Mission Impact Assessment for Cyber Warfare

Jared Holsopple, Shanchieh Jay Yang and Moises Sudit

1 Introduction

Cyber networks are used extensively by not only a nation's military to protect sensitive information and execute missions, but also the primary infrastructure that provides services that enable modern conveniences such as education, potable water, electricity, natural gas, and financial transactions. Disruption of any of these services could have widespread impacts to citizens' well-being. As such, these critical services may be targeted by malicious hackers during cyber warfare. Due to the increasing dependence on computers for military and infrastructure purposes, it is imperative to not only protect them and mitigate any immediate or potential threats, but to also understand the current or potential impacts beyond the cyber networks or the organization. This increased dependence means that a cyber attack may not only affect the cyber network, but also other tasks or missions that are dependent upon the network for execution and completion. It is therefore necessary to try to understand the current and potential impacts of cyber effects on the overall mission of a nation's military, infrastructure, and other critical services. The understanding of the impact is primarily controlled by two processes: state estimation and impact assessment. State estimation is the process of determining the current state of the assets while impact assessment is the process of calculating impact based on the current asset states.

Cleared for public release on 23 Apr 14, #88ABW-2014-1911

J. Holsopple · M. Sudit
Center of Multisource Information Fusion, CUBRC, Inc., Buffalo, NY, USA
e-mail: holsopple@cubrc.org

S. J. Yang (✉)
NetIP Lab, Department of Computer Engineering, Rochester Institute of Technology, Rochester, NY 14623, USA
e-mail: jay.yang@rit.edu

R.R. Yager et al. (eds.), *Intelligent Methods for Cyber Warfare*,
Studies in Computational Intelligence 563, DOI 10.1007/978-3-319-08624-8_11

In this chapter, we consider the case of a military computer network that could be subjected to external and insider attacks through various physical and virtual vulnerabilities. The goal is to provide an estimate of the *N*th-order impact of cyber threats while the missions are in operation. This is accomplished by a tree-based structure, referred to as a *Mission Tree*, which models the relationships between various missions, tasks, and assets. The relationships are modeled using Order Weighted Aggregators (OWAs),which provide for a diverse set of relationship types. The Mission Tree is different from other methods of modeling mission relationships in that it is capable of providing a quantitative estimate of impact by propagating the impacts "up", from the leaves to the root, through the tree.

Another key aspect of impact assessment is that missions or tasks will change during the course of warfare. This chapter will also explore how to dynamically change the mission tree to account for scheduled or non-scheduled changes and how those dynamic changes can affect how one performs impact assessment. This chapter will present a novel approach to address all of these aspects.

To ensure consistency of vocabulary, Sect. 2 provides a set of definitions, followed by a review of the existing methods for mission planning and assessment. Sections 4 and 5 discuss the mission tree structure and how it is used for mission impact assessment. Several examples and simulation results will be presented in Sect. 6 to demonstrate the utility of the mission tree.

2 Definitions

It is necessary that a state estimation process be executed first to determine asset damage, which is then fed into the mission impact assessment process. However, before discussing these processes in detail, it should be noted that there is no common vocabulary defined for mission impact assessment. In fact, vocabulary used for mission planning and impact assessment within one organization can sometimes conflict with the vocabulary used by another organization. Therefore, a set of definitions for various mission planning and impact assessment concepts is provided to maintain consistency throughout the chapter. The reader should take careful consideration of how these definitions differ from their organization's to maximize their understanding of the concepts presented in this chapter.

While this book focuses on computer security and cyber warfare, the concepts presented in this chapter are intended to be generic enough such that they are applicable to other application domains. Each definition given below will also provide a short description of how it applies specifically to cyber warfare.

Situation awareness—the state of knowledge that results from a process [1]. Situation awareness involves the detection and assessment of the threats to a computer network.

Situation assessment—the process that provides outputs that can be analyzed to gain situation awareness [2]. The situation assessment process is typically comprised

of a combination of sensors and data aggregation software manually analyzed by computer security experts.

Environment—the specific location in which events are being monitored and assessed. The protected computer network comprises most of the environment with respect to cyber warfare, but it can contain other physical or virtual elements such as buildings, rooms, or other networks.

Entity—something that has a distinct, separate existence, though it need not be a material existence [2]. Given this very general definition, it is important to define the granularity at which an entity is being defined for an application.

Object of Interest (OOI)—an entity or mission in the environment that should be monitored. In cyber warfare, this will generally refer to physical and virtual entities such as hosts, switches, services, and even communication links. The actual OOIs for a given network will depend heavily on how much is known about the network and what is considered critical or important enough to monitor.

OOI State—the condition of the OOI defined relevant to the application domain. An OOI may be in multiple states (e.g., *damaged* and *limited operations*).

Situation—a collection of activities and their effects on the environment at a given time.

Activity—something done as an action or a movement. They are composed of entities/groups related by one or more events over time and/or space [2].

Event—an occurrence that affects the environment [2].

Observable—one or more attributes of an event or object from a sensor or some form of intelligence.

Mission—a process by which a goal is intended to be achieved. It is possible that missions can be contained within other missions. It should be noted that we use the term mission in a very general sense throughout this chapter to represent any type of task that must be executed to achieve a desired goal.

Asset—an OOI that supports one or more missions whose state can be determined by a situation assessment algorithm. It should be noted that the granularities by which assets are defined with respect to the mission tree vary by application as well as the existing pre-processing mechanisms in place. We discuss this issue later in the paper.

Role—a function that an asset performs.

Impact—a quantitative assessment of how much a mission is affected by a given activity or situation. For consistency throughout this chapter we will use the term "impact" to imply "negative impact", unless otherwise specified.

Damage—a quantitative assessment corresponding to the state(s) a given asset is in with respect to its ability to perform a given role.

3 Mission Impact Assessment: A Brief Background Review

Mission impact assessment is not a new concept; however, it has traditionally been a manual approach. Grimaila and Fortson [3] argue the importance of fast and accurate

damage assessment, especially in "cyberspace where attacks can occur in milliseconds and may have a greater impact due to the complexity and interconnectedness of the information infrastructure. A failure to immediately detect, contain, remediate, and assess the impact following a cyber attack may result in other unforeseen higher order effects that may not be immediately apparent." The approach for impact assessment considered in this chapter can not only be used in the forensic phase of a cyber attack (i.e., after it has already occurred), but also during the attack. The ability to identify impacts that may not be "immediately apparent" *during* an attack can greatly improve the incident reports and enhance mitigation strategies by protecting assets that are not directly attacked. Grimaila and Fortson [3] also mention that a hurdle to fast and accurate damage assessment is the lack of asset documentation. This is problematic because the only way to estimate the direct and indirect impacts is to have an understanding of how assets interact with each other and are used for various missions and tasks. This understanding is also critical to the approach discussed in this chapter. In the past, the important components or concepts of a mission have been described in written language and/or diagrams. In some cases, including the use of cyber infrastructure to support military missions, the missions could be "implied" or undocumented by those who are assessing the health of the networked computing and storage systems. As such, modeling the mission in such a way that is understood by computers has been a daunting task.

Muccio and Kropa [4] describe cyber mission assurance as a four step process: (1) Prioritize mission essential functions, (2) map critical cyber assets, (3) vulnerability assessment of mission essential functions, and (4) mitigation of vulnerabilities and risks. This chapter will primarily focus on (2) and (3) to aid in (4). The prioritization of assets is also critical to mission impact assessment because it will determine the level of detail by which the models need to be developed.

Computer-aided mission impact assessment is part of a Decision Support System (DSS), which is a computer system/application that assists an analyst in evaluating the health of a system, task, or mission. A DSS can be traced back to 1982 when Ben-Bassat and Freedy [5] outlined the formal requirements for a generic DSS system that assessed the threat probabilities on various aspects of a given system.

Musman, et al. [6] discuss the evaluation of cyber attack impact on missions. They argued how critical it is to have mission models and descriptions stronger than what is available today. Their approach uses Business Process Modeling Notation (BPMN) and utilizes multiple information sources to develop the models necessary for impact assessment. Using cyber incident reports, they manually modify the mission model to produce new estimates, though the authors admit that this is a shortfall and are considering various alternatives to provide a faster impact estimate that is critical to effective impact assessment.

In the past decade or so, research has focused on modeling the mission dependencies to help facilitate computer-assisted analysis of current missions. D'Amico et al. [7] focused their research specifically towards computer networks by creating an ontology of the mission dependencies. Their approach focused on modeling how *cyber assets* provide a *capability* for each *mission*. While this approach provided the necessary modeling capability of mission relationships, it still required a graph-

ical analysis of the events to determine what was affected since it did not provide a numeric estimate for mission impacts.

Jakobsen [8] proposed the use of dependency graphs for cyber impact assessment. He also proposed the use of an "Impact Factor" to represent the capability of an attack affecting a given asset and the "Operational Capacity" that indicates the level to which the asset is compromised. These variables are similar to the impact scores and state estimation approaches discussed in this chapter. In addition, there has also been a wide array of other approaches to identify cyber attacks and their effects on computer networks, e.g., [9].

4 Asset State Estimation

The first step to determining mission impact is to determine a quantitative value of the damage to the objects of interest (OOIs) in the environment at a given time, which we will refer to as state estimation. The damage is calculated using inputs from an observable correlation process, which has grouped together observables in a meaningful way for a certain situation.

In the case of cyber warfare, the damage to an OOI corresponds indirectly to the level at which the information from hosts, services, and communication links can be trusted by the blue team. This section explores different ways that $d_i(t)$, the damage to an object of interest i at time t, can be calculated and used by the mission tree.

The state estimation process is not restricted to any single method of calculation. However, the following requirements are imposed for each value of $d_i(t)$:

1. Range of $d_i(t)$ is [0,1]
2. $d_i(t) = 0$ indicates that the OOI is operating normally and has no damage.
3. $d_i(t) = 1$ indicates that the OOI is unable to perform or be trusted for any of its tasks
4. $d_i(t) \in (0, 1)$ indicates that the OOI has some damage to it, but is still able to operate in a limited state. The extent to which the operations are limited tends towards the appropriate end of the range.

These requirements ensure that the damage scores are as consistent as possible across state estimation algorithms for different entity types. This in turn allows multiple state estimation algorithms to be used and potentially combined in various ways (see multiple state-space estimation, Sect. 4.4) to estimate $d_i(t)$ for a various types of OOIs.

4.1 Simple State Estimation

Simple state estimation is a direct calculation or assignment of $d_i(t)$ based on a set of rules or equations. As long as the rules or equations have known upper and lower

Table 1 Example rules for simple state estimation

Rule (Weight)	Criteria	Value
Firewall ($w_1 = 0.3$)	Traffic allowed	1 ($= mx_1$)
	Traffic forbidden	0
Attack type ($w_2 = 0.3$)	Reconnaissance	1
	User privilege escalation	2
	System (Root) privilege escalation	3 ($= mx_2$)
Connectivity ($w_3 = 0.4$)	Not connected	0
	Connected	1 ($= mx_3$)

bounds to the values, it can easily be normalized into the [0,1] interval. For example, consider three rules (Firewall, Attack Type, and Connectivity) that determine the current damage score of the state. Based on the maximum value of the score for each rule, each score can be normalized and combined with the other rules using a combination function such as weighted sum. In general, for R rules, each rule, r, will have a maximum possible value, mx_r, as well as a value, v_r, assigned to it. If we assign a weight, w_r, to each rule, we can calculate the weighted sum as:

$$d_i(t) = \sum_{r=0}^{R-1} W_r \frac{v_r}{mx_r}$$

The rules, criterion, and values are summarized in Table 1 which can be determined through various means applicable to each rule. For example, the firewall rule can utilize the firewall configuration to determine if certain IP addresses, protocols, and ports would be allowed. The attack type can be discerned by an alert aggregation tool that categorizes observables. The connectivity can be analyzed by using a known model of the network and routing tables to determine connectivity.

In this example, suppose a Reconnaissance observable is received and it reflects a penetration through the firewall to a connected target. In essence, this models a successful reconnaissance action on a target. Using a simple weighted sum, the aggregated asset damage score is as follows:

$$\begin{aligned} d_i(t) &= W_1 \left(\frac{1}{mx_1}\right) + W_2 \left(\frac{1}{mx_2}\right) + W_3 \left(\frac{1}{mx_3}\right) \\ &= (0.3)\left(\frac{1}{1}\right) + (0.3)\left(\frac{1}{3}\right) + (0.4)\left(\frac{1}{1}\right) = 0.8 \end{aligned}$$

In the equation above, it should be noted that each rule is normalized by its maximum value to obtain a final value on [0,1]. Also, while we have just presented the weighted sum as an example, any method to combine the values could be acceptable provided that the damage score calculations make sense for the rules.

4.2 Unbounded State Estimation

Recall that $d_i(t)$ must have a value on the [0,1] interval. However, some impact equations or rules do not have an upper bound. A simple example of this would be a score that increases by one every time a successful attack targets an OOI. In order to resolve this, one can simply define a maximum value corresponding to a value that is, for all intents and purposes, a value high enough that analysts would need to act upon. Once the maximum value is known, the damage score can be normalized into the [0,1] interval by saturating all values to a maximum then dividing by the maximum score. It should be noted that when exponential or special polynomial functions are used, the assessed damage scores can be close to the extreme values given the nonlinearity nature of the functions.

4.3 Single State-Space Estimation

This is a variant of the simple state estimation where, instead of the direct calculation, a qualitative state is determined then converted into a quantitative value. Such an approach provides a descriptive, language-based state for each OOI.

In cyber warfare one can consider the "Red" state space for an OOI, corresponding to the level of control or knowledge the red team may have on an OOI. To demonstrate this approach, consider the following five mutually exclusive states in the order of increasing severity:

- **Normal**—there is no indication of malicious or suspicious activity affecting the OOI. This is the default state for each OOI.
- **Attempted**—there has been at least one malicious or suspicious activity targeting the OOI, however, the targeted attacks have been unsuccessful.
- **Discovered**—there has been at least one successful attack targeting the OOI, however, the targeted attacks have only yielded information about the OOI.
- **Partially Compromised**—there have been successful attacks targeting the OOI that have given some control of it to the red team, however, the red team does not completely control the OOI.
- **Compromised**—there has been at least one successful attack targeting the OOI which has given the red team complete control of the OOI, thus is cannot be trusted by the blue team.

The determination of the qualitative state can be accomplished by establishing rules using various elements in the cyber environment such as, but not limited to, network connectivity, firewall rules, routing tables, sensor locations, and known vulnerabilities and services.

Once the state for an OOI has been determined, one can define a simple lookup table to calculate the damage score. An example lookup table is shown in Table 2. This example assumes a linear increase in damage score for each given state; however,

Table 2 Example states and damage scores for single state space estimation

State	$d_{i(t)}$
Normal	0.0
Attempted	0.25
Discovered	0.50
Partially compromised	0.75
Compromised	1.00

the values can be assigned logarithmically, polynomially, exponentially, or by some other method that adequately describes how the relative damage score increases:

4.4 Multiple State Space Estimation

This is an extension of single state-space estimation where an OOI can be evaluated across multiple state spaces. This method requires the combination of the $d_i(t)$'s for each state space. For example, suppose that, in addition to the Red State Space described above, there is an Operational State Space that defines the following four states: Operational ($d_i(t) = 0.0$), Maintenance (0.3), Degraded (0.7), Non-operational (1.0). It is therefore possible for an OOI to be in both the Degraded (0.7) and Partially Compromised (0.75) states simultaneously. Using this method, a new index, j, is added to the damage score, $d_{i,j}(t)$, to represent the state space the damage score is relevant to.

A combination function is needed to combine $d_{i,j}(t)'s$ into an overall damage score. Specifically, let J be the set of state-spaces, acombination function, $\bigoplus$, is such that $d_i(t) = \bigoplus_{j \in J}(d_{i,j}(t))$.

There are a few different combination functions that can now be considered:

4.4.1 Worst State

The simplest combination method is to use the worst state, which will be indicative of the highest damage score for a given state-space. This is defined by the max function:

$$d_i(t) = max_{j \in J}(d_{i,j}(t))$$

4.4.2 Weighted Sum

A different approach is to assign weights to each state space and combine them using a weighted sum. This approach only tends to make sense in a situation where some state-spaces may be more important than others when it comes to overall damage

Table 3 Example asset damage score combinations comparing DST with modified DST

Asset	$d_{i,1}(t)$	$d_{i,2}(t)$	$d_{i,3}(t)$	DST	Modified DST
1	0.1	0.2	0.3	0.001	0.3
2	0.1	0.1	0.8	0.047	0.8
3	0.1	0.7	0.8	0.509	0.903
4	0.9	0.7	0.8	0.988	0.988

and when all state spaces should have high damage scores to indicate a high impact.

$$d_i(t) = \sum_{j \in J} W_j d_{i,j}(t)$$

4.4.3 Modified Dempster-Shafer

Dempster-Shafer Theory (DST) [10] allows one to combine evidence from different sources to determine a degree of belief taking all pieces of evidence into account. DST uses a frame of discernment to define the possible states and mass functions (also referred to as basic probability assignment functions) to define the belief estimate from a given source. If we consider a source to be synonymous with a damage assessment for a given state space and a frame of discernment to be {D,N}, where D represents the belief that the asset is unable to perform its role ($D = d_{i,j}(t)$) and N represents the belief that the asset is able to perform its role ($N = 1 - d_{i,j}(t)$), we can use DST to combine the assessments into a singular assessment of the asset damage taking into account its damaged state for all state space.

One of the advantages of using DST to combine scores is that "high" scores can be combined into an even higher score. Likewise, "low" scores can be combined into an even lower score. This latter behavior is undesirable for calculating impact, so a piece-wise combination method can be used to combine the state spaces:

$$d_i(t) = \begin{cases} max_{j \in J}(d_{i,j}(t)) & , if\ all\ d_{i,j}(t) < 0.5 \\ \bigotimes_{j \in J, d_{i,j}(t) \geq 0.5}(d_{i,j}(t)) & , if\ \exists\ d_{i,j}(t) \geq 0.5, j \in J \end{cases}$$

With Dempster-Shafer a "high" score is greater than 0.5, and a "low" score is less than 0.5. As such, the above combination method uses the maximum score when all damage scores are low. This ensures that the final combined value will be at least as bad as the worst state. If at least one score is greater than 0.5, we use Dempster-Shafer combination to combine the high scores. This ensures that the combined score will be even higher than any single damage score.

Table 3 illustrates various combination scores for assets assessed across three different state spaces. It should be noted that in all cases, the modified DST score was at least as high as the DST score. Asset 1 had all low damage scores, and

the traditional DST calculation drove the final score even lower. This is undesirable because at a minimum we want to consider the lowest value. The modified DST calculation accounts for this by simply taking the maximum of the numbers, which is 0.3. Also note the disparity in the damage scores for assets 2 and 3. Having at least one state space with a high damage score should be of concern, so we ignore the low estimates and only combine the high ones. Note that asset 4's combined damage score is higher than any single state space.

4.5 Considerations for Damage Degradation over Time

For general mission impact assessment, it may be sensible for the impact or damage to decrease simply due to time. An example of this is an asset who was damaged, but has not received any impact changes for such a long time that the impact or damage is no longer relevant. As such, it may make sense to define an aggregation function that linearly, exponentially, or logarithmically discounts the impact or damage as time progresses, then "resets" itself once an event occurs to change the score. For example, we can define the following linear function such that *t'* is the last time state estimation changed the damage score due to an event and λ is the rate at which the damage degrades:

$$d_i(t) = \max(0, d_i(t') - \lambda(t - t')d_i(t'))$$

However, this concept could potentially lead to false negatives in the cyber domain, where "low and slow" attacks are very common. In a low and slow attack, a hacker can sometimes wait months between attacks on the network. The idea is that even if sensors or analysts do pick up their activity, the events will be too far apart to correlate with each other. In addition, if a hacker has successfully setup a backdoor undetected, the backdoor may be there long enough that it is also present in backups and may continue to be present even if a server or host is restored from a backup if the attack is detected. As such, we recommend that each "high" damage score be taken into consideration and if it is determined to be a false positive, the state estimation software should be able to allow for manual overrides to reduce the damage to the correct level. Automatic degradation may unnecessarily discount the damage if it goes unresolved over time.

5 Mission Tree Methodology

A mission tree is a tree-structure that captures the relationships between missions and assets. The mission tree is currently implemented as the impact assessment capability for Future Situation and Impact Awareness (FuSIA) [11]. As mentioned in Sect. 2, a "mission" is generically defined as any sort of task that must be performed, and an "asset" is a resource required for the execution of at least one mission. The mission

tree takes the damage scores calculated by state estimation and propagates them through the mission tree to provide mission impact estimates indicating the current health of a mission. The mission tree can also be used with various predictors or "what if scenario" interfaces to estimate future mission impact. The mission tree is intended to provide an estimate of mission impact, which is intended to trigger a deeper analysis for command and control activities. The intent of providing these estimates is to make more efficient use of an analyst's time since the impacts are a fast indicator of whether one or more events are adversely affecting a mission.

For example, a situation assessment tool such as FuSIA [11] will be able to suggest that a computer on a network has been compromised by a computer hacker using a Single State-Space Estimation. However, in order to determine how that impacts the other missions being executed, one must know not only how that computer supports each mission, but also be able to combine that information with the current state of other assets supporting that mission. In addition, that mission could be used to support other missions. The mission tree models these relationships and provides for various means of aggregating the data to provide a quick estimate of how far reaching the impacts are across other missions.

A static Mission Tree implementation would represent the state of the missions only at a specific point in time. In many applications, certain attacks may affect the mission more critical than others at different timeframes. This is even more challenging if new missions emerge over time, assets are re-assigned to assist in other missions, certain missions must be executed within a given timeframe, or the criticality of a mission or asset changes over time.

These dynamic and temporal changes must be taken into account in order to perform an accurate impact assessment in a timely and accurate manner. In Sect. 6, we will demonstrate how the mission tree is developed to incorporate dynamic and temporal mission changes through a fictional military computer network example.

Figure 1 shows the basic Mission Tree structure. A Mission Tree consists of three different types of nodes—assets, aggregations, and missions. Asset nodes must always be leaf nodes. An aggregation node is a node that performs a mathematical function to calculate the combined impact of all of the children nodes. Finally, the third node type is a mission node, which represents any type of task that needs to be executed either as the primary mission (the root node) or in support of another set of missions.

5.1 Asset Nodes

An asset node is defined by a 3-tuple *(i,e,c)* for a role r where i is the damage score for asset e in support of the parent mission with a criticality c. The criticality is used to describe the importance of a given node to the parent mission. An asset node must always be a leaf node and have an aggregation node as a parent.

Let $\boldsymbol{s_{a,r}}$ be a vector of probabilities such that $\boldsymbol{s_{a,r}(i) = p_i}$ where p_i is the probability that the function of the asset a is in state i. with respect to its role r. For example,

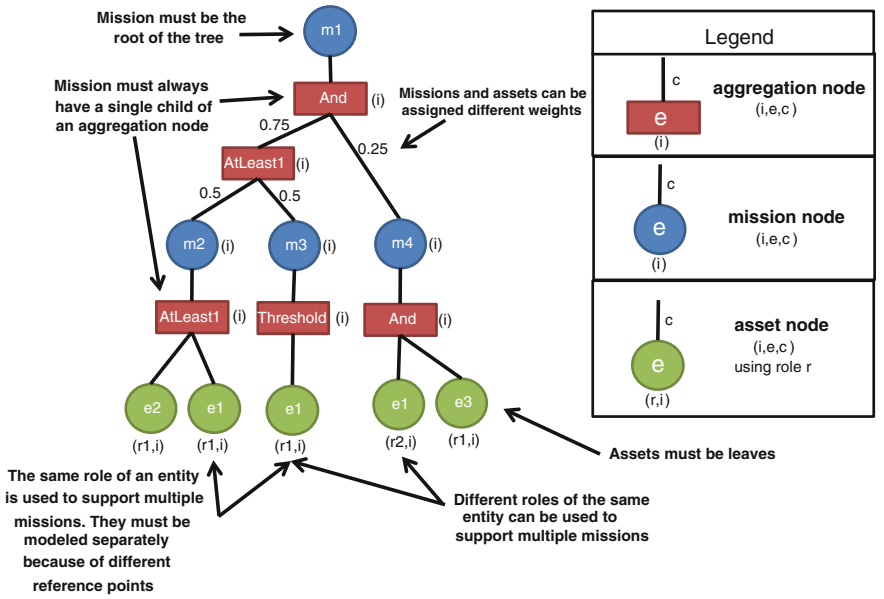

Fig. 1 Mission tree structure

$\boldsymbol{S_{a,r}} = (p_{normal}, p_{attempted}, p_{discovered}, p_{partially\_comp}, p_{comp})$ corresponds to the set probability that role r for asset a is in the Normal, Attempted, Discovered, Partially Compromised and Compromised states, respectively. $\boldsymbol{S_{a,r}}$ should be defined for every mutually exclusive set of states. If an asset only performs a single role, the r index may be removed.

Let $\boldsymbol{d}^*$ be a vector of the "numerical damage" for a given state. For example, we can define d such that $\boldsymbol{d}^* = (\boldsymbol{d}^*_{normal}, \boldsymbol{d}^*_{attempted}, \boldsymbol{d}^*_{discovered}, \boldsymbol{d}^*_{partially\_comp}, \boldsymbol{d}^*_{comp}) = (0, 0.2, 0.4, 0.6, 0.8)$. The $\boldsymbol{d}^*$ vector should be defined for every mutually exclusive set of states. We can therefore define $\boldsymbol{i} = \boldsymbol{d_{a,r}} = \boldsymbol{s_{a,r}} \bullet \boldsymbol{d}^*$ to be the damage score for an asset's role.

Using this definition, an entity can be represented across multiple missions by multiple asset nodes referencing the same entity. For example, if the network service "SSH" is an entity that supports three missions, three asset nodes referencing the SSH service are created. As a result, any change to the damage of the entity will trigger the recalculation of impact from each of these three asset nodes.

It is important to note that asset nodes merely *reference* the actual asset instance. So in essence, an asset node represents the damage score of the asset instance *with respect to* its parent mission. The reason for this distinction is to preserve a tree structure for an easier understanding of the impact score calculation process.

5.2 Mission Nodes

A mission node is also defined by a 3-tuple *(i,e,c)* such that *i* is the impact score for mission *e* in support of the parent mission with a criticality *c*. The criticality is used to describe the importance of a given node to the parent mission. Every mission node must contain a single parent (except for the root) and a single child aggregation node. For a mission node, *i= i(child aggregation node)*.

Recall that we generically refer to a mission as any task that needs to be completed to accomplish a given goal. Organizations may use strict mission structures comprising of multiple layers of missions, sub-missions, tasks, etc. However, this structure is not consistent across all organizations. So, for the sake of the mission tree, each of these elements are considered to be a "mission" since they are functionally treated the same for impact calculations. As a result, the depth of the mission tree can vary between organizations. A mission type is assigned to each mission to allow the user to distinguish which mission, sub-mission, task, etc. is being modeled.

5.3 Aggregation Nodes

Along with the tree-structure, the aggregation node enables the mission tree to not only model various relationships between its child assets, missions, or other aggregation nodes, but to also "propagate" impacts up the mission tree. The aggregation node enables mission-asset relationships through the definition of various functions that model various asset behaviors. An aggregation function shares mathematical similarities to a combination function used for state estimation. However, the combination of state spaces typically utilizes a single function, whereas a mission may utilize multiple functions forming more complex relationships aggregated together in a way meaningful to the mission-asset relationships.

Assets may have redundant behavior for missions, so "at least 1" of them may need to be un-impacted for the parent mission to execute. Assets could also have complementary behavior, meaning that "all" assets must be functional for mission execution. The aggregation nodes also provide for "at least N" relationships as well as "threshold" nodes. These various relationships all help to provide for an intelligent propagation of impact scores "up" the mission tree based on the estimated damage to each asset. The use of multiple aggregation nodes also allows for more complex relationships to be modeled.

An aggregation node calculates the combined impact of all children nodes and is defined by a 3-tuple *(i,e,c)* where *e* is an aggregation function with criticality *c* and *i=f(e)*. Every aggregation node must contain a parent that is either a mission node or an aggregation node. While an aggregation function can be any type of function, we adopt Yager's aggregation functions [12] due to their flexibility in defining various logical and mathematical relationships.

Yager's aggregation functions use a weighting vector multiplied with a *sorted* vector to perform various mathematical functions, such as maximum, average, and minimum. Due to their flexibility in function definition, they were chosen as the primary calculation means for the aggregation functions.

Each aggregation node is defined by a vector of weights, w, and a sorted vector, v_s, for Yager's aggregation calculations. The sorted vector is a vector sorted in descending order of all $\boldsymbol{i}^*\boldsymbol{c}$(impact multiplied by criticality) values defined by each child. The dot-product of each vector yields the impact score, i, for the aggregation node.

True Maximum is defined such that $w = [1, 0, \ldots 0]$. A true maximum calculation is equivalent to an "And" relationship between all of the children. This indicates that all of the children have complementary roles and must be fully functional to complete the mission, so the mission is "only as strong as its weakest link."

Weighted maximum is defined such that $w = [z, |ch|/(z-1), |ch|/(z-1), \ldots]$, $z <= 1$, where z should be close to one and $|ch|$ is the number of children. This can be used when it is highly desirable for all children to be fully functional, but the mission isn't completely non-functional when a child goes down.

True Minimum is defined such that $w = [0, \ldots, 0, 1]$. A true minimum calculation is equivalent to an "Or" relationship between all of the children. This indicates that all of the children are performing redundant roles. Thus only a single child needs to be operational for the mission to execute.

Weighted Minimum is defined such that $w = [|ch|/(z-1), \ldots, |ch|/(z-1), z]$. This can be used when the degradation of the functionality of the child will minimally impact the mission.

Average is defined with $w = [1/|ch|, \ldots, 1/|ch|]$. The average aggregator essentially captures a "consensus" between the children. However, it has not been found to be applicable to impact assessment calculations, so the use of this aggregator is not recommended.

It is also possible to model "At-Least-N" relationships that capture the situation where a minimum number of children are required to complete a mission. The "At-Least-N" operator is defined such that $w = [0, \ldots, 0, w_1, \ldots, w_n]$ where $w_n > \ldots > w_1$ and n is the minimum number of objects that must be operational. These are the primary aggregation functions used for the mission tree, and are mostly applicable to asset damage scores calculated by Single State Estimation. However, other aggregation nodes may be chosen dependent upon the state estimation technique used. For example, an impact score that grows exponentially may be skewed towards the higher end of the numbers. One could design an aggregation node that better distributes the scores evenly between 0 and 1. Finally, the modified Dempster-Shafer combination discussed in Sect. 4.4.3) could also be used as an aggregation function. This will allow the impacts to "grow" if multiple assets are impacted, thus being a good function for non-redundant assets. So in essence, the aggregation nodes can be used not only to model asset relationships, but to also change state estimation values as deemed necessary. Table 4 shows two examples of "at-least-2" operators.

Table 4 Two examples for "At-Least-2" operations

Sorted input	0.9	0.9	0.1	
Weight	0	0.75	0.25	Output
	0	0.675	0.025	0.7
Sorted input	0.9	0.6	0.4	
Weight	0	0.75	0.25	Output
	0	0.45	0.1	0.55

It should be noted that each aggregation node is defined independent of the number of its children. As such, when children are added or removed from an aggregation node, the new calculation vectors are automatically recalculated.

The threshold aggregation node is an example of one not using OWA. It is also a special aggregation node in that it can contain only a single child. A threshold aggregation node is a simple function that defines a minimum threshold of an impact value before it should be propagated upwards. This function simply returns a value if it is greater than the threshold, but returns 0 otherwise. This function can be important depending on the state estimation techniques used for the assets. If the "low" values for a state estimation technique are not important enough to propagate through to the parent missions, a thresholding node may be used.

These are the primary aggregation functions used for the mission tree, and are mostly applicable to asset damage scores calculated by Single State Estimation. However, other aggregation nodes may be chosen dependent upon the state estimation technique used. For example, an impact score that grows exponentially may be skewed towards the higher end of the numbers. One could design an aggregation node that better distributes the scores evenly between 0 and 1. Finally, the modified Dempster-Shafer combination discussed in Sect. 4.4.3 could also be used as an aggregation function. This will allow the impacts to "grow" if multiple assets are impacted, thus being a good function for non-redundant assets.

So in essence, the aggregation nodes can be used not only to model asset relationships, but to also change state estimation values as deemed necessary.

5.4 Calculating Impacts

A tree structure was chosen to represent the mission relationships because it naturally lends itself to "bottom-up" calculations. The calculations are performed using an optimized depth-first traversal of the mission tree to eliminate unnecessary calculations. For example, if a single asset's damage has changed, it is only necessary to recalculate the nodes on that particular branch. In addition, if the value on a node of that branch does not change, it is not necessary to continue the calculation since the parent nodes will not have changed.

After all of the calculations are performed on the mission tree, each asset and mission node has an impact score calculated for it, which corresponds to the fused effect the child nodes have on that node's ability to perform its task.

Initially we will assume the calculations to be specific to a given time *t*. This will ensure that the data structure is static for the analysis. For a given set of events at time *t*, we have one or more state estimation algorithms to determine the states of different assets within the environment.

When the calculated value of a node changes from its previous assessment, its parent is notified of the change, at which time the parent recalculates its new impact score. These recursive calculations continue until a calculated value does not change. After all of the damage scores have been determined, the parent aggregation nodes for all nodes whose damage scores have changed are then recalculated.

Due to the structure of the mission tree, every mission node has at most 1 child, which is an aggregation node. The value of this aggregation node also represents the impact score for the mission node.

5.5 Handling Mission Changes

Recall that we initially assumed the calculations for the mission tree to be performed for a single point in time. While this assumption was initially made for simplicity, it may not be a practical assumption in most applications, especially in the cyber domain where the life cycle of a cyber operation can be short. Over time, missions can be added or removed, assets may only be available at certain times, and missions may only be able to be executed within a given timeframe. As such, the assumption of the Mission Tree being defined for a specific time *t* may not be valid as temporal considerations would need to be taken into account. Therefore, the mission tree must be able to evolve with these changes.

In this section we will consider the different types of changes the Mission Tree can accommodate and how they can be triggered.

5.5.1 Types of Mission Tree Changes

There are various ways in which a Mission Tree can change. At first glance, how to handle the changes may seem as obvious as changing a node or a value within the tree, but there are subtle considerations to properly and efficientlymake the changes. In this sub-section, we will describe the different ways a mission tree can be modified and the considerations when making the change.

Adding an Asset

A new asset is typically created when a new resource becomes available for a given mission. The physical addition of the asset is as simple as adding the node as a child to an aggregation node. However, the following must also be taken into account:
Criticality—Determine how critical the asset is to the mission, relative to the other assets available.
Relationship to other assets—Determine whether the asset performs the same role as other assets supporting the mission. If it does, it should be added as a child underneath an "Or" or "At-Least-N" aggregator to indicate that it is a redundant asset. If the asset performs a unique and critical role to the mission, it should be added as a child to an "And" aggregator to indicate that this is a required asset. This is critical in ensuring that the aggregation nodes are correctly defining the mission dependencies. The aggregation nodes may need to be re-worked if the asset introduces a new mission dependency.

Adding a Mission

The considerations for adding a mission are very similar to adding an asset. However, unlike asset nodes, mission nodes are not leaves, we must also consider the tree structure "below" the mission.
When adding a mission, one must also consider which assets support the mission and also how those assets work together to perform that mission. This analysis will form the necessary structure for the aggregation nodes to capture these relationships.

Removing an Asset or Mission

When an asset or mission is removed, all of its children are also removed. As a result, careful considering must be taken into account so as to not make existing missions unable to perform their tasks. The impact of such node removals can become immediately evident when they are removed from the mission tree, which can be a benefit to the mission planner to assess the potential problems with removing a mission or asset. The obvious indicators of problematic removals are aggregation nodes with no children. Such nodes indicate that there is a specific dependency that is no longer modeled – and thus may indicate that a mission would be unable to perform. In addition, due to "At-Least-N" aggregators, it is possible that an aggregation node may not have enough children. These potential problems can be brought to the attention of the analyst so that they can be resolved immediately.

Re-Assigning an Asset

As missions evolve, assets may need to be re-assigned for various reasons. As such, the re-assignment of an asset is necessary to model. However, it is simply equivalent to the removal and subsequent addition of the mission and/or asset.

5.5.2 Change Triggers

We define a change trigger, as an event that causes a change to the mission tree. When a change is triggered, a list of actions, A, is executed to change the mission tree. The list of available actions were described in Sect. 5.5.1. Triggers are a critical inclusion to the mission tree in order to handle any changes to the mission tree, whether they are unplanned or planned.

Functional Trigger—These are one-time changes to the mission tree. As business or a particular task evolves, new missions may need to be created or existing missions are deemed unnecessary. These triggers are typically manual changes that must be made in order to accommodate changes that did not have a predictable time at which they became effective.

Absolute Temporal Trigger—These are one-time changes that are triggered by a single point in time. These changes typically represent a predictable change to the mission tree, such as a deadline for a given mission. When deadlines for missions have occurred, the mission is permanently removed from the mission tree. In addition, known or planned tasks in support of a mission can be created at the given point in time.

Cyclical Trigger—These changes are characterized by a predictable and cyclical change in the mission definition. These changes are typically caused by a business cycle, where certain assets may be more critical during normal business hours. In addition, due to resource availability, assets may only be available within certain timeframe, so there are only specific periods of time at which the assets are able to affect the mission. These changes result in a cyclic change of the mission tree.

5.5.3 Managing Mission Tree Changes

Depending on how frequently the mission tree changes, one may need to consider the best way to manage these changes. For a forensic analysis and to also consider past, or future, events in a "what-if" analysis, the mission tree needs to be stored at various points in time. While each calculation is calculated for a given time t, we need to be able to quickly assess the estimate of the mission for various times as events occur. As such, the proper storage of the tree is critical. There are two ways in which we can store a Mission Tree:

Absolute—This is the most straightforward approach, but also the most data intensive. In this approach, the entire mission tree is logged any time a change occurs. The time at which the change is effective for is logged in the database. When a calculation

is needed for a given time, the entire mission tree is defined. This storage approach may make more sense in a mission tree that does not change frequently.
Relative—In this approach, the mission tree is stored in its entirety every so often. Every time a mission tree changes, instead of storing the entire mission tree, only the action is logged. When a calculation is needed for a given time, the closest complete mission tree is first queried, followed by actions that were logged for modifying that base tree into the tree applicable for that time.

This approach requires much less storage space, but also is slightly more inefficient as a list of changes must be processed to determine the actual mission tree. This approach is best suited to applications where the mission tree is frequently changing.

6 Mission Tree Example

The mission tree was originally developed as a domain-agnostic approach to assessing mission impact to one or multiple application domains. This definition enables the impact assessment of missions that span multiple domains. For example, ground assets can be assessed alongside cyber assets to create a combined mission impact estimate. However, each domain has a specific way to "map" the mission tree. In this section, we will discuss how the idea of a mission tree is applied to cyber networks, assets, and missions.

6.1 Input Data

As has been mentioned earlier, there has been a cornucopia of work focusing on the identification of various types of cyber attacks, which can be leveraged as inputs to the Mission Tree. The approaches applicable to provide input data to the Mission Tree must be able to:

1. Correlate multiple cyber events together to represent a single cyber attack.
2. Provide a generic categorization of each event and, ideally, whether the event was successful in its execution.
3. Identify one or more assets affected in some way by each event.

With these three capabilities, we can determine the damage to each asset on a Mission Tree, which will then enable us to calculate mission impacts. In this example we will use the Single State-Space Estimation technique with the following states:

1. Normal (0.0)—nothing abnormal has occurred on or to the asset, and the asset is functioning normally.
2. Attempted (0.25)—an abnormal event has occurred on or to the asset, but the event did not seem to have any negative effects on the asset.

3. Discovered (0.5)—the adversary has some knowledge of the asset or attributes thereof. However, the asset seems to be functioning normally.
4. Partially Compromised (0.75)—one or more roles on the asset are under the adversary's control. Therefore, while the asset may appear to be functioning properly, the asset cannot always be trusted.
5. Compromised (1.0)—the asset is assumed to be under complete control by the adversary, so we cannot trust the asset and will therefore have the greatest impact to the mission.

6.2 Mission Tree Example

In this section, we will create a dynamically changing mission tree for a simple computer network for a fictitious mission to be executed by a nation's military. Some of the missions can only be executed at night, while some missions can only be executed during the day. In this example, we will demonstrate how compromised assets can affect not only the current missions, but also missions requiring the asset in the future.

Our example assumes that the "night" missions must be executed between 18:00 and 05:00, while the "day" missions must be executed between 05:00 and 18:00. The time in between the missions is intended for rest and/or maintenance. In addition, the CommServer is shared between the missions, but during the night is only exclusively available to the night missions and exclusively available to the day missions during the day. In addition, during the day, one of the day missions requires the use of at least 2 of the available workstations. It has also been found that the sensors monitoring these workstations trigger many false positives for discovery attacks. So any value less than or equal to 0.5 is most likely a false positive and should not be taken into account. The night missions require the use of at least 2 of 3 available field units that communicate valuable information back to the soldiers. As such, these are important cyber assets that also need to be protected.

We can create a *static* mission tree as shown in Fig. 2. Note that immediately above the "At Least 2" aggregator for the workstations is a thresholding function. This is defined to filter out the false positives that are known to be caused by the sensors monitoring the network. As such, only values above 0.5 will ever propagate up to Day Mission 1. Also note that the CommServer is shared across all four missions. However, this is not entirely accurate given that anything that affects the CommServer during the day will only affect the day missions. We will be using a modified Dempster-Shafer calculation for each of the "And" nodes.

We can define two cyclic triggers to modify the mission tree to give an accurate assessment based on the intended availability of the CommServer.

At the respective times, each of the triggers perform their actions. Which allows for a more accurate assessment of mission impact. The two cyclic triggers fundamentally create two slightly different mission trees show in Figs. 3 and 4. It should be noted that any mission or aggregation node with no children will always be assumed to have

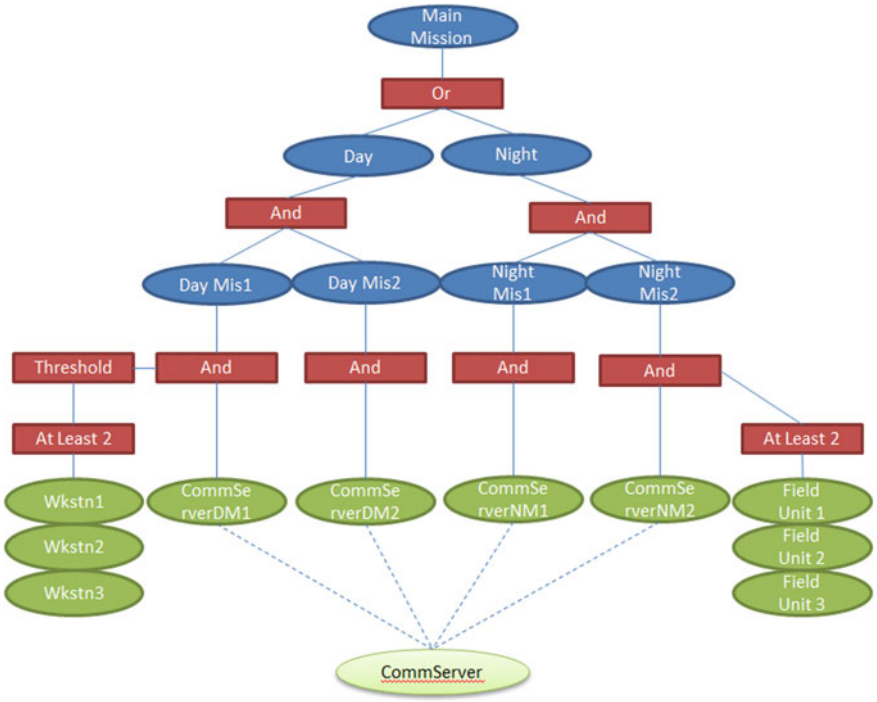

Fig. 2 Static mission tree for the basic day/night military operations

a zero impact. Also, the "And" aggregation nodes with singular children essentially just act as a pass through node. This minor detail is to ensure the data structure of the mission tree is maintained. It also allows for easier updates to the mission should any assets be added.

Figure 5 shows how the impact scores for each mission varied over time based on a set of cyber events defined in Table 5. Prior to the attack occurring, all missions had a 0.0 impact score. Note that due to the "at least 2" and thresholding aggregation nodes for the workstations, the impact to the workstations is not seen on Day Mission 1 until 13:00. It is important to note, however, that this does not imply that the events prior to 13:00 would have been ignored. It simply implies that it was not until 13:00 that the mission was negatively affected by the actions. In a realistic environment, the workstations would be monitored and the mitigation of workstation 1 should have immediately started when it was compromised. The different levels of the mission tree can be monitored by different people at the same time to try to prevent future events from causing higher level impacts.

By 15:00, the attacks to the workstations were resolved, so the impact decreases to 0. However, the impact spikes again when the CommServer is compromised. Note, however, that due to the dynamic nature of the missions, the night missions are not

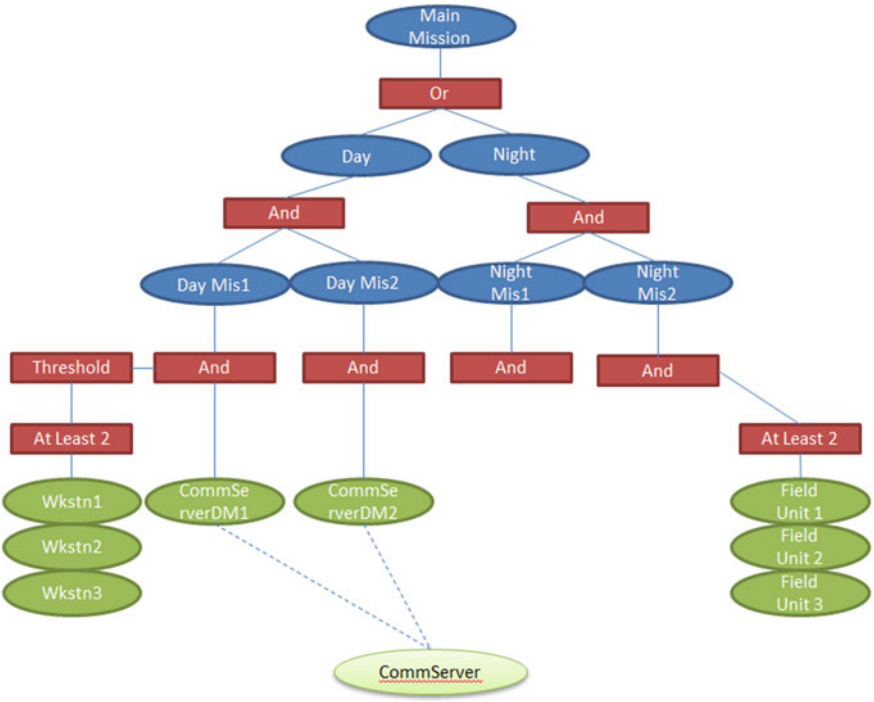

Fig. 3 Mission tree effective 8:00 M-F

yet affected. If the CommServer's compromise was mitigated prior to 18:00, there would have been no impact to the night missions.

6.3 Cyber Security Mission Tree Considerations

The previous section gives a simple example of a dynamically changing mission tree associated with a relatively small-scale computer network. The example demonstrates how a mission tree can be used to not only determine current impacts but also future impacts. Realistically, however, a typical enterprise network is comprised of hundreds, if not thousands of physical and virtual assets, so the practicality of the mission tree in such environments needs to be discussed.

Tools such as Snort® [13] or a more comprehensive tool such as HP's® Network Management Center [14] will be able to identify the states of assets which can then be fed to a state estimation process such as the simple state estimation process provided by FuSIA [11]. Each of these tools provides an initial data reduction to the point that the assessments indicate the damage to computers and their services.

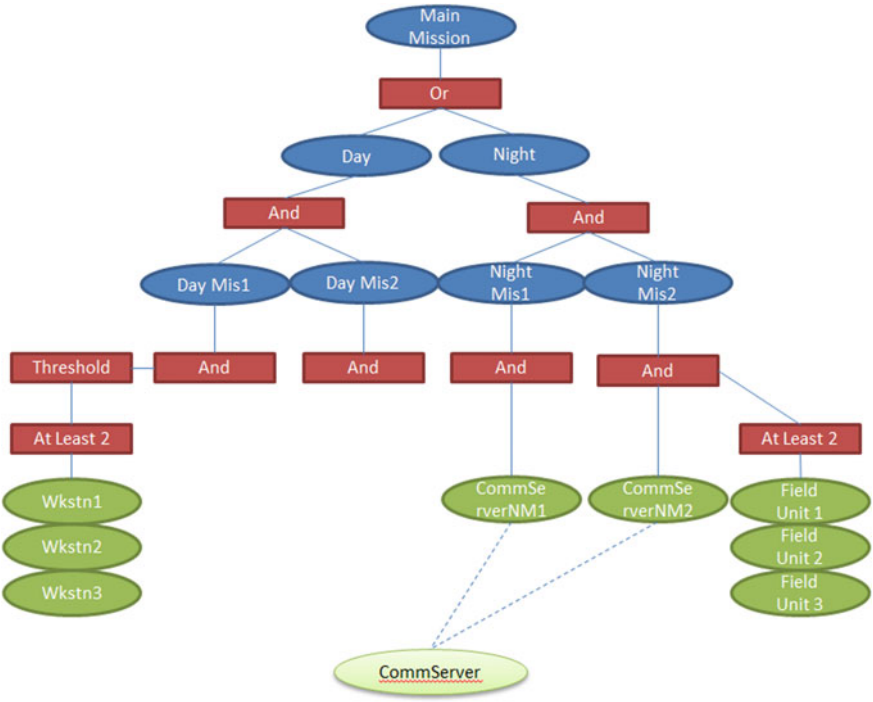

Fig. 4 Mission tree effective 18:00 M-F

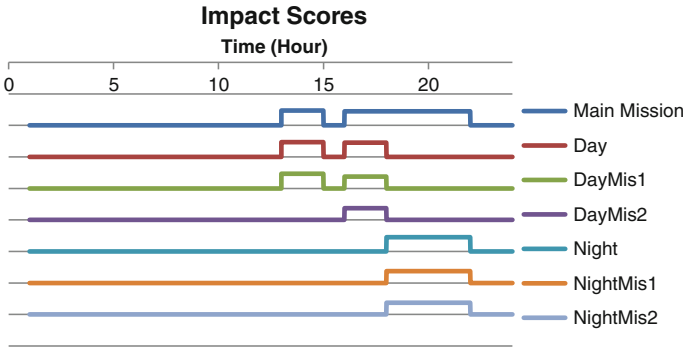

Fig. 5 Mission impacts esultant from Hacker attack over time

When modeling a mission tree for cyber warfare, an understanding of the computer network (e.g., the environment) is necessary for an adequate impact assessment. In addition, careful consideration must be taken into account to ensure that the granularity of the defined assets and roles is sufficient for accurate estimates, however, not so detailed that the provided assessments become unwieldy to manage.

Table 5 Triggers for mission tree

Trigger type	Time	Action list
Cyclic	8:00	Remove commServerNM1 Remove commServerNM2 Add commServerDM1 Add commServerDM2
Cyclic	18:00	Remove commServerDM1 Remove commServerDM2 Add commServerNM1 Add commServerNM2

Fundamentally, each analyst is looking to protect certain aspects of a computer network. Given the rapidly shifting focus to service-oriented architectures, whose resources can span one or more physical computers, we recommend modeling the services themselves as the assets—not the physical computers they reside on. Since many services can be provided redundantly, that can easily be modeled in the mission tree through the use of the "Or" aggregator where-ever that specific service is needed. Pre-processing of the events as described in the previous sub-section and knowledge of the computer network can help to facilitate estimates of impacts to specific services if only the physical computer being attacked is known. To further improve scalability, similar assets can be merged into a single asset to reduce complexity. For example, workstations (and their subsequent services) used by people of the same working group are often created from the same software image. Since each workstation would essentially be able to perform the same critical services, one could merge all of the individual workstation services into a single service on the mission tree.

If the "cyber missions" are ill-defined, the actual creation of the mission tree is not necessarily a trivial process. We recommend first building the top of the tree to mimic the structure of the organization, then creating specific tasks for each department and determining which services are necessary for the successful execution of these tasks.

It should also be noted that due to how generically the mission tree is defined, non-cyber assets can also be included on the mission tree. Although, of course, the assessment of those assets would require a different suite of situation assessment tools.

Using these considerations, a useful cyber mission tree can be developed that will allow for the quick estimate of mission impacts.

7 Challenges in Formal Evaluation of Mission Impact Assessment for Cyber Warfare

The previous sections presented a methodology and an example for estimating mission impact. There are, however, numerous challenges in practice to formally design an experiment and/or scenarios to evaluate the utility of this approach. This

Table 6 Cyber events

Time	Event	Damage
10:00	Discovery of Wkstn1	d(wkstn1)=0.5
11:00	Discovery of Wkstn2	d(wkstn2)=0.5
12:00	Compromise of Wkstn1	d(wkstn1)=1.0
13:00	Partial Compromise of Wkstn2	d(wkstn2)=0.75
15:00	Finished Mitigation of Wkstn1 & Wkstn2	d(wkstn1)=0.0 d(wkstn2)=0.0
16:00	Partial Compromise of CommServer	d(CommServerDM1)=0.75 d(CommServerDM2)=0.75
22:00	Finished Mitigation of CommServer	d(CommServerDM1)=0.75 d(CommServerDM2)=0.75

section details these challenges and suggests potential solutions so that one can work towards a better ability to not only evaluate the utility of mission impact assessment approaches, but also compare mission impact assessment approaches against each other.

7.1 Lack of Mission and Network Detail in Existing Cyber Warfare Scenarios

Perhaps the biggest challenge in evaluating mission impact assessment approaches is the lack of appropriate data. While there are some publicly available datasets to evaluate cyber warfare products, none of them currently provide any mission detail. This makes defining even somewhat accurate mission trees (and more specifically the aggregation nodes defining the relationships) impossible.Grimaila and Fortson [3] also argue that one of the limiting factors to perform damage assessment is due to the lack of documentation.

Even when it is available, the typical system information of an enterprise network is insufficient for accurate state estimation. For example, most data sets may provide a simple network diagram and a general list of services (not including version numbers). Most also do not include firewall or routing configurations. As such, state estimation may generate a large number of false positives, leading to poor perceived performance of the system—when in reality the poor performance is due to poor modeling.

Cyber warfare scenarios should include more detailed information about the computer networks, including (but not limited to) firewall rules, routing configurations, sensor placement, and as detailed of a list of services (including their versions) as possible. This information will significantly help the process of automated state estimation, thus leading to more accurate inputs to mission impact assessment. In

addition, a detailed mission description should be included so that the user may better understand the goals for each part of the network. In an ideal scenario, the mission description would be formulated in a graphical manner, however, at a minimum a textual description would be useful to define an appropriate mission model.

7.2 *Deviates Significantly from Current Cyber Defense Approaches*

Cyber warfare analysts are generally concerned with the current state of the protected network. As soon as they identify a potential threat to any asset on the network, they seek to mitigate it. However, cyber attacks often have effects beyond just the computer network. These effects are often not known at the time of the attack and require extensive forensic analysis to determine.

Defining the necessary impact assessment models such as the mission tree or the state estimation rules may not be a trivial process given that it requires more information than most organizations currently document and diagram. While diagrams and other documents may be available for a network, many details are often just known by the IT personnel or discovered on-demand, since the models described this chapter are not necessary to maintain a reliable computer network.

Mission impact assessment could aid in the forensic analysis by providing quick estimates of impacts to other missions that would help to better define a starting point for a more detailed analysis, while ignoring the less likely entities to have been impacted. In addition, mission impact assessment could also provide a similar starting point for the mitigation strategy.

In order to alleviate this difficulty, automated discovery tools must be used to help develop the models necessary for state estimation and impact assessment. While there are cyber defense packages available that integrate discovery tools together to gather some of the required information, they are often cost-prohibitive for many smaller and medium-sized organizations. In addition, each organization would also need to document and diagram their mission structure(s) to allow for the development of a useful mission tree.

7.3 *"Ground Truth" for Mission Impact Cannot Be Defined Numerically*

Even if a scenario provided sufficient enough detail for accurate state estimation and the development of the mission tree, defining ground truth for the mission tree outputs is relative and arbitrary. How is the ground truth defined for the impact scores? Would distance metrics potentially skew results for approaches that for all intents and purposes had equal performance (i.e., if ground truth defines an impact

of 0.8, and mission impact tools provide scores of 0.82, 0.76, and 0.99, are any of the scores really that much "better" than the others)? In addition, as evidenced in the simple scenario described above, the impact can vary significantly with time—even if no malicious or defense actions are taken in that period of time.

The nature of impact assessment often tends more towards educated opinion than facts, so defining ground truth for the purposes of collecting measures of performance may not be feasible due to the inability to quantitatively define the necessary values. As such, this problem is not well-suited towards any sort of formal "benchmarking", but there are other approaches focusing on the utility to the analyst that could be performed.

7.4 No Available Measures of Performance

Given that there is no ground truth that could be available for evaluating mission impact approaches, it is necessary to design an experiment for analysts identifying the threats and effects to the network. Such an experiment would require equally skilled analysts protecting the same network using different tools. Measures of effectiveness could then be collected to determine various metrics like attack response time and attack mitigation time.

8 Conclusion

In this chapter we have reviewed mission impact assessment for cyber warfare. Mission impact assessment can fundamentally be broken down into two processes: state estimation and impact assessment. The state estimation provides a damage score for each asset, and can be performed in multiple ways. The impact assessment provides quick estimates of mission health which can be calculated using a Mission Tree. Due to the dynamic nature of mission planning, the Mission Tree is also capable of dynamically changing over time to account for various mission changes. A simple military example was presented to discuss the potential utility for mission impact assessment. However, there still remain significant challenges in formalizing and deploying cyber warfare mission impact assessment processes due to the available technology and current network documentation processes.

References

1. Endsley, Mica R.: Toward a theory of situation awareness in dynamic systems. Hum. Factors J. **37**(1), 32–64 (1995)
2. Salerno, J.: Measuring situation assessment performance through the activities of interest score. In: Proceedings of the 11th International Conference on Information Fusion (2008)

3. Grimaila, M., Fortson, L.: Towards an information asset-based defensive cyber damage assessment process. In: Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications, (2007)
4. Muccio, S., Kropa, B.: Cyber Mission Assurance", http://www.wpafb.af.mil/shared/media/document/AFD-110516-046.pdf
5. Ben-Bassat, M., Freedy, A.: Knowledge requirements and management in expert decision support systems for (Military) situation assessment. IEEE Trans. Syst. Man Cybern **12**(4): 479–490 (1982)
6. Musman, S., Temin, A., Tanner, M., Fox, D., Pridemore, B.: Evaluating the impact of cyber attacks on missions. Mitre Corp. http://www.mitre.org/work/tech_papers/2010/09_4577/09_4577.pdf
7. D'Amico, A., Buchanan, L., Goodall, J.: Mission impact of cyber events: scenarios and ontology to express the relationships between cyber assets, missions, and users. In: Proceedings of 5th International Conference on Information Warfare and Security, Wright-Patterson Air Force Base, OH (2010)
8. Jakobsen, G.: Mission cyber security situation assessment using impact dependency graphs. In: Proceedings of the 14th International Conference on Information Fusion, (2011)
9. Yang, S.J., Stotz, A., Holsopple, J., Sudit, M., Kuhl, M.: High level information fusion for tracking and projection of multistage cyber attacks. Elsevier Int J Infor Fusion, Spec Issue High-level Inf Fusion Situation Awareness **10**(1), 107–121 (2009)
10. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press, Princeton (1976)
11. Holsopple, J., Yang, S.J.: FuSIA: future situation and impact awareness. In: Proceedings of the 11th ISIF/IEEE International Conference on Information Fusion, Cologne, Germany, (2008)
12. Yager, R.R.: Generalized OWA aggregation operators. Fuzzy Optim. Decis. Making **2**, 93–107 (2004)
13. Snort®, http://www.snort.org
14. HP® Network Management Center, http://www.hpenterprisesecurity.com/