Cyber-FIT Agent-Based Simulation Framework Version 4

Geoffrey B. Dobson, Kathleen M. Carley

November 17, 2021
CMU-ISR-21-113

Institute for Software Research
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

This work was supported in part by the AFRL under Cyber-FIT grant (FA86502126244). Additional support was provided by the center for Computational Analysis of Social and Organizational Systems (CASOS) and the Center for Informed Democracy and Social Cybersecurity (IDeaS) at Carnegie Mellon University. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the AFRL, or the U.S. government.

CASOS Center for the Computational Analysis of Social and Organizational Systems

Keywords: agent-based modeling, simulation, military, cyber warfare, Cyber-FIT

Abstract

This report describes the Cyber Forces Interactions Terrain (Cyber-FIT) Version 4, a simulation framework for computationally modeling cyber team performance. The projection of cyber mission forces into contested environments, and the simulation of the desired effects is very difficult. Military cyber teams are routinely deployed into environments with contested cyber terrain, but little is known about how well they performed. Cyber security training is already resource intensive in terms of both formal education and certification programs, yet cyber readiness and aptitude remain elusive to define. Cyber-FIT aims to help address these problems by computationally defining the performance measures of cyber teams. This model intends to be comprehensive and extensible. It is object oriented and modular in nature so new measures can be added over time without re-architecting the lowest level agent interactions currently in place allowing for new concepts and technological advances.

Table of Contents

1 Introduction

This report describes the current Cyber-FIT agent-based simulation model, version 4. Three previous versions have been developed aiming to create a framework that primarily allows researchers the ability to reason about the complex interactions of military cyber teams from both an individual and team level. It is an agent-based model describing the interactions between agent types based on rulesets defining stochastic behaviors and environmental changes. Cyber-FIT has been developed in a spiral development strategy, starting with the most basic agents and interactions, and adding complexity with each version. In version 1, the basic framework was created along with several proof of concept virtual experiments about the number of cyber forces needed to counter varying adversaries and types of attacks in different environments [1]. In version 2, more realistic adversary behavior was added to simulate movement through the cyber kill chain and how that affects cyber team defensive planning efforts [2]. In version 3, a cognitive model of cyber situation awareness was added in order to incorporate other types of theory development within cyber human-machine teaming research efforts [3].

Version 4, described within this report, adds two primary improvements. First, the model now incorporates knowledge, skills, and experience to defending cyber team behavior, whereas previous versions did not differentiate on those traits. Second, this version was completely redeveloped and architected using Argonne National Laboratory's Recursive Porous Agent Simulation Toolkit (Repast) Simphony [4].

2 Model Description

At the lowest level, Cyber-FIT is made up of agents and interactions. All agents are one of two main types: forces and terrain. From a military modeling and simulation perspective this is the highest-level categorization of agent types and allows for future model output porting and multi-modeling. Force agents represent military personnel in a conflict simulation and has three sub-types: defender, attacker, and friendly. Terrain agents represent the computer systems present in a cyber conflict simulation and has three sub-types: networking, serving, and host. Terrain agents, representing computers, are named as such due to the United States Department of Defense creating US Cyber Command and declaring cyberspace a terrain of war [5]. The interactions between agents are either force-to-force, force-to-terrain, or terrain-to-terrain.

2.1 Terrain Agents

The terrain agents represent cyber terrain: any computing machine that military forces depend on. This can include servers in a data center, a tablet used in field operations, laptops in a work center, etc. Terrain agents are the cyberspace assets that military cyber forces are vying to control. In this version, terrain agents are all owned by the defender agent side of the conflict. This simulates a deployment of a cyber team and focuses the development and computational modeling on performance measures defining success for that deployment. Terrain agent class behaviors and variables are defined in the following table.

Variables	
Name	Description
type	Type of cyber terrain, either networking, server, or host
status	Either operating normal, or compromised
vulnerabilities[]	List of vulnerability identification numbers that are currently present
payloads[]	List of payloads delivered by attacker agent currently present
missionID	Kinetic mission identification number this terrain agent is supporting
Behaviors	
Name	Description
step()	Every step a terrain agent will generate a random number of vulnerabilities that are now present, update its own terrain statistics, and then set its color for simulation visualization
generateVulnerabilities()	Each tick, a vulnerability might occur. Vulnerabilities are denoted by a vulnerability number between 0 and 99 that represents the severity of the vulnerability. The higher the number, the more severe the vulnerability, except for zero which represents a zero day vulnerability that can only be exploited by the most sophisticated adversaries
updateTerrainStats()	Update agent's own statistics
createConnection()	Connects to another terrain agent for computing purposes
addZeroDay()	Adds zero day vulnerability to itself due to attacker agent successfully developing and delivering a zero day vulnerability to it
sendMessage()	Connects to another terrain agent in order to send information message for defender or friendly agents
trySurvey()	Tries a survey operation initiated by a defender agent, which results in either a success, where terrain agent info is passed back to defender agent, or a fail, where the survey was not successful and no information was passed back to the defender agent
trySecure()	Tries a secure operation initiated by a defender agent, which results in either a success, where vulnerabilities identified by the defender agent have been removed, or

	a fail, where the identified vulnerabilities have not been removed
tryRestore()	Is in a compromised state, and tries a restoral operation initiated by a defender agent, which results in either a success, where the compromised terrain agent is restored to working or fail where the terrain agent is still compromised
tryAttack()	Is in an uncompromised state, and tries an attack where a payload that has been delivered by an attacker agent is either successful due to existing vulnerabilities, where the terrain agent becomes compromised, or a fail where the terrain agent continues working normally

Table 1: Terrain Agent Class Variables and Behavior Methods

2.2 Defender Agents

The defender agents represent the cyber forces deployed to the conflict with a mission to defend the cyber terrain that kinetic forces depend on to carry out their own missions. Defender agents are deployed to the simulated conflict as teams of any size, made up of members of any type, as denoted by the cyber forces configuration file. Once deployed, the defender agents will work together to share information about the cyber terrain, remove vulnerabilities from assigned terrain, and restore terrain that are compromised. All of their behaviors are based on some subset of their class variables, depending on the circumstances of the current run. Defender agent class behaviors and variables are defined in the following table.

Variables	
Name	Description
team	Cyber team identification number
squad	Represents the sub-team that the defender agent is assigned. There are three squad types: lead, network, and host. Lead represents the team leadership and intelligence operations. Network defender agents focus on vulnerable terrain that are networking and serving type. Host defender agents focus on vulnerable terrain that are serving and host type
knowledge	Knowledge level denoted as low, medium, or high
skill	Skill level denoted as low, medium, or high
experience	Experience level denoted as low, medium, or high
compromisedTerrain[]	List of terrain the defender agent believes to be compromised

vulnerabilitiesTerrain[:]	Table of terrain agents and vulnerabilities each terrain agent that the defender agent believes exist on that terrain
opType	Type of cyber operation currently working on
opTime	Current time working on current cyber operation
totalOps	Total number of cyber operations conducted
totalSurveySucces	Total number of successful survey attempts
totalSecureSucess	Total number of successful secure attempts
totalRestoralSuccess	Total number of successful restoral attempts
Behaviors	
Name	Description
step()	Every step a defender agent will either: continue restoral operations on compromised terrain, continue the current cyber operation they were working, or select a new cyber operation to begin
getNewOp()	Process defender agent goes through to select a new cyber operation to begin next step. The operations that the defender agent can select are one of seven types as defined CISA [6]: Analyze, Collect and Operate, Investigate, Operate and Maintain, Oversee and Govern, Protect and Defend, and Securely Provision.
continueOp()	Defender agent has not completed current cyber operation so it continues that cyber operation
interactWithForce()	Defender agent, based on their current cyber operation, needs to interact with another defender agent for communication purposes
interactWithTerrain()	Defender agent, based on their current cyber operation, needs to interact with terrain agents. This represents the cyber operations where a defender agent is attempting to survey, restore, secure, or message
surveyOp()	Defender agent, based on their current cyber operation, needs to use cyber terrain to survey other cyber terrain in order to update their cyber situation awareness
secureOp()	Defender agent, based on their current cyber operation, uses cyber terrain to connect to other cyber terrain in order to increase the cyber security of those terrain agents by removing vulnerabilities
restoralOp()	Defender agent is aware of compromised terrain and has been assigned to task of attempting to restore that terrain

hasCompromiseSA	Defender agent has become aware, or still is aware of compromised terrain agents that are compromised. Defender agent will share that information with other members of the cyber team
sendMessage()	Sends message to teammate based on current operation
sendMessageCompromised()	Has information about compromised terrain so shares that information with teammates or team lead

Table 2: Defender Agent Class Variables and Behavior Methods

2.3 Attacker Agents

The attacker agents represent the cyber forces assigned to attack the cyber terrain that the defending and friendly forces depend on for their military operations. In cyber war-gaming and exercises this is commonly referred to as OPFOR (opposing forces). Any number of attacker agents can be added to the conflict, with each attacker agent having a sophistication level as denoted by the simulation configuration files. The attacker agents work alone. Attacker agents work through the cyber kill chain as defined by Lockheed Martin [7] with the ultimate goal of compromising terrain agents. Once compromised, friendly forces cannot utilize those terrain agents to conduct kinetic operational missions. The modeling of how an attack works is based on the MITRE ATT&CK® framework [8]. Terrain agents must be vulnerable to an attacking technique by an attacker agent. As of this writing, ATT&CK has 215 techniques documented and described. In this version of Cyber-FIT, attacker agents have 100 techniques available. In real world operations, the 215 techniques could each exploit one to many vulnerabilities present on a computer network. To abstract that away, in this version of Cyber-FIT, vulnerability identification numbers and attack technique numbers are representing a similarity (attack matches vulnerability) that allows the attack to be successful. This level of complexity is by design so different taxonomies can be implemented in future versions. Attacker agent class behaviors and variables are defined in the following table.

Variables	
Name	Description
tier	Sophistication level of the attacker agent as defined by the Department of Defense [9].
phase	Current phase of the cyber kill chain that the attacker agent is engaged in
phaseTime	Amount of time spent in the current phase of the cyber kill chain
recons[]	List of terrain agent identification numbers that the attacker agent was able to successfully conduct cyber reconnaissance operations on
attacks[]	List of attack identification numbers that are currently available to the attacker agent

deliveredTo[]	List of terrain agent identification numbers that the attacker agent was able to deliver cyber payload to
attackAttempts	Number of terrain agents the attacker agent attempted an attack on during the current simulation
attackSuccesses	Number of terrain agents the attacker agent successfully compromised
Behaviors	
Name	Description
step()	Every step, an attacker agent continues on in whatever phase of the cyber kill chain it is in. If there has been an interruption, the attacker continues in "phase zero", simulating time between or before beginning an attack attempt
initialize()	Attacker agent initializes number of attacks and attack identification numbers available before starting the cyber kill chain and after every cyber kill chain attempt
reconPhase()	Attacker agent attempts to connect to terrain agents and discover vulnerabilities
weaponizationPhase()	Attacker agent spends time on one terrain agent preparing attacks to be delivered to other terrain agents
deliveryPhase()	Attacker agent delivers payload to terrain agents it believes to be vulnerable to that particular attack based on information gathered during recon phase
exploitationPhase()	Attacker agent waits as exploit is attempted by malicious code on terrain agent
commandAndControlPhase()	Attacker agent is able to interact with select compromised agents for the purpose of controlling that terrain agent and furthering attack objectives
actionsOnObjectivesPhase()	Attacker agent waits as further actions occur on own terrain

Table 3: Attacker Agent Class Variables and Behaviors

2.4 Friendly Agents

The friendly agents represent the military forces conducting kinetic missions associated with the simulated conflict. In order to achieve their objectives they depend on information and computing resources provided by the cyber terrain. Therefore, at any given time, friendly agents might connect to terrain agents associated with their mission to request information. These information requests are processed and, depending on the terrain agent status, fulfilled with a timing value or not fulfilled. Information requests have a mission assurance category level between one and three based on the Department of

Defense assigned criteria. Friendly agent class behaviors and variables are defined in the following table.

Variables	
Name	Description
missionID	Kinetic mission identification number this friendly agent is assigned to
infoRequests	Total number of information requests made during simulated mission
infoFulfills	Total number of information request fulfillments during a simulated mission
Behaviors	
Name	Description
step()	Every step a friendly agent may or may not connect to a terrain agent and make an information request

Table 4: Friendly Agent Class Variables and Behaviors

2.5 Force-Force Interaction Links

The force-to-force interactions are directed links representing force agents interacting within a simulated cyber conflict. The force-to-force agent interactions are informational in nature and related to either the cyber or kinetic operation currently being conducted by force agents. Force-to-force interaction link class variables and behaviors are defined in the following table.

Variables	
Name	Description
lifetime	Number of ticks this link will remain active
type	Type of link
Behaviors	
Name	Description
step()	Decrement lifetime value and if equal to zero link will die

Table 5: Force-Force Interaction Link Class Variables and Behavior Methods

2.6 Force-Terrain Interaction Links

The force-to-terrain interactions are directed links representing force agents interacting with cyber terrain agents during a simulated cyber conflict. Force-to-terrain agent interactions occur when any force agent (defender, attacker, or friendly) needs to utilize a terrain agent for any reason. Force-to-terrain interactions can occur because agents need to use terrain to message other agents, read information, update terrain, or send messages. Force-to-terrain interaction link class variables and behaviors are defined in the following table.

Variables	
Name	Description
lifetime	Number of ticks this link will remain active
type	Type of link
Behaviors	
Name	Description
step()	Decrement lifetime value and if equal to zero link will die

Table 6: Force-Force Interaction Link Class Variables and Behavior Methods

2.7 Terrain-Terrain Interaction Links

The terrain-to-terrain interactions are directed links representing terrain agents interacting with other terrain agents during a simulated cyber conflict. Terrain-to-terrain agent interactions occur when terrains are connecting to each other simulating all of the cyber operations and interactions built into this model. For example, when an attacker agent is using a cyber terrain agent to simulate the installation of malicious software onto a friendly agent's cyber terrain, an attacking type terrain-to-terrain agent interaction is created. Terrain-to-terrain interaction link class variables and behaviors are defined in the following table.

Variables	
Name	Description
lifetime	Number of ticks this link will remain active
type	Type of link
Behaviors	
Name	Description
step()	Decrement lifetime value and if equal to zero link will die

Table 7: Terrain-Terrain Interaction Link Class Variables and Behavior Methods

3 The Performance Measures of Cyber Teams

The primary design goal of this version of the Cyber-FIT Simulation Framework is to provide an apparatus to comprehensively and quantitatively measure the performance of a cyber team. This means after each run of the simulation all data is present in the output files that can be used to measure the simulated team performance, answering the question: How well did the team do? In order to create a list of performance measures, many conversations with subject matter experts have occurred. These conversations have occurred at cyber war exercises, cyber war-gaming sessions, cyber operations doctrine writing sessions, cyber security and simulation conferences, and through work at Carnegie Mellon University through CASOS Center events and workshops. Finally, a focus session was spent with a diverse group of military cyber operations planning experts validating the current model design and behaviors that lead to the collected data performance measures. The rest of this section is a detailed description of each measure. Each measure is explained as to how it is calculated, what behaviors affect the measures, what control variables affect the measure, and how this measure could be collected in operational systems. The following table defines all model specific terms referenced frequently throughout the remainder of this section.

Term	Description
Tick	A simulated time unit or period. Typically in agent-based modeling each tick represents a second, minute, hour, or other user defined time period.
Cyber Team	For the entirety of this technical report, cyber team refers to a group of defender agents assigned to the same team. Cyber-FIT allows for multiple team simulations but each performance measure is specific to a cyber team of defender agents.
Mission Defined	This refers to variables that are user defined in mission configuration files and context dependent. Mission defined can refer to expected mission outcomes, timing consideration, and details defining kinetic and friendly forces.

Table 8: Section 3 Common Terms and Descriptions

3.1 Terrain Vulnerability Rate

Terrain vulnerability level represents the total vulnerability level of a given network of computer systems (cyber terrain). From a modeling and simulation software perspective, this is an example of a very specific agent by agent measure that can be computationally quantified and aggregated to total terrain vulnerability level. In the Cyber-FIT model this means each terrain agent has a temporally changing list of vulnerabilities ranging in identification number from 0 – 99. Each identification number is also a proxy value representing the severity level of the vulnerability. The higher the number the more vulnerable this particular vulnerability makes the terrain agent. So, a terrain agent with vulnerabilities 90 and 80 is much more vulnerable than a terrain agent with vulnerabilities 9 and 8. This means the worst possible scenario for one terrain agent is that it becomes vulnerable to all attacks, or its list of vulnerabilities is all integers in the range [0,99] which is 4,950. Summing all vulnerabilities over all terrain agents gives total vulnerability level. Dividing by the number of terrain agents gives the terrain vulnerability rate.

3.1.1 Computation

Define T as the set of all mission terrain agents as a subset of all agents A

$$T \in A$$

Define V as the set of all vulnerabilities, V_i, that terrain agent T_J can have

$V_{T_j,i} \neq 0 \leftrightarrow T_j$ has vulnerability i

Then, total vulnerability level TVL of T is calculated by

$$TVL = \sum_{j,i} V_{T_j,i}$$

Finally, to normalize, *TVL* is divided by total possible vulnerability level for each terrain agent *j*, giving terrain vulnerability rate, *TVR*

$$TVR = \frac{TVL}{4{,}950j}$$

3.1.2 Operational Considerations

This military focused definition is similar to long-standing concepts of terrain based risk assessment. In conflicts, militaries will analyze, for example, land terrain positioning and determine where and how they are vulnerable to attack. This can be based on geographic considerations such as access to water, proximity to supply chains, difficulties with mountains, etc. In the newly emergent concept of cyber terrain, militaries will similarly conduct vulnerability assessments on this terrain type. Rather than analyzing physical components, the analysis is based on logical components, systems architectural, networking, software, and cyber security. When military cyber teams are deployed to protect networked systems, one of the first artifacts produced is a terrain vulnerability assessment. The assessment will touch upon aspects similar to those just mentioned. Terrain vulnerability level, at face value, is one of the most easily understood performance measures of a cyber team. The primary purpose of any military or corporate Information Technology (IT) department, is to make the network less vulnerable to attack (minimize terrain vulnerability level). This is done near continuously, every day, through system monitoring and updating. Most IT offices will have dashboards displaying vulnerability status of a wide array of systems. Those more vulnerable might be displayed yellow, and active problem systems could be red. Therefore, snapshots of system vulnerability level can be shown throughout the cyber team operations providing real-time quantified values of terrain vulnerability status. This measure is already reported on in real-world operations and arguably the closest to tracking the ground truth.

3.2 Terrain Vulnerability Change

Terrain vulnerability change builds upon the previous section by adding the change to vulnerability level over time. This represents a change measure at any given time period. In the Cyber-FIT model, this is measured by fitting a curve over a given period of ticks and then plotting the derivative of that function.

3.2.1 Computation

As previously defined, terrain vulnerability rate, TVR can be measured over time.

Therefore, define terrain vulnerability change, TVC calculated by

$$TVC = \frac{\Delta}{\Delta t} TVR$$

3.2.2 Operational Considerations

This performance measure is more indicative of mission success in that it is a clear measure of how more or less secure the assigned cyber terrain is, after a period of time has surpassed. Ideally, a cyber team assigned to secure a network of computer systems will cause the terrain vulnerability to decrease, which would be apparent, visually, by graphing and displaying terrain vulnerability change for the duration of the cyber operation. This measure, like terrain vulnerability rate is already regularly used in both military and industry cyber security operations centers and information technology offices. Systems are normally set to alert when a vulnerability rate is detected to change above an abnormal threshold, which essentially means the terrain vulnerability measure has increased too quickly or above a threshold value.

3.3 Terrain Compromise Rate

Terrain compromise rate represents the rate of compromised systems present on the network. This measure is one of the most direct measures of cyber team success, as preventing systems from being compromised is the primary goal. Reducing terrain vulnerability rate reduces the likelihood that terrain might become compromised, but ultimately system compromise is what the team is aiming to prevent. In the Cyber-FIT model, terrain compromise rate is computed by dividing number of terrain agents in a compromised state by total number of terrain agents at any given time in the simulation.

3.3.1 Computation

Define T as the set of all mission terrain agents as a subset of all agents A

$$T \in A$$

Define T_c as the subset set of all T that are in a compromised state

$$T_c \in T$$

Define terrain compromise rate TCR calculated by dividing the absolute value of the compromised set by the absolute value of the full set

$$TCR = \frac{|T_c|}{|T|}$$

3.3.2 Operational Considerations

The ideal state for any cyber team focused on securing an operational network is to have zero compromised systems. However, over a long enough time period some systems will inevitably become compromised, even if through non-malicious means. A system that is simply "down" due to outdated software, hardware failure, user error, system interruption, power issues, etc., will still likely be considered compromised, at least initially from an incident response perspective. With a large enough network, considering compromised systems that have become inoperable for unknown reasons, a compromise rate above zero is inevitable. This is another measure regularly known to real world operation centers at the current time. The state of technology already allows for the tracking, reporting, and analysis of this measure. Security information event management (SIEM) solutions are widely used in military and industry organizations, tracking system responses from health checks. Unresponsive systems are identified and alerts are sent to analysts. Tracking this measure over time is already built into SIEM capabilities.

3.4 Terrain Compromise Rate Change

Terrain compromise rate change builds upon the previous section and represents how terrain compromise rate is changing over time. In the Cyber-FIT model this means a curve is fit plotting terrain compromise rate over ticks and taking the derivative at every tick.

3.4.1 Computation

As previously defined, terrain compromise rate, TCR, can be measured over time

Therefore, define terrain compromise rate change $TCRC$ calculated by

$$TCRC = \frac{\Delta}{\Delta t} TCR$$

3.4.2 Operational Considerations

This measure is similar to terrain vulnerability change as it is very well understood as an indicator of a successful military cyber mission, or period of time in an industry cyber security operations center. Clearly, if the compromise rate decreases over time, the teams are performing well and the organization has a more secure posture. This measure is different than terrain vulnerability change in that it is much more challenging to measure. This is for the simple reason that SIEMs are much better at defining specific vulnerabilities, due to the industry-wide work that goes into vulnerability identification. Compromises are more difficult to define. However, if an organization assumes some noise will follow the compromise system signal, then there should be a pattern and moderate regularity to the noise. For example, if some number of systems per year appear down, due to a hardware failure, then that network behavior should fall into a steady state. The important consideration for actually measuring terrain compromise rate change is to keep track of all down systems over time, and visually manage. Performance dashboards tracking the terrain compromise rate change historically would be vital in order to know if the current state is better or worse.

3.5 Mission Compromise Time

Compromise time is a measure of how long computer systems are compromised before the cyber team can restore them. In the Cyber-FIT model, this means that a terrain agent has changed state to compromised due to a successful attack by an attacker agent. The time from state changing to compromised, until a defender agent becomes aware of the compromise and then restores the terrain agent to normal, is compromise time for that particular terrain agent. The total time amongst all terrain agents in a compromise state is compromise time for a given campaign simulation.

3.5.1 Computation

Define T as the set of all mission terrain agents as a subset of all agents A

$$T \in A$$

For each mission terrain agent T_i, define c_i as the cumulative compromise time

Define mission compromise time MCT as the sum of all cumulative compromise time over all mission terrain agents

$$MCT = \sum_i c_i$$

3.5.2 Operational Considerations

Compromise time is clearly an important measure of cyber team performance. The longer systems are compromised, the longer the attackers have to complete their own objectives. Usually these objectives include lateral movement within the network, exfiltrating data, causing damage to systems that result in other software or hardware controlled failures, etc. Therefore, a well performing cyber team should be able to first recognize when systems are compromised, and then restore those systems in a timely manner. This measure follows along with the previous sections' discussion. Determining when a compromise occurs is still very difficult, due to the advanced persistent threats present on real world systems all over the world. Similar to compromise rate change, if an organization is tracking down time for any systems, and visually graphing the metrics around that, they can begin to understand normal trends within their networks.

3.6 Time to Detect

Time to detect refers to the amount of time it takes for a cyber team to recognize a system has been compromised. In the Cyber-FIT model, this means a defender agent has interacted with a terrain agent to run a survey operation, and the terrain agent on end2 of the terrain-to-terrain interaction directed link has status of compromised. Time to react is the amount of time surpassed from terrain agent compromise until one of the defender agents of the cyber team reads the compromise information and adds it to the compromised terrain array variable.

3.6.1 Computation

Define T as the set of terrain agents as subset of all agents A

$$T \in A$$

Define C as the set of all compromises where compromise C_i occurs on T_j at time f_i

Define C_d as the set of all compromises that have been detected, where C_i was detected at time g_i

Therefore, average time to detect, TTD is calculated by subtracting compromise time from reaction time for all compromises and dividing by the number of successful restoral operations i

$$\overline{TTD} = \frac{\sum_{\{f_i, g_i\} \in C_d} g_i - f_i}{i}$$

3.6.2 Operational Considerations

Time to detect is a performance measure frequently referenced by subject matter experts and is specifically listed on the notional dashboard in the Defense Science Board Report calling for a performance measures dashboard [9]. The state of technology already allows for the tracking, reporting, and analysis of this measure. As stated previously, SIEM solutions are widely used in military and industry organizations. One of the primary purposes of SIEM systems is to alert on anomalous activity, especially compromised systems. This means there is a log, with a timestamp, of when a specific system became dysfunctional through malicious cyber activity. This is referred to as an incident. At this point an incident report is either automatically generated, or a cyber team member annotates one. The time between dysfunction and when the incident report is read and/or filed would provide the data for time to detect. This measure is something that is currently prioritized by cyber teams especially those of "cyber security center" type. Most corporations have a group of professionals that represent first line cyber defenders and are named something like “security operations center” or SOC. This team is a 24 hours/day, 365 days/year operation. The SIEMs they use are always on, and always monitoring. There is always someone on duty, or at least on call. People that have worked in this type of role will have stories about late night and vacation/holiday work sessions due to an operational security issue. Time to detect is a real measure that is vital for these types of teams.

3.7 Time to Restore

Time to restore refers to the amount of time it takes for a cyber team to restore compromised systems. In the Cyber-FIT model, this means a set of defender agents are running restoral operations on a compromised terrain agent that has status of compromised. Time to restore is the amount of time surpassed from when a particular terrain agent has had status change to compromised, until that terrain agent has status changed to uncompromised.

3.7.1 Computation

Define T as the set of terrain agents as subset of all agents A

$$T \in A$$

Define C as the set of all compromises where compromise C_i occurs on T_j at time f_i

Define C_r as the set of all compromises that have been resolved where c_i was resolved at time g_i

Therefore average time to restore is calculated by subtracting compromise time from restoral time, divided by total discoveries

$$\overline{\text{TTR}} = \frac{\sum_{\{f_i, g_i\} \in C_r} g_i - f_i}{i}$$

3.7.2 Operational Considerations

Time to restore is also a performance measure frequently referenced by subject matter experts and is specifically listed on the notional dashboard in the Defense Science Board Report calling for a performance measures dashboard [9]. The state of technology already allows for the tracking, reporting, and analysis of this measure. Like time to react, SIEM logging data can be used to quantify this measure for a cyber team. This measure would be simpler to compute than time to react because no human induced lag would be introduced. That is, the SIEM would detail the exact timestamps when the system went down, and was subsequently restored. Most cyber security centers use visual aids where systems that are down are clearly displayed. Time to restore is visually apparent through these types of SIEM visual management systems. Carrying from the time to react measure, in a real-world environment a security center type of team would likely be more tuned to time to react, while the team is assigned to fixing the problem (incident response team) is more focused on time to restore. Put simply, one sub-team is reacting while another sub-team is restoring. We see this exact attempt at team segmentation in how the original U.S. military cyber teams were constructed in doctrine. Each team was originally made up of 39 team members segmented into five different squads: mission protect, cyber readiness, cyber support, discovery and counter-infiltration, and cyber threat emulation [10]. Over time the makeup of military cyber teams has evolved but the concept of sub-segmentation remains.

3.8 Time to Survey

Time to survey refers to the amount of time it takes for a cyber team to complete a survey mission where they need to gain a full understanding of the architecture, dependencies, and vulnerability level of a specified network(s) of computer systems related to an operational function. This measure is modeled after the recent utilization of military cyber teams being tasked with "survey missions". While not all the same, typically, this means a cyber team is tasked to provide cyber security status to various commanders and stakeholders as it relates to operational interests. There are many types of missions cyber teams are tasked with, all of which could potentially be modeled in various forms using Cyber-FIT. Some measures, like this one, are considered "mission dependent measures", that is performance is partially defined by the mission. In the Cyber-FIT model, this means mission parameters are loaded and then terrain agents generate vulnerabilities over a pre-specified amount of time. Upon completion of that time, a team(s) of defender agents run survey operations until the mission parameters have been satisfied. Time to survey is the amount of time surpassed from the time the team of defender agents began survey operations until the survey mission parameters have been achieved.

3.8.1 Computation

Define T as the set of terrain agents as subset of all agents A

$$T \in A$$

Define mission terrain, MT as the set of terrain agents assigned to the cyber team as subset of T

$$MT \in T$$

Define surveyed terrain, ST as the set of all terrain agents that the cyber team has surveyed as a subset of MT

$$ST \in MT$$

Define time survey mission began t_x, where

$$ST = null$$

Define time survey mission complete t_y, when

$$ST = MT$$

Therefore time to survey t_{survey} is computed by

$$t_{survey} = t_y - t_x$$

3.8.2 Operational Considerations

Time to survey is a mission dependent performance measure and more applicable to military cyber teams at this point. Military organizations frequently task cyber protection teams (CPT) to conduct "survey missions" [11]. Generally speaking, in the scope of the mission, the team will be expected to deploy system security tools and sensors into a specified network of cyber terrain responsible for DoD objectives. The cyber team will use their tools to scan the network systems and then create an assessment of the architectural, network, and host based vulnerability level and overall security posture. Missions like this are ideal from a team performance measure perspective because there are specific objectives to meet and time frames to operate within. Cyber teams tasked with a survey mission will normally plan the mission, execute the mission, and then provide a report. Therefore, in current operational environments, leadership already knows: how long the mission took, and how thorough the report is. In Cyber-FIT, the time to survey parameter is the time that the simulated team reports all required assets have been assessed. Further real world considerations can be expanded upon as cyber teams continue to evolve. For instance, time to survey could be considered based on whether the team is onsite versus offsite for the operation. The number of personnel needed and tools available could also help define time to complete survey mission performance measures. Although this measure is simplistic from a mathematical standpoint, its impact is substantial on military cyber resources. This measure is also impactful in the cyber security industry at large. Cyber teams worldwide are surveying systems nearly continuously with the SIEMs attached to their networks. Organization managers would be greatly informed by understanding how much time and funding is needed to complete effective surveys. If a cyber incident occurs, and causes damage, how effective was the survey operations that have been ongoing? This type of analysis will inform resource allocation and risk management decisions.

3.9 Time to Secure

Time to secure refers to the amount of time it takes for a cyber team to complete a secure mission where they need to reduce the overall vulnerability level of a specified network(s) of computer systems related to an operational function. In the Cyber-FIT model, this means that mission parameters are loaded and then terrain agents generate vulnerabilities over a pre-specified amount of time, simulating time where the cyber team is not assigned to that terrain. Upon completion of the that time, a team(s) of defender agents run survey operations, building a list of vulnerabilities per terrain agent, and then secure operations, removing the vulnerabilities. Time to secure is the amount of time that has surpassed from the time the team of defender agents began survey and secure operations until the overall terrain vulnerability rate has been reduced to a mission defined value.

3.9.1 Computation

Recall previously defined Terrain Vulnerability Rate, TVR

Define m as the mission defined acceptable TVR

Define t_x as time secure mission began

Define t_y as time when $TVR = m$

Therefore time to secure is computed by

$$t_{secure} = t_y - t_x$$

3.9.2 Operational Considerations

Time to secure in an operational perspective is very similar to time to survey. Being a mission dependent measure, the same considerations carry over from the time to survey section, except that the parameters of success would be more difficult to define for time to secure. Identifying systems that must be assessed is considerably simpler than defining how to specifically "secure" the systems. From a host view, taking one computer system at a time, one could define secure as either free from vulnerabilities, or near to free. But at a network level, security is much more complicated and difficult to define. A cyber team might find that the placement of certain devices has caused the network to have a routing type vulnerability that would be expensive to change. So tradeoffs are assessed such as an expensive architecture change versus an added layer of security to overcome the vulnerability. This means that human interpretation will come into play more in defining time to secure success parameters. This is further complicated by the fact that most networks are in place and have been for some time, sometimes a very long time. So, a cyber team securing a network is almost always having to contend with decisions about how to handle legacy infrastructure, in other words someone else's design decisions that are affecting the current security posture for the organization.

3.10 Cyber Situation Awareness

One of the most widely used definition of situation awareness was developed by Endsley [12] describing it as the "perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future". This means that the situation awareness is context dependent, and individualized. Taking this concept to the team level, an aggregate of the individual's awareness would need to be contextualized. This is difficult because different individuals play different roles and therefore have different needs in terms of what they need specific to the cyber domain [13]. As definitions can vary so can computations of cyber situation awareness. For this version of Cyber-FIT the three basic parts of Endsley's original definition of situation awareness will be used, which can be broken down as 1) knowledge of current state, 2) comprehension of that state, and 3) projection of that state. This allows for calculations to occur in each tick of the simulation on each of those three parts. Since defender agent's primary goal in their operations is to decrease the vulnerability level of the terrain agents, vulnerabilities will be the mechanism to score cyber situation awareness for a team performance measure. That is, each agent's knowledge of vulnerabilities present, their comprehension of the vulnerabilities, and their projection of which operation to select next.

3.10.1 Computation

Define cyber situation awareness CSA, as a function of knowledge K, comprehension C, and projection P related to vulnerabilities

$$CSA = f(K, C, P)$$

Each value: K, C, P will represent a fraction on the range $[0,1]$

Each value is weighted by a mission defined weighting factor either α, β, γ assigned to each of K, C, P where

$$\alpha + \beta + \gamma = 1$$

Therefore

$$CSA = \alpha K + \beta C + \gamma P$$

Define T as the set of all mission terrain agents as a subset of all agents A

$$T \in A$$

Cyber team knowledge level K represents the ratio of surveyed vulnerability level of T to the actual vulnerability level of T

Define V as the set of all vulnerabilities v_i that terrain agent T_J can have

$V_{T_j,i} \neq 0 \leftrightarrow T_j$ has vulnerability i

Then, total vulnerability level TVL of T is calculated by

$$TVL = \sum_{j,i} V_{T_{j,i}}$$

Define V_s as the set of vulnerabilities v_k that have been discovered and the cyber team has current awareness of as a subset of V

$$V_s \in V$$

Then, surveyed vulnerability level SVL of T is calculated by

$$SVL = \sum_{j,k} V_{s_{k,T_j}}$$

Therefore, K is calculated by

$$K = \frac{SVL}{TVL}$$

Next, comprehension, C is defined. Comprehension represents a defender agent's understanding and implications of the combined vulnerability status of terrain agents they have surveyed.

Define D as the set of defender agents on the cyber team as a subset of all agents A,

$$D \in A$$

Where each D_l has an agent-level comprehension H_m described in the defender agent methods section. Cyber team comprehension is the average of the defender agent H_m

$$C = \frac{\sum_m H_m}{m}$$

Next, projection, P is defined. In the Cyber-FIT model, a defender agent is either conducting an operation, or not, which represents confusion or uncertainty. So, the ratio of agents not doing anything versus those actively engaged in operations is a proxy for their projection at any given time in the simulation.

Each D_l has operational variable o set to the following value

$$o \ = \begin{cases} 0, & operating \\ 1, & not\ operating \end{cases}$$

Therefore, P is calculated by

$$P = \frac{\sum_l o_{d_l}}{l}$$

3.10.2 Operational Considerations

Cyber situation awareness is the most abstract of the performance measures. The basic structure of combining the three elements that Endsley defined (knowledge, comprehension, and projection) are meant to act as a building block that can be altered by researchers in order to experiment with different ways to compute the K, C, and P values. Also, the weighting factors could be vastly different based on mission needs, or leadership mandates. In real world environments, applications are already being developed to gauge all three measures. Impromptu checks from software can be built into systems that simultaneously gauge awareness and guide the person through recommended steps [14], the data of which can be aggregated as a knowledge measure. U.S. Army researchers [15] have built an experimental tool to track user key strokes and general operating activity data that can be fed to machine learning applications and classify expert activities, which can be used to quantify both comprehension level and projection as team members move from one operation to another. The purpose of situation awareness measures is not to gain a perfect understanding (it's impossible to quantify exactly how every team member is functioning, cognitively) but instead a general understanding. However, this measure is far more likely to apply to the emerging discipline of human-machine teaming. While we can't computationally measure the human brain to determine why the team member thinks in a certain way about cyber terrain vulnerabilities, we can for machines. As an example, a bot operating on a network to alert the cyber team via email, when a certain flag value changes, is computationally defined. That bot example, actually defines knowledge (what data to look for), comprehension (what value represents an anomaly), and projection (send email to alert others because we need to take action).

3.11 Operational Efficiency

Operational efficiency refers to how well the team performs its operations in terms of resource utilization, and not wasting time. Generally speaking this means moving from task to task quickly and completing tasks quickly. Efficiency can be difficult to define because it's often difficult to prescribe specifically, in mathematical terms, what the ultimate output of the team is. Further, once the outputs are defined, it's very difficult to identify and then measure the input variables that may moderate the output. Sivasubramaniam et al reviewed efficiency measurement literature and analyzed which variables had the most effect on efficiency ratings in new product development environments. This research was able to identify nine distinct yet common independent variables that effect team efficiency. This type of analysis tracks closest to a cyber team efficiency measure due to the "input, process, output" (IPO) nature of the work environment [16]. In the Cyber-FIT model, tracking the timing of defender agent operation completions will be the mechanism to measure efficiency. That is, when each defender agent selects a new operation, that operation has both a severity level and time to complete requirement. The severity level is based on the mission assurance category (MAC) levels regularly used by Department of Defense leadership as a way to prioritize system acquisition and protection [17]. The higher the mission assurance category, the higher the importance of that particular system. MAC levels are one, two, and three, and so those three values will be used in the computation of operational efficiency.

3.11.1 Computation

Define operational efficiency, OE as a function of operational time parameter, completion time, and operation severity.

Define D as the set of defender agents on the cyber team as a subset of all agents A

$$D \in A$$

Each D_i has attempted a set of operations, O where each O_j has a severity s equal to the mission assurance category of the operation, time to complete requirement parameter p, and time completed c.

Therefore, each d_i has an efficiency rating, e_i where

$$e = \sum_{o_{j},s=1} \frac{p}{c} + 2\left(\sum_{o_{j},s=2} \frac{p}{c}\right) + 3\left(\sum_{o_{j},s=3} \frac{p}{c}\right)$$

As shown, the severity level of the operation weights the component calculation of the efficiency rating between one and three. Finally, team level operational efficiency would be the average of all individual agent efficiencies expressed as

$$OE = \frac{\sum_i e}{i}$$

3.11.2 Operational Considerations

Efficiency is a measure most cyber professionals can intuitively sense amongst their peers. This is clear by discussing team member performance where in most circumstances, it is well known who “gets stuff done”. In nearly any informational workplace like software development and cyber operations, those that are efficient are sought after by managers to work on teams where they have a vested interest. Moving from task to task without wasting time, or becoming distracted is a key skill for cyber productivity. There is also an art to being able to troubleshoot individually, without interrupting other team members. The aggregate of all of these decisions, skills, and abilities interact and manifest within cyber operational behavior. The current Cyber-FIT model is counting operational timing and severity, partially because it is modeling what is possible to measure in current real world environments. Most cyber teams will use some type of task management and/or incident response tracking system. These systems typically show who has taken ownership of a task (self-selecting or management assigned) and how long it took that individual to complete the task. Also, as activities occur related to the task, the system is updated with timestamps. For example, an individual is assigned a task to investigate a faulty system. The individual might upload a memory dump, then make comments about the incident, then upload an assessment report, then troubleshoot, and then restore the system. Each of these actions is saved and a picture of the incident from task assignment to resolution can be made clearly visible. This means that over time, a trend analysis can be completed to learn how well different individuals do on different types of tasks, what task categories are the most difficult, etc. Is the team getting faster? Are less team members needed per task? Is the team becoming more efficient?

3.12 Cyber Mission Capability Rate

Cyber mission capability rate represents how functional the cyberspace systems are to kinetic mission forces that depend on them. At a high level, for the purpose of describing this capability, military forces deployed to a conflict could be categorized into two groups: kinetic and cyber. The kinetic forces are conducting missions that are not cyberspace specific, but depend on cyberspace to complete their mission. The cyber forces are working only on cyberspace systems in order to enable the kinetic forces. This means the primary purpose of the cyberspace terrain (computer systems) is to provide information to the kinetic forces, when requested. In the Cyber-FIT model, cyber mission capability rate is the ratio of information requests that friendly agents successfully read to the total information requests, weighted by criticality of the mission, and within an acceptable time to read parameter.

3.12.1 Computation

Define R as the set of all kinetic force information requests R_i each with time to read requirement parameter, p and criticality parameter c

Define F as the set of all kinetic force information fulfillments F_i with time to fulfill parameter t

Define cyber mission capability rate $CMCR$ calculated by

$$CMCR = \frac{F}{R}$$

Then, each F_i is computed according to the following function:

$$F_i = \begin{cases} 1 * c, & t \leq p \\ \dfrac{3p - t}{2p} * \dfrac{1}{c}, & p < t < 3p \\ 0, & t \geq 3p \end{cases}$$

Each R_i is set to the criticality parameter c, so

$$R_i = c$$

Then find total F and R values by

$$F = \sum_i F$$

$$R = \sum_i R$$

3.12.2 Operational Considerations

A long-standing measure of the readiness of the U.S. Air Force is the aircraft mission capability rate (MCR). This measure is tracked by all flying units and reported to Congress periodically for review, where the Government Accountability Office prepares reports associated with MCR [18]. Aircraft mission capability rate, generally speaking, is a measure of the percentage of time an aircraft is available to fly missions. So, if an aircraft is damaged, or not available due to a safety mishap, then it is not available to perform a kinetic mission, operated by aircraft mission personnel. This correlates perfectly to the concept of cyber mission capability rate (CMCR). Just like the personnel responsible for making the aircraft available to the flight crew, the purpose of a cyber team is to make cyber terrain information systems available to kinetic mission forces. The performance of that cyber terrain in fulfilling information requests is the primary measure kinetic forces will judge the cyber terrain they depend on. This cyber team performance measure is incredibly difficult to quantitatively measure in real world operations and would be extremely difficult to actually implement. Most users in any military or industry setting of corporate IT systems have an intuitive sense of how the computer systems are working based on responsiveness they are experiencing when using and accessing computer systems. In that sense, a survey could be sent out periodically to get a qualitative assessment of cyber mission capability rate.

3.13 Time to Compromise

Time to compromise represents the amount of time it takes from when the attacker starts an attack campaign until targeted machines are compromised. In the Cyber-FIT model, this is measured from the time an attacker begins phase one of an attack campaign until, during the exploitation phase, a terrain agent changes state from operating to compromised.

3.13.1 Computation

Define O as the set of attacker agents on the cyber team as a subset of all agents A

$$O \in A$$

Each O_i has a total attack campaign time parameter t and number of successful attacks s

Define time to compromise TC for each O_i computed by

$$TC = \frac{t}{s}$$

3.13.2 Operational Considerations

This cyber team performance measure is attacker based. That means, the details of the attacker's activities must be available in order to compute it. Agent-based modeling and simulation software can provide an excellent mechanism to experiment with phenomenon like this. Obviously, it would be of great interest to military and industrial organizations alike to have a full understanding of when, where, and how cyber adversaries begin attack campaigns, and when they become successful and on what systems. The Cyber-FIT model, and agent-based systems in general, can be an excellent tool to try out ideas on what might be possible, in a computational and programmatic manner. Running simulations can lead to theories about what exactly is going on with real world systems. Then, the empirical data an organization actually has can be compared to simulated data. This can either validate, at some level, the simulation software, or give clues as to why the simulation is not outputting data that matches empirical data.

3.14 Compromise Success Rate

Compromise success rate represents how successful attackers are in an attack campaign. It is measured by number of successful attacks and number of attack attempts. In the Cyber-FIT model, this is measured by continuously counting, each tick, how many total attacks have been attempted by each attacker agent and how many of those attacks have been successful.

3.14.1 Computation

Define O as the set of attacker agents on the cyber team as a subset of all agents A

$$O \in A$$

Each O_i has a total number of attacks x during a campaign, and number of successful attacks s

Define compromise success rate CSR for each O_i computed by

$$CSR = \frac{s}{x}$$

3.14.2 Operational Considerations

Much like the previous measure, this is also attacker based. In real world systems, knowledge about how many attacks have been attempted is very difficult to quantify. In situations where an attack has been successful, in nearly all cases, there is very little in the way of how many other attacks the adversary launched that weren't successful. The concept of "covering your tracks" means attackers tend to be as careful as possible about not giving away their position and removing evidence as they go. This is detailed as an exploitation technique by the MITRE tracking system [19]. Also, in most circumstances, the cyber teams and organizational leadership do not initiate forensic investigation of attacks until well after the attacks have been executed. Going back in time through logs and SIEM data is extremely time consuming and resource intensive. Like time to compromise performance measure, compromise success rate is virtually unknown to the organization on the receiving end of attacks.

3.15 Force-Force Interaction Network Node Total Degree Centrality

Force-force interaction network node centrality total degree is a measure meant to detect key leaders within the cyber team. This is done by examining a dynamic network of communications within the cyber team where each team member is a node. In the Cyber-FIT model this is done by creating directed links from defender agent to other defender agents in order to share vulnerability and compromise information. At every tick, some number of defender agents may communicate with others, in which case a directed link between them forms for a random time period in order to communicate. Throughout the simulation, Cyber-FIT stores this date as a file of links. Post-simulation processing software converts the link data to a time period based dynamic network file that is imported into ORA for dynamic network analysis. ORA processes the data, runs network science algorithms on it, and provides a report detailing selected network measures, in this case node total degree centrality.

3.15.1 Computation

Define A as the input network with n nodes (each representing a defender agent's ego) and maximum link value v, representing the number of messages sent to other defender agents

Total-degree centrality for each defender agent node i, TDC calculated by

$$TDC_i = \frac{\sum_j \left(A_{i,j} + A_{j,i}\right) - A_{i,i}}{2v(n-1)}$$

3.15.2 Operational Considerations

This measure is one of two network science based cyber team performance measures, along with terrain-terrain interaction network density. As a cyber team works together, communication networks emerge and dynamically change over time. Capturing the network data in periods of time (minutes, hours, days, etc.) and then analyzing how the measures change over time is called dynamic network analysis. Dynamic network analysis has been used to cover a wide array of scientific questions, especially those in the social sciences where human interactions are the core data being considered [20]. There are many network measures that could be considered, in order to gain insights about how a team is performing, so in this version of Cyber-FIT, the measures were limited to two. This is so the efficacy of network science measures could be considered in a simulation system with two of the most frequently used measures. This measure, node total-degree centrality, is a popular measure used in many studies to identify nodes most important for the flow of information [21]. This relates directly to cyber team performance because in many cases, the key leaders of an operation are not readily apparent based on the formal organizational structure. In an operational environment very similar data to the Cyber-FIT simulation data could be extracted and analyzed. The easiest way to do this would be to export the chat data from a team messaging server, especially when a cyber operation is being conducted by team members not physically in the same space.

3.16 Terrain-Terrain Interaction Network Density

Terrain-terrain interaction network density represents how much of the computer network is connected at any given time. In the Cyber-FIT model, this is simulated when directed links are created between terrain agents as a result of defender, friendly, or attacker agent behavior. Each time, a defender, friendly, or attacker agent creates a directed link to a terrain agent (which is a force-terrain interaction), one or more subsequent directed links are created between that terrain agent and other terrain agents. Then, at any given tick, a network of directed links where each end is of type terrain agent, can be extracted from the simulation.

3.16.1 Computation

Define A as the binary input network of terrain-terrain directed links with m rows and n columns

Density D is computed by

$$D = \frac{\sum(A)}{m \times n}$$

3.16.2 Operational Considerations

This measure is the other network science type measure included in this version of Cyber-FIT. Networks will usually have similar traffic patterns over time, based on patterns of life and usage by human involvement. This is why network density shows indications of a potentially effective measure to use for computer network visualization and monitoring [22]. A corporate network with normal business hours will have very different traffic patterns at 2:00 PM versus 2:00 AM. Network density as a cyber team performance measure is less an indicator of performance, and more a corollary to overall mission metrics. That is, there will not be a specific network density measure the team is aiming for. Instead, network density can be used as an indicator of normal operations, versus adversarial anomaly detection.

4 Cyber Team Performance Simulation

A cyber team performance simulation is conducted in order to output all performance measures for analysis and implications. The cyber team will deploy to a simulated conflict involving 500 computer systems operating to support 4 kinetic missions that need to be protected from several adversarial cyber forces of varying tiers.

4.1 Simulation Setup

When initializing Cyber-FIT, three files must be configured to setup key variables along with mission information. The first file, called missions supported, defines the friendly kinetic mission cyber terrain to be defended, including number of forces and associated cyber terrain systems. The second file, called, defenders, defines the cyber teams that will deploy in terms of squad, knowledge, skill, and experience. The third file, called attackers, defines the adversarial forces in terms of numbers and sophistication level. The following three tables display the pertinent contents of each file.

Mission ID	Unit	Friendly Forces	Networking Terrain	Server Terrain	Client Terrain
0	Base	0	10	20	30
1	Command Post	25	5	10	50
2	Fires	75	5	6	225
3	Logistics	50	3	5	75
4	Security	75	2	4	50

Table 9: Summary of Simulation Missions Supported File

Cyber Team ID	Squad	Knowledge	Skill	Experience
1	1	2	2	2
1	2	1	1	1
1	2	2	2	2
1	2	2	2	2
1	2	3	3	3
1	3	1	1	1
1	3	2	2	2
1	3	2	2	2
1	3	3	3	3

Table 10: Summary of Simulation Cyber Teams File

Adversary Type	**Tier**
State	3
Criminal	4
State	5

Table 11: Summary of Simulation Adversaries File

The simulation is run for 14,400 ticks, with each tick representing one simulated minute of time. This represents a ten-day simulation of continuous cyber conflict. Each run of the simulation takes approximately twenty minutes to complete on a Dell computer running Windows 10 with an Intel Xeon 2.7 GHz processor and 32 GB of RAM. Each simulation produces approximately 6 MB of team performance data. This simulation was run ten times.

4.2 Simulation Results

This section presents the sixteen cyber team performance measures resulting from the cyber conflict simulation.

4.2.1 Terrain Vulnerability Rate and Change Results

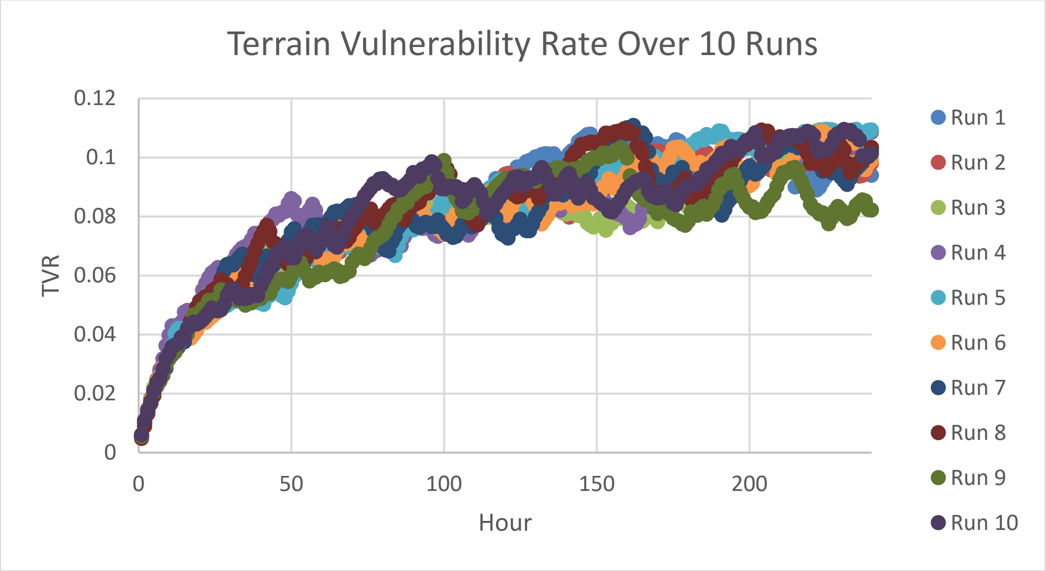

Figure 1: Terrain Vulnerability Rate

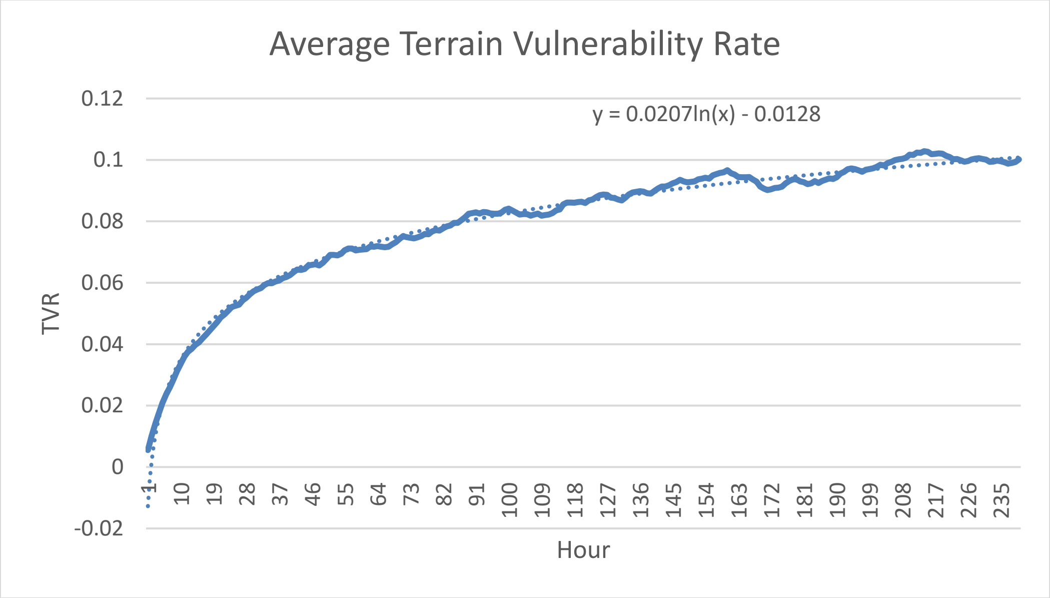

Figure 2: Average Terrain Vulnerability Rate

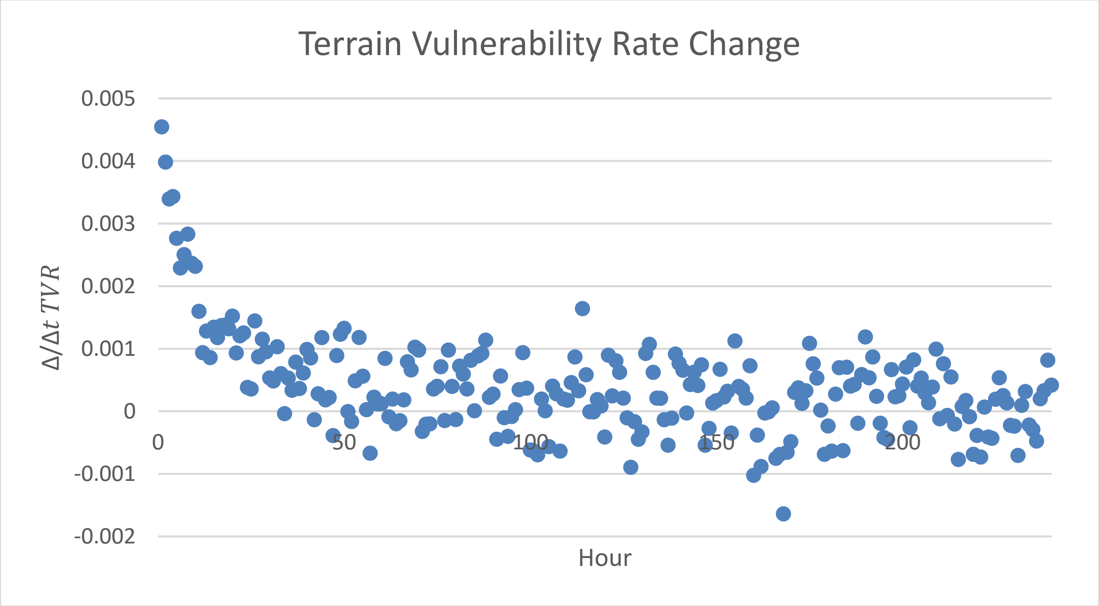

Figure 3: Average Terrain Vulnerability Rate Change

4.2.2 Terrain Vulnerability Rate and Change Discussion

As shown in Figures 1 and 2, the terrain vulnerability rate will increase rapidly in the early part of the simulation and then level off. A logarithmic best fit curve to the average terrain vulnerability rate, as shown in Figure 2, is $y = 0.0207\,ln(x) - 0.0128$, meaning the terrain vulnerability rate would not increase very much as the simulation continues under the configuration parameters. After tick 150, as shown in Figure 3, the change value hovers around zero. From a realism perspective, this is what is expected from most normal operations: the terrain vulnerability rate staying steady. In the last hour of the simulation, the lowest terrain vulnerability rate simulated is .082 and the highest is .109. The terrain vulnerability rate value realized in this simulation is approximately 0.1, which means a ten percent vulnerable state. This number is an abstraction and not meant to match real world operations perfectly. In real world situations, ten percent vulnerable is likely too high. However, in the simulation this means each simulated computer has approximately ten vulnerabilities at any given time, out of one hundred possible vulnerabilities. In real world operations, vulnerability state is well known through the use of network vulnerability management software where computers on the network report back about known vulnerabilities. Over time trend analysis can give a sense of how well the cyber team is managing vulnerabilities and therefore their own performance.

4.2.3 Terrain Compromise Rate, Change and Time Results

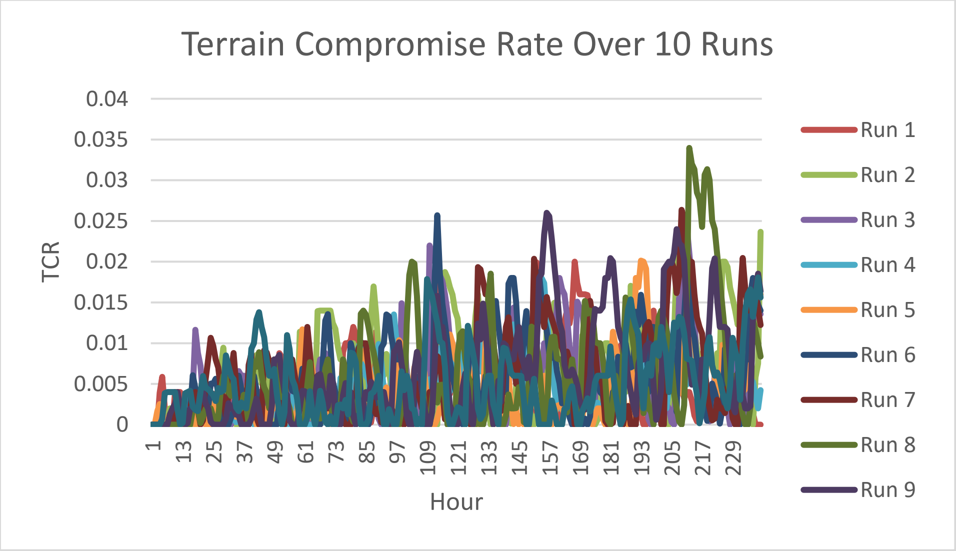

Figure 4: Terrain Compromise Rate

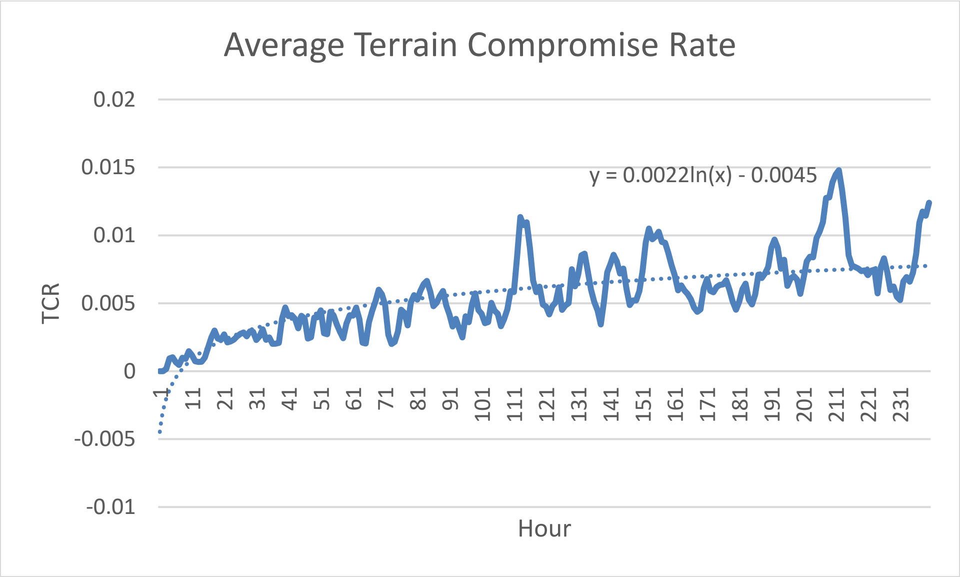

Figure 5: Average Terrain Compromise Rate

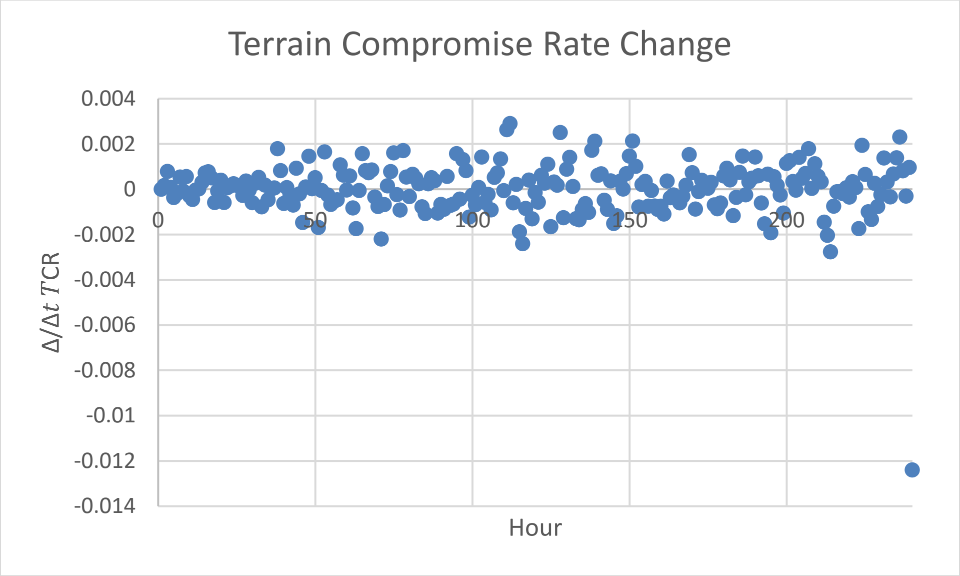

Figure 6: Terrain Compromise Rate Change

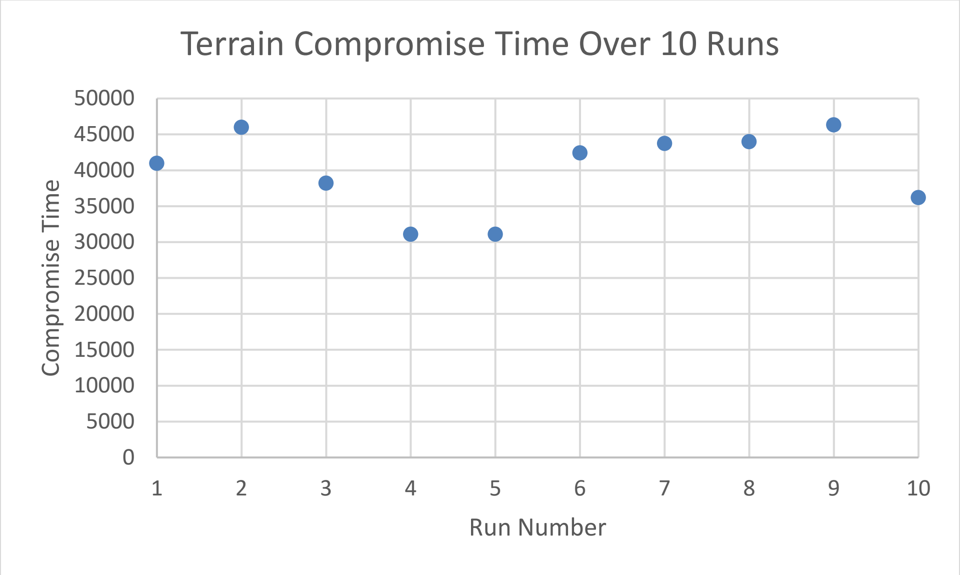

Figure 7: Terrain Compromise Time

4.2.4 Terrain Compromise Rate, Change, and Time Discussion

Terrain compromise rate, as shown in Figures 4 and 5, is more variable then terrain vulnerability rate. This is likely the case in real world operations as it is easier to detect vulnerabilities than compromised systems. Usually, a compromised system is unknown for some time, which is precisely what attackers are trying to do: compromise systems unbeknownst to the cyber team. Therefore, terrain compromise rate, time, and change will be harder to track in real world operations. The best fit logarithmic curve to average terrain compromise rate is $y = 0.0022ln(x) - 0.0045$. Like terrain vulnerability rate there is an initial increase, with the curve sloping up and then a leveling off. At that point, the attacking cyber team and defending cyber team are in a protracted battle where each side is fairly evenly matched. The attackers are able to exploit some systems, and the defenders are able to eventually identify and restore the compromised systems, which accounts for the variability in the curve. This is clearly shown in Figure 6, which is terrain compromise rate change, where the values hover around zero the entire simulation. In real world operations, this value would be near zero, nearly all of the time. In terms of this simulation, the model was designed to ensure some attacks would be successful, or else there would be nothing to analyze. Also, one of the primary purposes of an agent-based model is to conduct what-if analysis. This means searching for the combination of parameters that does cause differences in terrain compromise rate, and other dependent variables. Also, this is a ten-day (240 hour) simulation. A more realistic way to simulate successful computer compromises would be over a much longer time horizon. Figure 7 shows total compromise time for each run of the simulation. This value represents how long each computer (in aggregate) was off-line and inaccessible over the course of the ten days. The minimum value simulated was 31,068 minutes and the maximum was 46,315. In real world operations, this value is usually better known. That is, when a machine goes offline, and stops checking into servers, there is a log entry for when this occurs. So, total downtime for machines can be tracked fairly precisely. The more difficult part is attributing downtime to malicious activity. Some machines can go offline for completely benign reasons ranging from operating system error, user locking a computer out, hardware problems, infrastructure work, etc. Typically, downtime is monitored very closely, as most computers on a network are there for a purpose and when on, are needed to do some type of job. In the Cyber-FIT model, all downtime is related to malicious activity, in the real world this would have to be decoupled.

4.2.5 Time to Detect and Time to Restore Results

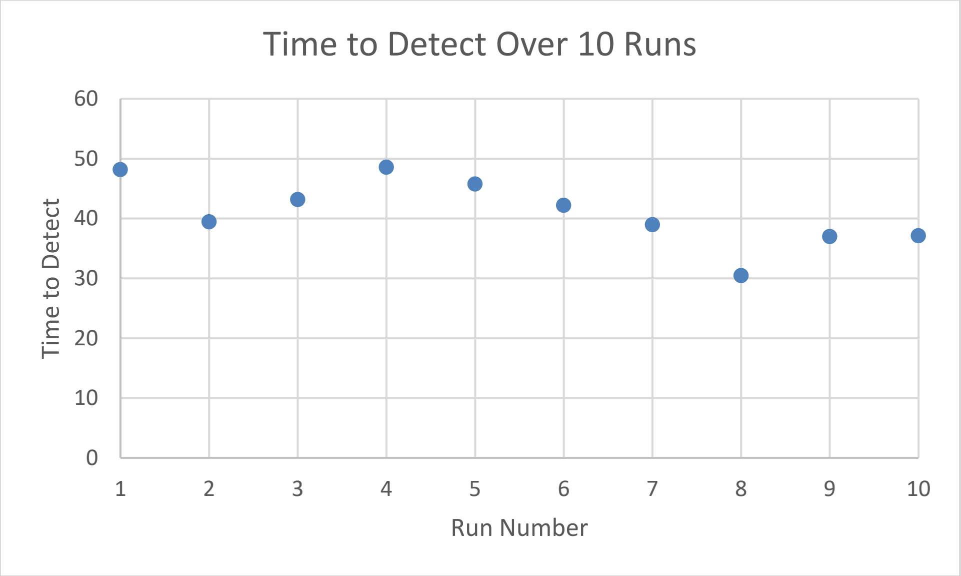

Figure 8: Time to Detect

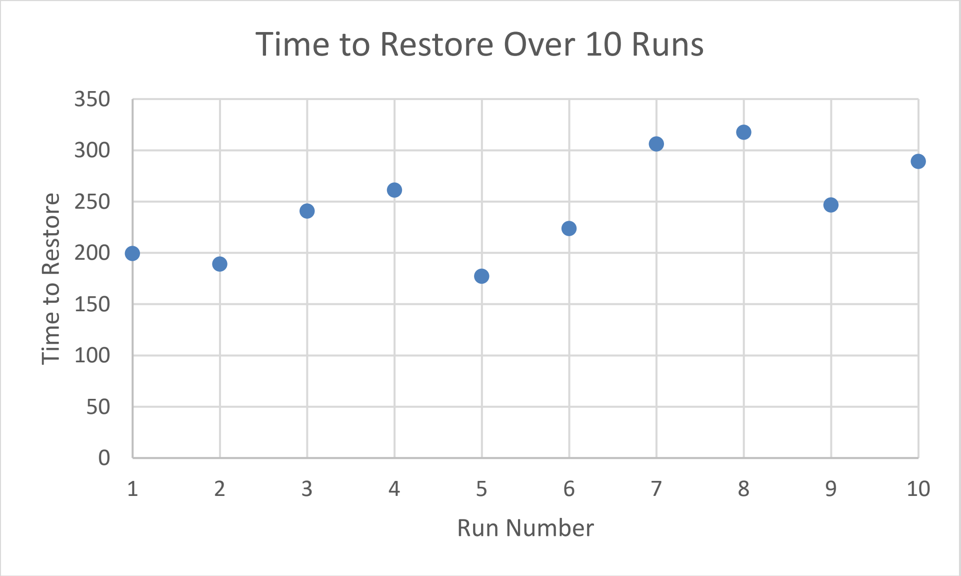

Figure 9: Time to Restore

4.2.6 Time to Detect and Time to Restore Discussion

Figure 8 shows the average time it took the cyber team to detect a machine was compromised over each run of the simulation. The minimum average time to detect was 30.44 minutes and the maximum average time to detect was 48.57. Figure 8 shows the average time it took the cyber team to restore compromised systems. The minimum time average time to restore was 177.16 minutes and maximum average time to restore was 317.56. Figure 9 shows the model results in higher variance for time to restore, which probably matches reality. In real world operations, these performance measures would both probably be high performing. Consider that the most devastating malicious compromises go unnoticed for long periods of time. This is why ransomware attacks are so popular (the cyber team cannot remove the malware) and exfiltration attacks (where large amounts of information is stolen) are so worrisome for corporations. In real world operations it is usually difficult to precisely determine how long a compromise went on unnoticed. This is largely due to the fact that it takes resources to investigate after the fact.

4.2.7 Cyber Situation Awareness Results

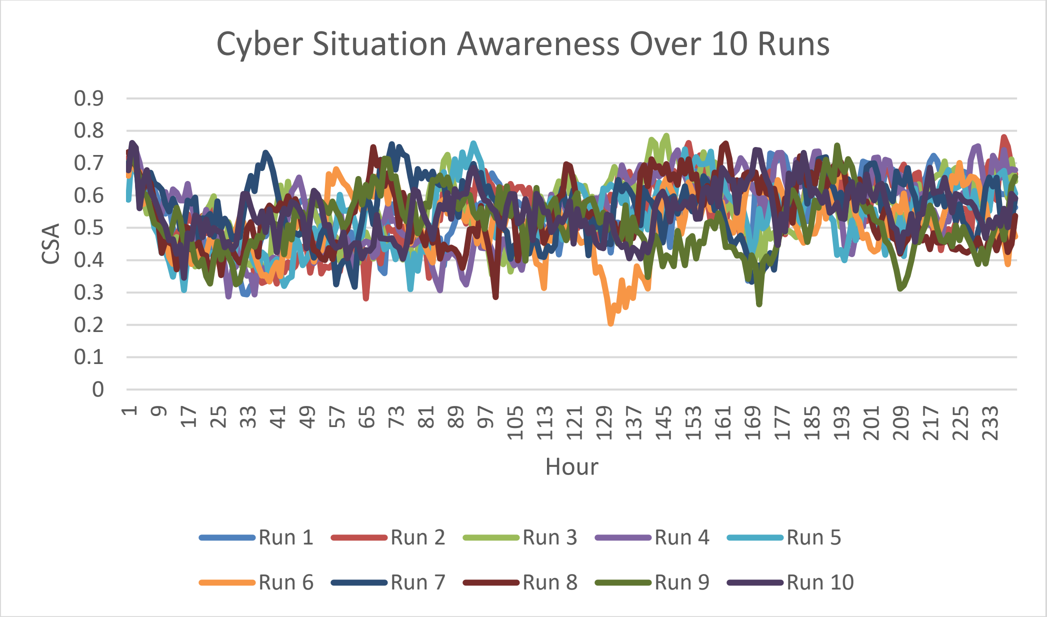

Figure 10: Cyber Situation Awareness

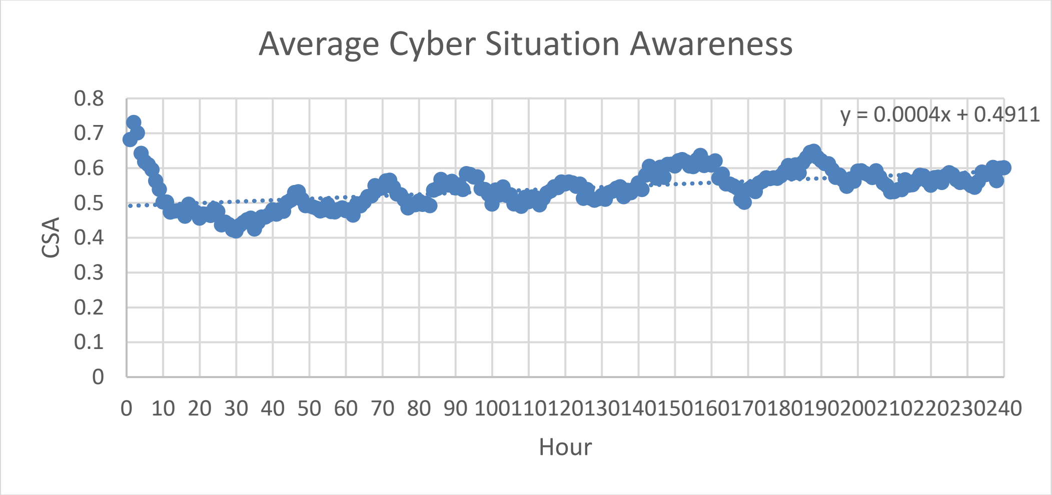

Figure 11: Average Cyber Situation Awareness

4.2.8 Cyber Situation Awareness Discussion

Figure 10 shows the model resulting in fairly high variance in cyber situation awareness over runs of the simulation. Figure 11 shows the average cyber situation awareness starting off higher and then decreasing (this is due to the computers becoming vulnerable and the cyber team needing to survey in order to find and become aware of the vulnerabilities). The values level off after the tenth hour. After this point, the minimum cyber situation awareness simulated is 0.204 and the maximum cyber situation awareness simulated is 0.785. The best fit linear curve to the average cyber situation awareness is $y = 0.004x + 0.4911$. The positive slope means that cyber situation awareness will continue to increase over time. This likely correlates well with real world operations due to teams sharing information over time and continuously communicating. This performance measure exists at this time in a theoretical sense only. That is, there are no real world teams actively tracking cyber situation awareness. It continues to be a concept understood, and sometimes discussed, and usually perceived, but not actively monitored.

4.2.9 Cyber Mission Capability Rate Results

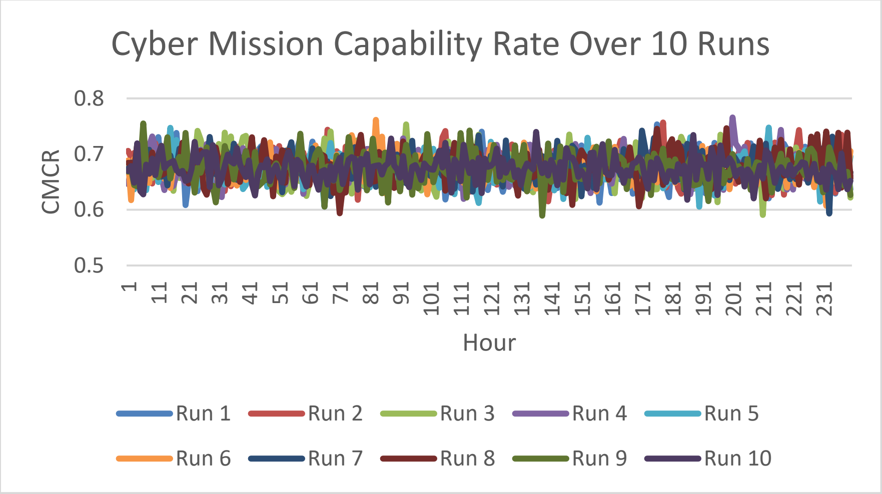

Figure 12: Cyber Mission Capability Rate

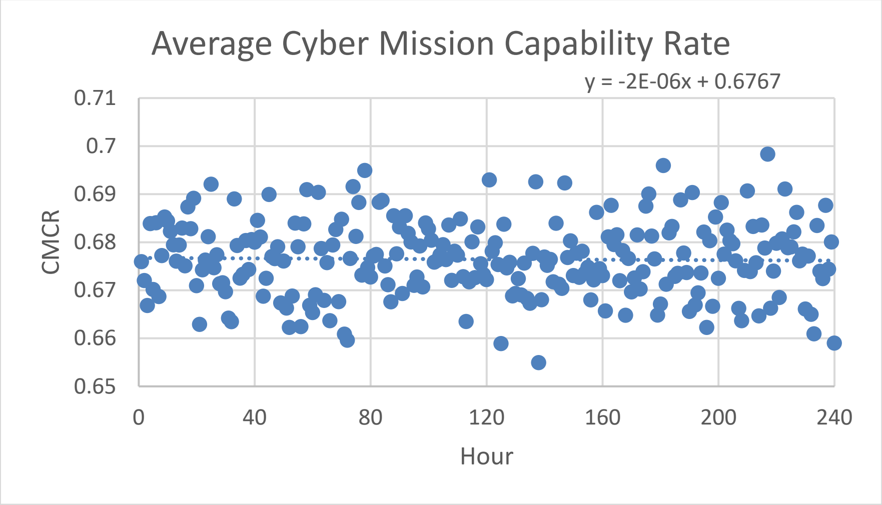

Figure 13: Average Cyber Mission Capability Rate

4.2.10 Cyber Mission Capability Rate Discussion

Figure 12 shows a highly variant but steady cyber mission capability rate over the ten runs of the simulation. Recall, cyber mission capability rate represents how well the computer network is providing the information requests needed by friendly forces to operate their own missions. This seems to be the most basic and important representation of what the purpose of the cyber team is: ensure the network moves information to those who need it. Like many cyber performance measures over time, this data is captured at hourly points throughout the ten-day simulation. The minimum cyber mission capability rate simulated was 0.589 and the maximum was 0.766. As shown in Figure 13, the slope of the best fit linear curve to the average cyber mission capability rate is very low and near zero, which means the team did not decrease or increase the cyber mission capability rate over the course of the simulated cyber conflict. This is expected due to the similar level of capabilities presented by both the attacking and defending cyber teams.

4.2.11 Cyber Operational Efficiency Results

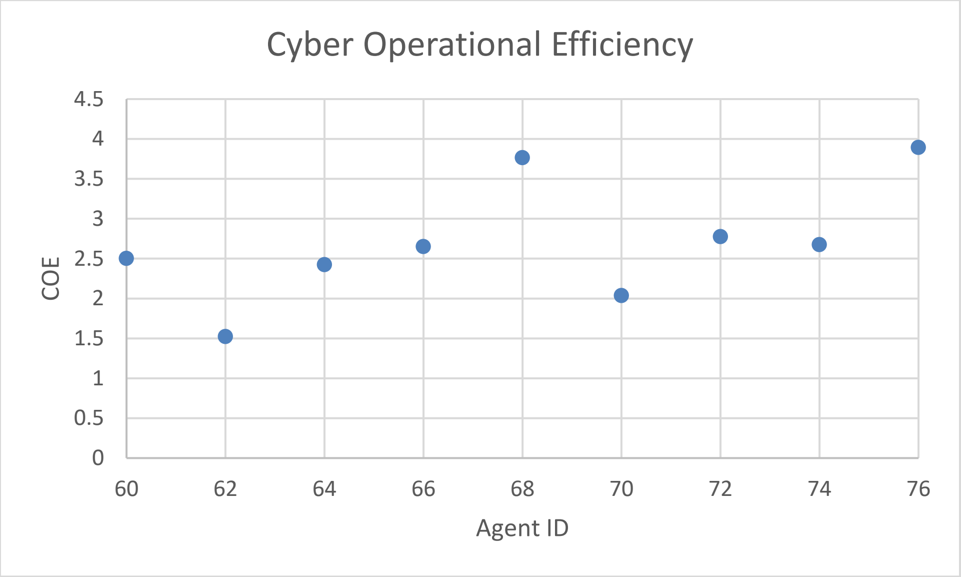

Figure 14: Agent Cyber Operational Efficiency

4.2.12 Cyber Operational Efficiency Discussion

The cyber operational efficiency performance measures simulated for each agent fell in line with what would be expected based on their knowledge, skill, and experience. In this simulation, the team was made up of eight hosts or network squad members, with knowledge, skill, experience (KSE) values of either all one, all two, or all three. These are agents 62 – 76. Agent 60 is the team lead, so while efficiency is tracked, it is not meaningful when comparing and contrasting with the other agents because the team lead tasks are abstracted into operations related to communication and management. Agents 62 – 76 are conducting survey and secure operations where they are actively searching for vulnerabilities, attempting to remove vulnerabilities, and attempting to restore compromised terrain when alerted. The resultant cyber operational efficiency measures, as shown in Figure 14, for each agent are lowest for KSE 1 (agents 62 and 70), middle for KSE 2 (agents 64, 66, 72, and 74), and highest for KSE 3 (agents 68 and 76). Taken altogether, as a team, the average cyber operational efficiency is 2.694. This value, by itself, is meaningless. Team cyber operational efficiency becomes meaningful when simulations containing different teams of varying size and KSE values are run, and then compared against one another.

4.2.13 Compromise Success Rate and Time to Compromise Results

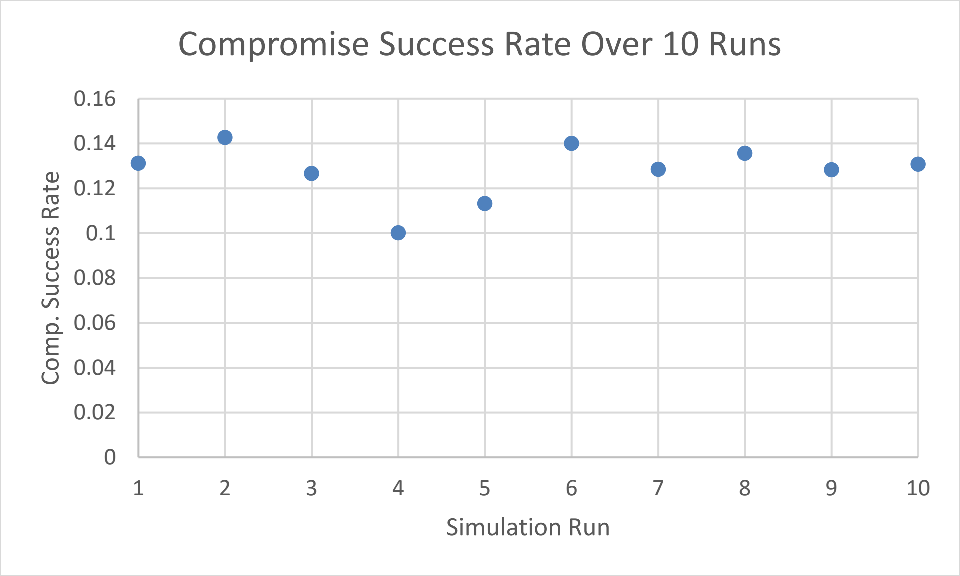

Figure 15: Attacker Agent Compromise Success Rate

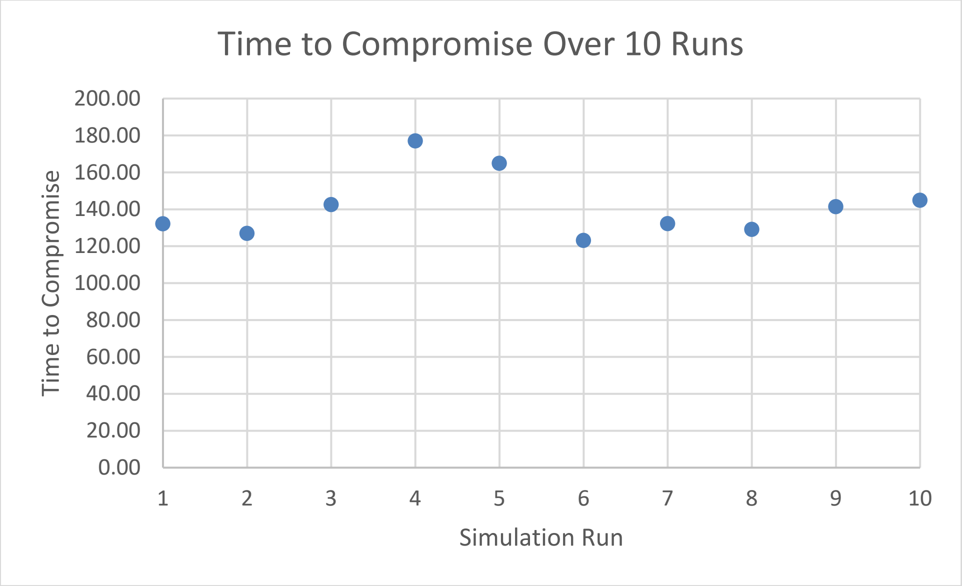

Figure 16: Attacker Time to Compromise

4.2.14 Compromise Success Rate and Time to Compromise Discussion

Compromise success rate and time to compromise are both measures of how well the attacking team is doing, and therefore the defending team (the focus of this model) is aiming to minimize the former and maximize the latter. Figure 15 shows the comprise success rate over the simulation runs. The minimum compromise success rate simulated was 0.100, the maximum was 0.143, and the average for all runs was 0.128. Figure 16 shows the time to compromise over the simulation runs. The minimum time to compromise was 123.099, the maximum was 177.024, and the average for all runs was 141.42. This data would be extremely difficult to compare with real world operations due to the limited information available at successful attacks. For compromise success rate, it would seem that the simulated values (approximately 0.128) are somewhat reasonable but likely higher than real world. Also, it would have to depend on the real world definition of success. In the Cyber-FIT model, the denominator includes all attempted attacks where the attacker agent attempts to deliver payload. In real world that could be expanded to starting with reconnaissance operations, or limited to only once payload is delivered.

4.2.15 Network Measures Results

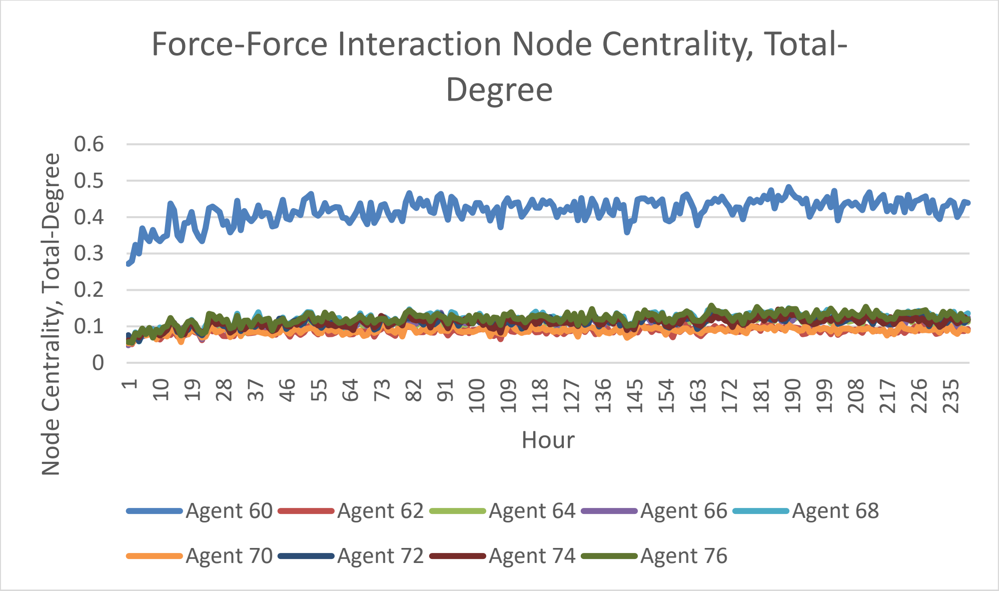

Figure 17: Force-Force Interaction Node Centrality, Total-Degree

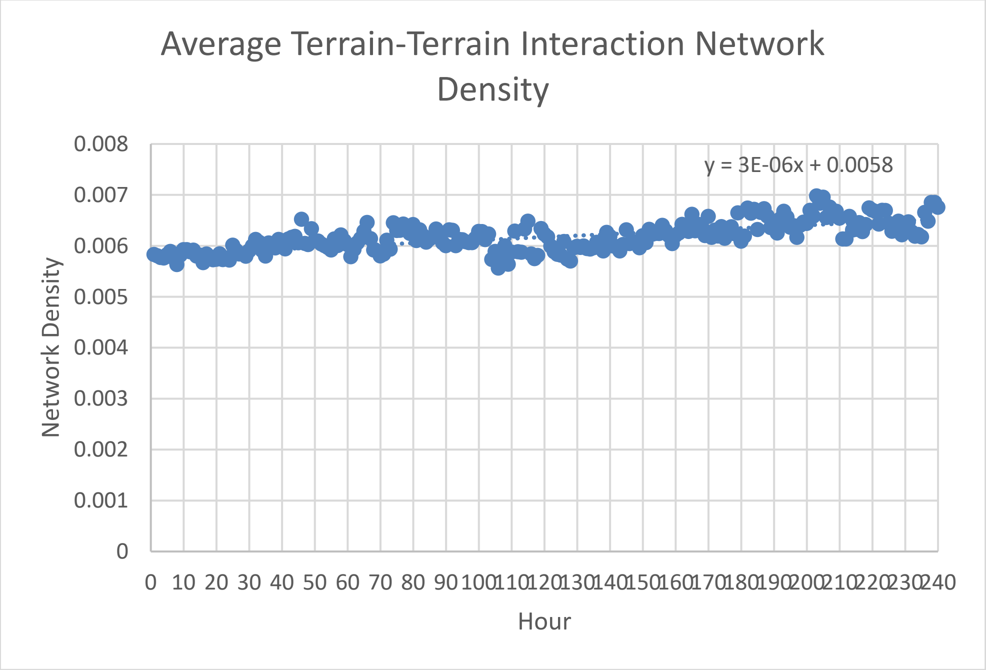

Figure 18: Terrain-Terrain Interaction Network Density

4.2.16 Network Measures Discussion

Two network measures were selected to collect and then use for dynamic network analysis. Using network measures for tracking and interpreting cyber team performance is less direct than the previous measures discussed. That is, there is no specific value a team would be aiming for in terms of network measures to gauge cyber team performance because not enough is known yet. Whereas, in the case of terrain vulnerability rate, the team is clearly working towards the lowest value possible, ideally zero. This means that in the case of network measures, trend analysis and over time correlation would be more appropriate. The dynamic network analysis for both measures was computed using ORA. Figure 17 shows a node level measure (centralization, total-degree) calculated on the collection of links at every hour. Clearly, agent 60, the team lead, has the highest node centralization total-degree during the entirety of the simulated conflict, which is expected. The other agents, on average, have similar values that vary within a small range throughout the simulation. Since the current version of Cyber-FIT doesn't have a wide range of behaviors, the agents will behave similarly. These two network measures are provided in this version as a proof of concept, which is shown to work at a basic level. Figure 18 shows the terrain-terrain interaction network density dynamic network analysis. The best fit linear curve to the average terrain-terrain interaction network density is shown with a slope of 0.000003. There is a very small increase over time, likely due to the slight increase in the number of vulnerabilities and compromises throughout the simulation, causing more activity amongst the team (which increases interactions amongst terrain being used for surveying and securing operations). Frequently, network visualization is coupled with network analysis to get a better sense of what is occurring. Figures 19 and 20 show a network picture of a randomly selected hour of the dynamic network analysis, produced by ORA.

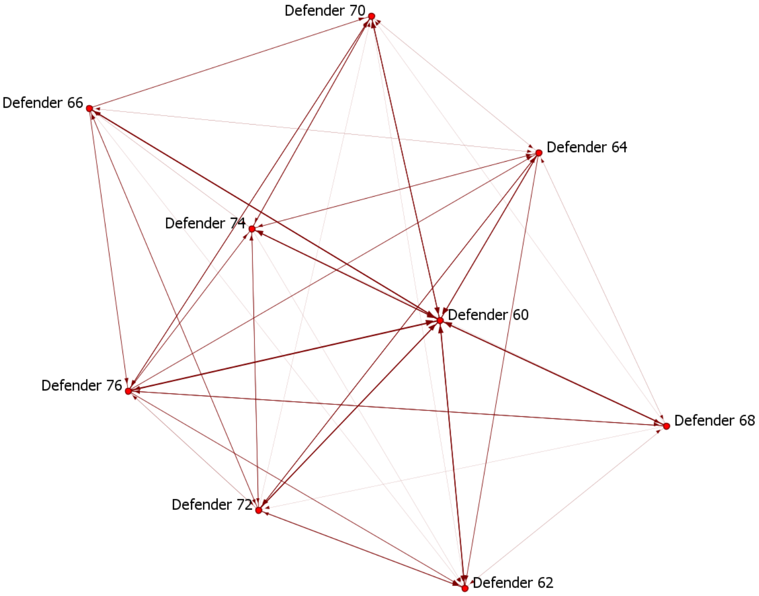

Figure 19: One hour visualization of Agent-Agent Link Network Node Centrality, Total-Degree

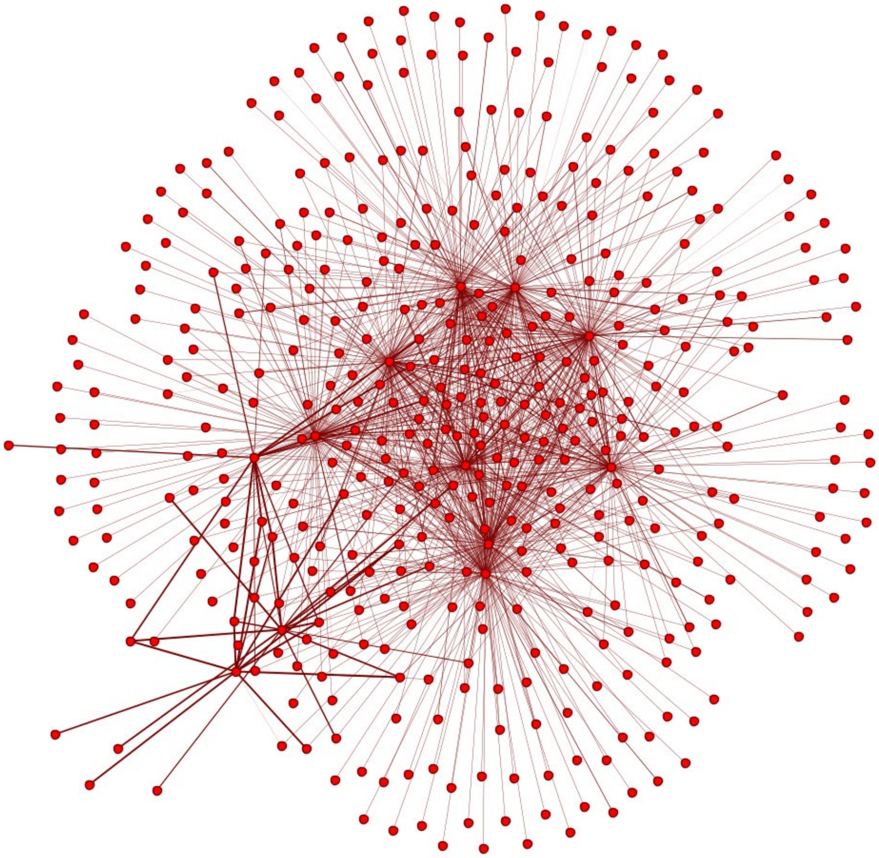

Figure 20: One hour network visualization of Terrain-Terrain Network Density

4.2.17 Mission Defined Measures Results

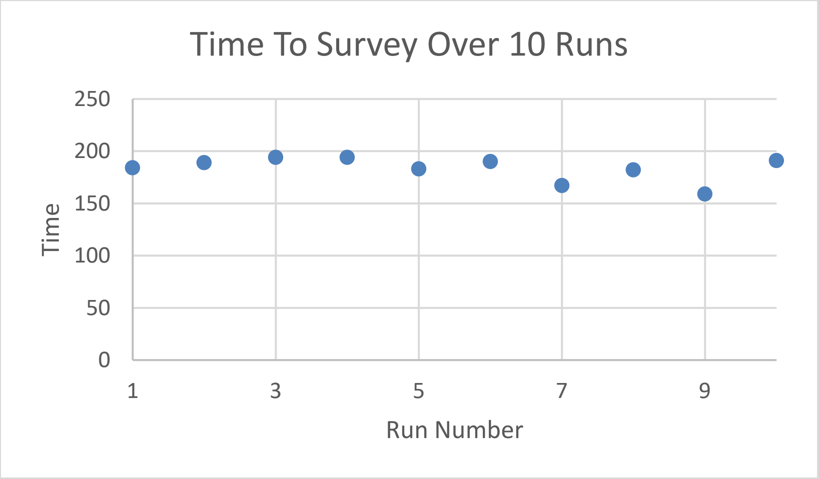

Figure 21: Time to Survey

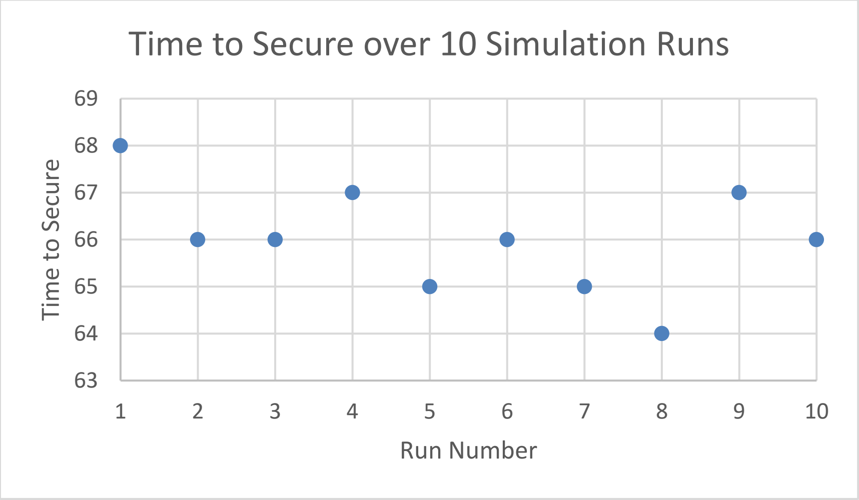

Figure 22: Time to Secure

4.2.18 Mission Defined Measures Discussion

As shown in Figure 21, the ten simulated missions resulted in the cyber team completing its survey mission between 159 minutes and 194 minutes. Figure 22 shows the ten simulated missions resulted in the team completing its secure mission between 64 and 68 minutes. Both of these missions were simulated with a two-day pre-deployment time, and then an eight-day mission with no attacker agents present (all other simulation setup variables were the same). This resulted in both measures having very little variance as can be seen in both figures. Since these two measures are mission-defined they can be set to include more complexity, which would result in more interesting results. For instance, the cyber team could secure terrain for a specific amount of time, achieving a very low terrain vulnerability rate, before attacker agents begin their operations, to see how low the team can keep the terrain vulnerability rate, with an active opposing force. Similarly, time to survey could be altered to include only the highest tier vulnerabilities (most severe and concerning) or only systems supporting the most important missions. This is exactly why leadership defines mission parameters in real world operations, because it is situational to what is occurring at what priority.

4.3 Team Performance Dashboard

A key motivation to this software simulation framework is helping move the state of the art in the direction of a comprehensive view of cyber team performance. Cyber-FIT Version 4 generates data in the form of comma separated value files reporting agent level data. These data are then post-processed and plotted into charts that were displayed in the previous section. Collectively, these charts can serve as a prototype of cyber team performance dashboard. The previously discussed Defense Science Board report [9], displayed a notional consideration of what a system performance dashboard should look like. At the time, none of those measures were formally defined. Cyber-FIT Version 4 defines the measures of performance, embedded in software, and then programmatically simulates and computes them. The Defense Science Board notional dashboard shown in Figure 23 can be compared and contrasted with the dashboard provided by Cyber-FIT shown in Figure 24.

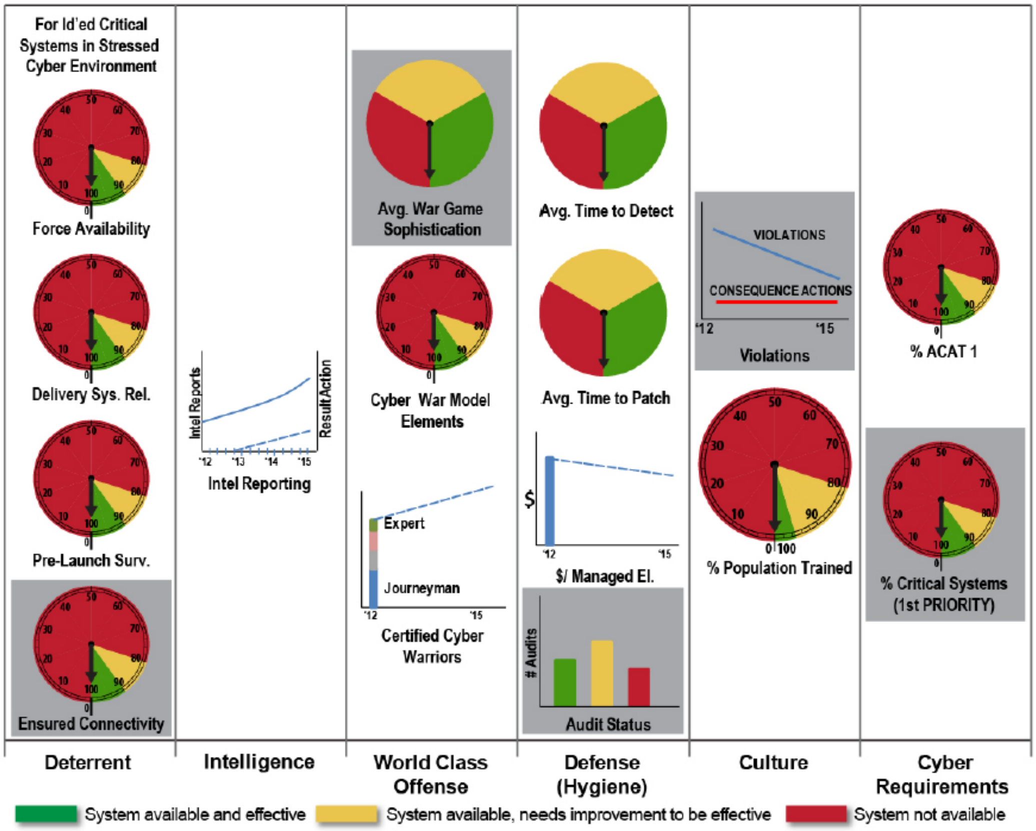

Figure 23: Defense Science Board Notional Cyber System Performance Dashboard

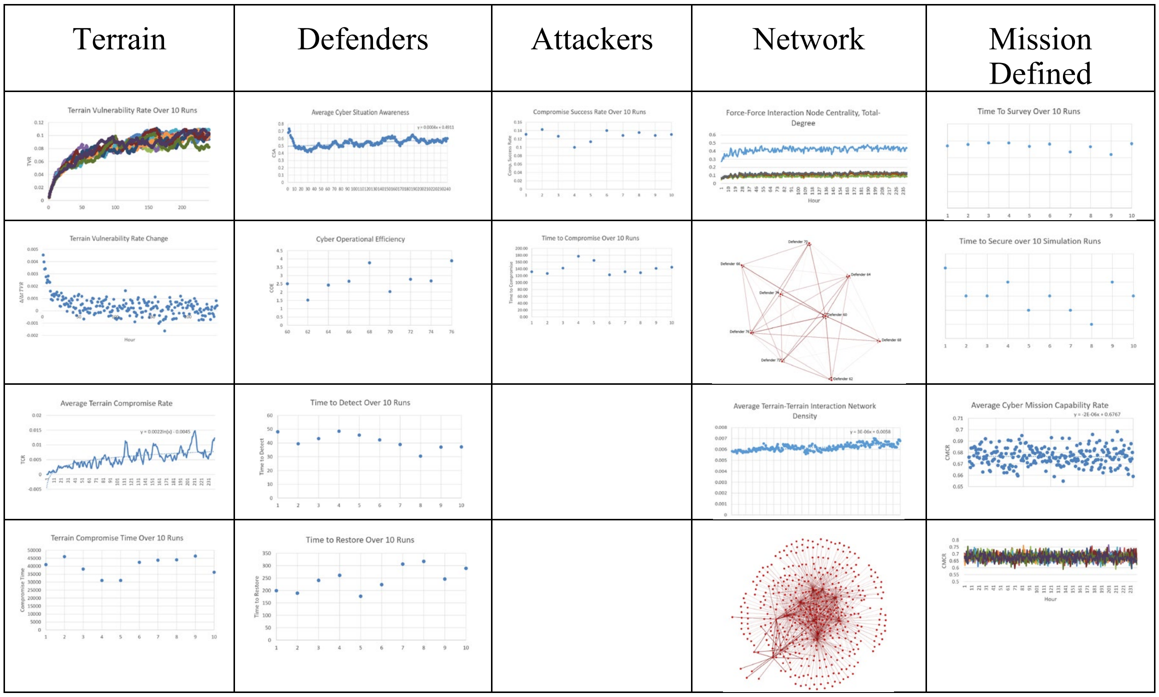

Figure 24: Cyber-FIT Simulated Team Performance Measures Prototype Dashboard

First, consider the categories of each. The notional dashboard breaks up the measures by: deterrent, intelligence, world class offense, defense, culture, and cyber requirements. Clearly, this dashboard is of a higher scope than the Cyber-FIT dashboard, which computes and displays team performance measures only. The design of the Cyber-FIT dashboard was based on the agent type aggregation of measures, which leads to fairly straight forward categories to group the measures: terrain, defenders, attackers, network and mission defined. Terrain measures are reporting on the cyber systems specifically – how vulnerable, available, and compromised they are. The defender category reflects the operational performance of the cyber forces tasked with defending the terrain, while the attacker category is the reverse of that. The network category provides network centric measures of interactions occurring on both the cyber systems and forces. These measures would have to be calibrated over time so change detection techniques can be utilized. Finally, the mission defined category would be set by leadership to track the measures specifically set by commanders. In comparing both dashboards, there are eight measures having a strong similarity between what was notionally proposed and what is being computationally modeled and simulated in Cyber-FIT: force availability, pre-launch survey, certified cyber warriors, average time to detect, average time to patch, audit status, percent ACAT 1, and percent critical systems. These are not perfect representations between dashboards, but a close enough approximation to fulfil some level of the vision proposed by the Defense Science Board.

5 Conclusion

In this technical report the Cyber-FIT Simulation Framework Version 4 is presented, described, and simulated with a realistic number of cyber forces and systems. This novel modeling tool is primarily used to advance the state of the science of computational and mathematical organization theory through focusing on cyber team performance. This includes how cyber team performance measures can be developed, computed, analyzed, and simulated. This model is novel in its agent-based approach where data is collected at both individual and team levels; and aggregated so over-time analysis can take place. This version lays the foundation for the most common behaviors of defensive cyber forces, offensive cyber forces and cyber terrain. This allows innumerable follow-on investigation to take place. With the core performance measures in place, and operating close to reality, new measures can be developed, new behaviors can be integrated, and new theory can be tested. Also, all of the control variables, along with stochastic variables can be experimented with to reason about which of the complexities inherent in cyber conflict are of most consequence. Appendix A describes the control variables that were set for the simulations run in Section 4 Cyber Team Performance Simulation. Each of these control variables took tuning, sometimes several simultaneously, in order to realize a response surface that mapped to the data that would be expected as a result of cyber conflict. Future work will continue that tuning by running virtual experiments where combinations of the variables are altered in a controlled way to understand how differently the outcome variables (selected performance measures) respond. Appendix B describes the Cyber-FIT model behaviors that are based on real world applications or existing frameworks. As Cyber-FIT was developed, many subject matter experts were consulted for how best to represent conceptual models of a more complex reality. Also, along the way, research efforts were identified that were applicable and then coded into how the agents would react to variable stimuli within the simulations. These behaviors can be experimented with, extended, and added to as the software matures.

Appendix A – Control Variables

Terrain Agent Class Control Variable	**Description**	**Values**
Vulnerability Growth Rate	Percent chance that a new vulnerability will appear on a terrain agent at any given tick	.01
Zero Day Vulnerability	Percent chance that a zero-day vulnerability will appear on a terrain agent at any given tick	.00001
Defender Agent Class Control Variables	**Description**	**Values**
Restoral Rate	Percent chance that a defender agent successfully restores a compromised system, at any given tick, based on skill of defender agent	{skill = 1: value = .001, skill = 2: value = .005, skill = 3: value = .01}
Restoral Effort	Percent of time that defender agent will devote to restoring terrain agents that are currently in a compromised state	.5
Stuck Time	Percent chance that a defender agent will get stuck any given tick that it's attempting to restore compromised terrain	{knwl = 1: value = .6, knwl = 2: value = .2, knwl = 3: value = 0}
Op Time	Time it takes for defender agent to complete operational task	{exp = 1: value = [3-150], exp = 2: value = [2-60] exp = 3: value = [1-30]}
Interaction Factor	Percent chance a defender agent interacts with other defender agents in order to complete operational task	{squad = 1: value = .75, squad ≠ 1: value = .25}
Reporting Factor	Percent chance a non-team lead defender agent reports information to the team lead rather than a squad mate	.5
Survey Delay	Percent chance that a defender agent is delayed for a tick due to technical issue surveying terrain	.1
Survey Span	Number of terrain agents a defender agent is able to scan per tick based on number of known compromised terrain agents	{skill = 1: values = [3-21], skill = 2: values = [6-42], skill = 3: values = [9 – 63]}

Attacker Agent Class Control Variables	**Description**	**Values**
Phase Zero Exit Time	Percent chance that the attacker agent leaves phase zero (pre-attack) of the cyber kill chain at any given tick when they are currently in phase zero	.02
Zero Day	Percent chance that an attacker agent that is tier 6 is able to develop a zero day during a tick in the weaponization phase	.00002

Total Attacks	Number of attacks an attacker agent has developed upon pre kill chain initialization	{tier = 1: value = 1,
		tier = 2: value = 2,
		tier = 3: value = 4,
		tier = 4: value = 8,
		tier = 5: value = 15,
		tier = 6: value = 16}

Phase Time	Amount of time attacker spends in each phase	Random value between [0 – 100]
Recon Connect	Percent chance that an attacker agent's computer (terrain agent) connects to contested cyber terrain at any tick during Recon Phase	.5
Payload Connect	Percent chance that an attacker agent's computer (terrain agent) connects to contested cyber terrain at any tick during Payload Delivery Phase to attempt to deliver attack	.5

Appendix B – Model Behaviors Based On Literature

Model Behavior/Ruleset	Source
Defender agent operation severity level	Mission Assurance Category (MAC) Levels [17]
Defender agent operation types	CISA Cyber Security NICE Framework [6]
Defender agent squad sub-categorization	Gaining Cyber Dominance Report [10]
Defender agent differentiation of knowledge, skill, and experience	US Air Force Broad Agency Announcement FA8650-20-S-6099
Attacker agent adversary tier level	Defense Science Board Report [9]
Attacker agent cyber kill chain behavior	Lockheed Martin Cyber Kill Chain [7]
Terrain agent vulnerability growth	Quantitative Vulnerability Assessment of System Software [23]
Terrain agent compromise actions	ATT&CK [8]

6 References

[1] G. B. Dobson and K. M. Carley, "Cyber-FIT: An Agent-Based Modelling Approach to Simulating Cyber Warfare," in *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, 2017.

[2] G. B. Dobson, A. Rege and K. M. Carley, "Informing active cyber defence with realistic adversarial behaviour," *Journal of Information Warfare,* vol. 17, no. 2, pp. 16-31, 2018.

[3] G. B. Dobson and K. M. Carley, "A computational model of cyber situational awareness," in *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*, 2018.

[4] M. J. North, N. T. Collier, J. Ozik, E. B. Tatara, C. M. Macal, M. Bragen and P. Sydelko, "Complex adaptive systems modeling with Repast Simphony," *Complex adaptive systems modeling,* vol. 1, no. 1, pp. 1-26, 2013.

[5] C. Pellerin, "Cyberspace is the new domain of war," American Forces Press Service, 2010.

[6] W. Newhouse, S. Keith, B. Scribner and G. Witte, "National initiative for cybersecurity education (NICE) cybersecurity workforce framework," National Institute for Standards and Technology, 2017.

[7] M. Cloppert, "Security Intelligence: Attacking the Cyber Kill Chain," 2009.

[8] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington and C. B. Thomas, "Mitre att&ck: Design and philosophy," MITRE, McLean, 2018.

[9] Defense Science Board, "Resilient Military Systems and the Advanced Cyber Threat," Resilient Military Systems and the Advanced Cyber Threat, 2013.

[10] G. Longo, "Gaining Cyber Dominance," Carnegie Mellon University Software Engineering Insititue, Pittsburgh, 2015.

[11] M. Pomerleau, "Air Force Squeezes New Cyber Defense Teams Out Of Its Communications Squadrons," 1 October 2021. [Online]. Available: https://www.c4isrnet.com/cyber/2021/10/01/air-force-squeezes-new-cyber-defense-teams-out-of-its-communications-squadrons/. [Accessed 6 January 2022].

[12] M. R. Endsley, "Design and evaluation for situation awareness enhancement," vol. 32, no. 2, pp. 97-101, 1988.

[13] M. R. Endsley and E. S. Connors, "Foundation and Challenges," in *Cyber Defense and Situational Awareness*, Springer, Cham, 2014, pp. 7 - 27.

[14] E. Peters and A. Kitsantas, "The effect of nature of science metacognitive prompts on science students' content and nature of science knowledge, metacognition, and self-regulatory efficacy," *School Science and Mathematics,* vol. 110, no. 8, pp. 382-396, 2010.

[15] A. Poylisher, M. Witkowski, V. D. Veksler, B. E. Hoffman and N. Buchler, "Recording Human Operator Data in Cyber Environments: User Activity Tracker (UAT)," Data and Analysis Center, Aberdeen Proving Ground MD, 2020.

[16] N. Sivasubramaniam, S. J. Liebowitz and C. L. Lackman, "Determinants of new product development team performance: A meta-analytic review," *Journal of Product Innovation Management,* vol. 29, no. 5, pp. 803-820, 2012.

[17] P. Campbell, "Information Assurance (IA) Implementation: A Retrospective," Sandia National Laboratories, Albequerque, NM, 2012.

[18] U.S. Government Accountability Office, "Weapon System Sustainment: Aircraft Mission Capable Rates Generally Did Not Meet Goals and Cost of Sustaining Selected Weapon Systems Varied Widely," U.S. GAO, Washington D.C., 2020.

[19] MITRE, "Indicator Removal on Host," MITRE, [Online]. Available: https://attack.mitre.org/techniques/T1070/. [Accessed 19 May 2021].

[20] M. K. Ahuja and K. M. Carley, "Network structure in virtual organizations," *Organization Science,* vol. 10, no. 6, pp. 741-757, 1999.

[21] M. K. Ahuja, D. F. Galletta and K. M. Carley, "Individual centrality and performance in virtual R&D groups: An empirical study," *Management Science,* vol. 49, no. 1, pp. 21-38, 2003.

[22] G. B. Dobson, T. J. Shimeall and K. M. Carley, "Towards Network Science Enhanced Cyber Situational Awareness," *International Journal on Cyber Situational Awareness,* vol. 1, no. 1, pp. 11-30, 2017.

[23] A. H. Omar and Y. K. Malaiya, "Quantitative vulnerability assessment of systems software," in *Annual Reliability and Maintainability Symposium*, 2005.