office of Management and Budget (OMB) Circular A-130, Management of Federal Information Resources. FISMA is the law, and OMB Circular is the OMB policy required of all federal agencies. The DoD Information Assurance Certification and Accreditation Process (DIACAP) is the Department of Defense (DoD) process to ensure that risk management is applied on information systems.

**Table 2: Dependency Matrix – Step 2. Generation of Dependency Matrix**

Step 2. Dependency Matrix	Components					
	WIM Host – PC	Mobile Platform Handheld	Reach Back Host (RBC)	Downstream IS (TC-AIMS II/AALPS)	WPA	Encrypted XML
Data Security Requirements						
Collection	√	√				
Processing	√	√				
Storage			√			
Transmission	√	√			√	√
Access for Analysis			√	√		

they have in satisfying said requirement. CSES also introduces and combines such measures as verification costs that consider the fact that certain requirements are easier to satisfy (and prove). Such costs depend on the system, the requirement, and the selected verification method.

By applying the step-wise process of CSES, we estimate the MFC of a system for a set of stakeholders. We initially identify and then maintain (from the discovery phase) the following information:

1. the set of stakeholders of the system, and
2. the set of security specifications and thus security requirements that are to be satisfied by the system.
3. For each stakeholder and each security requirement, the stake that the selected stakeholder attaches to the selected service (or conversely, the cost that the stakeholder incurs if the service is disrupted).

This information is provided by stakeholders.

4. For each component of a specific security requirement, the likelihood that the system provides that service as specified. This information is computed in light of the V&V measures (inspection, verification, testing, security measures, firewalls, vulnerability removal, threat mitigation, etc) that the system has been subjected to. In particular, estimating the likelihood of delivering a service requires that we determine to what degree the components involved in delivering a service have been validated.

Thus, following the CSES vertical process of the Metrics Engine proceeds in three steps (applying Stake Estimation to generate the Stakes Matrix, Bayesian

Analysis to generate the Dependency Matrix, and Threat Analysis to generate the Impact Matrix, as illustrated in the Table 1, 2, and 3 respectively), by the subject matter experts as described in the vertical Evaluation Engine components. CSES encompasses not only failure costs but also mitigation costs, specifically verification costs. Once the basic matrices are populated, a baseline for the particular instantiation of the CSES is established and all changes to the baseline are maintained in a way that track the enterprise's evolution to provide near real-time assessments [11].

The CSES Tool provides a formalized mechanical means to manage the readiness and validity of the various countermeasures identified against the

**Table 3: Impact Matrix – Step 3. Generation of Impact Matrix**

Step 3. Impact Matrix	Threats (Partial List)			
	Components	Malicious Insider	Unauthorized Penetration (Hack)	Read Encrypted Communications
WIM Host - PC	√	√		
Mobile Platform Handheld	√	√		
Reach Back Host (RBC)	√	√		
Downstream IS (TC-AIMS II/AALPS)	√	√		
WPA				√
Encrypted XML				√

likelihood of attack and the impact of the vulnerability. The CSES tool enables us to more formally/rigorously assess the complimentarity and exposure of countermeasures and critical assets respectively. For example, determining what security requirements are being reinforced and what defensive measures are being taken. CSES does not help (referring to Section 3.1) the defensive premise but gives some help to the qualitative premise and significant help to the Ad-Hoc premise.

### 3. Discussion and Future Direction

The term *space race* (or missile gap) was coined in the late fifties when the Soviets launched Sputnik, to

refer to the wide gap between the US's national capabilities in space and its national aspirations. The term *software gap* was coined by Dr. Harlan D. Mills in the mid eighties, at the height of the IT revolution, when it was clear that software technology was not keeping up with the demands of the national economy, as it was growing increasingly dependent on the safe, reliable operation of software systems.

We coin the term *Security Gap* to refer to the vast technological gap that exists today between available capabilities and the demands imposed by recent global developments. We submit that the security gap matches or exceeds the earlier gaps in terms of its stake, and in terms of its technical challenge. We further argue that not only is the security gap very wide, it is actually widening, making it even more urgent that we devote our attention to this technology. Figure 4 shows the three dimensions of security:

- *Criticality*. There is no need for security unless we have a critical resource to protect.
- *Threat*. There is no need for security unless there is a threat to acquire, to expose, or to damage the critical resource that we have.
- *Vulnerability*. Even in the presence of criticality and threat, there is no need for security unless the system that protects the critical resource is vulnerable, in the sense that it lends itself to attack/intrusion/exposure/damage, etc.

We claim that the security gap is widening nowadays, as we are faring worse on all three dimensions of security because of an:

- Increased threat, as a consequence of the emerging global tensions in the world today, as well as the increased sophistication of the perpetrators.
- Increased criticality because the emergence of the Internet has shifted more and more economic and social activity online, making security virtually synonymous with cyber security.
- Increased vulnerability because emerging computing paradigms such as networking, distributed computing, mobile computing, open wide security gaps that are hard to control.

In this paper, we argue that a quantitative, formal approach is needed for modeling system security, and briefly discuss a refinement-based approach that integrates security with other dimensions of dependability. Furthermore, we provide a basic outline of how this

approach may be applied to a pervasive computing system with stringent confidentiality requirements.

### 3.1. Managing Security: A Quantitative Approach

Current security management methods may be characterized by means of three broad premises:

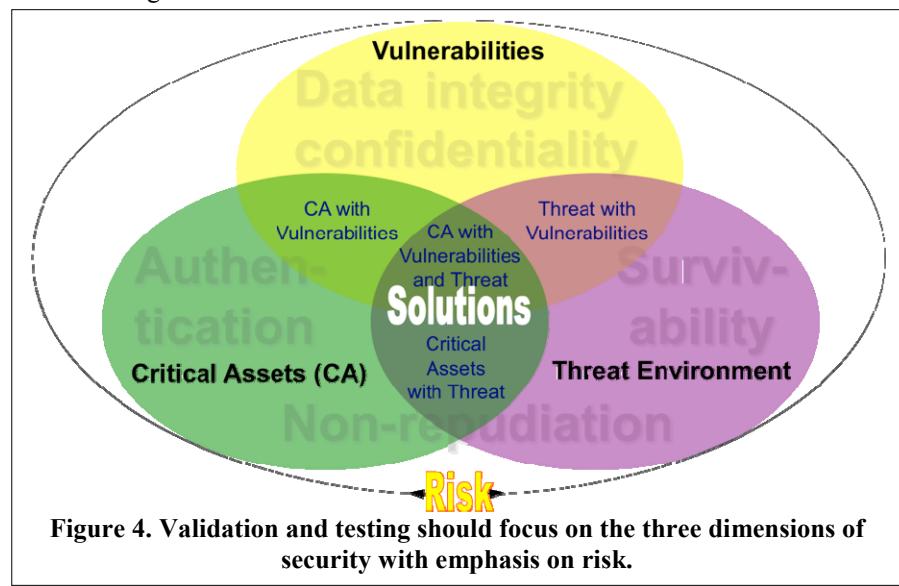
- *Defensive*. Traditional security measures are primarily defensive, in the sense that they are geared to predefined, pre-categorized, vulnerabilities and threats, and are unable to deal effectively with unknown threats.

- *Qualitative*. Security measures are taken because they are perceived to help with predefined threats; there is no quantification of how well they deal with the specific threat they are intended to mitigate, nor with any other variations on this threat.

- *Ad-Hoc*. When security requirements are perceived to be high (due to high criticality, vulnerability, or threat level), several measures are taken in concert; there is usually no acknowledgment of whether these measures are redundant or complementary, whether they overlap, whether together they ensure that some security goal is met, etc.

Clearly, security management requires greater discipline, which may be achieved by providing the following elements:

- A logic for specifying security requirements and verifying secure systems against such requirements.
- A model for managing system security by quantifying costs, risks, measures, and countermeasures.



- Automated tools that support security management according to the proposed models.

### 3.2 Modeling Security

Even though logically, system reliability is driven exclusively by the existence and possible manifestation of faults, empirical observations regularly show a very weak correlation between faults and reliability. In [1], Mills and Dyer discuss an example where they find a variance of 1 to 50 in the impact of faults on reliability; i.e. some faults cause system failure 50 times more often than others; while their experiment highlights a variance of 1 to 50, we have no doubt that actual variance is in fact unbounded. In addition, they find that they can remove 60 percent of a system's faults and improve its reliability by only 3 percent. In a study of IBM software products, Adams [12] finds that many faults in the system are only likely to cause failure after hundreds of thousands of months of product usage.

We argue that the same may be true for security: vulnerabilities in a system may have widely varying impacts on system security. In fairness, the variance may be wider for reliability than for security, because high impact vulnerabilities may be more attractive targets than lower impact vulnerabilities, but wide variances are still quite plausible. Wide variances, to the extent that they are borne out, have broad impacts on security management:

- In practice, security should not be defined as the absence of vulnerabilities any more than reliability is defined by the absence of faults (low impact vulnerabilities do not affect security in a meaningful way).
- In practice, security should not be measured or quantified by the number of vulnerabilities, just as it is widely agreed (as highlighted by Adams' [12] and Mills' [1] work that faults per thousand lines of code (KLOC) is an inappropriate measure of reliability).

• Security cannot be improved by focusing on vulnerabilities, as we have no way to tell whether a given vulnerability has low (1) or high (50) impact on security. Rather, security should be managed by pursuing a policy that leads us to the highest impact vulnerabilities first (a similar approach to usage pattern testing [1, 13-19]).

In light of these observations, we argue in favor of modeling security in a way that reflects its visible, measurable and observable attributes, rather than its hypothesized causes. We argue that a *logic for system security* is needed to be defined in terms of the following features:

- A notation for *security specification*, which details how to capture security requirements of a system.

- A formula for *security certification*, which formulates the condition under which a system (represented by its security abstraction) meets a given set of security requirements (represented by security specifications).

Note that in order to quantify reliability as the mean time to *failure*, we must define what it means to *fail*, which in turn requires that we define *specification* and *correctness*. Likewise, defining and quantifying security requires that we define the concepts of security specification and security certification.

### 3.3 Security as a Dimension of Dependability

It is customary [20] to define *dependability* as the aggregate of four attributes: *availability* (probability of providing services when needed), *reliability* (probability of failure free operation), *safety* (probability of disaster free operation) and *security* (probability of interference free operation). An important distinction is between *availability*, which deals with operational properties, and the other three, which deal with functional and behavioral properties. We argue in favor of a uniform model to capture the three behavioral properties, i.e. reliability, safety, and security. We submit three broad arguments to support our position.

- *Conceptual Argument.* Generalization is a problem-solving strategy that substitutes a specific problem with a more general problem, thereby abstracting away many irrelevant problem-specific details; though it is paradoxical, it is an effective problem solving strategy, as it tends to abstract away irrelevant detail, whence produce more elegant solutions. We argue that reliability, safety, and security lend themselves to analysis by generalization, by virtue of their commonalities. Reliability is defined in terms of faults, errors and failures, and is handled by means of a hierarchy of three methods: fault avoidance, fault removal, and fault tolerance. Safety is defined in terms of three concepts, hazard, mishap and accident, and is handled by means of a hierarchy of three methods: hazard avoidance, hazard removal and damage limitation. Security is defined in terms of three concepts, vulnerability, threat and exposure (or attack), and is handled by means of a hierarchy of three methods: vulnerability avoidance, vulnerability neutralization, and exposure limitation. We argue that these analogies are a strong hint to at least attempt to model these attributes in a uniform manner.

- *Pragmatic Argument.* Reliability, safety, and security are interdependent, in the sense that each property may depend on the others to hold. For example, all the claims of reliability and safety become void if an intrusion occurs and alters the system's

function or the system's state. Conversely, the security of a system is dependent on the reliability of the components that implement/enforce its security measures. Hence, in practice, having high values for one of these attributes is probably not meaningful unless we have commensurate values for the other attributes as well; it is inevitable that the proof of anyone of these properties will use hypotheses about the other properties. In addition, while the distinctions between reliability, safety and security are meaningful for the engineer, they are less meaningful for the user. From the standpoint of the user, it matters little whether a system failed because of violation of a reliability requirement, a safety requirement, or a security requirement.

### 3.4. A Refinement Based Approach

Whereas in the previous section we argued in favor of a uniform approach to the three behavioral attributes of dependability (reliability, safety, security) in this section, we argue in favor of a specific uniform approach, one based on refinement calculi. We submit three broad arguments to support our position.

- *Conceptual Agreement.* Refinement calculi have long been used to model correctness properties, hence form the basis for reliability analysis [21-25]. In [26] Mili et al., have shown how safety can be modeled by the same refinement mathematics, and how reliability and safety concerns can be addressed with the same recovery mechanism. We submit that the same refinement mathematics can be used to model (some aspects of) security. In particular, we argue that there is no difference between reliability and safety except in the quantification of failure costs (the violation of safety requirement costs more than the violation of a reliability requirement). We also argue that there is no difference between reliability and security except in the characterization of fault hypotheses (reliability deals with hardware or software faults where security deals with the faults caused by malicious intervention).

- *Pragmatic Argument.* In [27] Nicol et al., submit that it is virtually impossible to ascertain the security of a complex system, and argue that in order to enhance system security we must deploy a wide range of methods. We argue that many refinement-based methods of reliability are designed to achieve this very same goal; they can be generalized or adapted to deal with security. An important attribute of these methods is their ability to decompose verification goals and to compose verification claims, such as is used in the divide-and-conquer discipline.

- *Methodological Argument.* By merging various aspects of dependability and modeling them in

a uniform manner, we make it possible to build eclectic arguments of dependability.

### 3.5. Application of the CSES to the MAAP Protocol

MAAP defines a qualitative protocol for analyzing operational risk in work processes [28]. The appropriate components of CSES may be used to refine the MAAP protocol in a quantitative way (i.e., stake estimation). This refinement is preliminary, as the mathematical connections are not developed and are only described qualitatively as a mapping from the CSES to the MAAP. The following seven guidelines collectively form the foundation of MAAP: (1) determine mission objectives, (2) characterize all operations conducted in pursuit of the mission, (3) define risk evaluation criteria in relation to the mission objectives, (4) identify potential failure modes, (5) perform a root cause analysis for each failure mode, (6) develop an operational risk profile of the mission, and (7) ensure that operational risk is within tolerance.

## 4. Summary

The lack of sound and practical security metrics is severely hampering progress in the development of secure systems. The large number of potential threats and corresponding vulnerabilities that lurk in an information system represent a significant risk to any enterprise due to the potential of being exploited. As such, systems become even more complex and pervasive we must certainly find better ways to manage and account for these risks toward protecting our information and critical infrastructure assets. A well managed rigorous process will not only help us better understand and prioritize risk mitigation efforts, but should help to solidify vulnerability classification and rating efforts in a consistent structured and fine grain manner.

## 5. References

- [1] H. D. Mills, M. Dyer, and R. C. Linger, "Cleanroom Software Engineering," *IEEE Software*, vol. 4 (5), pp. 19-25, 1987.
- [2] A. Mili and F. T. Sheldon, "Challenging the Mean Time to Failure: Measuring Dependability as a Mean Failure Cost," in *HICSS '09, 42nd Hawaii International Conference on System Sciences*, 2009, pp. 1-10.
- [3] R. K. Abercrombie, F. T. Sheldon, and M. R. Grimalia, "A Systematic Comprehensive Computational Model for Stake Estimation in Mission Assurance," in *IEEE International*

- Conference on Social Computing / IEEE International Conference on Privacy, Security, Risk and Trust*, Minneapolis, MN, 2010, pp. 1153-1158.
- [4] "Federal Information Processing Standard (FIPS) 199, Standards for Security categorizations of Federal Information and Information Systems," ed, 2004.
- [5] M. Dam and P. Giambiagi, "Confidentiality for Mobile Code: The Case of a Simple Purchasing Applet," in *13th IEEE Computer Security Foundations Workshop*, Cambridge, U.K., 2000, pp. 233-244.
- [6] W. E. Burr, *et al.*, "Electronic Authentication Guideline, NIST Special Publication 800-63-1," ed, 2008.
- [7] M. Hefeeda and B. Bhargava, "On Mobile Code Security," CERIAS TR 2001-46, Purdue University, October 2001.
- [8] R. B. Ayed, A. Mili, F. T. Sheldon, and M. Shereshevsky, "An Integrated Approach to Dependability Management," in *Foundations of Empirical Software Engineering: The Legacy of Victor R. Basili*, St Louis, MO, 2005.
- [9] R. K. Abercrombie, F. T. Sheldon, and A. Mili, "Managing complex IT security processes with value based measures," in *Computational Intelligence in Cyber Security, 2009. CICS '09. IEEE Symposium on*, 2009, pp. 69-75.
- [10] A. B. Aissa, R. K. Abercrombie, F. T. Sheldon, and A. Mili, "Quantifying security threats and their potential impacts: a case study," *Innovations in Systems and Software Engineering*, pp. 1-13, 2010.
- [11] F. T. Sheldon, R. K. Abercrombie, and A. Mili, "Methodology for Evaluating Security Controls Based on Key Performance Indicators and Stakeholder Mission," in *HICSS '09, 42nd Hawaii International Conference on System Sciences*, 2009, pp. 1-10.
- [12] E. N. Adams, "Optimizing preventive service of software products," *IBM Journal of Research and Development*, vol. 28 (1), pp. 2-14, 1984.
- [13] S. A. Becker and J. A. Whittaker, *Clean room Software Engineering Practice*: IDEA publishing, 1997.
- [14] M. Dyer, *The Clean room Approach to Quality Software Development*: John Wiley and Sons, Inc., 1992.
- [15] R. C. Linger, "Cleanroom software engineering for zero-defect software," presented at the Proceedings of the 15th international conference on Software Engineering, Baltimore, Maryland, United States, 1993.
- [16] R. C. Linger and P. A. Hausler, "Cleanroom software engineering," in *25th Hawaii International Conference on System Sciences*, Kauai, Hawaii, 1992.
- [17] R. C. Linger, "Cleanroom Process Model," *IEEE Software*, vol. 11 (2), pp. 50-58, Mar./Apr 1994.
- [18] H. D. Mills, R. C. Linger, and A. R. Hevner, *Principles of Information Systems Analysis and Design*: Academic Press, 1985.
- [19] S. J. Prowell, C. J. Trammell, R. C. Linger, and J. H. Poore, *Cleanroom Software Engineering: Technology and Process*: SEI Series in Software Engineering, Addison Wesley, 1999.
- [20] I. Sommerville, *Software Engineering (9th Edition)*: Addison Wesley, 2010.
- [21] R.-J. Back, J. V. Wright, D. Gries, and F. B. Schneider, *Refinement Calculus: A Systematic Introduction (Graduate Texts in Computer Science)*: Springer Verlag, 1998.
- [22] J. Desharnais, A. Mili, and T. T. Nguyen, "Refinement and demonic semantics, Chapter 11," in *Relational Methods in Computer Science*, C. Brink, *et al.*, Eds., ed: Springer Verlag, January 1997, pp. 166-183.
- [23] P. Gardiner, "Data refinement of predicate transformers," *Theoretical Computer Science*, vol. 87, 1991.
- [24] M. B. Josephs, "An introduction to the theory of specification and refinement," Technical Report RC 12993, IBM Corporation, July 1987.
- [25] J. V. Wright, "A lattice theoretical basis for program refinement," Dept. of Computer Science, Abo Akademi, Finland, 1990.
- [26] A. Mili, F. T. Sheldon, F. Mili, and J. Desharnais, "Recoverability preservation: a measure of last resort," *Journal Innovations in Systems and Software Engineering*, vol. 1 (1), pp. 54-62, 2005.
- [27] D. M. Nicol, W. H. Sanders, and K. S. Trivedi, "Model-Based Evaluation: From Dependability to Security," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, pp. 48-65, January-March 2004.
- [28] C. J. Alberts and A. J. Dorofee, "Mission Assurance Analysis Protocol (MAAP): Assessing Risk in Complex Environments," Carnegie Mellon University, Pittsburgh, PA CMU/SEI-2005-TN-032, 2005.