Three Tenets for Secure Cyber-Physical System Design and Assessment

Jeff Hughes[a]

George Cybenko[b]

[a]Tenet3, LLC, Dayton OH jeff.hughes@tenet3.com
[b]Dartmouth, Hanover, NH gvc@dartmouth.edu

ABSTRACT

This paper presents a threat-driven quantitative mathematical framework for secure cyber-physical system design and assessment. Called *The Three Tenets*, this originally empirical approach has been used by the US Air Force Research Laboratory (AFRL) for secure system research and development. The *Tenets* were first documented in 2005 as a teachable methodology. The *Tenets* are motivated by a system threat model that itself consists of three elements which must exist for successful attacks to occur:

- system susceptibility;
- threat accessibility and;
- threat capability.

The Three Tenets arise naturally by countering each threat element individually. Specifically, the tenets are:

> *Tenet 1: Focus on What's Critical* - systems should include only essential functions (to reduce susceptibility);
>
> *Tenet 2: Move Key Assets Out-of-Band* - make mission essential elements and security controls difficult for attackers to reach logically and physically (to reduce accessibility);
>
> *Tenet 3: Detect, React, Adapt* - confound the attacker by implementing sensing system elements with dynamic response technologies (to counteract the attackers' capabilities).

As a design methodology, the *Tenets* mitigate reverse engineering and subsequent attacks on complex systems. Quantified by a Bayesian analysis and further justified by analytic properties of attack graph models, the *Tenets* suggest concrete cyber security metrics for system assessment.

1. INTRODUCTION

Our goal is to engineer and deploy systems that are more secure. This is in contrast with efforts that are aimed at designing systems that are "provably" secure according to some idealized formal model of security. In fact, we accept the position expressed in the 1999 National Academies study "Trust in Cyberspace":

> 'Security research during the past few decades has been based on formal policy models that focus on protecting information from unauthorized access by specifying which users should have access to data or other system objects. It is time to challenge this paradigm of "absolute security" and move toward a model built on three axioms of insecurity: insecurity exists; insecurity cannot be destroyed; and insecurity can be moved around'. (Page 247,[1])

This paper presents advancements beyond previous publicly released work by the Office of the Assistant Secretary Defense for Research and Engineering, ASD(R&E) and the US Air Force Research Laboratory (AFRL) under the Software Protection Initiative (SPI) from 2001-2011. All opinions expressed are those of the authors, not AFRL nor ASD(R&E). No USAF Objection to Publication: 88ABW-2013-3882

Cyber Sensing 2014, edited by Igor V. Ternovskiy, Peter Chin, Proc. of SPIE Vol. 9097, 90970A
© 2014 SPIE · CCC code: 0277-786X/14/$18 · doi: 10.1117/12.2053933

Given that insecurity exists and cannot be completely eliminated in real systems, we strive to reduce insecurity to mission-acceptable levels. We argue that this is an engineering problem that can be formulated in terms of sound analytic and scientific principles.

By analogy, industrial and military systems have been designed and assessed with respect to mission reliability requirements for many years. Reliability engineering is an accepted practice within the US military today[2] and is based on analytic principles drawn from reliability theory.[3] Just as reliability theory is a foundation for composing and assessing systems to be "reliable enough" with respect to given usage requirements, we believe that *The Three Tenets* can be the foundation for building information and cyber-physical systems that are "secure enough" within a given mission, operating environment, and threat context. (Generally speaking, reliability theory is based on failures due to independently occuring natural events and their cascading effects while security engineering is concerned with coordinated attacks by rational adversaries.)

We recognize that there has been considerable interest recently in developing a "Science of Security" but the goal remains elusive.[4,5] Some promising developments have been made in the areas of attack graph modeling and analysis[6–10] and probabilistic analyses of such attack graphs.[11,12] Security engineering would ideally be based on such analytic foundations. However, security engineering today is still largely based on case studies and the associated lessons learned.[13]

This paper proposes to start eliminating this gap between cybersecurity theory and practice by developing *The Three Tenets* directly from a system's threat model and the attacker's reverse engineering steps taken to understand that system.

We believe that employing *The Three Tenets* during system design and development will result in more secure systems that:

1. are compatible with an enterprise's mission;
2. can generally be built from commercial components with modest customization;
3. enable superior nation-state class threat mitigation.

In fact, over the past several years, *The Three Tenets* have been successfully applied to the development of Software Protection Initiative (SPI) cybersecurity systems fielded and maintained across the Department of Defense and the Defense Industrial Base. During the period of 2003-2011, the Anti-Tamper Software Protection Initiative (ATSPI) Technology Office, Sensors Directorate, AFRL, employed the *Tenets* as a secure system design methodology and also for system security evaluation metrics.

Before continuing, it is important to make clear that this work is aimed at complex system security. By 'complex system', we mean a system that:

(a) is not provably secure (systems that can be formally shown to be "secure" in some sense are relatively few and quite specialized at the present time although that may change as technology matures);

(b) is defined prima facie by its enumerated software/hardware components, its access points in an operating environment (both intentional and unintentional), and by data that transit those access points whose semantics imply the system's intended use or functionality;

(c) when viewed externally, the system's components appear to be selected from a large sample space with sufficiently similar but not necessarily identical functionality making details of the system's precise operations and susceptibilites appear random (this merely states that a complex system is a black box with unknown weaknesses which the attacker must infer).

In particular this work is not about programming practices that make software applications more secure (such as avoiding constructs that create buffer overflows and so on).[14] It is about building complex systems out of multiple software and hardware components, each component of which may or may not be "high assurance" in some sense.

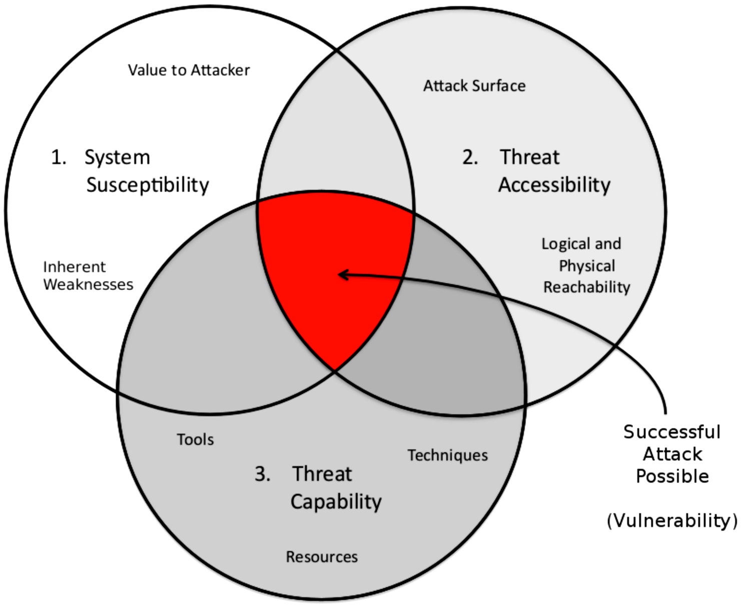


Figure 1. *The Three Tenets* threat model consists of the three elements required for a successful attack. Each *Tenet* will mitigate an element of this threat model.

Section 2 of this paper presents our underlying cyber security threat model. Section 3 develops *The Three Tenets* directly from the threat model and presents specific examples of the *Tenets* already in use. Section 4 contains implications of the *Tenets* including how to apply them to system security analysis. Section 5 is a discussion of quantitative security metrics derived from *The Three Tenets* formulation. Section 6 is a summary. Section 7 is the appendix with the mathematical formulation.

2. THE THREAT MODEL

Attackers will utilize available system features including access points and intended system functionality to achieve their goals. Our complex system definition (components, access points, functionality) suggests a threat model based on corresponding elements that we assert are necessary and sufficient for successful attacks to occur: 1) an inherent system weakness or susceptibility; 2) the threat's access to the weakness and; 3) the threat's capability to exploit the weakness. We further assert that only when these three threat elements are present does an actual vulnerability exist.

Murphy's Law* suggests that a system with vulnerabilities will be exploited given the appropriate operational environment.†. A threat model that supports reasoning about whether an inherent system weakness rises to the level of vulnerability is essential for cost effective system security engineering. This is an important perspective in our work, since security for security's sake is neither affordable nor desirable, and so vulnerabilities must be quantified and only mitigated to the degree necessary to prosecute the mission. This type of vulnerability decomposition aids in that process. Furthermore, this form of threat model has deep roots in the Electronic Warfare (EW) test and evaluation community.[16] That community shares a similar adversarial framework (measure-countermeasure) with cyber-physical system security. A version of this threat model has been suggested for EW vulnerability analysis since 1978 (called Data link Vulnerability Analysis or DVAL).[17] DVAL has four components in its vulnerability definition (susceptibility, interceptibility, accessibility, and feasibility). However, in contrast to DVAL, *The Three Tenets* threat model assumes 'feasibility' and 'interceptibility' are effectively merged into what we call 'capability'. In today's complex cyber-physical systems based on commercial-off-the-shelf technologies,

*"Any thing that can go wrong will go wrong."

†Murphy's Law has been associated with system engineering and risk minimization[15]

attackers can rehearse for almost any given operating environment (e.g. Stuxnet[18]) and develop a capability. The threat model elements are described next and their relationships are illustrated in Figure 1.

Threat Element 1 *System Susceptibility* - Absolute system confidentiality and availability cannot be simultaneously achieved. Therefore, all systems will have design trade-offs resulting in inherent weaknesses. In particular, we believe that confidentiality and availability are antithetical so that tradeoffs between them are necessary.[‡] Moreover, systems often contain unintentional design or implementation flaws. Both of these mechanisms are included in our notion of system susceptibility. The threat will attempt to discover and exploit these susceptibilities in order to compromise (modify, control, destroy, or steal) critical system elements or functions.

Threat Element 2 *Threat Accessibility* - A threat will probe and analyze a system in order to discover which susceptibilities are accessible and subsequently exploitable. Generally, the threat will use access points or services offered by a system to legitimate users as the original point of entry. Threat access is typically a superset of legitimate user access (since some access points may be undocumented and/or not of interest to legitimate users).

Threat Element 3 *Threat Capability* - After a thorough surveillance (either via remote observations or in situ instrumentation) of the system design and operation, an attacker will attempt to gain control, tamper with, and/or steal detailed system design knowledge or other critical data. This is often done using a known exploit or zero-day exploit determined after additional system reverse engineering. Skilled attackers typically employ a methodical approach to reverse engineering during which they expect to observe certain system behaviors. These system behaviors serve as exploitation guideposts and significantly aide the attacker. The degree to which the attacker is successful will depend on their level of system knowledge, their ability and resources to develop and use specialized tools given the system's functionality and operating environment, and their overall level of reverse engineering experience.

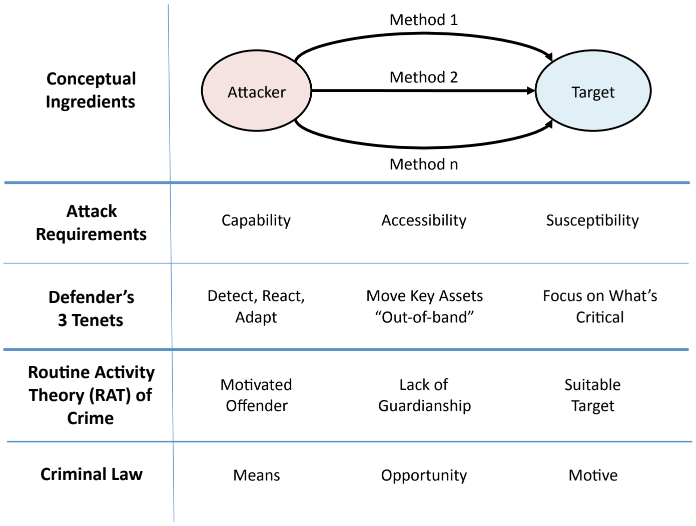
These three threat elements are closely related to classical criminological theories. For example, Routine Activity Theory (RAT), used in criminology,[19] posits that crimes occur when three elements coincide: 1) there is a motivated offender; 2) there is a lack of guardianship and; 3) there is a suitable target. The elements of RAT and the threat model elements listed above have a clear correspondence: capability – motivated offender, accessibility – lack of guardians, susceptibility – suitable target. Our threat model is also related to "Means, Motive and Opportunity" arguments necessary (but not sufficient) for convincing a jury of a suspect's guilt[20] . The point is that previous work in criminology is relevant and consistent with our approach to cyber threat modeling.

A graphical depiction of the attacker-access-target framework and the correspondences is shown in Figure 2. In anticipation of developing *The Three Tenets* in the Section 3, the figure also depicts correspondences between the *Tenets* (which are defensive in nature) and these necessary attack ingredients. Specifically, each tenet addresses a corresponding ingredient that is necessary for a successful attack to occur. The application of the *Tenets* can also be interpreted as shrinking each element of the threat model in Figure 1, thereby implying a reduction in system vulnerability. This is illustrated explicitly in Figure 3.

3. THE THREE TENETS

Taken together, *The Three Tenets* comprise a system security engineering approach consisting of both a secure design methodology and an assessment tool for security evaluation. The following are the heuristic definitions of the *Tenets* based on AFRL ATSPI Technology Office's experience. We will show in the following sections how the *Tenets* are related to a probabilistic model for reverse engineering a system.

[‡] "The only truly secure system is one that is powered off, cast in a block of concrete and sealed in a lead-lined room with armed guards - and even then I have my doubts."(Eugene H. Spafford)



Attack Requirements	Capability	Accessibility	Susceptibility
Defender's 3 Tenets	Detect, React, Adapt	Move Key Assets "Out-of-band"	Focus on What's Critical
Routine Activity Theory (RAT) of Crime	Motivated Offender	Lack of Guardianship	Suitable Target
Criminal Law	Means	Opportunity	Motive

Figure 2. Graphical depiction of successful cyber-physical system attack requirements and their relationship to *The Three Tenets* and classical theories of crime.

Tenet 1 *Focus on What's Critical* - The first *Tenet* instructs the designer to consciously and methodically focus on including only those system functions that are essential to the mission. This is an acknowledgement of Occam's Razor by the system designer. Adherence to this *Tenet* reduces the number of potential susceptibilities, and therefore, the paths between the attackers' starting state (the system access points) and goal states in which mission essential functions, critical security controls, or critical data are compromised. This *Tenet* eliminates those access points and susceptibilities associated with unneeded functionality.

Tenet 2 *Move Key Assets Out-of-Band* - The second *Tenet* instructs the designer to consciously differentiate between user access and attacker access for a given system's mission. This *Tenet* modifies system availability and is accomplished by moving the data/processes used by mission essential functions, their security controls, and associated access points out-of-band of the attacker either logically, physically, or both. By "out-of-band" we mean not accessible by the attacker through their preferred or available access methods. Adherence to this *Tenet* reduces threat access for a given mission (use case) and may enable unalterable observations of system state by a security control sensor. The extent and strength of access differentiation between the user and attacker is greatly influenced by the type of out-of-band mechanism employed and whether its done in software or hardware.

Tenet 3 *Detect, React, Adapt* - The third *Tenet* instructs the designer to employ dynamic sensing and response technologies (i.e a security control sensor or reference monitor) that mitigate the threat's capabilities and exploitation attempts through automated (preferably autonomic) system behavior. Adherence to this *Tenet* confounds the attacker's capabilities by making the system's defenses unpredictable (nonstationary) and adaptive (with penalties) instead of merely being passive.

Just as each ingredient of the threat model has grounding in EW and classical criminology theory, each of *The Three Tenets* has been advocated and practiced in one form or another by computer security researchers and developers in the past.[21] Furthermore, the *Tenets* mathematical model (Section 7) enables reasoning about complex system security in a probabilistic manner providing valuable insights consistent with lessons learned from work on formal methods.[22] We now present a few examples to illustrate common practices that are instances of the *Tenets* individually. A key contribution of *The Three Tenets* approach and this paper is the experiential recognition that applying all three together produces the most robust defenses.

3.1 Illustrative Examples of the Tenets in Use

Previous research has typically focused on cyber security approaches that are instances of just one of the *Tenets*. Several examples are presented below. When appropriate, those defensive approaches employing more than one *Tenet* will be identified.

Examples of Tenet 1: Focus on What's Critical

- One of the oldest and most enduring concepts in cyber security has been the "Principle of Least Privilege", first articulated by Jerry Saltzer and Mike Schroeder in 1975.[23,24] It states:

  > "Least privilege: Every program and every user of the system should operate using the least set of privileges necessary to complete the job. Primarily, this principle limits the damage that can result from an accident or error. It also reduces the number of potential interactions among privileged programs to the minimum for correct operation, so that unintentional, unwanted, or improper uses of privilege are less likely to occur. Thus, if a question arises related to misuse of a privilege, the number of programs that must be audited is minimized. Put another way, if a mechanism can provide 'firewalls,' the principle of least privilege provides a rationale for where to install the firewalls. The military security rule of 'need-to-know' is an example of this principle".

  One application of this principle is that if a service or functionality is not necessary for a mission, then users should have no access to it: that is, have no privileges to use it. If no users need to have such privileges, the service need not be implemented.

- Other, more direct, implementations of *Tenet 1* has to do with minimizing and/or monitoring the functionality of code[25] and reducing functionality of web broswers by disabling ActiveX and other capabilities.[26]

- Michael Howard has discussed the use of an attack surface metric for analysis of a system's relative risk.[27] He and his colleagues set aside the actual vulnerability of any given attack vector in favor of a simple enumeration of all the entry points to the system under study (in their case, successive versions of Microsoft Windows®). Their list of attack vectors includes open sockets, open named pipes, and other means by which an attacker could potentially introduce data. Howard proceeds to then rank the various editions of Windows® in terms of their attack surface: that is, their accessibility to an attacker. By focusing on the simple number of entry points into the system and reducing those entry points into the system, Microsoft could reduce the remote attack surface on Windows®, allowing developers to focus on a more limited number of potential attack vectors.

- This same approach can be applied at the network level in terms of firewall rules that allow traffic in from the outside. Closing off ports or disallowing traffic from certain subnets reduces the attack surface. This technique does not address any underlying vulnerabilities in the software, nor is it reducing the number of applications running on the network. However, doing this allows defenders to concentrate their resources on a smaller pool of potential vulnerabilities.

- Manadhata et al. discuss the concept of the attack surface in terms of both entry points to and exit points from the system, because an attack need not consist of the introduction of malicious commands or code, but may entirely comprise exfiltration of data.[9] This gets at moving assets out-of-band which is the next Tenet we consider.

Examples of Tenet 2: Move Critical Assets Out-of-Band

- One example of an exit point-only attack is that of eavesdropping on the electronic communication between a host and a connected monitor, in the manner described by Wim Van Eck, who was able to view the display on a monitor hundreds of meters away by using an antenna to capture the electromagnetic radiation emitted by the circuitry and cables.[28] Van Eck offered several solutions, one of which was to reduce the radiation level: use switching circuitry only as fast as necessary, keep cables short, keep circuit board traces short. In doing so, the asset is moved physically out-of-band by reducing the physical area in which an attack can occur. Another method of moving the system physically out-of-band would be to place it inside a Faraday cage, a conductive shell around the equipment that effectively blocks electromagnetic signals.

- Employing encryption to protect data at rest or in transit is a commonly used method for moving critical information out-of-band with a computable estimate of adversary work factor. Software code obfuscations that mitigate reverse engineering of binary code or implement changes in the runtime conventions of programs are also out-of-band techniques (albeit with varying strengths). Software out-of-band methods also include virtual machines and hypervisors. Software obfuscations typically do not support a priori adversary work factor estimates. Other obfuscation technologies exist. For instance, Address Space Layout Randomization (ASLR) is an obfuscation implemented to confound attackers by making it more difficult to predict target addresses for buffer overflow exploitations.[29] ASLR based protection is usually strengthened with other techniques such as Data Execution Prevention (a predominately *Tenet 1* technique because it limits system functionality available to the software executable). ASLR can still be bypassed because other access points are available to the attacker.[§]

- The Trusted Platform Module (TPM) is an example of using specialized hardware to store critical information in an out-of-band manner. Another example of moving critical assets out-of-band is the use of Smart Cards, SIM cards and USB tokens that use tamper-resistant hardware.[30] The out-of-band construct can be applied in a graduated fashion against an attacker to increase the work factor involved in gaining access. Many out-of-band hardware mechanisms also employ *Tenet 3* in a challenge-response paradigm.

Examples of Tenet 3: Detect, React, Adapt

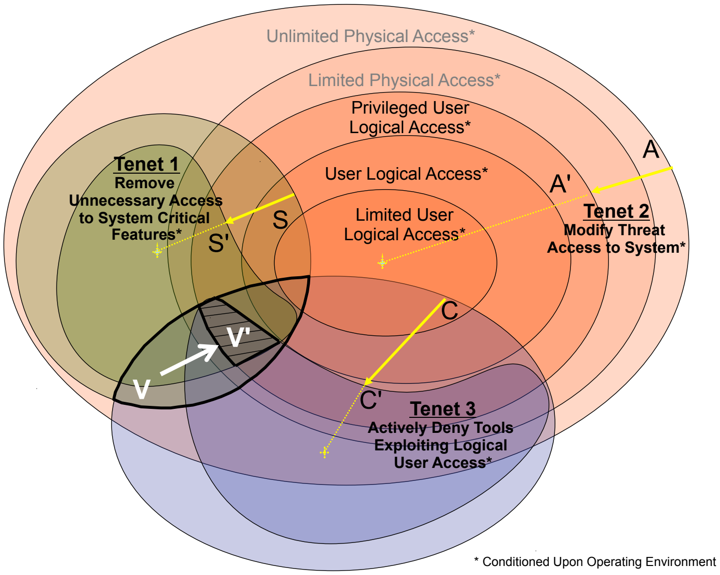
- Automated brute-force username/password dictionary attacks have been effective in the past and can be very quick. A commonly used simple “detect, react” mechanism is to cut off login attempts after a certain number of unsuccessful tries forces the attacker to conduct further reconnaissance or constantly switch usernames and/or originating IP addresses. "Adaptation" would involve changing the "reaction" over time perhaps even learning from previous attacks.

- Windows® and other operating systems have implemented stack cookies (canaries) as a stack smashing "detect, react" method. This *Tenet 3* approach lacked robustness by itself due in part to the attackers ability to co-opt additional unprotected "in-band" functionality by overwriting related stack data namely the exception handler record. One countermeasure to this attack on stack canaries was to create an additional “detect, react” security control protecting the exception handler function pointer (SafeSEH in the case of Windows®) and move associated data into the executable’s metadata and further out-of-band of the stack smashing attack. While SafeSEH increased attacker workload, it also impacted system availability/flexibility/compatibility for legitimate users. So Microsoft introduced an updated approach to exception handler protection called Structured Exception Handler Overwrite Protection (SEHOP)[¶]. SEHOP is also an example of *Tenet 3* protection. “Defense in depth” approaches to security can be challenging to analyze in complex systems. They are complicated by trade-offs with desired user functionality (i.e. a confidentiality vs. availability trade). We believe a *Tenet* based analysis can help book-keep and quantify these trade-offs and provide useful insights to the potential robustness of security controls.

- It is often desirable to detect a human user versus a ‘bot’. A typical scenario is the creation of guest or free email accounts. Von Ahn, et al introduced the CAPTCHA approach in order to reduce the capability of automated programs to create such accounts by requiring a user to solve one of a class of problems that are easy for most humans but are considered hard to solve via machine pattern recognition.[31] Recognizing words in images, recognizing objects in pictures and recognizing spoken words are among the CAPTCHA tests that have been developed. In terms of an adversary’s attack capability, CAPTCHAs are a reaction to the use of simple automated tools that allow for rapid repeated attacks. CAPTCHA’s have significantly impacted attacker workload by requiring the adversary to invest much more in automated pattern recognition

[§] http://www.zdnet.com/pwn2own-down-go-all-the-browsers-7000012283/

[¶] http://blogs.technet.com/b/srd/archive/2009/02/02/preventing-the-exploitation-of-seh-overwrites-with-sehop.aspx



Unlimited Physical Access*
Limited Physical Access*
Privileged User Logical Access*
User Logical Access*
Limited User Logical Access*
A
A'
Tenet 2
Modify Threat Access to System*
Tenet 1
Remove Unnecessary Access to System Critical Features*
S
S'
V'
V
C
C'
Tenet 3
Actively Deny Tools Exploiting Logical User Access*
* Conditioned Upon Operating Environment

Figure 3. This figure depicts the event space necessary for a vulnerability and therefore successful attack to occur. Application of the *Tenets* shrinks the event spaces and results in a smaller set of vulnerabilities, V'.

tools or in more automated human intervention[‖]. It should be noted that since CAPTCHAs are a detection mechanism they can readily incorporate a penalty (a severe reaction) and change that reaction in a temporal fashion (adaptation). Since its challenge-response detector is implemented out-of-band of the client device, CAPTCHAs exhibit differentiation between intended user access and that of the attacker. Hence CAPTCHAs also employ *Tenet 2* .

4. IMPLICATIONS OF THE TENETS

The above discussion develops *The Three Tenets* from a principled analysis of the threat circumstances that allow vulnerabilities to exist - namely the co-occurence of three necessary ingredients: susceptibility, accessibility, and attacker capability. Each *Tenet* is a mechanism to diminish the corresponding ingredient set so that the intersection of the three ingredient sets, which is the set of resulting vulnerabilities as depicted in Figure 1, is consequently reduced. Examples of how the *Tenets* reduce vulnerabilities are depicted by an Euler diagram in Figure 3, where the susceptibility set S, accessibility set A, and capability set C are each smaller resulting in a smaller Vulnerability set V' as well.

While this graphical depiction is intuitive and clearly illustrates how and why *The Three Tenets* work, more formal consequences of the *Tenets* can be derived as well. In this section, we show how the *Tenets* relate to:

- reverse engineering methodology;
- security analysis of systems with partial knownledge;
- empirical system knowledge, and;
- attack graph analysis.

Additional details are provided in the Appendix.

[‖] http://www.troyhunt.com/2012/01/breaking-captcha-with-automated-humans.html

4.1 The *Tenets* and Reverse Engineering of Complex Systems

Prospective attackers typically have to reverse engineer the details of a complex system that defenders have built and deployed (recall our definition of "complex system" in the introduction). The AFRL ATSPI Technology Office's experience with reverse engineering over several years suggests that the process can be decomposed into several canonical steps.

To understand the reverse engineering process, it helps to identify the steps in the forward design process. Given an operating environment and a mission definition, one may construct a system using the following 'forward' steps:

(1) Determine the system's required functionality and map it onto available components drawn from a large set of possible components thereby establishing candidate solutions;

(2) Refine the solutions and specify input and output (I/O) data types and processes required to support the functionality in the operating environment;

(3) Design specific access points (communication channels) between the system and the environment where I/O data (including control data) supporting the functionality transits.

Reverse engineering a system's composition and functionality involves the following corresponding "inverse" steps:

(1) Perform observations to enumerate the system's access points for the given environment and operating conditions. This is a direct measure of accessibility. Call this evidence set, E_a.

(2) Perform observations of data I/O and system processes viewable from these access points. This provides insights into the underlying system semantics and functionality. The attacker's ability to collect this evidence is a direct measure of attacker capability. Call this evidence set, E_c,

(3) Estimate system composition and functionality from the collected evidence E_a and E_c by testing (heuristically or formally) hypotheses about system construction, $H_1, ..., H_k$ where the H_j are mutually exclusive hypotheses about hardware/software component sets and the manner in which they are assembled in the system. We assume that knowledge of a system's composition is a direct measure of susceptibility.

The reverse engineering "inverse problem" is therefore reduced to evaluating the conditional probabilities $P(H_i|E_a, E_c)$ and comparing them for different H_i, selecting the maximum. This is the maximum likelihood estimate of how the system is constituted and allows the attacker to improve the success of any attack they mount. In Appendix 7.1, we show that these probabilities can be computed and combined to yield probabilities of system vulnerabilities. Application of *Tenets 2* and *3* impedes the first two reverse engineering steps explicitly. Application of *Tenet 1* during the system development process optimizes and informs the use of *Tenets 2* and *3* by the system designer and lowers the overall costs of applying protections. If done properly for complex systems, applying *Tenet 1* reduces system susceptibilities, saves the developer money in the application of security controls, but offers the attacker no advantage during their hypothesis analysis in the third reverse engineering step.

4.2 Analyzing Security Using Partial Knowledge of System Construction

Starting with some reasonable system assumptions based on operating environment, mission, and available commercial components, we can posit what a system contains, calling such a hypothesis H_i as before. Given an H_i, we can simulate, emulate, or build and measure it to estimate $P(E_a|H_i)$ and $P(E_c|H_i)$.

One way to quantify the uncertainty about H_i is by empirical sampling and analysis. Denote an actual system, H_i, by a set of n-tuple samples representing various possible system assemblies of components that exhibit a particular vulnerability. This is consistent with the attacker's point of view, where the possible system construction details may appear randomly chosen from a library of available technologies that implement a particular functionality. The attacker must often find specific vulnerabilities of an n-tuple instance by 'fuzzing'. We assert that all members of the system set H_i are vulnerable to the same class of attacks.

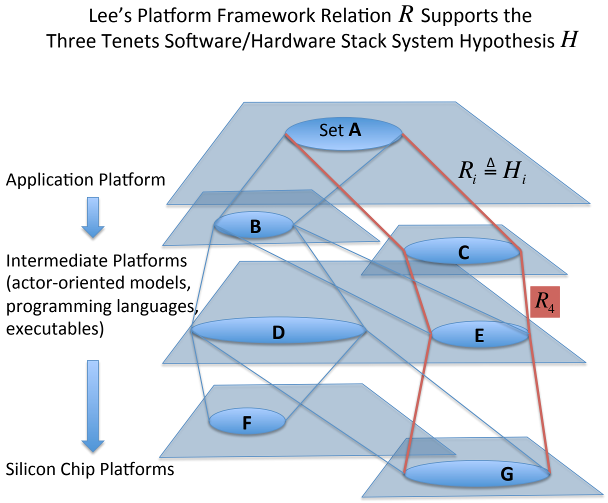


Figure 4. Edward Lee's platform framework[32] is employed to estimate the system H and subsequently $P(H_i)$ for *The Three Tenets* metric. The cartesian product R_4 of sets A, C, E, and G is shown highlighted in red.

In order to estimate $P(H_i)$, we build from Edward Lee's work in cyber-physical system analysis.[32] Lee's actor-oriented design describes sets of system designs called "platforms" representing hierarchies of collections of system design instances (levels of design abstraction). From these platforms he traces the design instantiation from a high level abstraction to a low level physically realizable instance in order to reason about various properties of the cyber-physical system (including timing constraints). Figure 4 is a simplified version taken from Lee's Figure 3.[32] Functional composition sets called "relations" are defined as the cartesian product of subsets of platforms. Relations effectively trace the design functionality from the abstract to a concrete realization.

We make use of this construct to reason about our system, H_i, through knowledge collected about the construction at various levels in the hierarchy. This approach is more likely to be used by a system integrator desiring security estimates. The integrator has insight into the assemblage of components, but only partial knowledge on the details of the components from suppliers. This construction analysis is important for cyber security practioners as vulnerabilities may result from flaws in a design or its instantiation.

Appendix 7.2 contains details about how to use this framework to estimate $P(H_i)$ for different hypotheses about system construction. The impact of *Tenet 1* on $P(H_i)$ is self evident in this context.

4.3 Analyzing Security by Observing the System

Security analysts are often given a system to evaluate and assess. A typical approach is to instrument the system and observe it while operating. Our *Three Tenet* probabilistic model provides a method to make use of those observations and reason about the system and its security properties.

Generally speaking, by allowing more access points and data observations to be available to the attacker, the attacker will be able to reduce uncertainty about the system composition and therefore its possible vulnerabilities. The uncertainty about hypotheses, H_i, can be quantified by the entropy, $\mathcal{H}$, of the probability distribution, $P(H_i|E_a, E_c)$[33] and is described in Appendix 7.3. Application of *Tenet 2* and *3* impedes attacker observations and the collection of evidence sets E_a and E_c.

4.4 The *Tenets* and Attack Graphs against Complex Systems

We argued that *The Three Tenets* are directly related to the reverse engineering process typically practiced by attackers. It is consequently derived how application of the *Tenets* increases uncertainty about system structure so that the attacker's task is more difficult, imprecise and overt.

It can also be argued that the *Tenets* can be derived from an analysis of classical attack graph models,[34] probabilistic attack graph models[11] and Markov Decision Process (MDP) attack graphs.[12] We develop the relationships explicitly in Appendix 7.4.

5. THE *TENETS* AND QUANTITATIVE CYBER-PHYSICAL SYSTEM SECURITY METRICS

These previously detailed considerations provide a quantitative basis for the following security metrics which are merely illustrative of more comprehensive and quantitative metrics that are possible:

- **System Susceptibility Metric:** In it's simplest instance, this system construction metric instructs us to minimize the number of access points to system critical functions. This "reachability" metric is a direct consequence of the first *Tenet*, to identify, implement, and protect only what is mission critical.

- **Access Point Metric:** Minimize the amount of I/O and system processes visible to an attacker. This metric is a direct consequence of the second *Tenet*, to move critical data "out-of-band". Enumeration of "in-band" vs "out-of-band" access points is one way to measure application of the second *Tenet*.

- **Threat Capability Metric:** Minimize useful insight into system operations in the sense that data observed at one time may or may not be similar or consistent with data observed at another time or on another system by the attacker. This "evidence variability" metric is a direct consequence of the third *Tenet*, to detect, react, and adapt and is referred to by cyber security practitioners as a "moving target defense".

These metrics can be readily measured by an enumeration of access points and data I/O or process observations together with determinism of system behavior components. Moreover, there are clear economic and effectiveness tradeoffs between, for example, implementing *Tenet 3* (detect, react and adapt) and *Tenet 1* (implementing only critical mission functionality). These tradeoffs can be addressed through a QuERIES-type methodology and are the subject of ongoing work.[12]

6. SUMMARY, DISCUSSION, AND FUTURE WORK

A threat driven, secure system design and assessment methodology has been developed and used by AFRL in secure system development for the last eight years. The methodology, called *The Three Tenets* of cyber security, is based on threat models and threat mitigating techniques.

We start with a three part system threat decomposition defined by: 1) system susceptibilities (axiomatically described as logical and physical system access points); 2) threat accessibility to those access points and their data transits; and 3) threat capability to exploit those access points. This decomposition was used as the threat framework for deriving the *Tenets*.

The Three Tenets are: 1) deliberately focus on including only mission essential functions in the system under development to minimize inherent susceptibilities; 2) move data/processes for the essential mission elements and security controls associated with those functions "out-of-band" either logically, physically, or both and; 3) employ detect, react, and adapt technologies via automatic/autonomic system response that mitigate threat exploitation attempts.

We observe that adherence to*The Three Tenets* results in more secure systems that are compatible with the enterprise, can be built from commercial components, and result in superior nation-state class threat mitigation. Quantitative measurements of Tenet adherence are presented and can be applied in either the system design phase or after-the-fact in a system assessment process.

Extensive research and experimentation (using QuERIES,[12] black hatting, and red teaming) has been underway to validate this approach. Investigations into which *Tenets* are necessary but not sufficient conditions for nation-state threat mitigation are ongoing. *The Three Tenets* also show promise for effectively composing secure systems consistently with national cyber strategy approaches.[35]

REFERENCES

[1] Schneider, F. B., [*Trust in Cyberspace*], National Research Council: National Academy Press, Washington, D.C. (1999).

[2] TM5-698-4, [*US Army Technical Manual 5-698-4: Failure Modes, Effects And Criticality Analysis (FMECA) For Command, Control, Communications, Computer, Intelligence, Surveillance, And Reconnaissance (C4ISR) Facilities*], Headquarters, Department Of The Army (USA), Washington, D.C. (2006).

[3] Gertsbakh, I., [*Statistical Reliability Theory*], Marcel Dekker, New York (1989).

[4] Evans, D., [*NSF/IARPA/NSA Workshop on the Science of Security*], Also see http://sos.cs.virginia.edu/., University of Virginia (2008).

[5] JASON, "Science of Cyber-Security," Tech. Rep. JSR-10-102, JASON Program Office, MITRE Corporation via funding by ODUSD(AT&L) (2010).

[6] Wang, L., Singhal, A., and Jajodia, S., "Toward measuring network security using attack graphs," in [*QoP '07: Proceedings of the 2007 ACM workshop on Quality of protection*], 49–54, ACM, New York, NY, USA (2007).

[7] Sheyner, O., Haines, J., Jha, S., Lippmann, R., and Wing, J. M., "Automated generation and analysis of attack graphs," *Security and Privacy, IEEE Symposium on* **0**, 273 (2002).

[8] Jha, S., Sheyner, O., and Wing, J. M., "Minimization and reliability analyses of attack graphs," tech. rep., IEEE SYMPOSIUM ON SECURITY AND PRIVACY (2002).

[9] Manadhata, P. K., Tan, K. M. C., Maxion, R. A., and Wing, J. M., "An approach to measuring a system's attack surface," Tech. Rep. CMU-CS-07-146, Carnegie Mellon University (2007).

[10] Ingols, K., Lippmann, R., and Piwowarski, K., "Practical attack graph generation for network defense," in [*Computer Security Applications Conference, 2006. ACSAC '06. 22nd Annual*], 121 –130 (Dec 2006).

[11] Mehta, V., Bartzis, C., Zhu, H., Clarke, E., and Wing, J., "Ranking attack graphs," in [*Proceedings of Recent Advances in Intrusion Detection (RAID)*], (September 2006).

[12] Carin, L., Cybenko, G., and Hughes, J., "Cybersecurity strategies: The QuERIES methodology," *IEEE Computer* **41**, 20–26 (2008).

[13] Anderson, R. J., [*Security Engineering: A Guide to Building Dependable Distributed Systems*], Wiley Publishing, Hoboken, NJ (2008).

[14] Chess, B. and West, J., [*Secure programming with static analysis*], Addison-Wesley, Upper Saddle River, NJ (2007).

[15] Bell, T. E., "Managing murphy's law: Engineering a minimum risk system," *IEEE Spectrum* , 24–27 (June 1989).

[16] Defense Systems Management College, [*Test and Evaluation Management Guide*], U.S. Department of Defense (1988).

[17] Guzie, G. L., "Vulnerability Risk Assessment," Tech. Rep. ARL-TR-1045, Army Research Laboratory (2000).

[18] Falliere, N., Murchu, L. O., and Chien, E., "W32.Stuxnet Dossier," tech. rep., Symantec (2011).

[19] Cohen, L. E. and Felson, M., "Social change and crime rate trends: A routine activity approach," *American Sociological Review* **44**(4), 588–608 (August 1979).

[20] Schum, D. A. and Martin, A. W., "Formal and empirical research on cascaded inference in jurisprudence," *Law & Society Review* **17**(1), pp. 105–152 (1982).

[21] Williams, D. J., "A guide to understanding security modeling in trusted systems," in [*Aqua Book*], Patrick R. Gallagher, J., ed., *Rainbow Series*, NCSC (1992).

[22] Landwehr, C. E., "Formal models for computer security," *ACM Computing Surveys* (September 1981).

[23] Saltzer, J. and Schroeder, M., "The protection of information in computer systems," *Proceedings of the IEEE* **63**(9), 1278–1308 (1975).

[24] Schneider, F., "Least privilege and more [computer security]," *IEEE Security & Privacy* **1**, 55 – 59 (Sept.-Oct. 2003).

[25] Whittaker, J. A. and De Vivanco, A., "Neutralizing windows-based malicious mobile code," in [*Proceedings of the 2002 ACM symposium on Applied computing*], *SAC '02*, 242–246, ACM, New York, NY, USA (2002).

[26] Dormann, W. and Rafail, J., "Securing your web browser." http://www.cert.org/tech_tips/securing_browser/ (2008).

[27] Howard, M., "Fending off future attacks by reducing attack surface," (2003).

[28] van Eck, W., "Electromagnetic radiation from video display units: An eavesdropping risk?," *Computers & Security* **4**(4), 269–286 (1985).

[29] Shacham, H., Page, M., Pfaff, B., Goh, E., Modadugu, N., and Boneh, D., [*On the Effectiveness of Address-Space Randomization*], 298–307 (2004).

[30] Guyot, V., [*Smart Card Security*], 329–351, ISTE (2010).

[31] von Ahn, L., Blum, M., Hopper, N., and Langford, J., "Captcha: Using hard ai problems for security," in [*Advances in Cryptology, EUROCRYPT 2003*], Biham, E., ed., *Lecture Notes in Computer Science* **2656**, 646–646, Springer Berlin, Heidelberg (2003).

[32] Lee, E., Neuendorffer, S., and Wirthlin, M., "Actor-oriented design of embedded hardware and software systems," *Invited paper, Journal of Circuits, Systems, and Computers Version 2* (November 2002).

[33] Cover, T. M. and Thomas, J. A., [*Elements of Information Theory*], John Wiley and Sons, New York (1991).

[34] Sheyner, O. et al., "Automated generation and analysis of attack graphs," in [*In Proceedings of the IEEE Symposium on Security and Privacy*], (May 2002).

[35] ODNI, "Federal Cyber Security Research Program." Federal Networking and Information Technology Research and Development Program http://www.nitrd.gov/fileupload/files/NITRDACSAC2010.pdf (2010).

7. APPENDICES

7.1 The *Tenets* and Reverse Engineering of Complex Systems

As discussed in 4.1, the reverse engineering 'inverse problem' is reduced to evaluating the conditional probabilities $P(H_i|E_a, E_c)$ and comparing them for different H_i, selecting the maximum. As in many inference problems, such as image or speech recognition, it is "easy" to empirically or analytically evaluate $P(E_a, E_c|H_i)$ from observations of systems configured as H_i. For example, an attacker can build a system specified by H_i and run experiments resulting in various observations E_a and E_c of the system.

Converting $P(E_a, E_c|H_i)$ into $P(H_i|E_a, E_c)$ is conventionally done using Bayes' Theorem according to:

$$P(H_i|E_a, E_c) = \frac{P(E_a, E_c, H_i)}{P(E_a, E_c)} = \frac{P(E_a, E_c|H_i)P(H_i)}{P(E_a, E_c)}$$

where further

$$P(E_a, E_c) = \sum_{i=1}^{k} P(E_a, E_c, H_i) = \sum_{i=1}^{k} P(E_a, E_c|H_i)P(H_i)$$

for each $i = 1, ..., k$ because H_i is treated as a random sample space according to our construction. (That is, only one H_i can be the outcome of the experiment and, from the attacker's point of view, the different possible system constructs are random.)

Assuming conditional independence of E_a and E_c, namely

$$P(E_a, E_c|H_i) = P(E_a|H_i)P(E_c|H_i)$$

(the Naive Bayes modeling assumption), the problem further reduces to estimating the three quantities:

$$P(E_a|H_i)\ ,\ P(E_c|H_i)\ ,\ P(H_i).$$

These quantities may be used in Section 5 to support quantitative metrics about system security. Furthermore, estimating these three quantities is required to understand a system's vulnerabilities and exploits. Referring back to Figure 3 and employing our definitions of the reverse engineering process we define:

$$V_i \triangleq E_a \cap E_c \cap H_i.$$

Hence the probability of a complex system vulnerability can be expressed as:

$$P(V_i) = P(E_a, E_c, H_i) = P(E_a|H_i)P(E_c|H_i)P(H_i).$$

7.2 Analyzing Security Using Partial Knowledge of System Construction

In this section of the Appendix, we derive a technique for estimating the probability that a system is constituted in a specific manner, H_i.

Recalling the framework developed in 4.2, Lee's "relations" can be understood by examining the simplified platform framework in Figure 4. The lettered sets identified in Figure 4 are subsets of their respective design platform. Set A may represent a set of functional designs for a specified system application over a platform of all possible system applications. The relation, R_4, expresses what Lee calls the "design framework", a set of n-tuple design mappings that connect the functional design at one level of abstraction through other platform design sets (C and E) to a physically realizable design set G. We note that for the platforms of Figure 4 the relations are

$$\begin{aligned} R_1 &\subseteq A \times B \times D \times F \\ R_2 &\subseteq A \times B \times D \times G \\ R_3 &\subseteq A \times B \times E \times G \\ R_4 &\subseteq A \times C \times E \times G \end{aligned}$$

This expression of system construction (and hence functionality) can be used to estimate $P(H_i)$. For the example in Figure 4 we have

$$P(H_i) \approx \frac{|R_i|}{\sum_{k=1}^{4} |R_k|} \ ,$$

where naturally $|R_i|$ is the count of n-tuple members of that set. It is our assertion that the number of members in the platform sets can be reasonably estimated based on analysis of open source technology reuse, commercial technology licensing, and/or the number of vendors offering proprietary technology solutions. As indicated earlier, with analysis of H_i, we can simulate the system to estimate $P(E_a|H_i)$ and $P(E_c|H_i)$ and compute $P(V_i)$.

7.3 Analyzing Security by Observing the System

As proposed in Section 4.3, the uncertainty about hypotheses, H_i, can be quantified by the entropy, $\mathcal{H}$, of the probability distribution, $P(H_i|E_a, E_c)$.[33] In our case, entropy is by definition

$$\mathcal{H}(P(H_i|E_a, E_c)) = \sum_{e_a \in E_a, e_c \in E_c} \mathcal{H}(P(H_i|e_a, e_c))P(e_a, e_c)$$

where e_a and e_c are specific evidence values obtained within the evidence class of sets E_a and E_c. The uncertainty in hypotheses about system composition and functionality is larger when the entropy is larger and in fact the error in predicting which hypothesis is true has a lower bound expressible in terms of the entropy (see Fano's inequality for example[33]).

The basic conditional entropy inequality

$$\mathcal{H}(P(H_i|E'_a, E'_c)) \leq \mathcal{H}(P(H_i|E_a, E_c))$$

where $E_a \subset E'_a$ and $E_c \subset E'_c$ states that the more evidence there is about access points, system processes, and data I/O, the less uncertainty there is in hypotheses about the system.[33]

This clearly suggests that the number of access points, namely the number of elements in E_a, is inversely related to the uncertainty in hypotheses about the complex system under study. Similarly, the number of system processes or data I/O observations, namely the number of elements in E_c, is also inversely related to the uncertainty in hypotheses about the system.

7.4 The *Tenets* and Attack Graphs against Complex Systems

In this part of the Appendix, we develop the relationship between the *Tenets* and various attack graph formalisms as suggested in Section 4.4.

In the probabilistic attack graph framework,[11,12] we let S denote the set of states (nodes) within the attack graph and Γ denote attack techniques. For $\alpha, \beta \in S$ and $\gamma \in \Gamma$, let $p^{\gamma}_{(\alpha,\beta)}(t)$ be the probability density function for the random variable, t, which is the time it takes the attacker being modeled to proceed from state α to state β using attack technique γ. Clearly, $p^{\gamma}_{(\alpha,\beta)}(t)$ depends on the system and the attacker's capabilities.

For some α, β and γ, $p^{\gamma}_{(\alpha,\beta)}(t)$ may not be classically defined because γ cannot advance the attacker from state α to state β. We cover that situation by extension, defining $p^{\gamma}_{(\alpha,\beta)}(\infty) = 1$ so that the probability of any finite-time attack being successful is zero. Techniques for estimating $p^{\gamma}_{(\alpha,\beta)}(t)$ using information markets and dynamic programming have been developed.[12]

1. *Tenet 1: Focus on What's Critical* - This tenet asserts that if β_1 and β_2 are two states corresponding to the compromise of two different mission critical components, then the expected time, T_M, to compromise/defeat the mission satisfies

   $$T_M \leq \min\{T_{\beta_1}, T_{\beta_2}\}$$

   where

   $$T_{\beta_i} = \min_{\gamma} \int_0^{\infty} t \cdot p^{\gamma}_{(\alpha,\beta_i)}(t)dt$$

   is the minimal expected time to reach state β_i using any attack technique. That is, if the attacker has more choices for reaching a state which compromises mission functionality, his expected success time cannot increase. This follows by observing that the minimum of a set of numbers does not increase as the set increases.

2. *Tenet 2: Put Key Assets Out-of-Band* - This asserts that $p^{\gamma}_{(\alpha,\beta)}(\infty) = 1$ for all γ when the channel from α to β is inaccessible to an attacker; that is, there is no technique for reaching attack state β from α. The more such degenerate states there are in this sense, the fewer paths there are in the attack graph that can be used to minimize as argued in *Tenet* 1 above.

3. *Tenet 3: Detect, React, Adapt* - This *Tenet* is an assertion about the nonstationarity of $p^{\gamma}_{(\alpha,\beta)}(t)$, namely that it depends on the total time, and attack path as perceived by the defender. For instance, the defender can implement defenses that reset the time to 0 forcing an attacker to restart or backtrack their attack effort.