LOCKHEED MARTIN

# A Threat-Driven Approach to Cyber Security

*Methodologies, Practices and Tools to Enable a Functionally Integrated Cyber Security Organization*

Michael Muckin, Scott C. Fitch
Lockheed Martin Corporation

## Abstract

Contemporary cyber security risk management practices are largely driven by compliance requirements, which force organizations to focus on security controls and vulnerabilities. Risk management considers multiple facets – including assets, threats, vulnerabilities and controls – which are jointly evaluated with the variables of probability and impact. Threats cause damage to information systems. Threats utilize vulnerabilities to enact this damage, and security controls are implemented to attempt to prevent or mitigate attacks executed by threat actors. The unbalanced focus on controls and vulnerabilities prevents organizations from combating the most critical element in risk management: the threats. This unbalanced condition is manifested as incident response processes rather than threat intelligence management in the analyst realm, adherence to predefined standards and policies in security architecture and engineering practices, and compliance verification in the operational domain.

A functionally integrated cyber security organization is structured to place threats at the forefront of strategic, tactical and operational practices. Architects, engineers and analysts adhere to a common methodology that incorporates threat analysis and threat intelligence across systems development and operational processes. This ensures security controls are implemented, evaluated and adjusted over time per the most impactful threats and attack vectors. The resultant risk management practices are enhanced due to a higher fidelity of information regarding current state security postures. This drives improved resource allocation and spending, and produces an agile and resilient cyber security practice. When this threat-driven approach is implemented along with tailored compliance processes, organizations can produce information systems that are both compliant and more secure.

**Keywords**: threat modeling, attack trees, threat profiles, threat intelligence, threat and risk, security controls, cybersecurity, compliance

## Table of Contents


Abstract ..... 1

1. Introduction ..... 3

2. The Threat-Driven Approach ..... 4

Elements of the Threat-Driven Approach ..... 5

Threats-Assets-Controls Relational Model ..... 5

A Common Threat Analysis Methodology ..... 6

Cyber Security Thesis ..... 7

There Are No Idle Threats – They Attack ..... 7

Integrating IDDIL/ATC ..... 9

Threat Analysis Practices and Tools ..... 12

Categorizing Threats ..... 12

Threat Models ..... 13

Attack Trees ..... 17

Threat Profiles ..... 19

Summary of Practices ..... 23

Threat Intelligence ..... 23

Tactical Analysis Integration ..... 24

Focus on the Largest Threats ..... 25

3. Controls ..... 26

Current-state Challenges ..... 26

The Integrated Solution ..... 28

Selecting and Implementing Controls ..... 28

Functional Controls Hierarchy ..... 28

Evaluating Controls Effectiveness ..... 32

Attack Use Cases ..... 32

Controls Effectiveness Matrix ..... 32

Controls Effectiveness Scorecard ..... 34

Architectural Rendering ..... 34

4. The Integrated Threat-Driven Approach ..... 35

5. Risk Management ..... 38

Risk Lifecycle ..... 40

Risk Management and Risk Assessment ..... 40

6. Summary ..... 41

Definitions of Terms ..... 42

References ..... 43

# 1. Introduction

Current-state architecture, engineering and operational practices in the cyber security domain focus largely on compliance to one or many regulations, directives, policies or frameworks. Some organizations augment these practices by incorporating traditional information security concepts and principles, and attempt to "build security in" to the development of IT systems, while the operational domain provides security services, detects and responds to incidents, and analyzes collected data to identify trends and patterns to improve existing security controls and services. Mature operational organizations adhere to the Cyber Kill Chain® (CKC) or a similar practice and leverage the Intelligence Driven Defense® [1] (IDD) approach to combat cyber threats.

Three primary gaps in this current state limit its effectiveness:

1. The behaviors, culture and the excessive amount of resources allocated to implementing and adhering to compliance requirements
2. The lack of formalized threat modeling and analysis practices that scale vertically and horizontally
3. The lack of institutionalized integration between the architecture/engineering functions and the operational/analyst functions.

Expanding on these limitations, compliance-driven strategies most often result in a controls-first mindset where systems architecture and foundational processes are driven by known sets of security controls or control frameworks. The results of this approach are described below:

- Compliance with a list of controls – although mandated by appropriate authority – does not assure a secure system or environment, propagating a false sense of security
- Resources are wasted on controls that do not address actual threats
- Measurement of controls effectiveness is often evaluated as a binary condition
- Analysis that would identify these issues is not performed
- Residual risk is elevated

Additionally, there is often excessive emphasis of effort on vulnerabilities, or a vulnerability-driven approach. A vulnerability-driven approach has the following deficiencies:

- Indicates a highly reactive operational environment
- Vulnerabilities and incidents are handled at a micro level rather than addressing larger scale threat scenarios and patterns
- Only known vulnerabilities can be corrected; unknown vulnerabilities or systemic design flaws are neglected
- Vulnerability metrics are misinterpreted without additional context, driving unnecessary behaviors and improper resource allocation
- Leads to gaps in architecture and operations in the areas of detect, respond and recover – due to an unbalanced focus on prevention

Threats (whether defined as people or events) are what do damage to systems and assets. Therefore, threats must be the primary driver of a well-designed and properly defended application, system, mission, environment or enterprise. This is labeled the *threat-driven approach,* the approach advocated in this paper. This approach will provide detailed guidance that will enable organizations to place threats at the forefront of planning, design, testing, deployment and operational activities.

## 2. The Threat-Driven Approach

The threat-driven approach is a *methodology*, a set of *practices* and a *mindset*. The primary purpose of this approach is to enable organizations to allocate the commensurate level of resources to defend their assets, to develop the inherent skills needed to support these efforts, and to align groups and teams into functional roles that will implement this approach. As presented in Figure 1, the architecture/engineering and operations/analyst functions are typically isolated from each other, preventing effective intelligence sharing, fragmenting strategic cyber security efforts, failing to provide adequate markers to drive roadmaps and strategic programs, and fostering a culture that desires to address cyber threats head-on but is unequipped to do so.

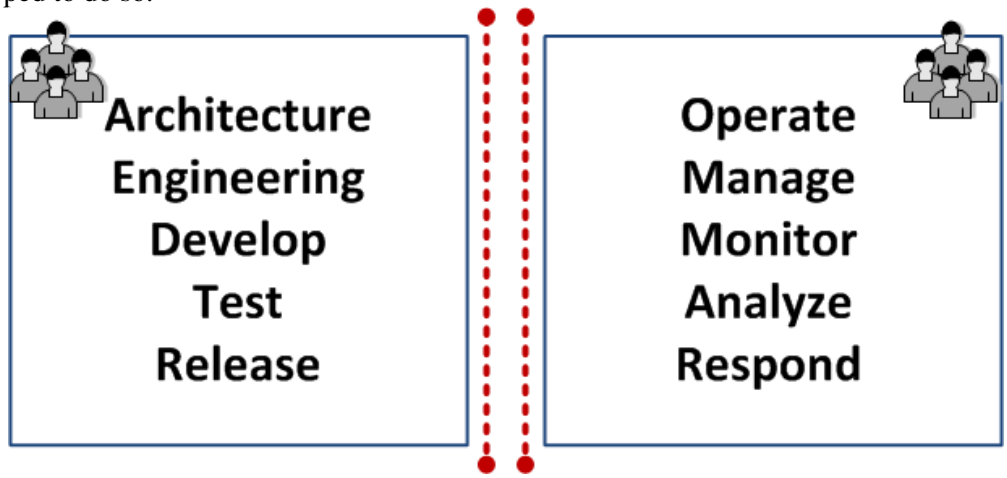


Figure 1 - Segmented Cyber Functions

Figure 1 illustrates the typical hard boundaries that exist – functionally and organizationally – between architecture/engineering and operations/analysts. These boundaries must be broken down and replaced with an integrated approach that links the most relevant threat-related elements from each respective domain into the reciprocal domain. Figure 2 depicts this preferred state. Ideally this crossover linkage would be accomplished via organizational and functional alignment within the enterprise and supported at all levels of management.

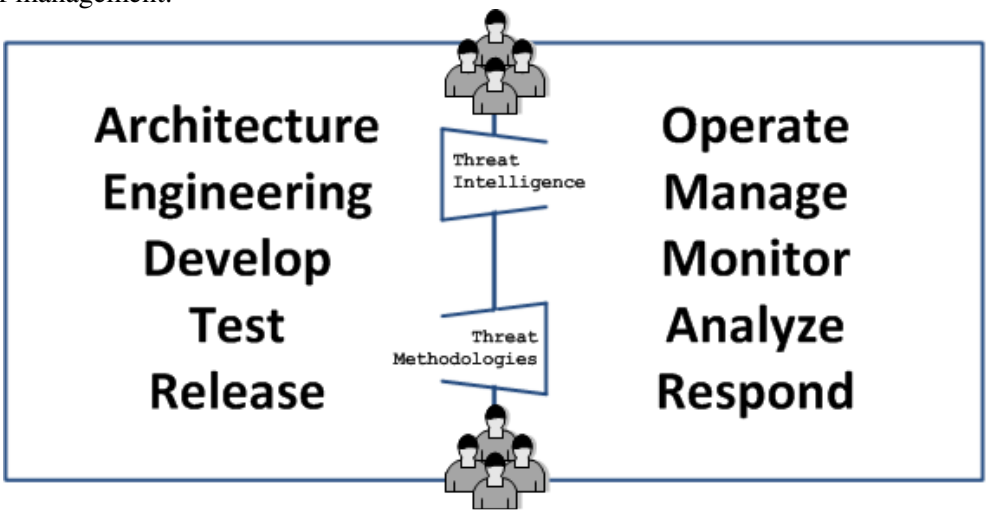


Figure 2 - Integrated Threat-Driven Approach

Figure 2 shows the necessary crossover elements and from which functional domain they are sourced. The operations domain feeds relevant threat intelligence into the architecture and engineering practices, and the architecture and engineering domain consumes that intelligence and adds threat models and

analysis (i.e. threat methodologies) to evolve the infrastructure, operational services/capabilities and overall security posture. Applying these concepts bridges the gap between these segmented functional domains and enables a robust, agile and proactive set of cyber security capabilities. Loosely speaking, this could be considered a "DevOps[1]" approach to cyber security.

## Elements of the Threat-Driven Approach

The *methodology* presented will provide guidance on bridging the gap between these two domains of practice and establish a set of unified threat analysis touchpoints.

The *practices* described will provide guidance on performing threat analysis activities in support of systems' development, threat/risk assessment projects, incident analysis, or evaluation of the effectiveness of security control sets. Within these practices, numerous tools will be presented and described.

The *mindset* espoused here – when adopted – will drive change in the cyber security/information security industry by adjusting the behaviors resulting from compliance-driven practices which have proven ineffective and inefficient in defending against the onslaught of current and future cyber threats. The goal is to produce systems that are *secure and compliant*.

Any discussion of cyber security threat practices must have one ultimate goal: *effective risk management* at all levels – from a single application to the entire organization. This paper will provide detailed guidance on how this can be accomplished.

Finally, since high quality threat analysis work is equal parts art and science, this paper will include both *descriptive* and *prescriptive* guidance.

## Threats-Assets-Controls Relational Model

The conceptual foundation of the threat-driven approach is a model of the relationship between threats, assets and controls. See [2] for definitions of terminology used in this paper.

This relationship model, as illustrated in Figure 3, is described as follows: threats target assets, which are almost universally found in one or more components of technology (within the cyber and networked systems[3] context). The threat actor(s) gain access to the assets via attack vectors and vulnerabilities present in the technology components that house or provide direct access to the targeted assets. Security controls are applied to the technology components with the intent to counter or mitigate the vulnerabilities and/or attack vectors used by the threat actors, thereby protecting the assets. This relationship highlights the significance of the threat perspective within this model.

[1] DevOps defined: http://theagileadmin.com/what-is-devops/

[2] Definitions of Terms section of this paper

[3] Networked systems includes all technology stacks and implementations outside of traditional IT systems

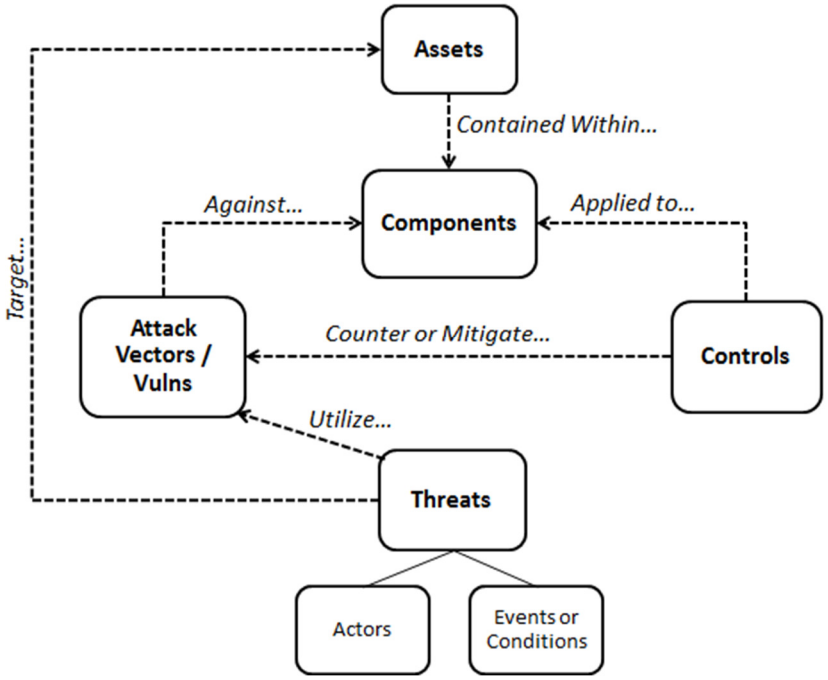


**Figure 3 - Threats, Assets and Controls Relationship Model**

Given these relationships, threat actors do not (or very rarely) directly access the targeted assets; they must interact with and circumvent other elements of the system to obtain their objectives against the assets. Therefore, controls are not directly aligned to assets. Instead, controls must provide a security function, directly aligned to the identified threats, attack vectors and vulnerabilities that provide access to the components that contain the assets. This is a fundamental principle: controls must be selected and implemented to address threats and attack vectors by performing one or more *functions*[4]. When threat intelligence is included in this model, architects, engineers and analysts can work in unison to identify potential gaps and assess degrees of effectives, thereby continuing to enhance the security posture of systems and infrastructure. When threat modeling and analysis is introduced in this model, potential areas of exposure and impact are highlighted which enhances the selection and implementation of controls.

## A Common Threat Analysis Methodology

The two primary goals of threat analysis are:

1. To provide a clear and thorough articulation of assets, threats and attacks to facilitate business/mission-relevant dialog and decision-making actions regarding risk level determination and risk management practices.
2. To select, implement, evaluate and determine gaps in security controls at the application, system, infrastructure and enterprise levels.

Numerous threat analysis practices and tools exist in today's cyber domain [2] [3] [4] [5] [6] [7] [8]. Some are tailored for development/engineering activities [2] [4] [6] [9], some are more appropriate for assessment work [5], and still others are applicable to operational defense and analysis [1] [8].

The methodology introduced in this paper was developed from the experiences collected and refined over a span of almost two decades. These experiences include countless information security architecture/engineering projects, and threat and risk assessments performed on software development

[4] Control functions will be described in Controls section of this paper

projects, complex IT systems, large scale data centers and non-IT networked systems. Tactical support of incident response activities and threat intelligence development is also a major portion of the experience base that shaped this methodology. This unified methodology spans all these use cases, and scales equally well vertically and horizontally.

### Cyber Security Thesis

Classic Systems Engineering practices do not effectively translate to cyber security practices. Development of secure systems – per the threat-driven approach – is very closely related to FMEA/FMECA (failure mode effects analysis/failure mode effects and criticality analysis) and other *fault analysis practices* used for quality and reliability engineering [10]. This supports the belief that highly secure systems are a corollary indicator of high-quality systems, a viewpoint the authors of this paper advocate.

### There Are No Idle Threats – They Attack

There is a mnemonic to help remember this methodology: "*There are no idle (IDDIL) threats – they attack (ATC)*". There are two phases of work within this methodology: IDDIL is considered the *discovery phase* and ATC is considered the *implementation phase*. The phases and their corresponding activities are listed below:

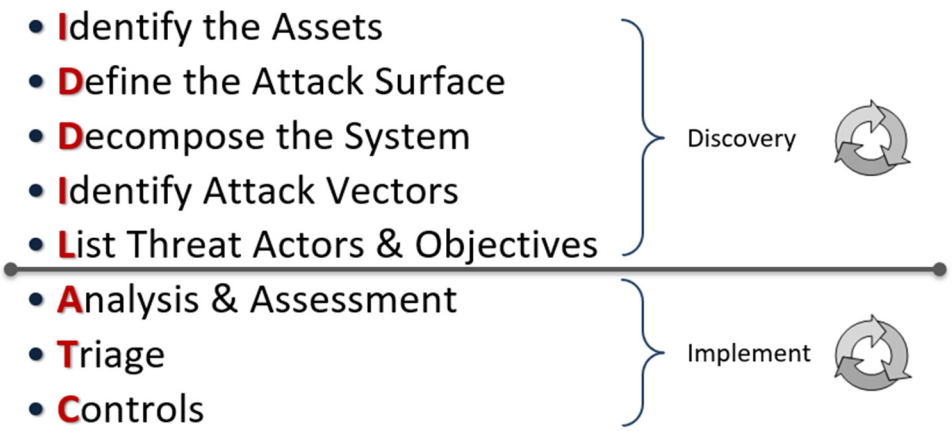


**Figure 4 - The IDDIL/ATC Methodology**

Before describing the detailed activities for each phase, the following general guidelines on how to perform the work are provided:

- **Business/mission context** – Ensure there is an understanding of the business/mission context and impact to business/mission objectives when performing this work.
- **Mindset** – The team performing the threat analysis must have the skills and capacity to *think like an attacker*. This trait is critical and directly corresponds to the mindset element presented in the description of the threat-driven approach.
- **Iterative** – These activities do not need to be sequential. An iterative approach is recommended, and some tasks can be performed in parallel. Completion of all tasks in the methodology is more important than the order in which they are performed. When considered from an enterprise/program/organization perspective versus a discrete project, iterative activities dictate a longer cycle of time and a deeper degree of analysis and integration.
- **Brainstorming** – To be effective and thorough, the methodology must be a group exercise with proper representation from business, mission and technology stakeholders. Assumptions will be

necessary and should be documented for follow up. Capture all suggested ideas regarding attacks and weaknesses – they will be prioritized later.

- **Time-bounded** – Limit the length of time of both individual sessions and overall assessment activities to maximize value versus time spent. This timeframe will vary based on scope and criticality of projects. However, it is necessary to establish time limits for effective project management.

### *Discovery Phase (IDDIL)*

- Identify the Assets – Assets include two major elements: 1. Data, components or functionality that are essential for the business mission of the system are known as business assets – and 2. Data, components or functionality that is of special interest to an attacker are known as security assets. They may not always be the same. Identify the assets of the system by soliciting input from the appropriate role(s) that provide business context. Obtain current threat intelligence about adversary targeting efforts and objectives. Document asset types and specify the locations where these assets reside within the system or environment.

- Define the Attack Surface – Once the assets have been identified, map out at a macro level the components/elements of the application/system/environment that contain, communicate with or otherwise provide some form of access to the assets. Follow the assets identified in the first step as a guide to determining the attack surface. The attack surface will help define system and trust boundaries, span of control and responsibility, and drive what is in and out of scope for any particular piece of work. Typically a data flow diagram (DFD), or a set of DFDs, or any equivalent type of diagram that best represents the system or environment under analysis, is produced during this phase of work

- Decompose the System – Use the information gathered in the first two activities to decompose the application/system/environment into a layered view. Use the as-designed use cases of the system to drive the discussion. Include technology details such as devices, interfaces, libraries, protocols, functions, APIs, etc. to complete the descriptions. Identify components or services responsible for security functions such as inventory, collect, detect, protect, manage and respond. Review existing effectiveness ratings[5] for security controls (either conceptual for design-time or existing for assessments) that are within the scope of work.

- Identify Attack Vectors – Leverage the documented attack surface, decomposed system and primary use cases to document paths of attack. Capture the components and areas of functionality included in these paths, including existing security controls and services. In addition to the physical or logical paths, consider multiple methods of attack utilizing the same pathways. Ensure current information and intelligence regarding exploits, vulnerabilities and threat actors is included in this phase. *Attack Trees*[6] are an ideal practice/tool to employ for this work. At this point in the process, categorize the threats and attacks using taxonomy appropriate to the system and organization.

- List Threat Actors/Attack Agents & their Objectives – Determine what entities would want to attack this system, and why. Include characteristics such as motivation, skill levels, resources and objectives in this analysis and list them accordingly. Consider how the different threat actor types would attack the target assets. Current threat intelligence is essential for this step.

[5] Control's effectiveness rating are discussed in the Controls section of this paper

[6] Attack trees are covered in the Threat Analysis Techniques and Tools – Attack Trees section of this paper

### *Implement Phase (ATC):*

Whereas the discovery phase identified the assets, threats and attacks, in the implement phase, a thorough analysis is performed. Using the data captured from the discovery phase, a detailed analysis and assessment is performed. This analysis will result in a prioritized listing of items to be addressed, with a full accounting of aggregate impact and business context. All discovery, analysis and prioritization activities will feed the selection of appropriate security controls to counter or mitigate the identified threats and attacks – or identify where gaps may exist in controls effectiveness coverage.

- Analysis – Ensure the *cause* of each threat/attack is well understood. Determine the *impact* those successful compromises produce. Revisit and update assumptions captured during discovery activities. Include any available *threat intelligence* or *indicators*. Ensure the scope and impact discussions include worst-case scenarios, as applicable. Mechanically, this is where threat models, attack trees/graphs, the Cyber Kill Chain® and any other practice or technique are employed as pertinent artifacts. The time and effort devoted to analysis activities is on par with the critically of the assets and business impact of the system under analysis.

- Assessment & Triage – These activities produce a prioritized listing based on the evaluations of the analysis referenced against business/mission objectives, impacts to the critical assets and threat intelligence. This listing includes *resultant conditions* expressed in both business/mission or technical contexts so that risk discussions can be conducted appropriately. Impact is given greater weight than probability at this point in the discussion. Probability is a key element of the overall risk management discussion, which will be discussed later in this document in the *Risk Management* section. However, it is worth noting that probability is considered a constant, and *active threat intelligence* is leveraged as one of the primary feeders to the probability variable.

- Controls – The final activity of IDDIL/ATC is to select and implement security controls to remove, counter or mitigate the threats and attack vectors identified during development or engineering work; or, in instances of assessment work, to evaluate and improve the effectiveness of existing controls. Selection of controls from a predefined list (regardless of how valid or mandatory that list may be) without the previous activities defined in this methodology, will fail to ensure that controls effectively address the threats. By contrast, following this methodology will ensure that the proper controls are selected, and – perhaps more important – *implemented to address the actual threats faced* according to the threat analysis outputs and threat intelligence inputs. Additionally, controls will exhibit certain *functionality*, and the characteristics of that functionality assist the engineer and analyst in determining any given control's degree of effectiveness. The control functions described in this paper are: inventory, collect, detect, protect, manage and respond. Lastly, by implementing this methodology, it is much more likely that gaps in controls coverage will be properly identified, whether these gaps are technological, process related, organizational or an industry-wide gap. The identification of these gaps allows improved identification of potential risk items, which translates into enhanced risk management practices.

## Integrating IDDIL/ATC

This section describes the integration of the IDDIL/ATC methodology into standard engineering and operational processes, including the alignment of the tasks associated with the functional cyber roles of architect, engineer and analyst. This description provides the detailed practices and procedures represented by the crossover integration elements in Figure 2. It is not the intent of this paper to go into specific detail in either domain, since these topics are exhaustively documented in the industry. Rather, this paper will present an overlay of the methodology's integration within these practices and provide detailed descriptions of those integration points.

### *Development and Engineering Integration*

Figure 5 presents a generic engineering lifecycle. This lifecycle does not represent any singular engineering methodology (i.e. waterfall, spiral, agile) but includes the typical phases that comprise any engineering discipline. The IDDIL/ATC phases and activities are overlaid on the engineering lifecycle to illustrate the integration with engineering phases. The integration of *Threat Modeling & Analysis* activities begins at Concept and extends through all phases into Operations. *Threat Intelligence* is garnered from Operations and fed as far back into the engineering phases as possible and practical. The inclusion of threat intelligence input highlights how the threat-driven approach enhances systems engineering and architecture practices. The *Threat Modeling & Analysis* practices and the *Threat Intelligence* practices are continuously evolving as the organization and the threats against it evolve.

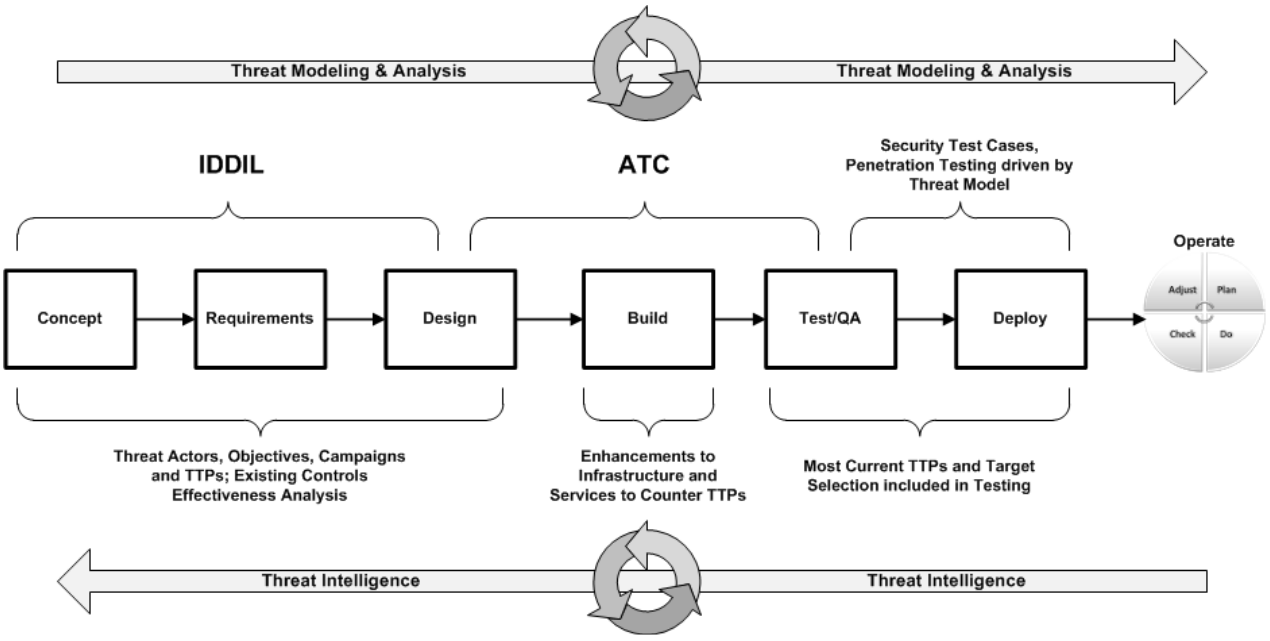


**Figure 5 - Integration of Threat Driven methodology and practices within engineering lifecycle phases**

The IDDIL methodology activities align with the Concept, Requirements and Design engineering phases. Full integration of the IDDIL activities with the corresponding engineering phases is necessary to produce high quality analysis outputs. These activities cannot be back-filled at some future phase without a significant amount of rework. The ATC activities align with the Design, Build and Test/QA phases; architecture decisions regarding security are solidified and refined, - and integrated security controls and services are specified, built and tested. The Triage activity drives the level of rigor, prioritizes design and control decisions and seeds discussions regarding risk acceptance and design tradeoffs. Threat Intelligence is integrated into these phases as specified in the Figure 5 diagram. Additional details regarding the data obtained from Threat Intelligence is provided in the Threat Intelligence section of this paper.

There are several key points to highlight in Figure 5:

- The *Controls* activity (IDDIL/ATC) integrates into the *Design*, *Build and Test/QA* engineering phases, but does not align with *Requirements.* Controls are designed and built, but by themselves are not requirements.
- Security testing and penetration testing are guided by the threat model. A good threat model is a blueprint for a penetration test. Additionally, relevant techniques, tactics and procedures (TTPs)

and/or targeting data available from threat intelligence must be included in testing activities

- Inclusion of current threat intelligence data into concept, requirements and design is a critical component of the threat-driven approach.

### *Affordability and Cost Impact*

It is a well-known fact that it is more expensive to correct an issue the farther right on the engineering timeline that the issue is discovered and resolved. This is especially true for cyber/information security issues. This highlights another major advantage of including threat modeling, analysis and intelligence data as early as possible into the engineering lifecycle: *major design flaws and systemic issues can be identified and corrected earlier, reducing the long term cost of building and maintaining the system*. Once a system is built, it is very difficult, if not impossible, to correct design flaws. No vulnerability scanner or compliance checklist will uncover a systemic design flaw. Unfortunately, most security related activities are initiated only in the latter phases of the engineering lifecycle, which produces a doubly dangerous scenario of potentially higher development and operational costs for the organization, and a much less secure environment.

### *Project Integration*

In addition to integration with engineering processes, the IDDIL/ATC methodology's phases and activities can directly drive a project's tasks, milestones and deliverables. Figures 6 and 7 illustrate example project plan templates based on the IDDIL/ATC activities. Figure 6 is a baseline template and Figure 7 presents a notional implementation.

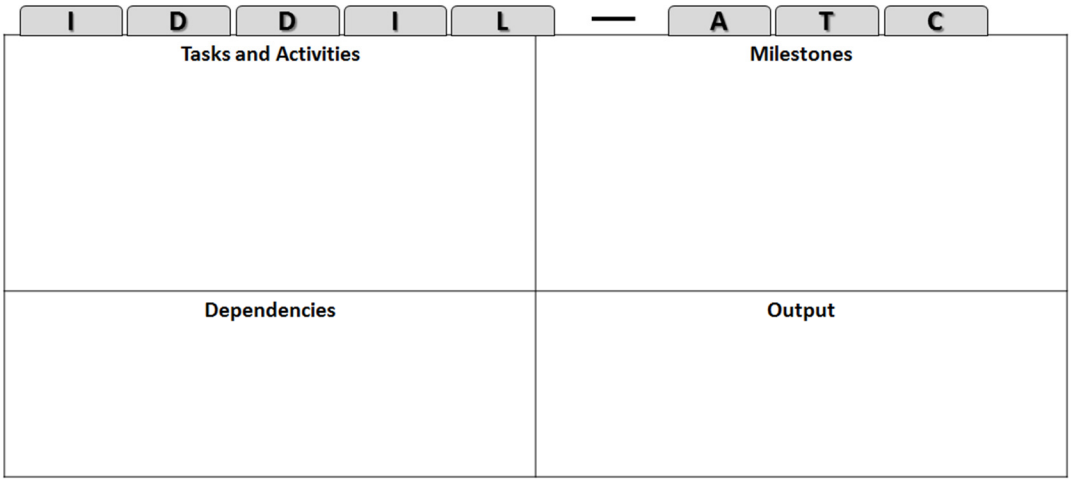


**Figure 6 - IDDIL/ATC Methodology Project Plan Template**

Figure 7 is an example notional second phase of a project, which – in this case – includes the IDDIL/ATC activities: Decompose, Identify attack vectors, List threat actors, Analyze and assess, and Triage. The corresponding first phase would have captured the assets and defined the attack surface, and the final phase would include controls allocation and design. The flexibility of IDDIL/ATC allows the phases and activities to be apportioned to almost any engineering or assessment work; e.g. phase one could have included the **IDD**IL/ATC activities and the second phase would perform the IDD**IL/AT**C activities. Or the project's activities can be divided according to the discovery (IDDIL) and implement (ATC) phases. The methodology can be structured to meet the project's needs.

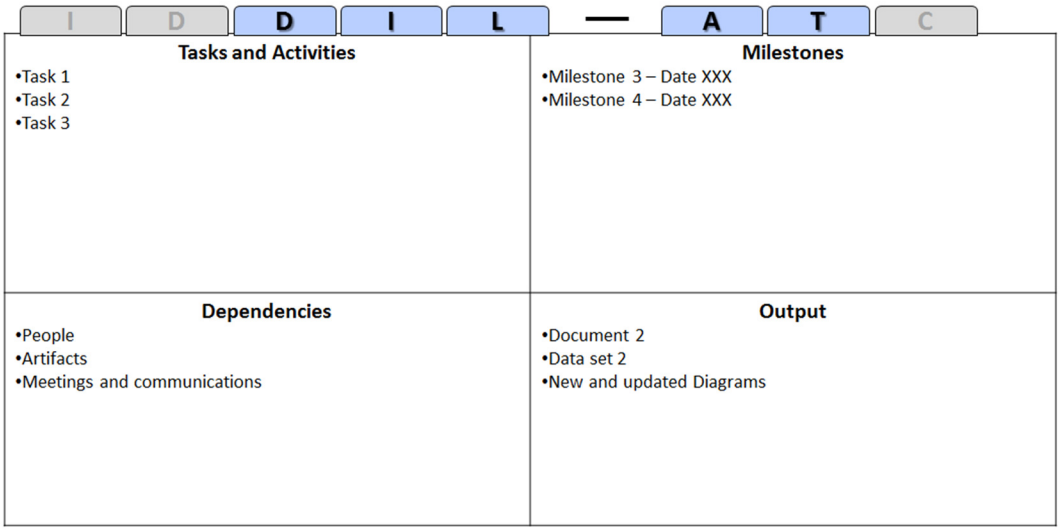


**Figure 7 - Notional "Phase 2" Project Plan integration**

## Threat Analysis Practices and Tools

IDDIL/ATC is considered an overarching threat analysis methodology, as it is encompasses the common activities that must be performed regardless of which practice, technique or tool used. Those supporting practices, techniques and tools include: threat models, attack trees, threat profiles and the CKC. Case studies will be presented to reinforce the concepts and describe the preferred usage of each practice based on comparative analysis. The concept of threat categorization will be described, including how to account for both the *cause and effect of threat types and attack vectors*. This section will not discuss the detailed mechanics of how to perform any of these practices, as that information exists in many locations and is beyond the scope of this paper.

### Categorizing Threats

A common practice performed during threat modeling and analysis is the categorization of threats. Microsoft developed STRIDE [11], while many organizations adhere to the confidentiality, integrity, availability (CIA) descriptors. Beyond these two, which are the most well-known, are various publications and taxonomies that attempt to attribute threats and attack characteristics [12] [13]. The STRIDE model considers the *effect* of each threat type and assumes the cause of each threat will be uncovered during analysis activities. Conversely, MITRE's CAPEC [12] provides details on the *causes* of many common attacks, but provides no guidance on effect. It is the responsibility of the individuals performing threat modeling and analysis to discover and describe *the cause and effect* of threats and attack vectors, which requires the unique context of a system under analysis.

The authors of this paper and the teams they work with have successfully used the STRIDE model in many projects, but have enhanced it by adding one additional threat type: *Lateral Movement (LM)*. STRIDE is primarily focused on software engineering and development, so the concept of lateral movement – which is primarily a system-of-systems type of threat – was not included. The LM addition to STRIDE follows the pattern of describing the effect of a threat. This new threat categorization is labeled STRIDE-LM.

**Table 1 - Threat Categorization, Security Properties and Controls**

| STRIDE-LM | Threat | Property | Definition | Controls |
|---|---|---|---|---|
| **S** | Spoofing | Authentication | Impersonating someone or something | Authentication Stores, Strong Authentication mechanisms |
| **T** | Tampering | Integrity / Access Controls | Modifying data or code | Crypto Hash, Digital watermark/ isolation and access checks |
| **R** | Repudiation | Non-repudiation | Claiming to have not performed a specific action | Logging infrastructure, full-packet-capture |
| **I** | Information Disclosure | Confidentiality | Exposing information or data to unauthorized individuals or roles | Encryption or Isolation |
| **D** | Denial of Service | Availability | Deny or degrade service | Redundancy, failover, QoS, Bandwidth throttle |
| **E** | Elevation of Privilege | Authorization / Least Privilege | Gain capabilities without proper authorization | RBAC, DACL, MAC; Sudo, UAC, Privileged account protections |
| **LM** | Lateral Movement | Segmentation / Least Privilege | Expand influence post-compromise; often dependent on Elevation of Privilege | Credential Hardening; Segmentation and Boundary enforcement; Host-based firewalls |

Table 1 presents the STRIDE-LM categorization model, which includes *definitions*, the corresponding security *property* and default *controls* associated with the threat type. The threat model will identify the primary threat types, which then directly reference a corresponding property and control type. For example, the primary threat of *Information Disclosure* (STR**I**DE-LM) has the corresponding security property *Confidentiality* and the control types *Encryption or Isolation*. The default entries in the Controls column will directly reference the *Functional Controls Hierarchy*[7] and the corresponding listings for *Category* and *Implementation* within this hierarchy. This direct reference from threat categorization to controls is a foundational concept of the threat-driven approach.

## Threat Models

A threat model is a visual representation of four main elements:

1. The assets within a system (**I**DDIL)
2. The system's attack surface (I**D**DIL)
3. A description of how the components and assets interact (IDD**I**L)
4. Threat actors who could attack the system and how the attack could occur (IDD**IL**)

The adoption of threat modeling is gaining momentum in the industry. Many enterprise organizations and government agencies perform a version of this practice [3] [9] [7] [14] [15]. However, there is not yet a standard format for documenting and communicating threat models nor a consensus on how threat modeling is applied to multiple types of projects – i.e. software development vs. building a data center. Additionally, there are numerous tools to aid in the development of threat models [16] [17] [18]. This paper is not advocating any one technique, tool or taxonomy. The selection and usage of any one tool, and learning the tool – vs. the methodology – constrains the effectiveness of the threat modeling and analysis activities. Each organization/group/team should use the techniques, tools or taxonomy that best fit its business/mission needs, while leveraging IDDIL/ATC as the parent methodology. The IDDIL/ATC methodology provides flexibility and the ability to scale as needed when conducting threat analysis projects.

[7] Covered in the Functional Controls Hierarchy section of this paper

Structurally speaking, threat models can take on a wide variety of formats, those formats determined by a combination of a) the tool being used to construct the model, and b) the scope, context and intent of the output of the model. Figures 8, 9 and 10 are threat model case studies. The threat model in Figure 8 is a larger scale "system of systems" top-level threat model, Figure 9 is a system-level model built using a standard data flow diagram (DFD) format with one of the more popular threat modeling tools [16] and Figure 10 is a threat model of a software application [19]. Although each of these threat models appears unique, there are commonalities across them:

- The assets are articulated
- The flow of sensitive data/assets determines the structure of the diagram
- Trust boundaries, attack surface and attack vectors are clearly identified
- In Figures 8 and 10, threat actors are enumerated

The threat model in Figure 8 was an assessment project, Figure 9 was a systems integration project performed during system design, and Figure 10 was a software development project utilizing the agile process. None of these diagrams resemble a network diagram or other systems architecture diagram. While other architecture diagrams are extremely valuable in gathering the relevant information to complete a threat model, they are not preferred as the actual artifact for documenting and communicating the threats and attack vectors within a system.

Threat models provide great value in identifying the intercommunications of major components of the system and illustrate where critical data can and will exist within the system. By identifying the functional and logical (vs. physical) interfaces between components, attack vector enumeration becomes apparent. Although not illustrated here, UML (unified modeling language) component and sequence diagrams have been used in threat model development where analysis of complex algorithms and state changes require full decomposition. The key takeaway is: use the most appropriate diagraming tool for the threat analysis that needs to be performed, and ensure all activities within the IDDIL/ATC methodology are completed.

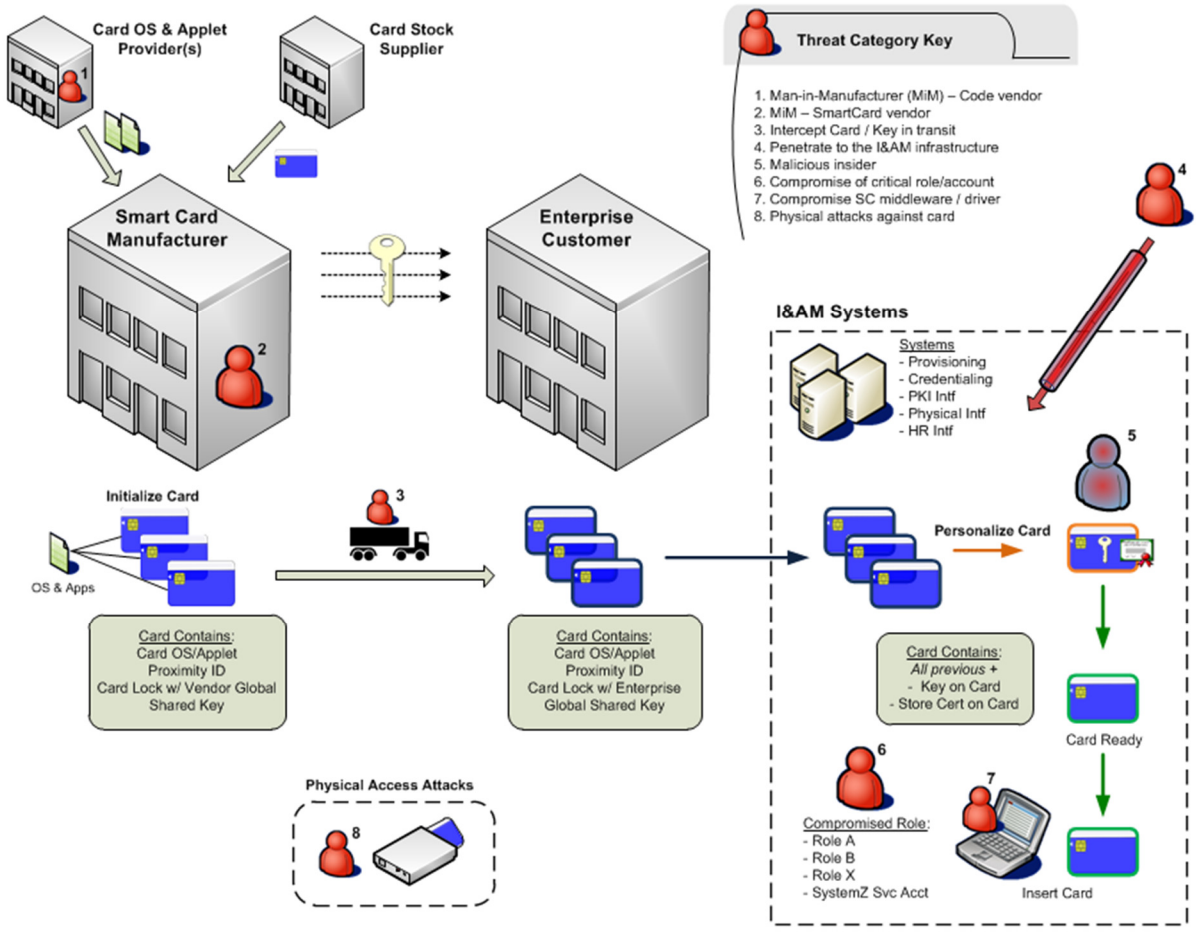


Figure 8 - Example Threat Model of a smart card ecosystem

Figure 8 presents a threat model of a smart card ecosystem. This is a top-level diagram that is further supported by numerous, sub-level diagrams that contain more granular decomposition and analysis (not shown). This threat assessment's results led to significant enhancements to this environment's infrastructure security controls, modified key operational processes and triggered penetration testing activities to determine the presence and magnitude of potential flaws in specific components. Beyond this, it enabled an informed decision on risk management at the executive level regarding the "man-in-the-manufacturer" threat and attack vector.

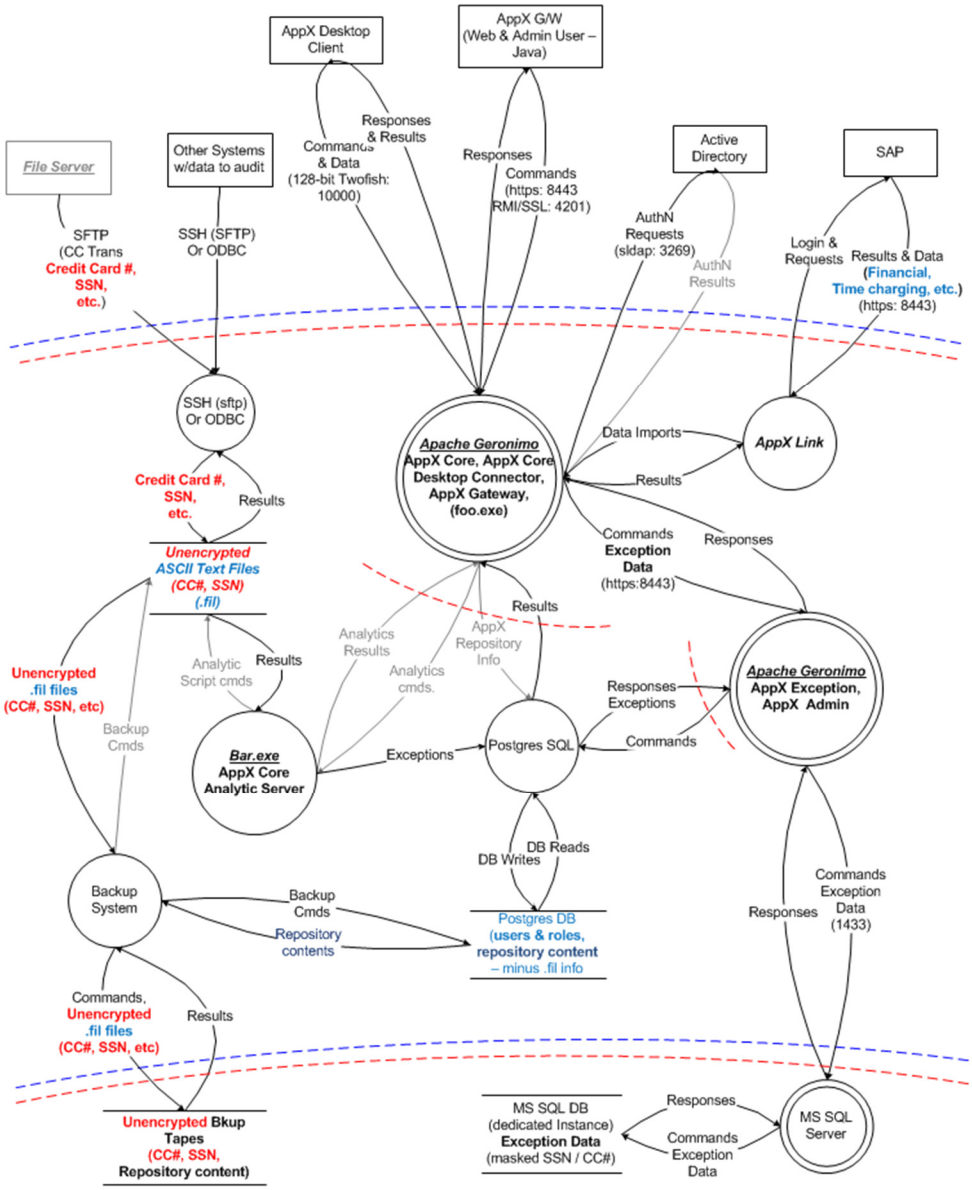


**Figure 9 - Classic DFD Threat Model**

Figure 9 illustrates a threat model of a financial audit system. This threat model was built during a systems development and integration project. It enabled the engineering team to identify and address the most significant threats. The resultant threat-driven controls included a combination of technical security controls and procedural controls. This particular model also demonstrated the *value of a threat model as an artifact*. Three years after the original threat model was built, the corresponding system was migrated to a new hosting environment and none of the original engineering team was associated with the system at that time. However, this threat model was included in the system's documentation library, and it allowed the migration engineers to specify the same types of controls in the new environment without unnecessary rework. Threat models are extremely valuable as historical artifacts. As historical artifacts, they establish a baseline for any future analysis, which could include changes to the system, environment, or the nature of the threats and attacks against the system.

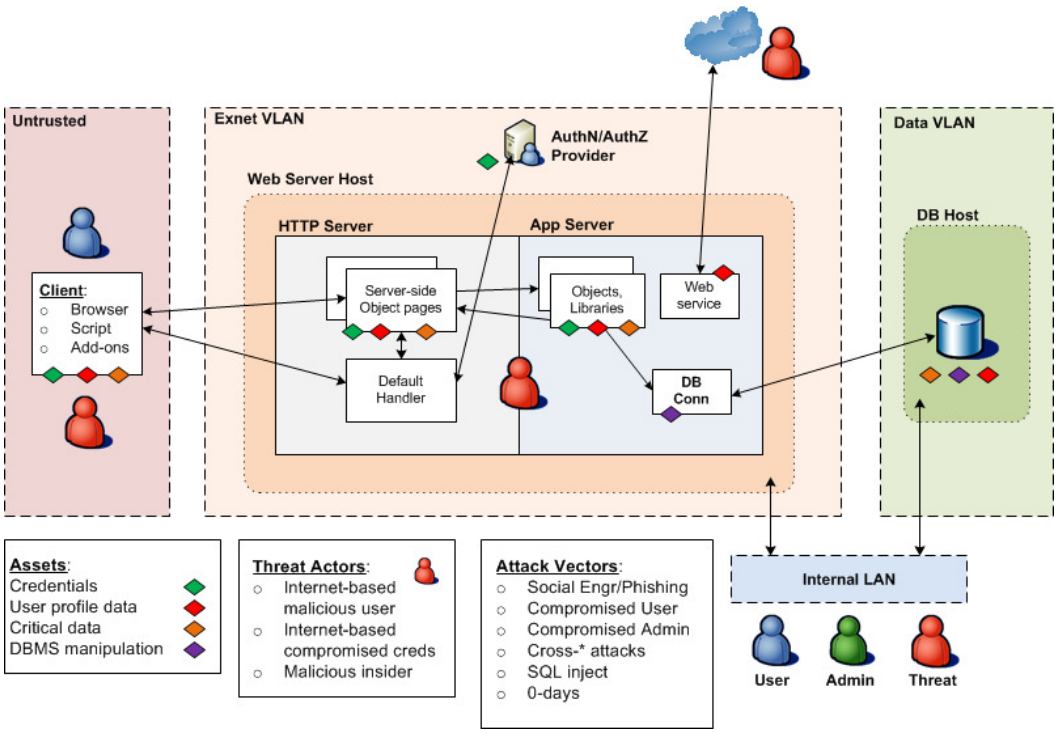


**Figure 10 – Web Application Threat Model**

Figure 10 presents a threat model of a common n-tier web application software development project. Note the inclusion of each of the IDDIL elements of in this diagram. Software development threat models allow development teams to identify where the key assets exist within the system, and how they traverse the system, so the proper controls can be built/implemented at the appropriate locations. Missing from this diagram is the assumed management infrastructure provided by the hosting environment. This frequently occurs in application level threat models and it is one of the reasons that *lateral movement* threats are often overlooked. Proper implementation of the I**DDIL/**ATC activities will prevent this critical oversight from occurring and further emphasizes the need for the integrated threat-driven approach advocated in this paper.

## Attack Trees

Attack Trees were borrowed from fault analysis trees used in other engineering disciplines and were first introduced into the information security industry in 1999 [20]. Attack trees are an excellent tool for decomposing and identifying attack vectors (ID**DI**L). In general, when comparing a threat model to attack trees, the threat model will describe the "*what*" and the attack tree will provide details on the "*how*". It should be stated that in the IDDIL/ATC methodology, usage of both threat models and attack trees as complementary modalities is assumed, except in cases where it would provide no additional value to do both.

Construction of an attack tree will produce a logical, hierarchical, graphic decomposition of attack paths and conditions necessary for a particular threat to be realized or attack to be successful. The branches and nodes on the tree represent paths of attack with chained events required for the full attack to occur. This enables the engineer or analyst to fully understand the attack vectors and unique conditions necessary for the attack to succeed. In this manner, security controls are directly aligned to branches/nodes in the tree to counter or mitigate the corresponding attack vector at the optimal locations (**A**T**C**). A limitation of attack trees is scale: each top node/end node of a tree is a singular effect of a realized threat or result of an attack. The more attacks to analyze, the more trees are needed. There are exceptions to this stated

limitation [21], but these examples had atypical allocation of resources and time to produce trees of this scale.

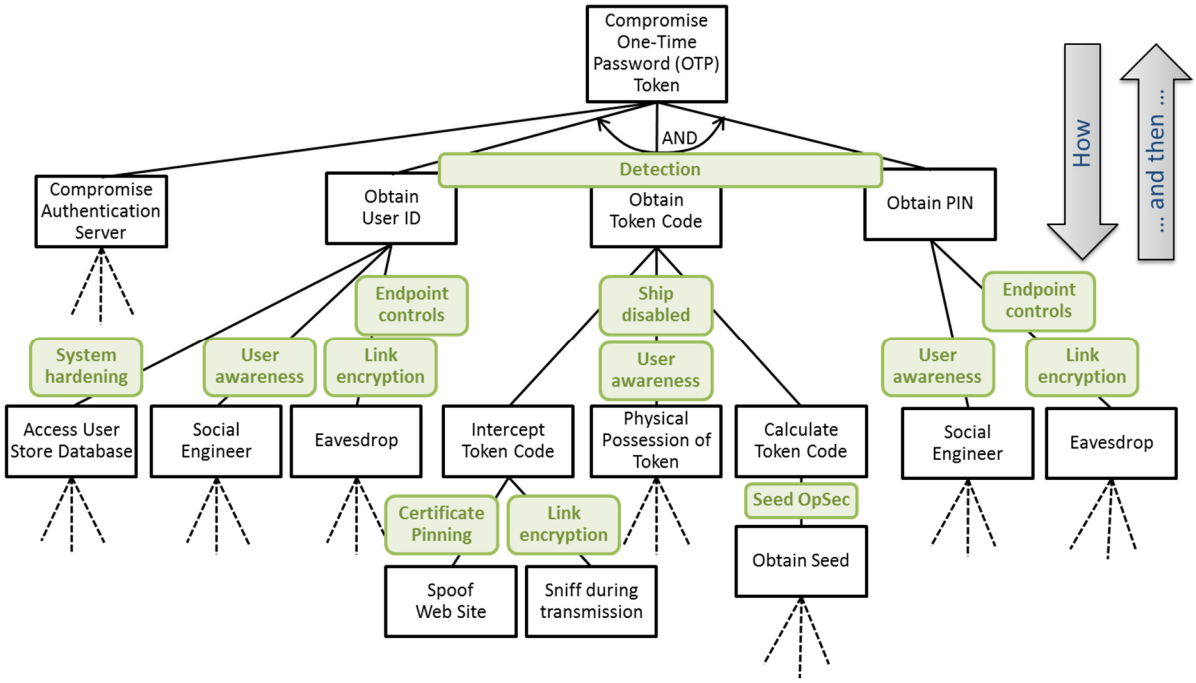


**Figure 11 - Attack Tree for One-Time Password tokens**

Figure 11 is an attack tree that decomposes the conditions necessary for a successful compromise of a one-time-password (OTP) token credential. Each branch contains a set of step-wise conditions that must occur in sequence. Multiple individual branches can be "chained" with Boolean AND logic – represented by the arc arrows in the diagram. Branches without the arc AND arrows are considered an OR condition. A completed attack tree should follow these guidelines: as the tree is built downward, each branch and subsequent condition on the branch should describe "How" the events chain together to realize the top-node threat condition – and when working from the bottom upward, each node should support the above level node by a logical progression of "… and then…". The green boxes overlaid on this attack tree are where specific security controls are allocated to remove, counter or mitigate that particular attack path. If a tree has a complete branch without controls, this is obviously a gap that requires risk management discussions. The allocation of controls in this manner also provides the means to determine the efficacy of controls in both a broad and granular manner and is a foundational method of control effectiveness evaluations discussed in the Controls section of this paper. Architects can leverage completed attack trees to identify the scope of discovered gaps and ineffective controls and decide whether infrastructure, services or roadmaps require adjustment.

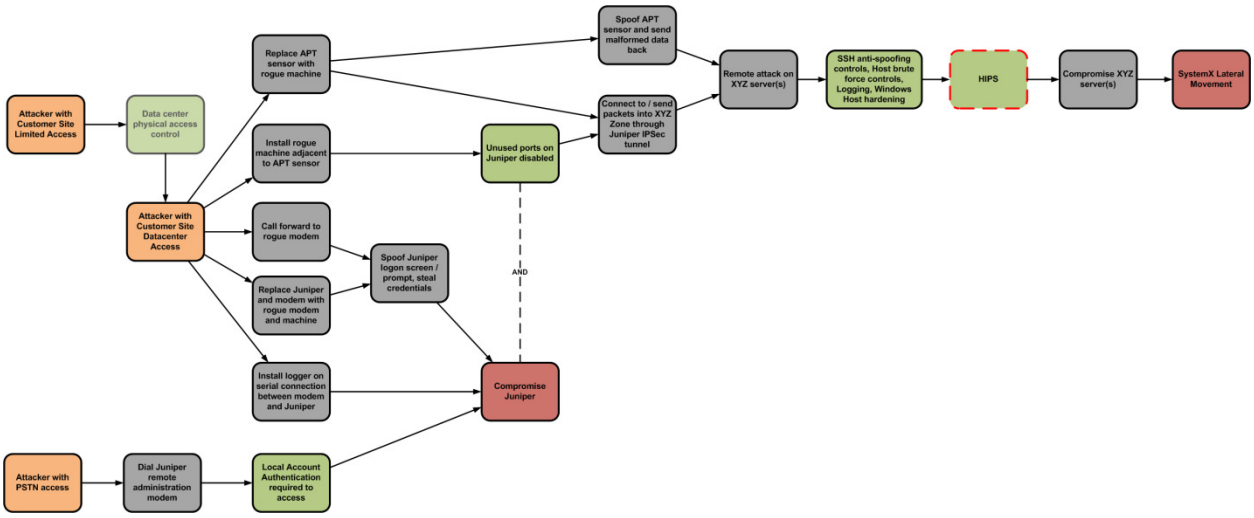


**Figure 12 - Example Attack Tree for VPN partner connection**

The attack tree in Figure 12 is a horizontal tree that adheres to the guidelines discussed in the previous tree's description. In this tree, the grey nodes are the conditional nodes of the tree, the red nodes are resultant conditions, green nodes are security controls and the orange nodes are threat actors/attack agents. This tree is shown as a comparison to the tree in Figure 11 – to illustrate the flexibility and creativity possible with attack trees.

## Threat Profiles

For the purposes of this paper, a *threat profile* is defined as a tabular summary of threats, attacks and related characteristics. The tabular format allows for a large degree of flexibility in terms of the columns and rows selected for each profile. Threat profiles are an excellent tool for communicating the results of completed threat models and/or attack trees. They also allow for consolidation, sorting, pivoting and other common data manipulation functions. Providing detailed descriptions of both the cause and effect of threats and attack vectors is an ideal use case for threat profiles.

For comparison, some assessment practices use the term *threat profile* to describe a decomposition that is much more analogous to an attack tree [5], while others are more congruent with the description provided in this paper [13]. More recently, the term threat profile has been used in the development of cyber threat intelligence.

Table 2 presents a generic threat profile template. Table 3, along with Listing 1, and Table 4 illustrate case study threat profiles. The differences in scope and applicability between Tables 3 and 4 are described as follows: Table 3 is a more granular profile and Table 4 is an example of a large-scale, detailed analysis presented as a summarized and aggregate data set. These two examples demonstrate the flexibility and scalability of threat profiles.

**Table 2 - Threat Profile template**

| | Description |
|---|---|
| Asset/Threat Object | The thing the attacker wants or that the owner needs to protect |
| Threat Types | STRIDE-LM; CIA; Others |
| Attack Surface | The components, interfaces, etc that will be initially attacked |
| Attack Vectors | The path or technique the attacker uses to realize the threat |
| Threat Actors | The entity who is trying to realize the threat against the asset |
| Resultant Condition | Describe what happens if the threat is realized |
| Vulnerabilities | Any known vulnerabilities (there may not be any) |
| Controls | Things that will help mitigate or counter the attack |

**Table 3 Example Threat Profile - Concept Phase using STRIDE-LM**

| | **Description** |
|---|---|
| **Asset/Threat Object** | Virtual computing infrastructure "Gold" images |
| **Threat Types** | S **T** R **I** D **E** - **LM**<br>Tampering<br>Information Disclosure<br>Elevation of Privilege<br>Lateral Movement |
| **Attack Surface** | • Virtual Disk image file<br>• File system where image file resides<br>• OS that manages file system<br>• Disk/Storage device that contains OS, FS, image<br>• Network services/protocols that access the image<br>• Applications that manage the image file |
| **Attack Vectors** | • Gain admin/root credentials on OS that manages Gold image<br>• Exploit vulnerability on OS that maintains Gold image |
| **Threat Actors** | • APT with foothold on corporate or management networks<br>• Malicious insider<br>• Corporate espionage |
| **Compromise Result** | • Adversary can gain complete control of the virtual compute infrastructure deployed from the Gold image<br>• Adversary can steal or tamper data in the Gold image to disrupt business or sabotage critical plans |

The threat profile presented in Table 3 was included as part of an Operational Concept document in support of the design of a new extranet hosting service; hence the lack of vulnerabilities and controls at this point in time. However, high-level controls were allocated based on the primary threat types identified using the STRIDE-LM model, resulting in the following: S **T** R **I** D **E** – **LM**

- Tampering

- Information Disclosure
- Elevation of Privilege
- Lateral Movement

Each primary threat type has a corresponding security property (as presented in Table 1) that assists the engineer with the selection and implementation of corresponding security controls. Each primary threat type and the corresponding controls are described in Listing 1 below:

Tampering → Control = Integrity and Access Controls
Design:
- Periodic hash check on Gold Image files
- Only virtual compute admins granted access, ACLs on network and file system

Information Disclosure → Control = Encrypt or Isolate
Design:
- Too costly to Encrypt
- Isolated Storage filers for Gold Image files

Elevation of Privilege → Control = Authorization
Design:
- Specific roles of different levels of access (RO, Full, etc.)
- Provisioning/de-provisioning process to restrict authorized users and remove unauthorized users

Lateral Movement → Control = Credential Hardening and Segmentation
Design:
- Privileged account password management
- Separate administrative accounts
- Administrative access authentication gateways

**Listing 1 - Control allocation design per threat type**

To provide a larger scale example of using a threat profile, Table 4 is provided below. This threat profile lists a subset of items from the complete table. This profile communicated the results of a comprehensive threat analysis, which included multiple threat models and attack tree diagrams. It was primarily used as a communications tool for managers to enable risk management decisions and initiate an action plan. It was also used to track triaged issues to completion and allowed for other valuable data analysis functions. From a tactical perspective, the first column in Table 4 identifies the *cause* of the issue and the second column describes the *effect* of the issue. The cause-effect linkage is a necessary element in threat analysis work and threat profiles provide an effective facilitation and communications tool for this purpose.

Table 4 - Threat Profile, large scale summary analysis

| Threat /Finding/ Topic | Resultant Condition | Threat Types(s) (STRIDE-LM) | Counter / Mitigation Principles |
|---|---|---|---|
| Overall Architecture Issues | | | |
| Insufficient isolation of services (forest level) | malicious domain owners can control services in other domains | S,T,R,I,D,E,LM | Create separate forests |
| Insufficient isolation of data (forest level) | malicious domain owners can access restricted data in other domains | T,I | Create separate forests |
| Specific Issues | | | |
| Insufficient isolation of authentication (Interdomain) | users can authenticate on systems in other domains | I,E,LM | Setup firewalls between domains within a forest |
| Insufficient auditing of malicious activity (device) | Inappropriate activities are not identified | S,T,R,I,D,E | Windows auditing should be configured and enabled in all devices in all domains |
| SID exploitation - User (Interforest) | User impersonation in other trusted environments Unauthorized use of Security ID (SID) information | S,T,I,E,LM | SID Filtering Quarentining |
| Authentication Exposure between forest (Interforest) | Increase authentication exposure footprint between forests | S,T,I,E,LM | ● Selective Authentication ● Disable DomainInfo or TopLevelName record |
| Authentication Exposure within or between forests (intra or Interforest) | Stolen credentials, traffic eavesdropping | S,T,I,E,LM | Set LDAPServerIntegrity to 2 to deny LDAP simple binds |

The obvious flexibility of threat profiles make them a preferred option whenever communications, tracking, sorting or additional data pivoting of analysis results is necessary.

### Summary of Practices

Threat models, attack trees and threat profiles have been discussed to this point. A summary of these techniques' descriptions and primary use cases is provided below:

**Table 5 Summary of Threat Analysis Practices**

| Technique | Description | Best used for |
|---|---|---|
| **Threat Models** | Depict data flows, data stores, interfaces, processes, system and trust boundaries | • Typically the starting point of threat analysis work<br>• Brainstorming<br>• Defining attack surface<br>• System-level analysis<br>• Component-level analysis<br>• New technology or systems |
| **Attack Trees** | Decomposition of *how* a threat against an asset can be realized | • Detailed analysis of attacks<br>• Aligning controls to branches or leaf nodes<br>• Generally narrow in scope |
| **Threat Profiles** | Tabular summary of threats against a system or component | • Communications<br>• Decision making<br>• Describes cause and effect<br>• Conceptual phase support<br>• Often assumes prior analysis has been done<br>• Time constraints do not allow a full threat model or attack tree analysis |

With this summary of threat analysis and modeling practices, techniques and tools completed, the integration of the IDD methodology and CKC practices, which produce the resultant *threat intelligence*, is discussed next.

## Threat Intelligence

The IDD methodology is well established in the industry and is based upon the CKC practices. To extract the greatest value from the CKC, it must be leveraged as a *comprehensive analysis and synthesis framework* and not applied in the limited scope of an incident response process. When utilized as an analysis and synthesis platform, organizations can become an *intelligence producer*, and channel that intelligence into strategic program objectives and enhancements to core infrastructure, thus realizing the benefits of a threat-driven approach. The organization's resiliency to cyber-attacks is maximized, funding and resourcing of enterprise protections are properly informed and allocated.

Figure 13 illustrates the analysis and synthesis capabilities achieved by extending the kill chain steps across multiple threat campaigns. In lieu of halting analysis at the step where a block occurred, analysts continue to decompose the TTPs used across campaigns and then reference them forward to identify where additional blocks could have happened, as well as referencing backward to identify where blocks should have occurred. Steps where no potential blocks exist represent gaps where investments could be considered. TTPs, attributions and other characteristics discovered during analysis/synthesis are fed into an *intelligence management system* to enhance reporting, historical analysis and predictive capabilities.

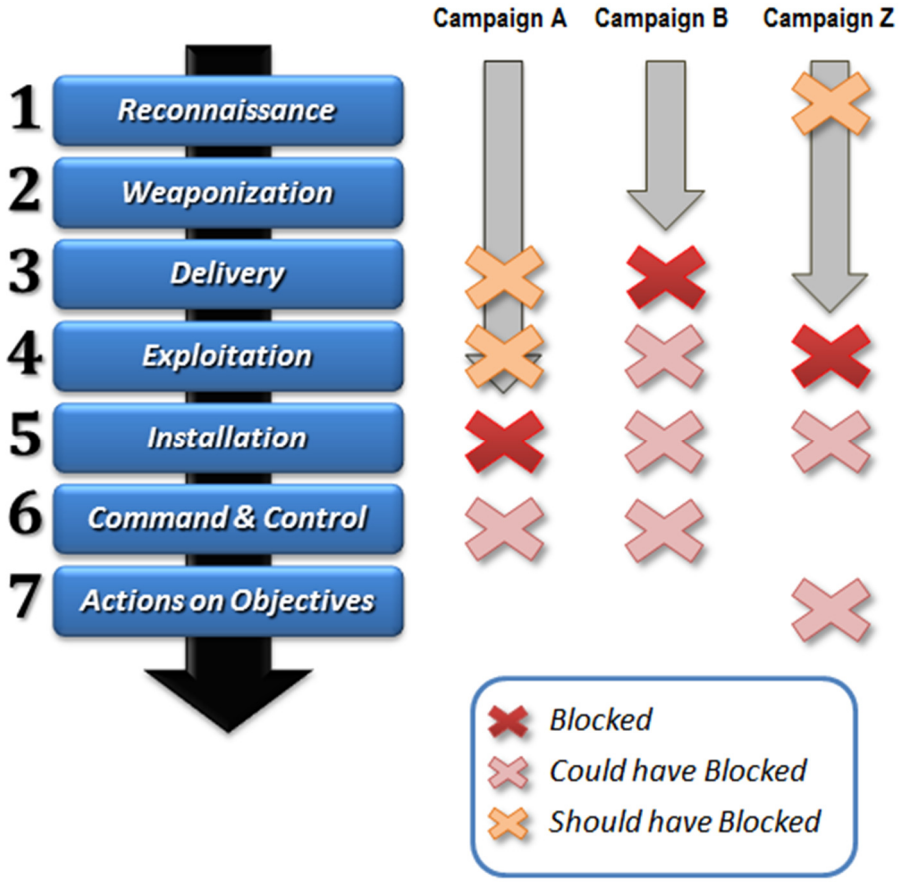


Figure 13 - the Cyber Kill Chain as a Framework for Intelligence Driven Defense

The CKC should be used by architects and engineers and not only by analysts doing incident response and intelligence management. When architects/engineers understand the implications of each of the 7 steps of the kill chain, they can deliver a more thorough and high quality threat analysis artifact. When architects/engineers receive and implement threat intelligence inputs, a more agile sustaining engineering environment is enabled and architectural gaps are more easily identified and resolved. This concept will be revisited in the Controls section of this paper and will describe how this same analysis can be leveraged to evaluate controls and shape systems architectures. Conjointly, analysts and operational roles can benefit from threat modeling and analysis artifacts to uncover potential new attack vectors, especially in newer technologies and environments for which little intelligence will exist. Blending these complementary practices produces an agile and resilient cyber security practice and propels the organization to a more mature security posture.

### Tactical Analysis Integration

The CKC is the de facto standard for computer network incident response and is the platform for threat intelligence analysis. In cases where threat intelligence is lacking or when threat actors are exhibiting new TTPs, the IDDIL/ATC methodology has proven to be an effective complement to the kill chain. The Heartbleed vulnerability [22] is an excellent case study. Since there were no advanced indicators or intelligence regarding the Heartbleed vulnerability, there was a period of discovery that was necessary when the Heartbleed exploit was first detected. Lockheed Martin's integrated response team therefore applied the IDDIL/ATC methodology to enhance the overall response efforts, as described below:

- Identify the assets: Using a combination of existing systems management tools and prior scan

reports (*Inventory* and *Manage* control functions[8]), as well as active scanning, the team was able to rapidly identify which systems had vulnerable versions of OpenSSL.

- Define the attack surface: Locate the identified vulnerable OpenSSL systems and determine which applications and services they support (*Inventory* and *Manage* control functions). Business managers, system owners and administrators were contacted early in the process to confirm findings and enable a more efficient process for future activities.
- Decompose: Determine the interfaces to/from the identified vulnerable systems; determine the types of vulnerable systems – i.e. VPN servers, Web servers, browsers, etc. and the types of data/accounts available within those interfaces and systems. This provided granular enumeration of potential exposure of critical or sensitive data. This enumeration is a continuation of *Identify the assets*.
- Identify attack vectors: Open source information combined with threat intelligence data and red team research led to discovery of additional attack vectors. These activities included validation of publicly disclosed reports and the acquisition of additional details regarding how the attack actually worked. This information allowed tool enhancements to identify additional vulnerable assets in a compressed time frame.
- List threat actors/attack agents: Using the knowledge gained from the previous activities, active monitoring was performed to correlate known adversaries, security researchers, and unknown single actors.
- Analysis: Determine the full scope of exposure based on IDDIL discovery activities. All vulnerable assets are identified and potential exposure of critical or sensitive data is documented. Analysis focuses on evidence of any compromise or exfiltration of critical or sensitive data.
- Triage: Findings captured from **IDDIL/A**TC allowed immediate actions and controls to be implemented against the *reconnaissance*, *delivery*, *exploit* and *actions on objectives* steps of the kill chain. These controls were applied at appropriate locations, as guided by the attack surface, decompose and attack vector findings. Remediation efforts were prioritized so that actions and controls were first performed on externally facing systems, then on partner facing systems, and lastly internal systems.
- Controls: The IDDIL/ATC methodology provided detailed and actionable information in a highly compressed timeframe, which enabled the implementation of different security control functions at multiple locations. These control functions include *Protect*, but also include *Collect*, *Detect* and *Respond*. The pre-existence of adequate *Inventory* and *Manage* control functions increased the efficiency and effectiveness of the IDDIL/ATC activities. These same Inventory and Manage functions were enhanced as part of the overall Heartbleed response activities. Tactically, by gaining a full understanding of the attack surface and vectors, multiple controls were deployed at appropriate architectural layers to enhance detection and correlation of Heartbleed activity. The derived intelligence was applied against adversary scanning activity itself as a means to ensure the controls were appropriately implemented.

As a whole, the full cycle of activities surrounding the Heartbleed vulnerability validated the combined threat-driven methodologies and Defendable Architectures [23] concepts.

### Focus on the Largest Threats

There is another way to apply the threat-driven approach: *focus on the largest threats to your environment* and continuously apply the IDDIL/ATC methodology against those threats. In this context, threat is used as an adverse effect vs. a person or group (i.e. threat actor). One of those large threats, as discussed previously, is lateral movement. Although lateral movement has several variants, one of the most damaging is the ability of an adversary to pivot and expand influence within an environment once that

[8] Control functions are described in the Functional Controls Hierarchy section of this paper

adversary has a foothold. This corresponds to step 7 (actions on objectives) of the CKC. One of the primary mechanisms of lateral movement is Windows pass-the-hash [24] (PTH) attacks.

Lockheed Martin created a cross-functional team comprised of members from the LM-CIRT (Lockheed Martin computer incident response team), Red Team and Security Architecture/Engineering groups to fully understand these attacks and explore if new or enhanced collection, detection or protection controls could be developed. Merging threat intelligence and IDDIL/ATC practices, using open sources [25] [26] and adding original research [27], the team identified several root cause issues common to multiple threat actors' methods to extract hashes from memory structures: abusing certain APIs in a manner they were not intended. (*Note*: since PTH is a post-exploit tactic, the research focused exclusively on extraction techniques and did not focus on delivery or exploitation). Antivirus, memory protections and other endpoint controls have proven largely ineffective at preventing this particular attack. Re-designing administrative/privileged account provisioning processes, adding administrative access gateways and protecting privileged account passwords provide a moderate degree of protection. Microsoft has recently added operating system enhancements [28] [29] that reduce the attack surface and mitigate some attack vectors. However, as long as password hashes are stored in memory on Windows systems, this threat will remain. Therefore, the joint team developed the following set of new controls:

- A custom HIPS (host intrusion prevention system) rule to log and block API calls of CreateRemoteThread() or NtCreateThreadEx() into lsasrv.dll or lsass.exe
- Removal of the debug libraries on critical, high-impact servers: symsrv.dll and dbghelp.dll
- Log and trace outbound requests to the Microsoft Symbol Server: (http://msdl.microsoft.com/download/symbols)

These controls have proven effective at mitigating – and in many cases, preventing – attempts at password hash extraction. At a minimum, they provide high fidelity visibility to threat intelligence processes when threat actors first attempt to move laterally.

Research on this issue continues and additional enhancements to controls are under investigation; as adversaries' TTPs evolve so must the controls. The PTH attack vector has had many evolutions in the arms race of attacker versus defender. Current similar attack vectors [30] are also being included in this ongoing research to determine if similar controls can be designed and deployed.

# 3. Controls

Security controls are a *designed response* against the actual threats and attack vectors present in any given application, system or environment. Security controls either remove, counter or mitigate threats or attack vectors. They must be designed, implemented and assessed per these postulations. Controls exist as a technology, a process or a policy. Controls can also be considered logical, physical or administrative. A logical progression would state: "*since controls exist to remove, counter or mitigate threats and attack vectors, then threats and attack vectors should determine where, why and how security controls are selected, implemented and assessed*". The thesis attached to the threat-driven approach assumes this stated progression is the exception vs. the rule.

## Current-state Challenges

Compliance policies and frameworks [31] [32] have driven behaviors where organizations consider a listing of controls mandatory. These control listings contain examples of *potential* controls, and are themselves the cumulative result of the industry's long-term threat and vulnerability analysis and assessment efforts. When these lists are taken as the primary source for cyber security engineering or assessment activities, it blinds or impairs an organization from implementing the controls that will actually *protect* and *defend* the assets within that organization's unique environment. The controls are

translated into system requirements without requisite analysis to ensure the controls selected and implemented will accomplish the function they are intended to perform. Identification of gaps in controls coverage rarely occurs. This condition and the associated perceptions surrounding compliance requirements must change. The same policies and frameworks that have driven these behaviors contain guidance which allows and *recommends the inclusion of threat analysis* in support of controls allocation and evaluation, as well as risk assessment and risk management practices. Unfortunately, broad awareness of this guidance does not yet exist amongst the organizations that are subject to compliance mandates. As described in [33] SA-8, [34] Task 2-1, [32] Step 2.c, [9] Section 3.1.3, and [35] Tasks 1-1, 2-1, 2-2 and 4-1, descriptions of implementing threat analysis activities, as well as guidance on tailoring the standard compliance process to satisfy business/mission objectives, exists. As presented in [36], these compliance challenges are considered as an opportunity, along with a recommendation that supports the threat-driven approach. However, additional work within the industry is required to modify these policies and frameworks to more explicitly and more broadly adopt the threat-driven approach.

A typical practice for implementation of security controls per compliance requirements is to link controls to technologies or components – i.e. when a web server is deployed, then certain controls are automatically selected; the same for routers/switches, database servers and so on. This creates two issues: 1) too narrow of a focus, where the macro and/or peripheral impacts are not accounted for – and 2) the selection of the controls does not account for the threats and attack vectors that are *present for that system/environment as a whole*. Selection and implementation of security controls per discrete technologies – as specified from a pre-defined list – is considered *engineering to policy* and is not the preferred approach.

Consider an internet-facing web application as example: an organization could implement every mandated control per their compliance lists for every web server, database server, firewall, and network device deployed. However, the environment that hosts the web servers was designed to include administrative and management interfaces that permit *lateral movement into the internal network if any one server is compromised*. This is what is known as a *single point of compromise*, and this condition is frequently discovered in controls-first environments. Adequate compensating controls do not exist, because this threat and attack vector was not considered during the design, deployment and sustaining engineering phases for this environment. The environment would have been deemed 100% compliant, yet remain extremely vulnerable to a high-impact attack. A proper implementation of the IDDIL/ATC methodology would have identified these issues and allocated the proper controls, including compensating controls.

It is acknowledged that certain controls are effectively "mandatory" given certain technologies, parameters or environments. Also, there are numerous baseline controls that should be ubiquitous in any IT organization. Examples include password complexity policies, encryption standards, endpoint protections and vulnerability and patch management practices – to name a few. These baseline and "mandatory" controls should be considered a least common denominator, along with the realization that while they are necessary, they are far from adequate given current and future threat actor capabilities.

Management, maintenance, audit and evaluation of deployed controls – per compliance requirements – consume large amounts of resources and time. These resources could provide greater value to the organization if they were focused on combating threats versus compliance. If an organization is effectively and actively combating threats, it must also manage controls within the scope of compliance activities; the difference is the threats drive how controls are assessed and managed, not a checklist. Automated and repeatable verification of compliance requirements would be an enabling solution that would improve the compliance burden. Modified organizational and policy structures would also increase efficiency and streamline compliance activities: configure the organization with integrated cyber roles and

define policy standards so that compliance requirements are easily auditable, and the minutiae of control implementations is separated and managed at appropriate levels.

## The Integrated Solution

The adoption of the threat-driven approach and corresponding methodologies unifies the practices associated with selecting, implementing and evaluating security controls. That is, the same threat analysis and threat intelligence practices that are used to allocate security controls are also used to assess the effectiveness of security controls. This common set of practices streamlines the compliance and audit burdens and allows architects, engineers and analysts to realize the integrated model presented throughout this paper.

There are two foundational concepts that enable this integration:

1. The threat categorization → security property → security control relationships
2. The Intelligence Driven Defense® [1] Courses of Action Matrix

Merging these two concepts produces a framework to select, implement and evaluate security controls' effectiveness. This merged framework results in a flexible and scalable set of tools and practices, listed below:

- A Functional Controls Hierarchy
- Attack Use Cases
- A Controls Effectiveness Matrix
- A Controls Effectiveness Scorecard
- A Data-driven Architectural Controls Coverage diagram

## Selecting and Implementing Controls

Threat analysis artifacts articulate threats and attack vectors and categorize them using taxonomy appropriate to the business/mission. This paper includes the STRIDE-LM model for threat categorization; others exist. Regardless of how threats are categorized, the model must include corresponding security properties and available controls per each threat type. This establishes a direct relationship between threat analysis outputs and the selection and implementation of security controls. The controls provided in the threat categorization table are typically generic in scope and refer to a *category* of a control; this is to keep the categorization table manageable and focused on the threats during analysis phases. When a project, program or service arrives at a phase of work when security controls are selected, a more detailed and context-specific implementation mechanism is required. The Functional Controls Hierarchy is the extension of the Controls column in the threat categorization model, and provides the implementation mechanism.

### Functional Controls Hierarchy

Organizations typically maintain a listing or catalog of available security controls and services, which is built from its native security services and capabilities referenced against one or many information security frameworks or standards. A catalyst for establishing a more effective mechanism to align controls to threats is to view security controls as *accomplishing a unique function* and selecting controls per these baseline functions. A Functional Controls Hierarchy (FCH) will deliver this capability as described in Table 6 below. The concept of functional controls and the corresponding tools discussed in this section of the document are similar to the Framework Core presented in [37]. There are few other similarities between that framework and this paper.

**Table 6 – Functional Control Hierarchy description**

| Function | Category | Implementation | Effectiveness |
|---|---|---|---|
| **<the primary objective of a control set>** | <A member of a set of the control type specified by function and aligned to a more granular capability. Instances of Category should align to either the Property or Control columns in the Threat Categorization table.> | <An instance of a service, technology, product or process that corresponds to the Category type either as the service that is providing the capability or the technology area where the control is applied. Instances of Implementation should align to the Controls column in the Threat Categorization table.> | <An indicator of this implementation's ability to achieve the corresponding control's objective; only applicable when assessed against actual threats or attack cases/TTPs> |

The FCH is described as follows:

- The Function column contains the highest-level security capabilities.
- The Category column will contain instances of generic security capabilities that correspond to a function but do not indicate a discrete technology, service or product.
    - Instances of the Category column should align to either the Property or Controls columns of the Threat Categorization table (ref Table 1).
- The Implementation column will contain discrete instances of a technology, service, product or process per its corresponding Category. It could also list the technology component where the instance is applied.
    - Instances in the Implementation column should align to the Controls column in the Threat Categorization table (ref Table 1)
- From a policy/framework perspective – Category and Implementation instances should typically have a clear mapping to the compliance or governance model used by the organization

Expressing controls as *functions* is a subtle, yet important shift in adopting the threat-driven approach. Each function has a defined purpose; each category describes a particular aspect of its corresponding function; each implementation of a category is the service or control that provides the explicit functionality. Table 7 presents an example FCH with the Function and Category columns populated. The Implementation column is not shown here since each organization's implementations will be unique.

Table 7 - Functional Controls Hierarchy

| Function | Category |
|---|---|
| Inventory | Discovery and Reconcile |
| | Catalog and Organize |
| | Rogue Detection |
| | Decommission/Shutdown/Remove |
| Collect | Traffic/Data Discovery |
| | Logging System and Storage |
| | Logging Sources |
| | Traffic Decryption |
| | Logging Requirements |
| | Logging Integrity |
| Detect | Endpoint Signature |
| | Endpoint Heuristic |
| | Endpoint Behavior |
| | Network Signature |
| | Network Heuristic |
| | Network Behavior |
| | Application Signature |
| | Application Heuristic |
| | Application Behavior |
| | User Behavior |
| | E-Mail Systems (non-endpoint) |
| Protect | Authentication |
| | Authorization |
| | Application Input & Protocol Validation |
| | Malware Prevention |
| | API Hardening |
| | Least Privilege Enforcement |
| | System Hardening |
| | Application Whitelisting/System Integrity |
| | Data Integrity |
| | Data Encryption (at rest) |
| | Data Encryption (in transit) |
| | Data Encryption Enablement |
| | Isolation and Boundary Enforcement |
| | Reputation Services |
| | Availability and Resiliency |
| | Awareness and Education |
| Manage | Baseline Definition |
| | Configuration Control |
| | Verification and Audit |
| | Vulnerability Management |
| | Patch Management |
| Respond | Selective Alerting |
| | Incident Handling & Forensics |
| | Intelligence Management System |
| | Intelligence Feeds |
| | Adversary TTPs |
| | Response Plans for Top *N* scenarios |
| | Recover & Restore |

This FCH represents a current-state baseline; it will change over time. Each organization should create a hierarchy that is appropriate for its unique needs. However, this listing in Table 7 is a reasonable baseline to adopt, modify or extend.

These functions will appear similar to other taxonomies and catalogs [37]; however, in this hierarchy, the *Collect* function is unique. As an intelligence-producing entity, the *collect* function takes on a special significance and must therefore account for technical security control characteristics as well as threat

intelligence requirements. The IDD methodology and Defendable Architectures objectives drive implementations of the collect function to provide a more robust detection and forensics capability, to provide historical analysis data which enhances intelligence production, and to support fine-tuned recovery capabilities.

### *Architect Role*

Architects build and maintain the FCH. From the architect's perspective, the FCH is a synthesized model of available security controls and services, aligned to the respective organization/environment's technology stacks and deployed networks/systems. The architect, with regular input from engineers and analysts, modifies and updates the category and implementation entries to reflect current and future states. Architects implement the FCH as an enterprise controls coverage platform, which can be visually expressed in the *Architectural Rendering* section of this paper.

The FCH will identify duplication of controls and services at the category and implementation levels. By identifying duplicate functionality, the organization can balance its business/mission needs against risk management and budget constraints pertaining to the duplicate services and potentially remove the overlapping controls, thereby reducing costs. Conversely, the functional hierarchy is an effective tool for architects to identify gaps in controls coverage. These gaps can exist in currently deployed controls and services, or these gaps can be identified and labeled as either a gap for which a solution exists but has not been implemented, or the gap is an industry wide gap that requires roadmaps, exploration, development, and/or vendor negotiations to close the gap. Known gaps are deliberately included within the FCH so they can be effectively tracked, communicated and dispositioned.

#### Control Functions and Defendable Architecture Characteristics

Defendable architectures [23] describe, at a strategic level, the core characteristics of *visibility, manageability and survivability*. There is a direct relationship between these characteristics and the functional control hierarchy as presented in Table 8.

**Table 8 - Defendable Architecture characteristics and Control Functions alignment**

| Defendable Architecture Characteristic | Control Hierarchy Function |
|---|---|
| Visibility | Inventory<br>Collect<br>Detect |
| Manageability | Inventory<br>Manage<br>Respond |
| Survivability | Detect<br>Protect<br>Respond |

The defendable architecture characteristics are strategic drivers for an organization's cyber/information security initiatives and programs. The functional control hierarchy is an enabler as well as a barometer for realizing the defendable architecture goals.

### *Engineer Role*

Engineers leverage system/project-level threat analysis artifacts and threat intelligence to correlate the identified threats and attack vectors and select appropriate security implementations. The link between the threat categorization table to FCH categories and implementations is the baseline for controls selection and implementation. Additionally, the engineer determines what threat/attack vector/control combinations

are most critical for testing and develops security test plans accordingly.

### *Analyst Role*

Analysts use the FCH as a tool to support threat actor campaign analysis and intelligence management activities. Analysts then provide the results of these activities to the architect to ensure the FCH entries are current and accurate, and to identify potential gaps in controls coverage.

## Evaluating Controls Effectiveness

This section presents a method and a tool for determining the effectiveness of controls *given the context of a specific threat, attack use case, or other pattern identified from threat intelligence or threat modeling activities.* With this tool, a control's efficacy is not examined as a binary condition (i.e. the presence of the control is the evaluation criteria). Rather, the analysis performed via IDDIL/ATC and threat intelligence produce descriptions of the *characteristics and behaviors of the threats* and the *explicit manner of how attacks will be executed*. It is from this perspective that controls are then evaluated.

### Attack Use Cases

Attack use cases are similar in concept to threat intelligence campaign analysis, however, *attack use cases* aggregate and describe patterns of the *most common and damaging attacks*, regardless of the adversary – or group of adversaries – behind the attacks. Attack use cases are analogous to enterprise or organizational level threat modeling, and are built and assessed by the architect and analyst roles. An attack use case is not a list of vulnerabilities or exploits; a list of this type would be unmanageable and at too low level of detail. Attack use cases contain the following elements:

- The associated attack surface elements (**IDD**IL/AT**C**)
- The most common, damaging attack vectors and TTPs (IDD**IL**/A**T**C + current threat intel)
- The assets/objectives under attack (**I**DDIL)

Examples of attack use cases include the following:

- 3rd Party compromised credentials
- DoS/DDoS against a critical site or application
- Off-corporate network device attack

This is not a comprehensive list; it is provided to demonstrate some common cases and to provide context for further explanations. However, any organization's full list of attack use cases should be manageable, and not be too large; the intent of creating attack uses cases is to identify and analyze the "*Top N*" attack patterns per each organization. As an example, this presentation [21] identified 7 items that are analogous to attack use cases.

### Controls Effectiveness Matrix

The Controls Effectiveness Matrix requires a specific threat, attack vector or – preferably - an attack use case to achieve optimal results. The matrix is an extension of the FCH, with an added column for effectiveness, which captures the results of the analysis. This tool is used by the functional roles of architect, engineer and analyst. There are four qualitative ratings used in this matrix:

- The control fully achieves its intended functional objective
- The control partially achieves its intended functional objective
- The control does not achieve its intended functional objective
- A gap exists for the intended functional objective

If a security control is present, given the attack use case, it is evaluated and given one of the first three ratings. The final rating is reserved for any case where a particular control function could have been applied, but where that control's capability is not present in either the organization's available

controls/services portfolio, or the industry has no control of this type.

Table 9 presents a notional set of ratings for controls within the Inventory function and a given attack use case. Analysis is only performed on control function implementations within the scope of the attack use case. High-quality threat analysis artifacts and threat intelligence are critical inputs to obtaining accurate and actionable effectiveness ratings.

**Table 9 - Controls Effectiveness Matrix**

| Function | Category | Implementation | Effectiveness |
|---|---|---|---|
| Inventory | Discovery and Reconcile | System X | ● |
| | | Control Q | ○ |
| | | Process 1 | ○ |
| | Catalog and Organize | System X | ○ |
| | | Technology T | ● |
| | | System Z | ○ |
| | | Process 2 | ● |
| | Rogue Detection | System M | ○ |
| | | Technology R | △ |
| | Decommission/Shutdown/Remove | Process 2 | ○ |
| | | System T | ○ |

| | |
|---|---|
| ● | Fully achieves control function's objective |
| ○ | Partially achieves control function's objective |
| ○ | Does not achieve control function's objective |
| △ | Control function capability gap |

This matrix can be used to produce aggregate ratings for control Implementations, Categories and Functions, by summarizing the ratings of a particular control across multiple threats, attack vectors and attack use cases. Regardless of how the analysis is performed, the resultant qualitative findings can be given quantitative values and other attributes and then fed into a database to become part of the intelligence management system, or those quantitative values can be rendered as architectural diagrams.

### *Architect Role*

The controls effectiveness matrix is a direct extension of the FCH, and the architect role is responsible for the construction and maintenance of the FCH, as described in the previous section. Additionally, the architect will identify the appropriate assets and components to be included within the analysis, per the given attack use case or attack vectors.

### *Engineer Role*

Engineers can leverage the controls effectiveness matrix in support of control selection or evaluation. If recent ratings exist for candidate controls that align to equivalent threats and attack vectors of an engineering project, the selection process can be streamlined. From an evaluation perspective, engineers can use the matrix as a baseline reference for the creation of security test cases.

### *Analyst Role*

The analyst is the key role for the controls effectiveness matrix. The analyst is responsible for incorporating threat intelligence from the intelligence management system to define the scope of attack use cases. The analyst also performs the assessment of control's effectiveness along with the engineer and architect.

## Controls Effectiveness Scorecard

This scorecard presents a dashboard view of enterprise controls coverage against the most relevant attacks faced by the organization. These scorecards are a valuable communications tool for executive management as well as providing architects and analysts a common tool for threats, attacks and controls coverage. Structurally, the Controls Effectiveness Scorecard is a consolidated, point-in-time analysis of an attack use case against in-scope controls from the FCH. The vertical axis contains the control functions and the horizontal axis displays the attack use case's corresponding attack surface/vector components, presented in architectural component groupings[9]. Table 10 presents a Controls Effectiveness Scorecard for the 3rd Party Compromised Credentials attack use case; which includes attack surface/vector components such as Internet gateways, VPN circuits, authentication directories and mechanisms, network zones and applications available to 3rd parties, as informed by the threat model and current threat intelligence. The qualitative rating system is the same as the Controls Effectiveness Matrix, with the exception of some elements being not applicable.

**Table 10 - Controls Effectiveness Scorecard**

| **Attack Use Case:** *3rd Party Compromised Credentials* | | **Attack Surface/Vector** | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | User | Network | OS | App Server | App Client | Storage | Memory |
| **Function** | Inventory | ○ | ○ | ● | --- | ○ | --- | ○ |
| | Collect | ○ | ● | ○ | ○ | ○ | ○ | ○ |
| | Detect | ○ | ● | ○ | ○ | ○ | △ | ○ |
| | Protect | ○ | ○ | ● | △ | ○ | △ | ○ |
| | Manage | ● | ● | ● | --- | ○ | ○ | --- |
| | Respond | ○ | ● | ○ | ○ | ● | △ | ○ |

| | |
|---|---|
| ● | Fully achieves control function's objective |
| ○ | Partially achieves control function's objective |
| ○ | Does not achieve control function's objective |
| △ | Control function capability gap |
| --- | Not within scope of this attack use case |

Scorecards are built per attack use case, so the number of scorecards that need to be maintained should be relatively small. Architects and analysts collaborate continuously to provide current scorecard analysis results. The analyst provides input and updates to each scorecard from the intelligence management system and the architect validates attack surface components and controls coverage from the FCH.

## Architectural Rendering

The outputs of the Controls Effectiveness Matrix and Controls Effectiveness Scorecard tools provide a rich data-set. Combined with input from the Intelligence Management System, this data-set is a valuable commodity to architects. This combined, context-relevant information allows architects to understand the organization's controls/services portfolio efficacy, and identify and manage gaps in coverage or capability. Architectural diagrams are built utilizing the same data-set, thus rendering the analysis results at an enterprise level (or whatever scope/context desired) in an architecturally relevant artifact. Figure 14

[9] These architectural component groupings may change based on the attack surface of each attack use case

illustrates the architectural rendering of this concept. This diagram consumes data-sets from the matrix and scorecard analysis and annotates the diagram with overlays that reflect (in this example) the effective coverage of *Protect* function controls. This visual reference becomes a powerful assessment, communications and planning tool.

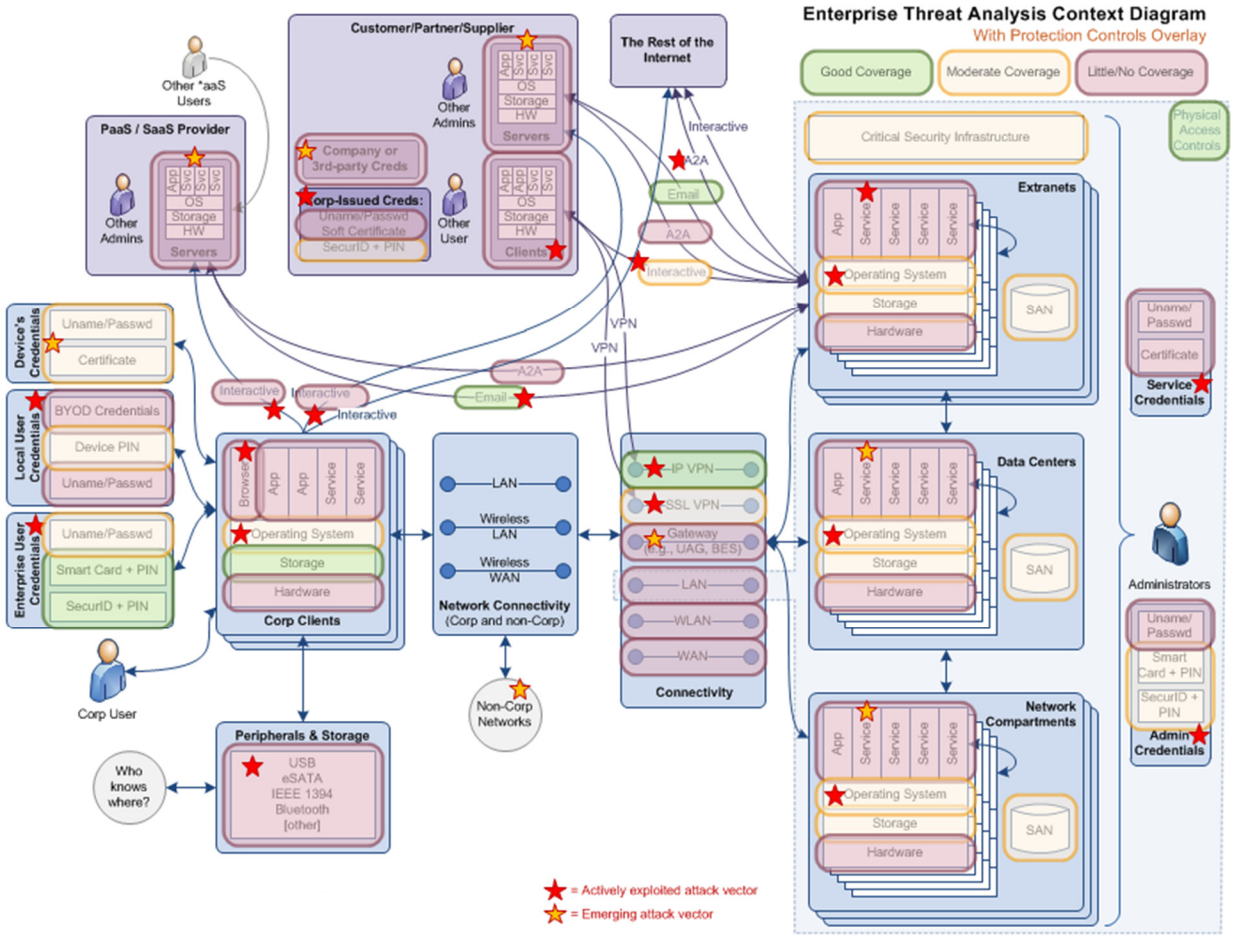


**Figure 14 - Data-bound Architectural Controls Effectiveness Mapping**

A large degree of flexibility and numerous potential use cases exist for the combined toolset of the Functional Controls Hierarchy, Controls Effectiveness Matrix and Scorecard and data-bound architectural coverage diagrams. This paper is introducing the concepts, methodology, practices and tools that enable these artifacts to be realized and is providing guidance on how to achieve the threat-driven approach with tangible examples. However, a full exploration of all use cases and applications of these tools is beyond the scope of this paper.

## 4. The Integrated Threat-Driven Approach

In Section 2 of this paper, an integrated, threat-driven approach to cyber security was presented. The intervening sections have presented the means to achieve this integrated approach: the methodologies, practices and tools. Adoption of these methodologies, practices and tools will enable organizations to realize the current and future benefits of the threat-driven approach. A summary of the key elements of the threat-driven approach are presented below:

The methodologies:

- IDDIL/ATC
- Intelligence Driven Defense®

The practices:

- Threat Modeling, Attack Trees, Threat Profiles
- The Cyber Kill Chain®
- Controls Effectiveness Ratings

The tools:

- Threat Categorization table
- Functional Controls Hierarchy
- Controls Effectiveness Matrix and Scorecard
- Intelligence Management System
- Architectural Renderings

Figure 15 presents a full view of the integrated Threat-Driven Approach. This diagram illustrates the implementation of the methodologies, practices and tools in a functional relationship model, including the correlating cyber security roles' (architect, engineer, and analyst) alignment to the relevant practice/tool. This model is the exploded view of the introductory concept presented in Figure 2.

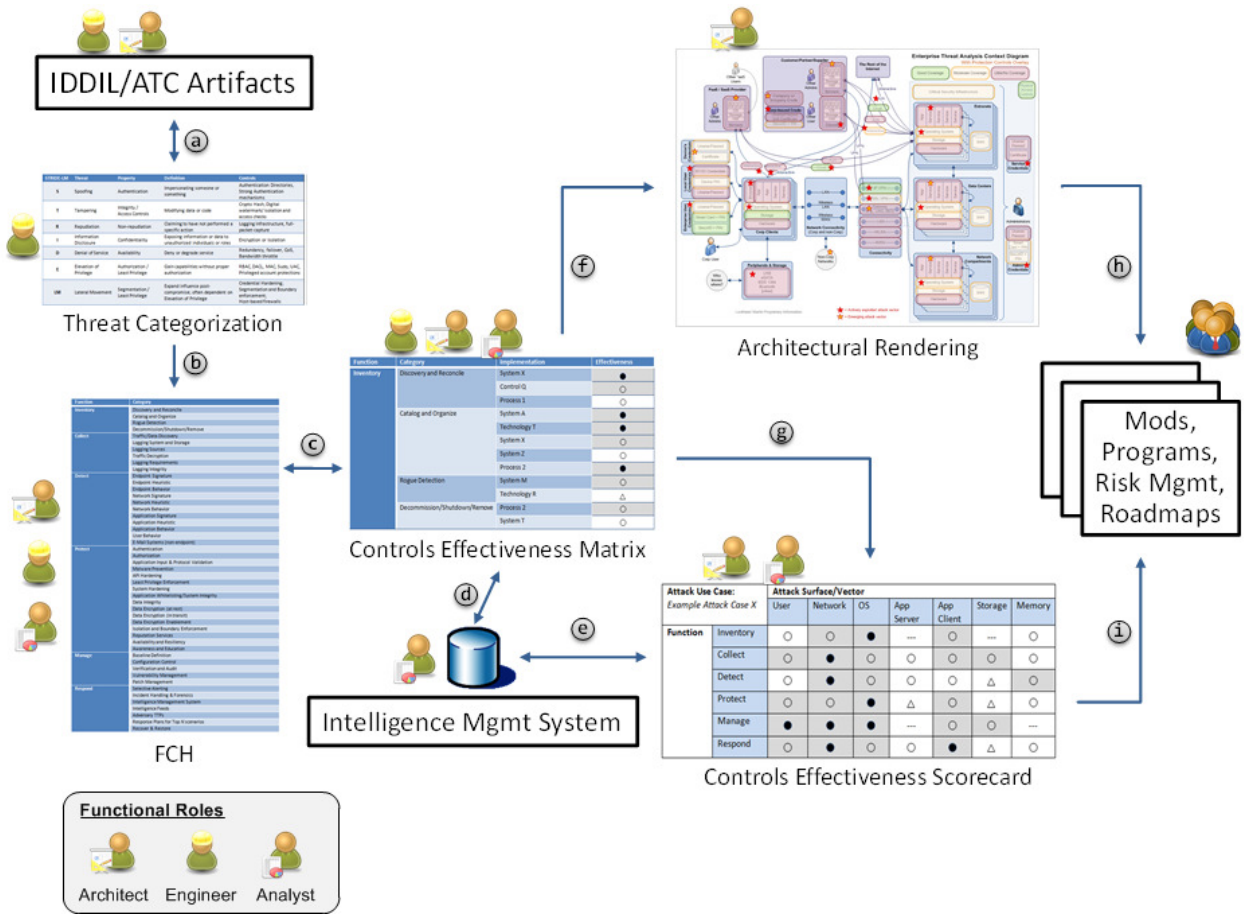


**Figure 15 – The Integrated Threat Driven Approach**

At the far right of Figure 15, the primary outputs of this threat-driven approach are presented. Modifications to existing controls, services and architectures; strategic programs instantiated or continued to address critical security needs; risk management based on high quality threat analysis and threat intelligence and balanced against controls ratings; roadmaps established to achieve strategic goals, to address critical gaps and to align with strategic programs.

Although each of the respective practices and tools, and how they interface, have already been described in this paper, a summary description of the integrated process flows in Figure 15 is described below:

(a) – IDDIL/ATC ← → Threat Categorization
Threat modeling and analysis artifacts will describe threats and attack vectors that will correspond to the security properties and controls in the threat categorization table; thus enabling a threat-driven selection of controls per the IDDIL/ATC activities. The scope of work at this stage can be application, system, environment or enterprise level.

(b) – Threat Categorization → Functional Controls Hierarchy
The controls column in the threat categorization table aligns to corresponding references in the current portfolio of controls/services in the Functional Control Hierarchy.

(c) – Functional Controls Hierarchy ← → Controls Effectiveness Matrix
The controls effectiveness matrix is an extension of the FCH. The FCH establishes the potential

controls to be evaluated. Once evaluation is performed within the matrix, results are returned back to the FCH to validate existing entries and ratings, to document gaps discovered during analysis, or update a control/service based on deltas in threats, attack vectors or architecture.

(d) – Controls Effectiveness Matrix ← → Intelligence Management System
A threat-driven evaluation of controls using the matrix requires current threat intelligence, which is provided by the intelligence management system. Analysis results are either validated against the intelligence management system or trigger updates to the system.

(e) – Intelligence Management System ← → Controls Effectiveness Scorecard
This interface is functionally equivalent to the previous interface. However, the context is more specific to the identification of attack use cases and an aggregate analysis of control functions.

(f) – Controls Effectiveness Matrix → Architectural Rendering
The qualitative results from the matrix are given quantitative values which are then bound to architectural component models to visually communicate the analysis performed. The scale of the rendered diagrams reflects the scope of analysis from the matrix.

(g) – Controls Effectiveness Matrix → Controls Effectiveness Scorecard
The controls effectiveness scorecard presents the aggregate summary of analysis results from the controls effective matrix, per specific attack use cases. This is the same data-set that is used in interface (f) to build the architectural renderings.

(h) – Architectural Rendering → Outputs/Actions
The architectural rendering diagrams are an essential communications tool to drive strategic cyber activities. Coupled with the controls effectiveness scorecard, these diagrams become an authoritative record for enterprise/organizational controls coverage.

(i) – Controls Effectiveness Scorecard → Outputs/Actions
The scorecard presents the same data-set results as the architectural rendering diagrams, displayed as a coverage graph of functional controls vs. attack surface. The graph display assists in the communication of the analysis results. Coupled with the architectural diagrams, a more complete message can be communicated to business, technology and cyber security decision makers.

The linkages between threat analysis, threat intelligence, and controls effectiveness ratings are clearly identified. The full scope of potential applications of the Threat-Driven methodologies, practices and tools is considerable. This paper has provided working examples based on cyber security practice refinements over a six year time frame. The examples are provided to demonstrate the implementation details of a fully integrated cyber security practice. This integrated functional approach has established a current-state agile and resilient cyber security posture, which is equally well-positioned for future cyber security challenges.

## 5. Risk Management

The terms “threat” and “risk” are oft-used phrases in cyber/information security. They are at times incorrectly used interchangeably. For the purpose of this paper the following definitions of threat and risk are used:

- Threat: an event, condition or consequence that produces adverse effects or undesired results
- Risk: a relative value produced by the analysis of *probability* and *impact* of a threat being realized or an exploit occurring

In general, a threat is relatively static over time and risk is a more dynamic value based upon the changing environment variables, current TTPs, existing controls, business/mission objectives and adversarial interest. If the assets of the business/mission remain constant, then the threats against them remain constant; it is the relative level of risk that will vary over time.

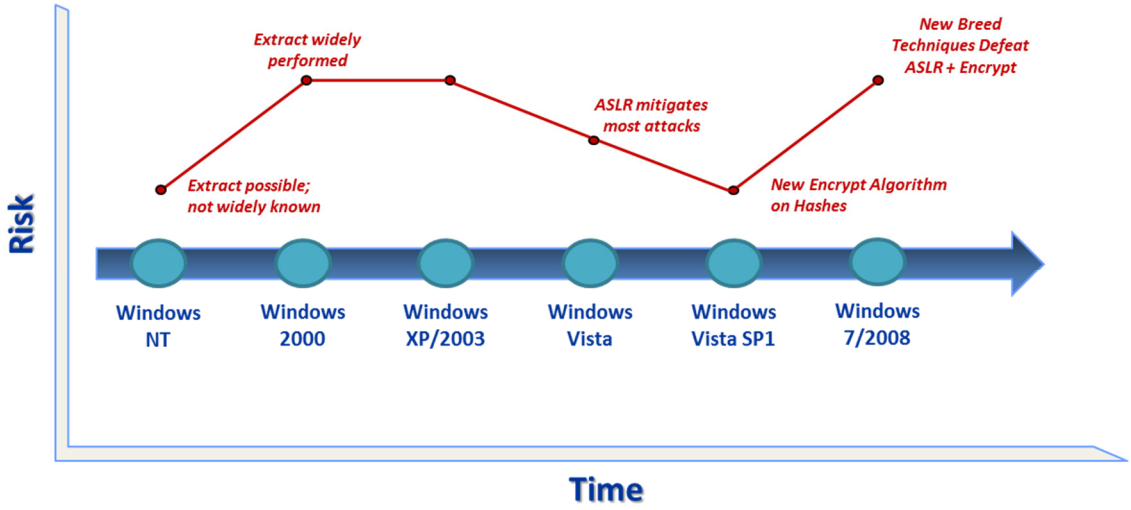


Figure 166 - Threat vs. Risk: Windows password hash extraction

To illustrate this concept, Windows password hashes provide a great example. The *threat* of extraction of Windows password hashes from memory has existed since Windows NT. Applying IDDIL/ATC and STRIDE-LM to this scenario produces the following:

- IDDIL/ATC:
  - Asset: password hashes/password (**I**DDIL)
  - Attack surface: protocols, libraries, memory structures (I**DD**IL)
  - Attack vectors: arbitrary code exploit + abuse native APIs (IDD**I**L)
- STRIDE-LM
  - Information disclosure of password hashes (STR**I**DE-LM); primary threat
  - Elevation of privilege (STRIDE-LM); resultant threat
  - Lateral movement (STRIDE-**LM**); resultant threat

The *risk* of successful extraction has varied over time – based on probability and impact – because of the continuous arms-race related to the tools available to perform the extraction of the hashes, the awareness and implementation of these techniques by larger and more advanced groups of threat actors, and the evolution of protection mechanisms to prevent/disrupt this extraction. Figure 16 illustrates this in a relative manner: the constant, horizontal blue line represents the threat and the fluctuating thin red line is the relative risk, corresponding to changes in attack techniques or protection enhancements over time, and is more weighted towards probability in this example. Impact analysis would require the context of an existing system or environment.

This paper will not attempt to add a new risk management framework or formula to the many already available; rather, it will elaborate on the following key topics regarding risk:

- The role of threats and threat analysis in risk level determination and risk management
- A suggested approach for the *probability variable* in risk discussions
- The phrase “risk acceptance” in its proper context

## Risk Lifecycle

Most risk management frameworks use the word "risk" in multiple contexts. If risk management is to be done effectively, a clear articulation of the various aspects of *risk* must exist. Risk in the cyber security domain has several different contexts and usages. There is a clear "risk lifecycle" progression. The architect, engineer or analyst must understand which part of that lifecycle applies to the work s/he is performing, and how each different phase in this lifecycle will impact the others. The risk lifecycle phases are presented below:

- Project or program inception: At this point in time a risk level rating or categorization (usually weighted towards impact) is assigned. This rating is intended to drive selection and allocation of certain types of controls as well as determine the level of engineering rigor required.
- Systems development/engineering: During these phases, risk ratings or labels are expanded and updated based on discoveries during design, build and test, or reflects changes to the baseline concept and requirements. This is identification and communication of potential risk.
- Transition to production: This is often where terms such as residual risk, aggregate risk, risk acceptance and risk management are utilized.
- Operations: Cyber risk management becomes part of the overall business/mission risk management process.
- Assessments: Risk assessments are performed on existing, operation systems and environments

### Risk Management and Risk Assessment

The role and function of risk assessment activities and how they fit into risk management practices requires granular clarification. Risk assessment is always a point-in-time evaluation. It has little to do with true risk management. Risk management is constant. While periodic evaluations are necessary for an organization to understand how to make broad adjustments, true risk management is best informed through the threat methodologies presented in this paper. If risk management is the continuous evaluation of the impact and probability of any undesired condition (i.e. threat) occurring, then high-quality threat analysis artifacts and current threat intelligence are the most accurate data elements to steer risk management discussions and decisions within any security related domain. The challenge – as it has always been in cyber/information security – is to articulate and evaluate the variables of *probability* and *impact* in a repeatable and reliable manner to effectively manage and reduce risk.

For organizations that have a mature threat intelligence capability, *probability* should be heavily influenced by threat intelligence. By aggregating assets and attack surfaces, the probability of attack easily reaches 1.0. If an organization asks the question: "Will we be attacked by some cyber adversary?" – the answer must be "yes". Therefore, organizations must leverage their threat analysis artifacts to understand the potential exposure to its assets, and threat intelligence to provide visibility into how those attacks will occur.

Probability or likelihood may also fail to drive appropriate risk management practices for occurrences of high-impact vulnerabilities. In the last few years, what probability value would cyber professionals have assigned to: compromise of RSA SecurID, OpenSSL (Heartbleed) and the Bash shell (Shellshock)? Most cyber security professionals would have given each of these vulnerabilities a (very) low rating, suggesting a low risk. Yet all of these events occurred. Using probability of vulnerabilities is not an effective risk management practice. Focusing on the constant threat is what enables improved risk management.

It is more effective to leverage threat analysis artifacts and threat intelligence – which includes the *functional controls effectiveness* practices – to make informed risk management decisions. Threat analysis inherently provides decision makers with the necessary data points, historical analysis and potential impact evaluations. One common problem in cyber/information security is the "must prevent" mindset; i.e. all attacks must be prevented. This limited viewpoint fails to consider compensating controls and also

neglects the concept of *protect and defend*, as presented in [23]. Defendable architectures and compensating controls are enablers of risk management and risk mitigation.

Coinciding with the misguided "must prevent" mindset is the concept of *risk acceptance*. Doing business means accepting risk – from the smallest home-office business to the largest organizations. Cyber risk is another facet that technologists, managers and executives must disposition and manage. Risk acceptance and risk management criteria must be determined per each scenario, and shift appropriately as the business/mission objectives, assets, threats and risk variables vary over time. Empowered decision-makers need to have the best information available to make educated decisions concerning risk acceptance and management. From the cyber/information security perspective, the most relevant and impactful information is produced by the threat methodologies described in this paper.

## 6. Summary

The combined threat-driven methodologies of IDDIL/ATC and Intelligence Driven Defense® empower organizations to unify architecture, engineering, operations and analyst roles in security engineering and cyber security domains. This unified approach drives organizational and functional alignment to enable a more mature and resilient defensive security posture. This threat-driven approach needs to complement and supplement the compliance-driven behaviors evident in contemporary IT and information security practices. Evidence of this more mature cyber capability is indicated by defendable architectures, which the threat-driven methodologies facilitate.

Security controls selection, implementation, evaluation and assessment should be viewed through the functional lens. This produces stronger cyber defenses and a more efficient workforce of cyber professionals. The common data sets resulting from functional security control analysis and attack use cases is relevant to analysts, engineers and architects – enabling tactical, operational, strategic and architectural enhancements simultaneously.

Risk determination, risk acceptance and risk management are all informed with a higher degree of fidelity by the threat methodologies outlined in this paper, allowing business decisions in information security, cyber security or any security related domain to have greater confidence in the ongoing risk management process.

## Definitions of Terms

**Asset:** any resource worth protecting (e.g., data, functionality, services, people, physical resources)

**Attack surface:** the collection of components and interfaces that a threat actor could use to realize a threat against an asset

**Attack tree:** a hierarchical, dependent-branched graph that describes the unique sequence of conditions required to accomplish a specific attack or realize a threat. (Note: attack trees were adapted from fault analysis trees)

**Attack use case:** aggregate representation of a set of related attack vectors

**Attack vector:** a specific sequence of exploits utilizing components within the attack surface to realize a threat against an asset

**Component:** any discrete element of a system (e.g., technology, processes, users, administrators)

**Control:** a technology, process, or policy that removes, counters, or mitigates one or more threats, attack vectors or vulnerabilities

**Control Effectiveness:** an assessment of a security control's ability to perform its intended function given the context of specific threats, attack vectors, and attack use cases

**Exploit:** the execution of an attack that takes advantage of one or more vulnerabilities in a system

**Risk:** Relative value produced by the analysis of probability and impact of a threat against an asset being realized

**Threat:** an undesired event or condition that would impact an asset

**Threat actor:** an entity that would attempt to impact an asset

**Threat intelligence:** knowledge about threat actors, their campaigns, objectives, tactics, techniques, and procedures

**Threat model:** a logical presentation of an application, system or environment's assets, attack surface and threat actors, decomposed to illustrate the flow of data and the threats against them

**Threat profile:** a tabular summary of threats, attacks and corresponding attributes

**Vulnerability:** a specific weakness or flaw in a component or system that can be used to perform unintended actions

## References


[1] E. M. Hutchins, M. J. Cloppert and R. M. P. Amin, "Intelligence-Driven Computer Network Defense," [Online]. Available: http://www.lockheedmartin.com/content/dam/lockheed/data/corporate/documents/LM-White-Paper-Intel-Driven-Defense.pdf.

[2] Microsoft Corporation, "Microsoft Security Development LIfecycle (SDL)," [Online]. Available: http://www.microsoft.com/security/sdl/default.aspx.

[3] BSIMM - Building Security In Maturity Model, "BSIMM," [Online]. Available: http://bsimm.com/.

[4] OWASP (Open Web Application Security Project), "Open Security Assurance Maturity Model," [Online]. Available: http://www.opensamm.org/.

[5] Carnegie Mellon Computer Emergency Response Team (CERT), "OCTAVE," [Online]. Available: http://www.cert.org/resilience/products-services/octave/index.cfm.

[6] Cigital Corporation , "Architecture Analysis," [Online]. Available: http://www.cigital.com/services/architecture-analysis/.

[7] Google, Inc. , "Google End-to-End Threat Model," [Online]. Available: https://github.com/google/end-to-end/wiki/Threat-model.

[8] ThreatConnect, "Diamond Model for Intrusion Analysis," [Online]. Available: http://www.threatconnect.com/methodology/diamond_model_of_intrusion_analysis.

[9] Defense Information Systems Agency (DISA), "Application Security and Development STIG," 24 October 2014. [Online]. Available: http://iase.disa.mil/stigs/app-security/app-security/Pages/app-security.aspx.

[10] US Department of Defense, "Military Standard 1629A," 24 November 1980. [Online]. Available: http://www.fmea-fmeca.com/milstd1629.pdf.

[11] A. Shostack, "STRIDE Chart," 11 September 2007. [Online]. Available: http://blogs.microsoft.com/cybertrust/2007/09/11/stride-chart/.

[12] MITRE, "Common Attack Pattern Enumeration and Classification (CAPEC)," [Online]. Available: https://capec.mitre.org/.

[13] M. Mateski, C. M. Trevino, C. K. Veitch, J. Michalski, H. J. Mark, S. Maruoka and J. Frye, "Cyber Threat Metrics," Sandia National Laboratories, March 2012.

[14] WebSense, "Security Threat Modeling: Six Steps to Success," 6 March 2014. [Online]. Available: http://community.websense.com/blogs/websense-insights/archive/2014/03/06/security-threat-modeling-six-steps-to-success.aspx.

[15] Ponemon Institute, LLC, "Exposing the Cyber Cracks: A Global Perspective," Sponsored by WebSense, 2014.

[16] Microsft Corporation, "Threat Model Tool 2014; Cyber Trust Blog," [Online]. Available: http://blogs.microsoft.com/cybertrust/2014/04/15/introducing-microsoft-threat-modeling-tool-2014/.

[17] MyAppSecurity , "ThreatModeler Tool," [Online]. Available: http://myappsecurity.com/.

[18] OctoTrike.org, "Trike Tools," [Online]. Available: http://octotrike.org/tool.shtml.

[19] A. L. Estes, *Secure Software Engineering for Designers and Architects,* Lockheed Martin.

[20] B. Schneier, "Attack Trees: Modeling Security Threats," *Dr. Dobbs Journal,* December, 1999.

[21] I. Green, "Extreme Cyber Scenario Planning and Fault Tree Analysis - GRC T17," in *RSA Conference*, San Francisco, CA , 2013.

[22] Codenomicon, "Heartbleed," April 2014. [Online]. Available: http://heartbleed.com/.

[23] S. C. Fitch and M. Muckin, "Defendable Architectures," 2015.

[24] H. Ochoa, "Pass-the-Hash Toolkit for Windows," in *Hack-in-the-Box Security Conference*, 2008.

[25] H. Ochoa, "WCE Internals," in *RootedCon*, 2011.

[26] M. Muckin, "Windows Vista Security Internals," in *BlackHat*, 2009.

[27] M. Muckin, "Exploration Initiative: Windows Password Hash Protections," in *DC3 DCISE Technical Exchange*, 2013.

[28] A. Ionescu, "The Evolution of Protected Processes Part 1: Pass-the-Hash Mitigations in Windows 8.1," November 2013. [Online]. Available: http://www.alex-ionescu.com/?p=97.

[29] Microsoft Corporation, "Credentials Protection and Management," [Online]. Available: http://technet.microsoft.com/en-us/library/dn408190.aspx.

[30] B. Delpy, "mimikatz - Golden Ticket," [Online]. Available: http://rycon.hu/papers/goldenticket.html.

[31] National Institute of Standards and Technology (NIST), *Special Publication 800-37 - Guide for Applying the Risk Management Framework to Federal Information Systems,* NIST.

[32] US Department of Defense, *Risk Management Framework - Instruction 8510.01.*

[33] National Institute of Standards and Technology (NIST), *Special Publication 800-53 Revision 4: Security and Privacy Controls for Federal Information Systems and Organizations,* NIST.

[34] National Institute of Standards and Technology (NIST), *Special Publication 800-30: Guide for Conducting Risk Assessments,* NIST.

[35] National Institute of Standards and Technolog (NIST), *Special Publication 800-39: Managing Information Security Risk,* NIST.

[36] G. Peterson, "Friend of the Devil and the Shostack Code," 12 March 2014. [Online]. Available: http://1raindrop.typepad.com/1_raindrop/2014/03/this-is-part-three-on-looking-at-governance-and-compliance-in-the-first-post-we-looked-at-charlie-mungers-comments-on-gove.html.

[37] National Institute of Standards and Technology (NIST), "National Institute of Standards and Technology (NIST) Cybersecurity Framework," February 2014. [Online]. Available: http://www.nist.gov/cyberframework/upload/cybersecurity-framework-021214.pdf.