2022-03

TESTING DECEPTION WITH A COMMERCIAL TOOL SIMULATING CYBERSPACE

Drew, Sasha K.; Heinen, Charles W.

Monterey, CA; Naval Postgraduate School

http://hdl.handle.net/10945/69636

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

TESTING DECEPTION WITH A COMMERCIAL TOOL SIMULATING CYBERSPACE

by

Sasha K. Drew and Charles W. Heinen

March 2022

Thesis Advisor: Armon C. Barton
Co-Advisor: Neil C. Rowe

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** March 2022	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE TESTING DECEPTION WITH A COMMERCIAL TOOL SIMULATING CYBERSPACE	**5. FUNDING NUMBERS**
6. AUTHOR(S) Sasha K. Drew and Charles W. Heinen	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Marine Forces Cyberspace Command, Fort Meade, Maryland 20755	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.	
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

Deception methods have been applied to the traditional domains of war (air, land, sea, and space). In the newest domain of cyber, deception can be studied to see how it can be best used. Cyberspace operations are an essential warfighting domain within the Department of Defense (DOD). Many training exercises and courses have been developed to aid leadership with planning and to execute cyberspace effects that support operations. However, only a few simulations train cyber operators about how to respond to cyberspace threats. This work tested a commercial product from Soar Technologies (Soar Tech) that simulates conflict in cyberspace. The Cyberspace Course of Action Tool (CCAT) is a decision-support tool that evaluates defensive deception in a wargame simulating a local-area network being attacked. Results showed that defensive deception methods of decoys and bait could be effective in cyberspace. This could help military cyber defenses since their digital infrastructure is threatened daily with cyberattacks.

14. SUBJECT TERMS Artificial Intelligence, Cyber Course of Action Tool, defensive cyberspace operations, distributed denial of service, Department of Defense, empirical game-theoretic analysis, offensive cyberspace operations, program of record, Soar Technologies	**15. NUMBER OF PAGES** 59
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

TESTING DECEPTION WITH A COMMERCIAL TOOL SIMULATING CYBERSPACE

Sasha K. Drew
Chief Petty Officer, United States Navy
BS, American Military University, 2013

Charles W. Heinen
Chief Petty Officer, United States Navy
BS, University of Maryland University College, 2016
M, Colorado State University-Global Campus, 2018

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED CYBER OPERATIONS

from the

NAVAL POSTGRADUATE SCHOOL
March 2022

Approved by: Armon C. Barton
Advisor

Neil C. Rowe
Co-Advisor

Alex Bordetsky
Chair, Department of Information Sciences

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Deception methods have been applied to the traditional domains of war (air, land, sea, and space). In the newest domain of cyber, deception can be studied to see how it can be best used. Cyberspace operations are an essential warfighting domain within the Department of Defense (DOD). Many training exercises and courses have been developed to aid leadership with planning and to execute cyberspace effects that support operations. However, only a few simulations train cyber operators about how to respond to cyberspace threats. This work tested a commercial product from Soar Technologies (Soar Tech) that simulates conflict in cyberspace. The Cyberspace Course of Action Tool (CCAT) is a decision-support tool that evaluates defensive deception in a wargame simulating a local-area network being attacked. Results showed that defensive deception methods of decoys and bait could be effective in cyberspace. This could help military cyber defenses since their digital infrastructure is threatened daily with cyberattacks.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

AI	artificial intelligence
CCAT	Cyber Course of Action Tool
DDOS	distributed denial of service
DOD	Department of Defense
EGTA	empirical game-theoretic analysis
SoarTech	Soar Technologies

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First and foremost, I would like to thank my family and friends for the motivation, patience, and encouragement given to me during this time at NPS. Thanks for always being loving and available for me when I needed it the most. I especially appreciate the continued support throughout my naval career.

Thank you to my advisors, Dr. Rowe and Dr. Barton, for guidance and technical expertise throughout the thesis process. I would also like to thank Justin Green for planting the seed for this project and providing valuable feedback. A special thank-you to Robert “Bob” Bixler for your patience, hard work, and dedication in ensuring the CCAT tool was tested with a realistic scenario. Your efforts dynamically impacted the results of this research. Lastly, thank you, Charles Heinen, for your tenacity and humor throughout the MACO program.

—ITC Sasha K. Drew

I would like to thank my family for their support throughout this past year and for everything they have done to support me through this program. They provided me with the motivation and encouragement I needed to complete this degree and this thesis. Thank you for putting up with all the late nights and day classes! I would not have been able to complete this without my youngest son sitting with me and “helping” me with school in 95% of my classes.

Additionally, I would like to sincerely thank my advisors, Dr. Rowe and Dr. Barton, for their guidance, patience, and help in the completion of this thesis. I would also like to thank Mr. Robert Bixler from Soar Technologies for helping Sasha and me through every step of the process and dedicating the time to setting up our experiments. Thank you to GySgt Justin Green for bringing Sasha and me onto this project to continue his work.

—CTNC Charles W. Heinen

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Now that cyberspace is a warfare domain like air, land, sea, and space, military organizations are thinking about how defense and offense can apply there. Although cyberspace is evolving and changing with the influx of new Internet-connected devices daily, few Department of Defense or commercial wargame simulations cover cyberspace operations planning. A big challenge in defensive cyberspace operations is understanding, interpreting, and acting on the large amounts of data used in our networks. Deception has historically been an important tool in warfare for both offense and defense. In cyberspace, deception can waste an attacker's time with decoys, or can send malicious software to an attacker. The purpose of this research was to test simulated deceptions in cyberspace as defensive methods.

This research experimented with modeling cyberwar tactics in a wargame designed to prepare operators for defending cyberspace. We tested a product from Soar Technologies that can customize and fight realistic scenarios including deception. Soar Technologies was funded by the Defense Advanced Research Projects Agency to support research for military decision-making in the cyberspace domain; previous experiments with their wargame resulted in an NPS thesis (Green, 2020). The work described here is successor work to Green's thesis.

Deception within cyberspace can help protect defenders, enhancing the defender's rate of success and limiting the attacker's ability to disrupt activities. Deception technology could prevent a malicious user that found a way to infiltrate a network from doing any significant damage (Parks & Duggan, 2011). One tactic could be generating deceptive decoys that imitate legitimate technology assets throughout the infrastructure. In general, deception within cyberspace could add protection to defenders by enhancing the defender's rate of success while limiting the attacker's ability to disrupt activities. Deception methods can also use artificial intelligence and machine learning to ensure techniques are flexible and dynamic.

Deception can be tested in wargames. These can serve a broad range of purposes and can use military rules and probability theory in simulating conflict (Shang et al., 2019). Wargames allow policy makers and operators to test strategies and tactics without real-world penalties. Unexpected scenarios can occur in wargames just as in actual warfare; seeing and experiencing these surprises in a wargame prepares a player to respond efficiently in real life. The U.S. Department of Defense uses wargames extensively.

The experiments in this research involved runs of a wargame with random choices. This randomization was used for both offensive and defensive tactics by allowing players to explore possibilities better. For defenders, better tactics could strengthen cybersecurity programs. The work produced in this thesis designed a cyber wargame using a commercial tool. Chapter II discusses previous work on cyberspace deception including honeypots, game theory, artificial intelligence, and attack modeling. Chapter III describes our experimental methodology, including tested scenarios in collaboration with Soar technologies. Costs, benefits, offensive strategies, and defensive strategies will be measured for the attacker and defender for each scenario. Chapter IV summarizes the results from the scenarios. Chapter V discusses conclusions and recommendations for future work.

II. LITERATURE REVIEW

A. DEFENSE IN CYBERSPACE

The original Internet protocols were designed to treat all machines the same and to accept connection requests from anyone on the network. The tight-knit community of people who invented the Internet trusted one another, so they avoided security controls that would have increased the complexity of the protocols (Garzke & Lindsay, 2015). That same openness later facilitated distributed denial-of-service attacks that would flood servers with more connection requests than they could handle. Cyber criminals, and autonomous botnets under their control, regularly scan networks in search of vulnerable systems to co-opt (Trassare et al., 2013). Military and more sophisticated adversaries also scan and map networks as part of reconnaissance and intelligence gathering.

Cyber defense is becoming harder in cyberspace because of the increasing number of vulnerabilities and increasing sophistication of attacks using Internet connectivity. Automated tools can help with many defensive tasks such as testing software for bugs or confirming security settings (Gartzke & Lindsay, 2015). Attacks are increasingly automated, so defense must be automated as well. One useful form of automated response could be implementing cyberspace deception within a network (Anitha et al., 2016). Automated deception can allow defenders to conceal true-value targets while encouraging attackers to go after low-value targets. It provides the defenders more time to successfully identify attacks and figure out how to fight them, providing a strategic advantage for defenders over their attackers (Changwook-Park & Kim, 2019).

Penetration testing or “red teaming” can test defensive strategies. It can show network and system weaknesses, and it can identify vulnerable or exploitable systems that require remediation (Randhawa et al., 2018). It can be part of a wargame and can be done by an organization different from the targeted-network owner to provide an unbiased test. Penetration testing can be automated. An example is Trogdor (Randhawa et al., 2018), an automated system that uses a model and critical-node analysis. It compiles a visual picture of the defenders and vulnerable resources within the network.

B. DEFENSIVE CYBER DECEPTION

Traditional deception practices can apply to cyberspace (Rowe & Rothstein, 2004), although methods differ between cyber defense and offense (Rowe & Rrushi, 2016). Cyber-deception strategies for the defenders include misinformation and disinformation to sabotage the early stages of attack reconnaissance. They can also use camouflage and fakes. Camouflage can be hiding or concealing honeypots within the defender's network. One kind of fake is a honeypot, a decoy for cyberattackers designed to detect and study their tricks and types of attacks. It acts as a potential target and reports to defenders about attempts to access that system. Attackers wish to avoid honeypots, so effectively disguising one gives a defender an edge.

A deception in the form of fake documents can also help the defender throw attackers off the trail of useful information. False information can be planted in documents to waste the time of attackers, while alerting defenders about potential malicious activity (Rowe & Rrushi, 2016); fake network nodes can provide a similar effect. One study generated believable fake documents using a genetic algorithm (Karuna et al., 2021); a fake document can affect the time and effort for an attack to succeed. For measuring text comprehensibility, they used principles of psycholinguistics and reading comprehension: connectivity, dispersion, and sequentially. Another project created false code similar to real code but having no function (Park & Stolfo, 2012). Fake network nodes or devices can provide an effect similar to that of fake files (Rowe & Rrushi, 2016), but could give the defender more time to react to an attacker.

There are several ways to accomplish defensive deception. Second-order deception adds another layer of deception behind existing deception (Rowe & Rrushi, 2016). This type of deception can increase the overall cost that the attacker incurs because of the time spent working through the multiple layers of deception. Resource deception messages deceive the attacker into believing that what they need to perform their attack is unavailable, ultimately leading them to leave the network (Rowe & Rrushi, 2016).

Some work has tested deceptive response frameworks such as (Goh, 2007). This work categorizes the various types of responses used against the attackers and shows how intrusion deception fits in. A different project used technologies and tools from third-party vendors such as VMware, Snort, and honeypots in a sandbox environment open to attacks from the Internet (Chong & Koh, 2018). The results showed that attackers had some interesting reactions to deceptions such as a SSH brute force attack after several port scans in four days. It took the attacker approximately seven hundred attempts to successfully brute force a user account. Another project studied a deep-learning method for planning automated deceptive cyber defenses (Matthew, 2020).

Deceptive responses can be used in network defenses at the application layer by adapting to Web traffic. Red-teams consists of security professionals disguised as adversaries to prevail over cybersecurity controls. They have been using deception for phishing, fake documents, social engineering, and more attacks. One project examined the value of human interactions and teamwork in red-team deception using cognitive and behavioral testing (Bruggen et al., 2019). This research concluded that deceiving the red team might work better on an operational network by providing confusion through complexities and anomalies (Bruggen et al., 2019). Another project used "moving decoys" (Sun et al., 2019) to constrain attackers and forward the malicious commands to decoy servers, providing the attackers with credible results from their attacks.

A project implemented deception techniques within a Web-based honeypot built with Tanner and Snare software, as well as a secure-shell SSH honeypot built with Cowrie software (Chong & Koh, 2018). The techniques included fake files, defensive camouflage, and false excuses. Most attackers did fingerprinting and vulnerability scanning of the honeypots, but did not respond to customized deception and showed minimal interactions. A related experiment tested and evaluated a Web honeypot tool Glastopf and an SSH honeypot tool Kippo (Rowe & Yahyaoui, 2015). Including deception in the Web honeypot encouraged interest by attackers in the additional linked Web pages and interactive features.

C. GAME THEORY

Game theory studies situations where two parties compete, alternating actions between them (Ross, 2019). In a game model for cybersecurity, defenders try to secure the network, and attackers try to circumvent those defenses. Both defender and attacker actions have costs. Costs vary between sites because each target has different information assets, risk factors, and security strategies. A measure of how well the attacker did in the game is the difference between the attacker's net cost (cost minus benefits achieved) and the defender's net cost.

Game theory has been used for projects with the defender using decoys to hide assets. One project used game theory to plan a cyberattack scenario where a defender used lightweight decoys to conceal and defend actual hosts (Major et al., 2019). The defender and attacker played a game with information assets consisting of decoy and real systems, possible actions for each player, and a method for evaluating and defining individual player tactics. This work used multiple game trees and assumed each player's understanding of the game playoffs and structure. This approach appeared to be effective.

Deception games are "imperfect-information stochastic games" in the literature. Network interdiction can be viewed as a game between an attacker and a network defender, where the attacker seeks to degrade network operations while the defender counteracts the attacker. One project studied network interdiction where the attacker has imperfect knowledge of the network topology but can learn about it by monitoring network operations (Zheng and Castanon, 2012). The network observed the attack, and chose to conceal parts of the network to hide information from the attacker. These game models study strategies for both attackers and defenders in a simulated environment, typically using minimax game theory. This theory can minimize the worst-case potential loss by having the attacker and defender consider their opponent's response to each of their strategies and select the method that should give the best payoff (Stanford, 2021).

Game modeling for cybersecurity requires attack and defense models. One project built attack models using deep learning with numerical simulations to verify accuracy (Najada et al., 2018). A different one generated an agent taxonomy using topological data analysis and analysis of simulation outputs (Swarup & Resasadegan, 2019). Another option is the Malicious Activity Simulation Tool, a scalable, flexible, and interoperable architecture for training specialists in cyber security (Swiatocha, 2018). Attack modeling can incorporate movement throughout the network to establish footholds for malicious activities (“lateral movement”) (Bai et al., 2019).

Defenders can make decoys look like legitimate network assets to confuse the attacker (Amin et al., 2020). Previous indicators of compromise can help network defenders recognize attacks and plan deceptions. One way is by finding the attackers’ shortest paths from source to targets and anticipating that they will follow those routes (Wilkens et al., 2019). Honeypots are good for detecting lateral movement in networks since anyone accessing them is suspicious. Another project used game theory to manage lightweight decoys to conceal and defend actual hosts on the same network (Major et al., 2019).

Cyber criminals and autonomous botnets under their control regularly scan networks in search of vulnerable systems to exploit; military and more sophisticated adversaries may also do this. They can collect data from logs, records, and alerts and use this information to correlate events on the network (Balbix, 2020). A system can recognize network attacks by intrusion detection looking for patterns in the data (Anitha et al., 2016). It can also use deception for defense by presenting a false network topology (Trassare et al., 2013).

D. ARTIFICIAL INTELLIGENCE

Artificial-intelligence (AI) technologies can enhance security by helping to plan defenses. It can support curated collection data from logs, records, and alerts and use this information to correlate events on the network (Balbix, 2020). An example of using intrusion detection to find patterns in the data is (Anitha et al., 2016). Artificial-intelligence technologies can process large amounts of data to identify patterns in data. They can also be used in several tasks in cyber defense (Trifonov et al., 2020).

AI techniques in machine learning are useful for cyber defense because cyber threats advance yearly in sophistication and automation. Machine learning could provide much faster protection than what is available with manual identification of threats (Rieck, 2011). In one study, two machine-learning algorithms using reinforcement learning played the attacker and defender in a game (Zhu et al., 2014). Each tried to maximize their cumulative reward as they learned good actions in an environment over many runs. Reinforcement learning can also address other cybersecurity problems (Nguyen & Reddi, 2020).

III. METHODS

A. OVERVIEW

We collaborated with Soar Technology Inc. (SoarTech) to use their proprietary tool, the Cyberspace Course of Action Tool (CCAT). The CCAT simulates a computer network including some deceptions and permits the testing of security measures. In particular, it can test the impact of deception within a network. This provides actionable insights and helps the design of high-security local-area networks. Our experiments with CCAT used three realistic scenarios, variants of those developed in previous experiments (Green, 2020). Deception effectiveness was then measured using metrics based on costs and rewards as discussed later in this chapter.

The CCAT was designed as a cyber wargame between attackers and defenders. Both the attacker and defender can choose actions depending on the specific situation. The scenarios are created with simulated nodes and assets. Empirical game-theoretic analysis (EGTA) is used with reinforcement learning to train agent (attacker or defender) behavior based on results. During training, the agent becomes less likely to choose an action that made it more costly to achieve its objectives, and more likely to choose an action that made it less costly to achieve its objectives.

B. SCENARIO COMPONENTS

The base network map used for our experiments is in Figure 1 (Green, 2020) and represents a local-area network with segmentation and security. It has simulated servers, workstations, a firewall, and other network devices commonly found in military networks, organized in three virtual local-area subnetworks (VLANs). Front-end servers support Web, mail, and applications that access backend servers. Two simulated program-of-record servers, two domain controllers, two Web database servers, two file servers, and one router for application and mail backend servers are in the server's virtual local-area network.

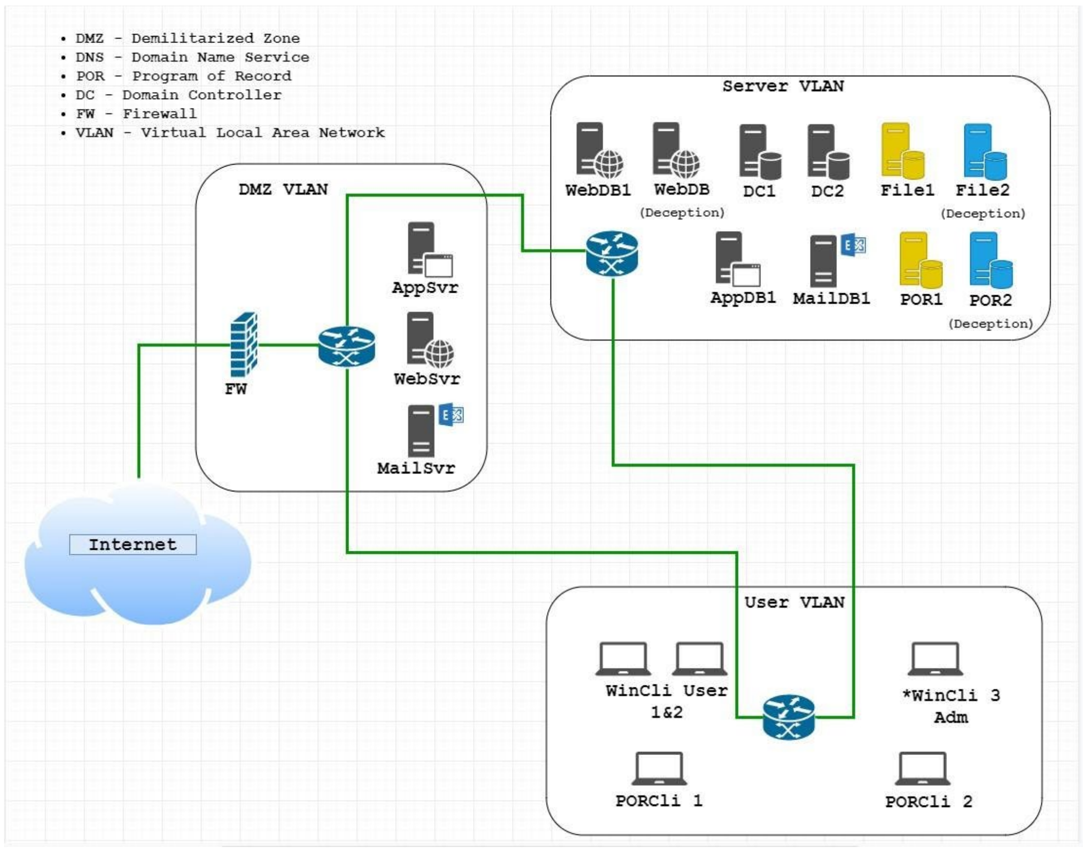


Figure 1. Network Map: Source: Green (2020).

Figure 2 shows the possible actions of a defender in the blue boxes. The defender's goals are to identify attack actions and prevent the attacker from accomplishing their goals. At the start, the defender can choose file monitoring, process monitoring, auditing logon events, auditing intrusion logs, or checking firewall alerts. If an attack has occurred, the defender can stop it from progressing by preventing exfiltration of data or preventing the server from being destroyed. The defender also can create decoy servers, create decoy programs of record, change file names, and change hostnames.

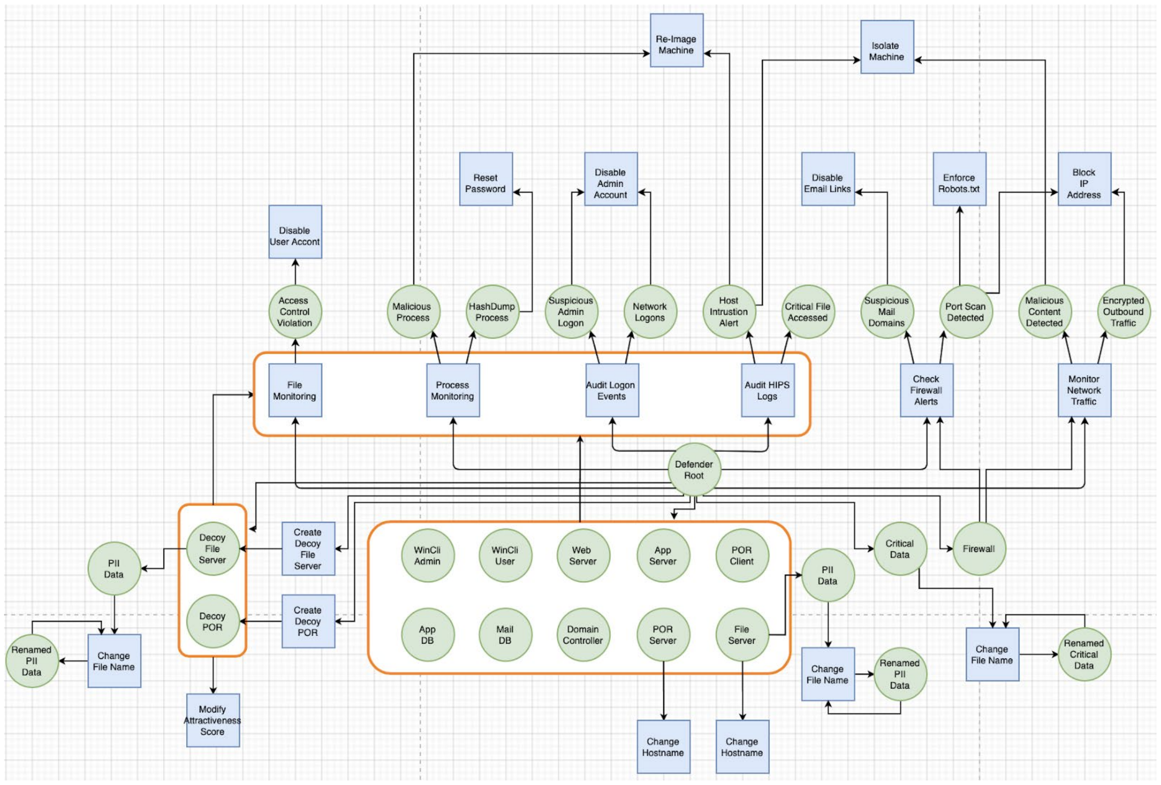


Figure 2. Defender Map. Source: Green (2020).

Figures 3 through 5 show the attacker plan options in the yellow boxes. Attackers start the game by selecting "Detect Decoy" or "Maintain IP Address," a waiting activity. The next action is "Map Network" which simulates a network scan on the defender network, resulting in open ports on the servers in the DMZ. The attacker can then select a specific attack based on the security conditions discovered from the "Map Network" phase. These include spear-phishing attacks, exploiting public-facing applications through servers, and Web crawling to collect a list of addresses. The attacker can then discover local networks, users, and processes, and search local data or determine decoys. This action gains them useful information such as usernames, personal information, or network connections. Then they can act based on the result of the attack such as "Exfil Data" or "Server Destroyed."

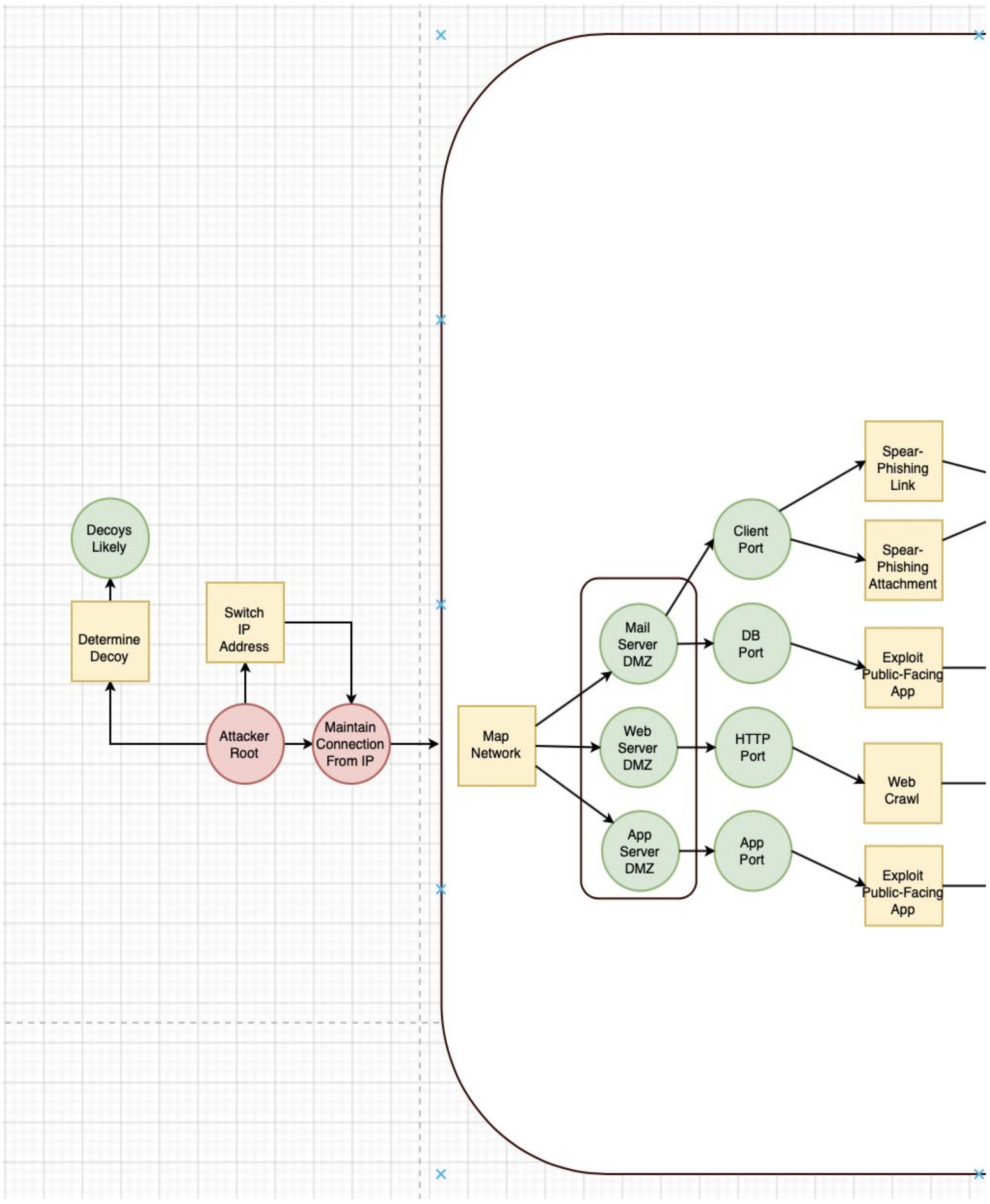


Figure 3. Attack Map (Part 1). Source: Green (2020).

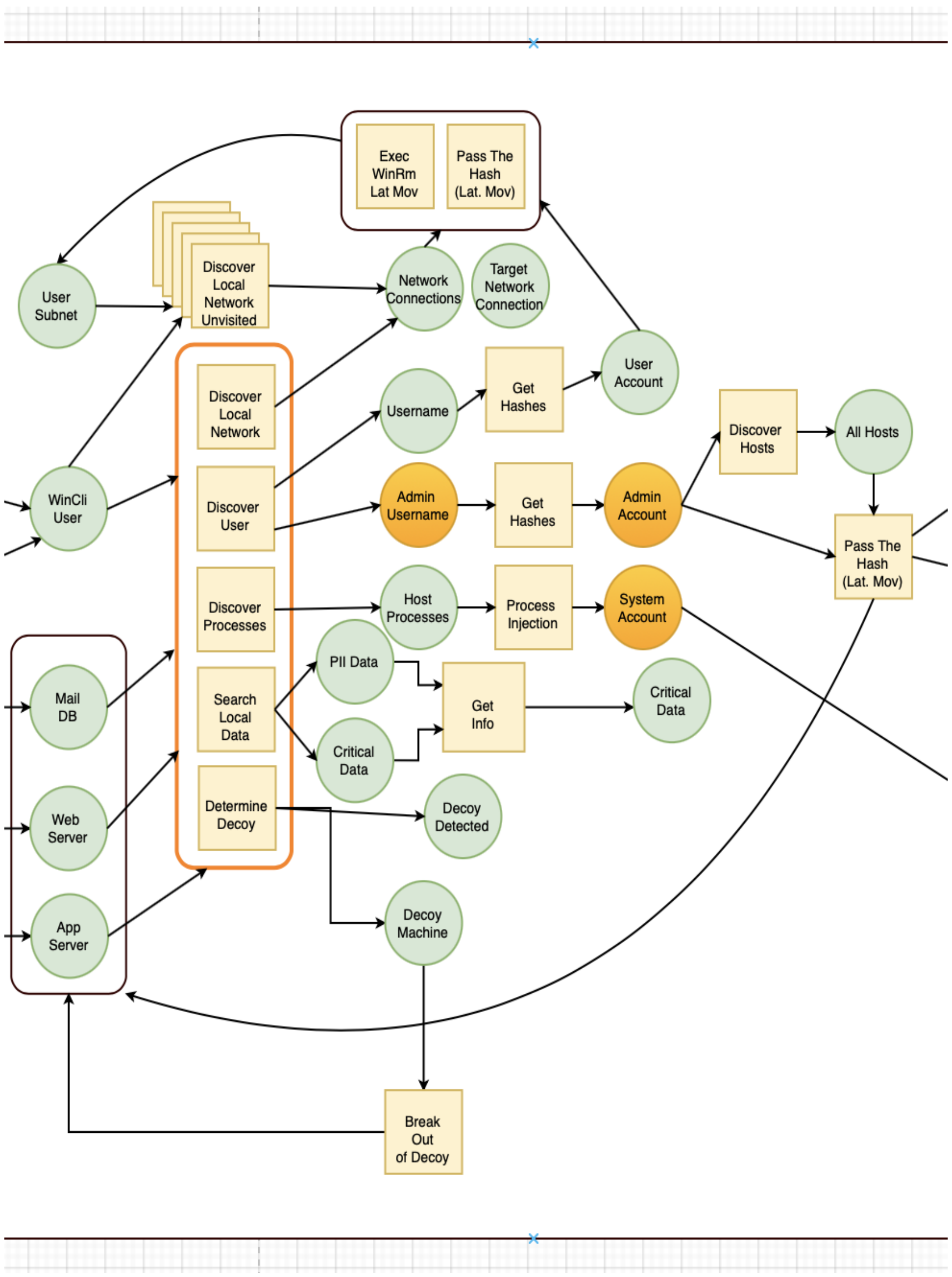


Figure 4. Attack Map (Part 2). Source: Green (2020).

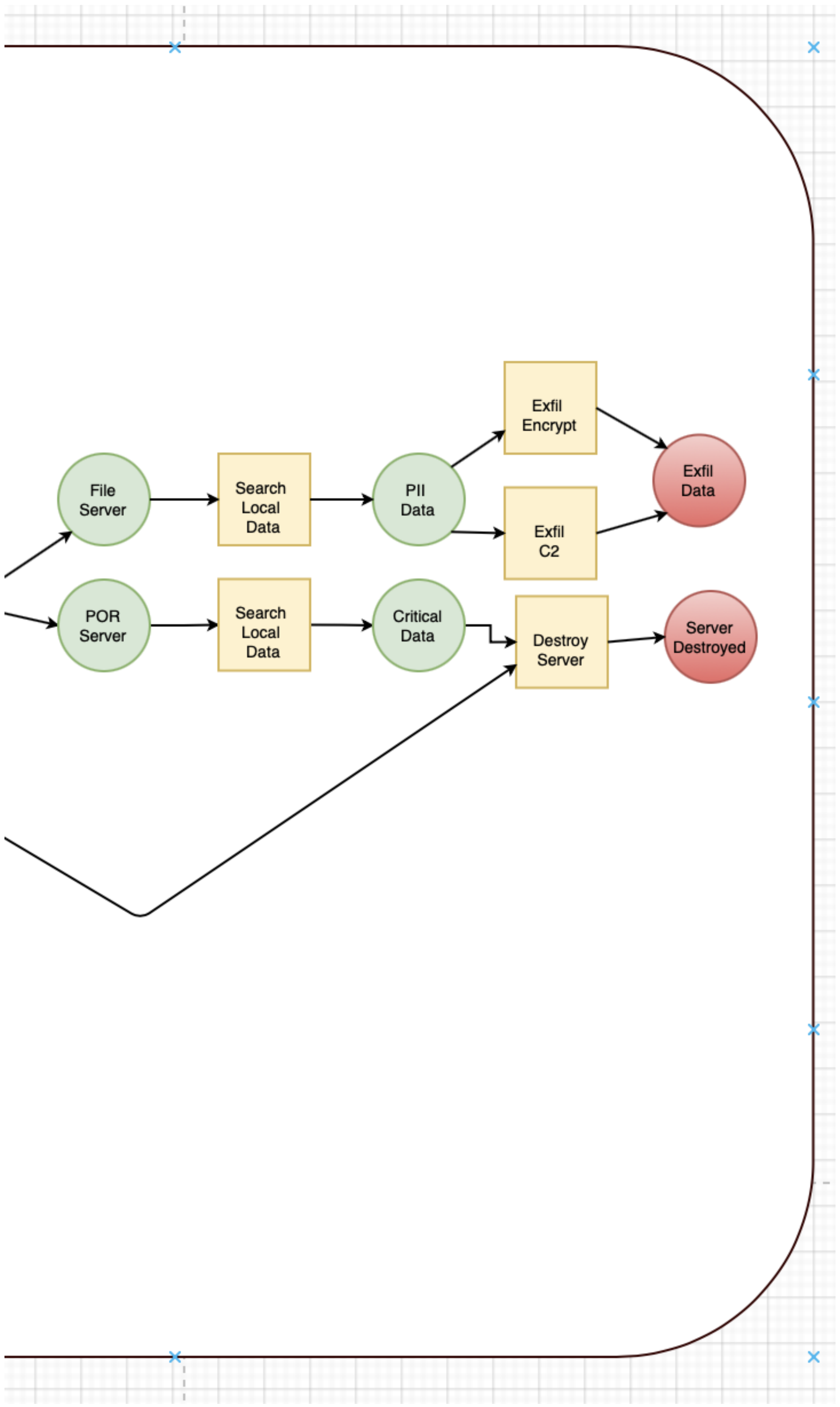


Figure 5. Attack Map (Part 3). Source: Green (2020).

C. SCENARIOS TESTED

The first scenario added ten machines to the baseline experimental topology to observe their impact. Some machines represented hosts without anything of interest on them for the attacker, while others had information that the attacker may consider interesting. To add complications for the defender, the first scenario sent a phishing email to the defender, and the defender had to disable the links and attachments in the email. The second scenario added 100 more generic machines as attack targets to the experimental topology. For the first two experiments these machines added density to the network, slowing the attacker in accomplishing their objectives. The third scenario used a distributed denial-of-service (DDoS) attack on an isolated part of the network to distract the defenders from an exploit-based attack against their email, application, or Web server. The defender can blacklist the IP address, do denial-of-service mitigations, or install tools to prevent additional common attacks. To make the scenario more challenging, the attacker spoofed a trusted IP address that did not require the usual degree of authentication; the defender must then choose the correct address to block.

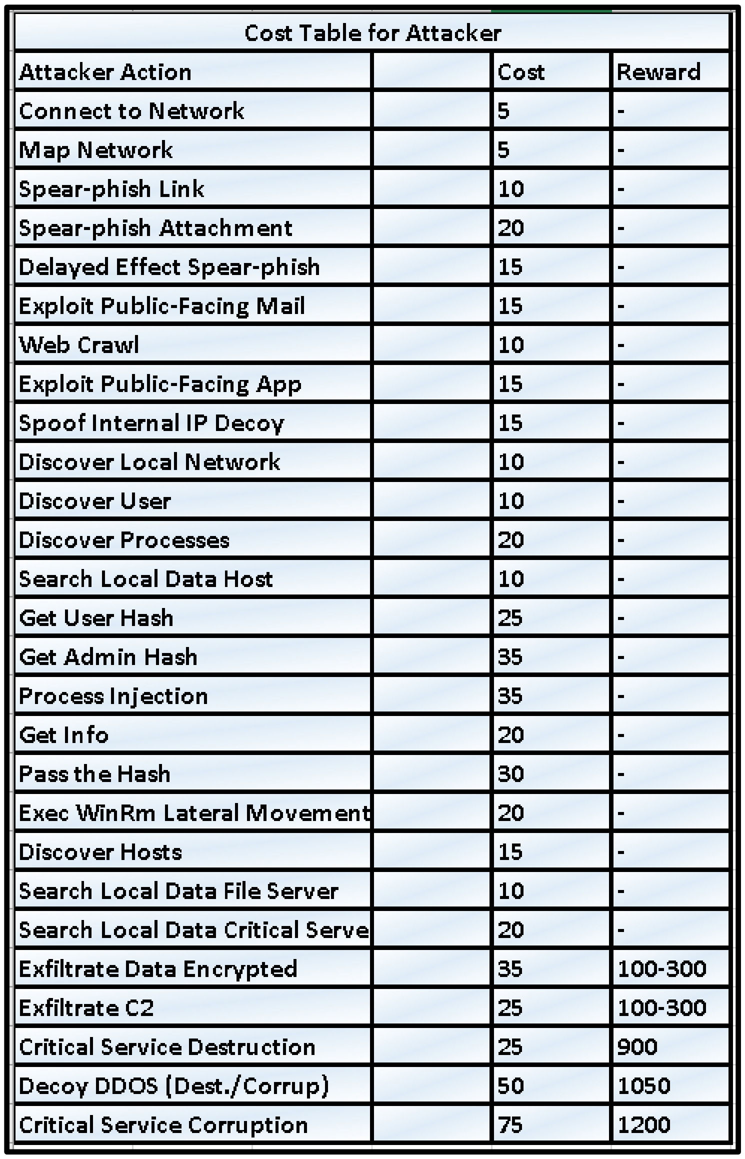
D. ASSIGNING COSTS AND BENEFITS TO THE SCENARIO OPTIONS

Costs and benefits are associated with each action that the attacker and defender select. We used the already established values from (Green, 2020) shown in Tables 1–5. The game scoring metric for the attacker was the total cost spent minus the rewards received for all agents over the entire game. The game score for the defender was the total cost spent by defender agents. The total game score was the score for the defender minus the score for the attacker. Thus, positive values of the metric indicated success of the attacker, and negative values indicated success of the defender. Attacker and defender actions were logged for further analysis.

The score for an attacker is the sum of the rewards it received for achieving its goals minus the cost of all the actions it has taken. The defender score is the sum of the rewards the attacker received and the costs the defender incurred. The attacker tries to maximize their score, and the defender tries to minimize their score, but the defender can never do better than a zero score.

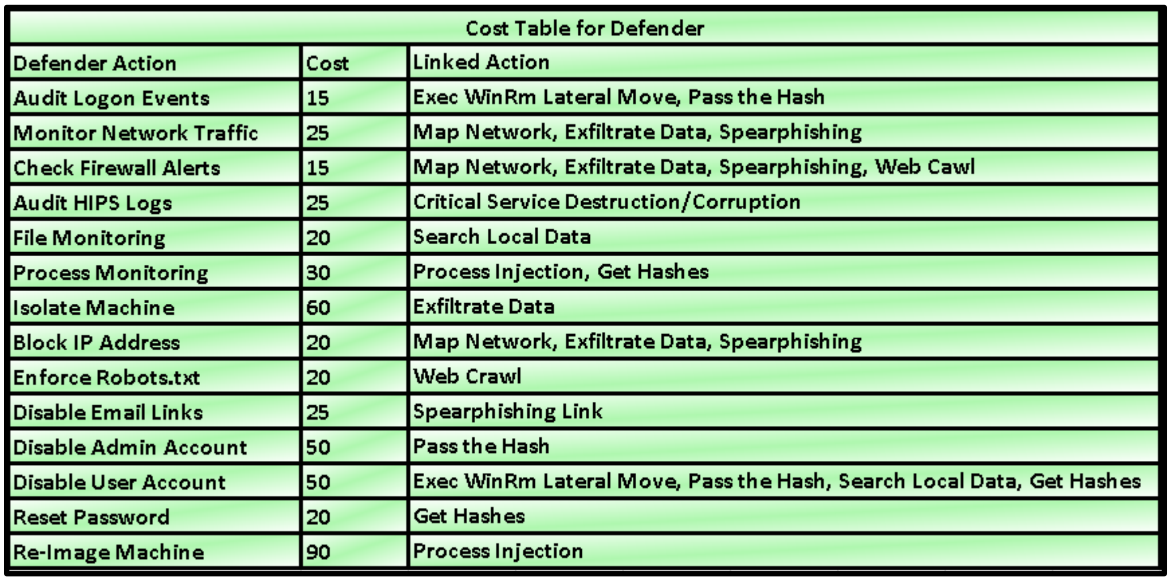
The values we used for the costs and rewards and how the simulation environment was set up were based on estimates and heuristics, so similar results in the real world are not guaranteed. However, if the defender score is low using deceptive techniques, it suggests that the deceptive techniques work.

Table 1. Cost Attacker. Adapted from Green (2020).



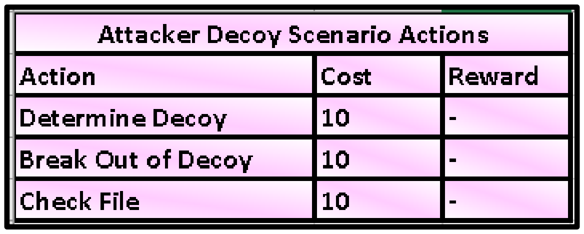
Cost Table for Attacker			
Attacker Action		Cost	Reward
Connect to Network		5	-
Map Network		5	-
Spear-phish Link		10	-
Spear-phish Attachment		20	-
Delayed Effect Spear-phish		15	-
Exploit Public-Facing Mail		15	-
Web Crawl		10	-
Exploit Public-Facing App		15	-
Spoof Internal IP Decoy		15	-
Discover Local Network		10	-
Discover User		10	-
Discover Processes		20	-
Search Local Data Host		10	-
Get User Hash		25	-
Get Admin Hash		35	-
Process Injection		35	-
Get Info		20	-
Pass the Hash		30	-
Exec WinRm Lateral Movement		20	-
Discover Hosts		15	-
Search Local Data File Server		10	-
Search Local Data Critical Serve		20	-
Exfiltrate Data Encrypted		35	100-300
Exfiltrate C2		25	100-300
Critical Service Destruction		25	900
Decoy DDOS (Dest./Corrup)		50	1050
Critical Service Corruption		75	1200

Table 2. Cost Defender. Adapted from Green (2020).



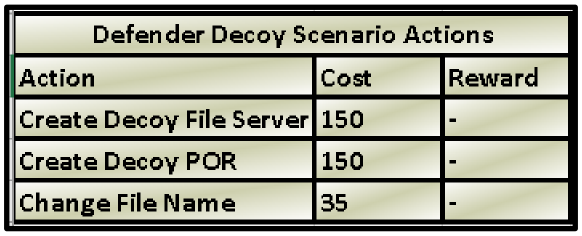
Cost Table for Defender		
Defender Action	Cost	Linked Action
Audit Logon Events	15	Exec WinRm Lateral Move, Pass the Hash
Monitor Network Traffic	25	Map Network, Exfiltrate Data, Spearphishing
Check Firewall Alerts	15	Map Network, Exfiltrate Data, Spearphishing, Web Cawl
Audit HIPS Logs	25	Critical Service Destruction/Corruption
File Monitoring	20	Search Local Data
Process Monitoring	30	Process Injection, Get Hashes
Isolate Machine	60	Exfiltrate Data
Block IP Address	20	Map Network, Exfiltrate Data, Spearphishing
Enforce Robots.txt	20	Web Crawl
Disable Email Links	25	Spearphishing Link
Disable Admin Account	50	Pass the Hash
Disable User Account	50	Exec WinRm Lateral Move, Pass the Hash, Search Local Data, Get Hashes
Reset Password	20	Get Hashes
Re-Image Machine	90	Process Injection

Table 3. Attacker Decoy Scenario Actions. Adapted from Green (2020).



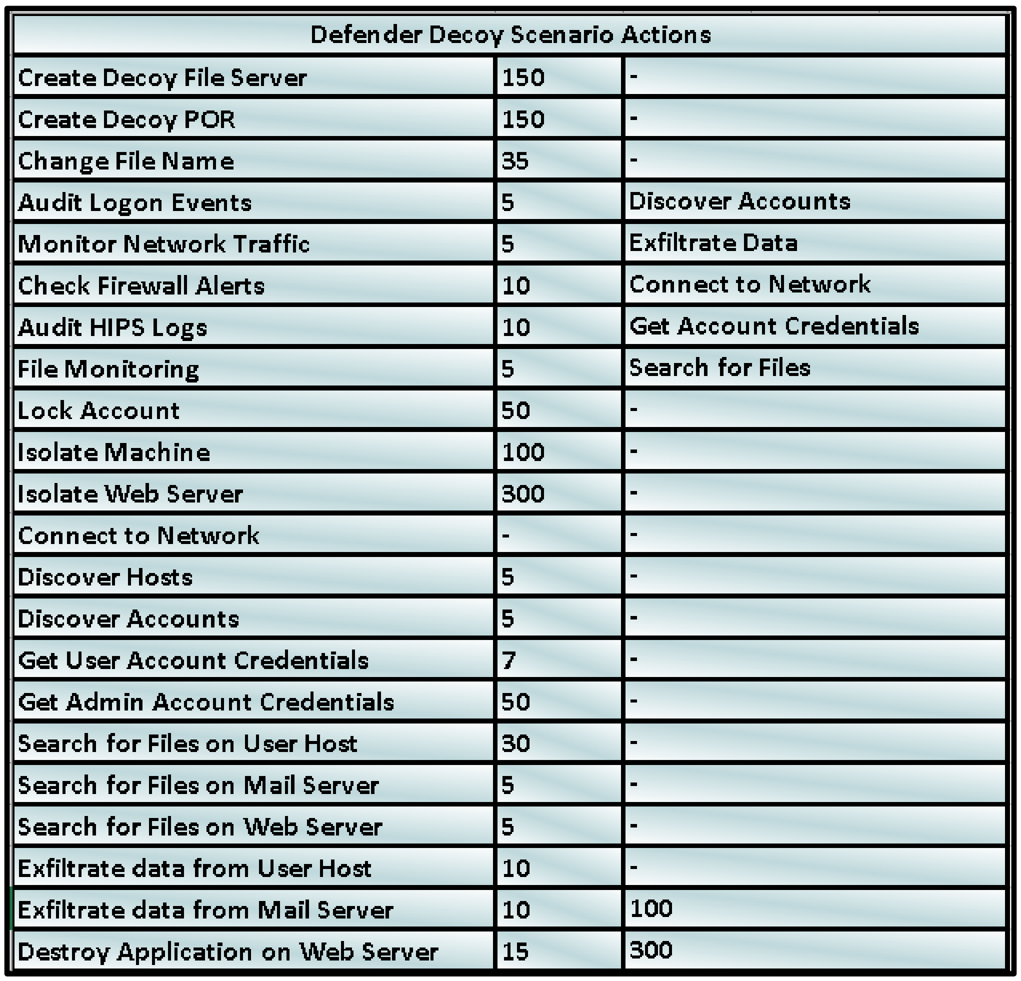
Attacker Decoy Scenario Actions		
Action	Cost	Reward
Determine Decoy	10	-
Break Out of Decoy	10	-
Check File	10	-

Table 4. Defender Decoy Scenario Actions. Adapted from Green (2020).



Defender Decoy Scenario Actions		
Action	Cost	Reward
Create Decoy File Server	150	-
Create Decoy POR	150	-
Change File Name	35	-

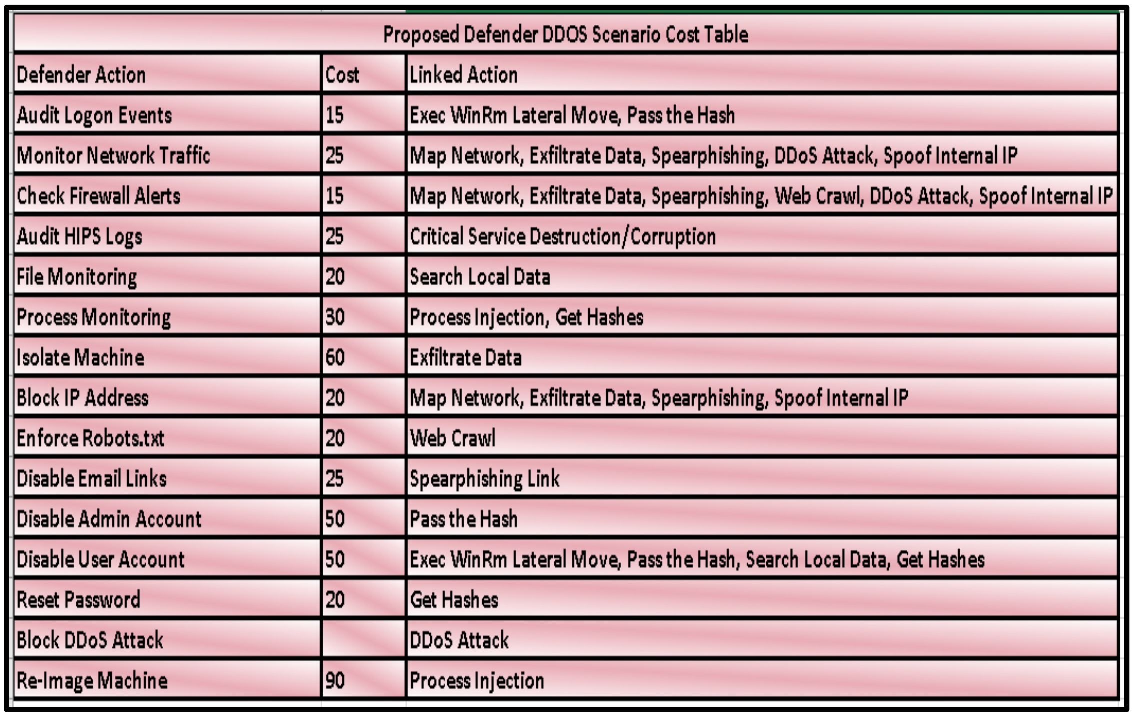
Table 5. Defender Decoy Scenario Actions. Adapted from Green (2020).



Defender Decoy Scenario Actions		
Create Decoy File Server	150	-
Create Decoy POR	150	-
Change File Name	35	-
Audit Logon Events	5	Discover Accounts
Monitor Network Traffic	5	Exfiltrate Data
Check Firewall Alerts	10	Connect to Network
Audit HIPS Logs	10	Get Account Credentials
File Monitoring	5	Search for Files
Lock Account	50	-
Isolate Machine	100	-
Isolate Web Server	300	-
Connect to Network	-	-
Discover Hosts	5	-
Discover Accounts	5	-
Get User Account Credentials	7	-
Get Admin Account Credentials	50	-
Search for Files on User Host	30	-
Search for Files on Mail Server	5	-
Search for Files on Web Server	5	-
Exfiltrate data from User Host	10	-
Exfiltrate data from Mail Server	10	100
Destroy Application on Web Server	15	300

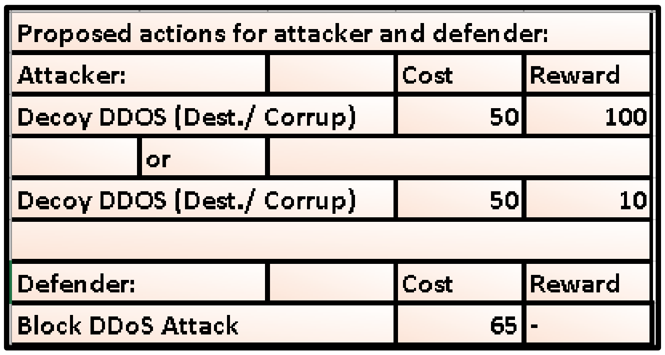
Specifically, for the denial-of-service scenario, the defender's reward for detecting and addressing the first attack is minimal since it is a diversion. It is more important that the defender prevent the attacker from gaining critical-system access. The longer a denial of service occurs, the easier it is to determine how to block it, so there will be a small reward if the defender does block it. A fairer cost-benefit accounting is that the attacker gains points for each period of denial of service, following Table 6 and Table 7. The costs and benefits for the military-adversary attacker could be different since their goals could be sabotage and espionage.

Table 6. Proposed Defender DDoS Scenario Cost Table



Proposed Defender DDOS Scenario Cost Table		
Defender Action	Cost	Linked Action
Audit Logon Events	15	Exec WinRm Lateral Move, Pass the Hash
Monitor Network Traffic	25	Map Network, Exfiltrate Data, Spearphishing, DDoS Attack, Spoof Internal IP
Check Firewall Alerts	15	Map Network, Exfiltrate Data, Spearphishing, Web Crawl, DDoS Attack, Spoof Internal IP
Audit HIPS Logs	25	Critical Service Destruction/Corruption
File Monitoring	20	Search Local Data
Process Monitoring	30	Process Injection, Get Hashes
Isolate Machine	60	Exfiltrate Data
Block IP Address	20	Map Network, Exfiltrate Data, Spearphishing, Spoof Internal IP
Enforce Robots.txt	20	Web Crawl
Disable Email Links	25	Spearphishing Link
Disable Admin Account	50	Pass the Hash
Disable User Account	50	Exec WinRm Lateral Move, Pass the Hash, Search Local Data, Get Hashes
Reset Password	20	Get Hashes
Block DDoS Attack		DDoS Attack
Re-Image Machine	90	Process Injection

Table 7. Proposed Actions for Attacker and Defender



Proposed actions for attacker and defender:			
Attacker:		Cost	Reward
Decoy DDOS (Dest./ Corrup)		50	100
	or		
Decoy DDOS (Dest./ Corrup)		50	10
Defender:		Cost	Reward
Block DDoS Attack		65	-

E. TRAINING AND TESTING

SoarTech ran the scenarios we configured at their site in Michigan. We mostly followed the parameter recommendations of our SoarTech contacts. To determine the best strategies for attack and defense, SoarTech engineers trained the attacker and defender agents used in their CCAT tool for us before testing it.

After that agent has been trained, that agent is played against previous agents of its opponent. For example, in round 10 of the "outer loop," we will have trained 10 different defender agents, 1 in each previous round. Each round of the outer loop consists of the 1 million games (so, 10 million total). The attacker and defender are trained separately, so each round has 2 million games, 1 million for the newest defender training against one of the previous attackers, and 1 million for the newest attacker training against one of the previous defenders. Across the million games, only the newest defender and newest attacker are trained, and the previous attacker and previous defender are unchanged. At the end of each round after 1 million games have been played, we run a few games with each pair of currently developed attacker and defender agents against all of the previous defender or attacker agents, respectively. At that point we compute the Nash equilibrium across all of the games, and get a score for each agent. Running a few games against previous agents is beneficial to demonstrate how quickly it takes the attacker and defender to choose an action. It also demonstrates which action is chosen the most. This information is compared to the most recent agents that incorporate the experience of all previous agents.

We have a stopping criterion for the outer loop that is either a certain level of regret, or a certain number of rounds. We pick whichever agent had the lowest regret (for each of the attacker and defender agents) and play the 100 games with that agent. For all of the games, we used the same parameter values. The maximum number of steps allowed per game was 45; at that point, games were ended, and scores computed even if more options were available to the players.

The game is not a zero-sum game because some choices are costly for both sides (Zhang, 2009). The defender gets penalized while the attacker is rewarded when the attacker achieves a goal. So, to perform its best, the defender should take as few actions as possible while also preventing the attacker from achieving as few goals as possible.

The "base scenario" was the simplest game tested. SoarTech used the costs and actions from the previous experiment (Green, 2020) with the additions described such as a delayed spear-phishing capability. Results on the base scenario became a baseline for the other three scenarios.

For the scenarios we tested, we added the following attacker and defender actions beyond those of (Green, 2020):

1. Attacker options of “spear-phish link” and “spear-phish attachment” were added for the first scenario.

2. Attacker action “decoy DDOS” and defender action “terminate excess network connection” were added for the second scenario. Also, the second scenario setup ensured the DDOS was always selected.

3. The third scenario allowed the attacker to use a spoofed IP address.

4. The first and third scenarios permitted a variety of attacker actions, but in the second scenario, the attacker always selected distributed denial of service.

5. The attacker and defender alternated turns and their actions did not overlap. Each agent chose its action with a neural network with an initial random initialization of weights; its inputs were the state of the simulation.

F. SCENARIOS NOT TESTED

Due to the limits of time, several good deception ideas for deceivers and attackers in (Rowe & Rrushi, 2016) were not tested but should be.

Deceiver deception ideas:

- Defensive deception methods can be as simple as providing messages when the attackers connect or have advanced as embedding malware within the files that an attacker takes. Decoys can be used for active defense by providing fake content. Discovery of the content stored on a system is a typical first stage of cyberattacks, and misleading content may trick an attacker into wasting time or following the wrong path in the next stage of their attack. Providing random false error messages on tries to connect to the decoys is effective because this showed inconsistency and test the attacker’s overall flexibility.

- Deception can inject malicious software into a system. Messages can have attachments which conceal malicious software. Almost any type of file can be attached to an email message, so attackers have considerable freedom.
- Another way that a defender can deceive is by creating large file directories generated automatically from grammar rules. A grammar could be inferred from examples. This will confuse attackers and make it harder for them to find key files. A grammar could use regular expressions like “/[a-z]+/” that can match many things, and could account for contextual parameters such as a user’s name.
- Address redirection enables defenders to make network reconnaissance harder for attackers. Connecting the attacker to an unexpected and unrelated website will impede attackers from mapping to the network.
- Decoy content can be used to detect malicious behavior since any access attempts to it are suspicious by default.
- A defender can send the attacker taunting messages saying how much they know about the attacker. This does require quickly recognizing the attacker’s tactics, techniques, and procedures. Every attack has an objective, such as computing resources or data theft, and the attack typically requires multiple steps to reach that objective, so these could be identified by defenders in a message to the attacker.
- Embedding malicious code within files requested by the attacker, or having website requests redirected to malicious websites, could impede an attack. The simulation could calculate an expected cost to the attacker to recognize and delete the malware. Not only will this slow the attacker by failing to provide what they are looking for, but it could also give the defenders a way into the attackers’ device, helping identify future attacks.
- Throttling limits, the number of sequential or concurrent user actions to prevent the overuse of resources. If a user gets throttled frequently, defenders can use a code to end the users access to the network or devices

while claiming that an attacker has accessed too many sites or has made too many downloads. Malicious traffic can be deliberately throttled with false messages.

- Batch files (".bat" files) help automate the execution of recurring command sequences. They can be used as bait for the attacker with misinformation in them to waste the attackers' time and resources.

Attacker deception ideas:

- Attacker deceptive methods aim at tricking the defender into providing access and deceive the defender by having them focus their attention on the wrong aspects of an incident. Attackers can use schemes and scams to try to deceive the defender to harvest user and password information, lock a computer with ransomware, or launch malicious code.
- Attackers can change traffic destinations by modifying the network settings. For example, once the default gateway on a device is modified, an attacker can redirect traffic leaving the network to a device the defender controls.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. RESULTS

The average defender and attacker scores for the 100 test iterations of the base scenario were -3943.54 and -1551.27, respectively, for a net defender benefit (defender minus attacker) of -2392.27 (Figure 6). Table 8 in the Appendix shows the overall defender scores for each test run, and Table 9 shows the overall attacker scores. The scores are compared to the baseline iteration.

The "base scenario" was the simplest game tested. SoarTech used the costs and actions from the previous experiment (Green, 2020) with the additions described such as a delayed spear-phishing capability. Results on the base scenario became a baseline for the other three scenarios. It is based on 100 runs as shown in Table 8 and Table 9. Table 10 shows the counts of attacker actions taken and Table 11 shows the counts of defender actions taken. Table 12 shows how many game steps it took the attacker to achieve their goals per game during the 100 test iterations when they did eventually achieve them. The blanks, denoted by DNF, are cases where the attacker did not achieve their goals in 45 steps. The average number of attacker game steps required to complete the base scenario's goals was 25.66, and it was reached 88 out of 100 times.

The average defender score on the first scenario was -3636.12 and the average attacker score was -1682.05 over the 100 test iterations, for a net defender benefit of -1954.07, an improvement on the baseline. Adding ten more machines to the network made it easier for the attacker to find what they wanted since we did not increase the number of data goal items that would reward the attacker. This scenario had an 84% completion rate (Figure 7), and the attacker took an average of 26.01 game steps over the 100 test iterations (Figure 8).

The second scenario added 100 more machines to the network, and it resulted in an average defender score of -3309.61 and the average attacker score of -1739.53, for a net defender benefit of -1570.08, a larger improvement on the baseline. This was the lowest (best) score for the defender compared to the other scenarios and the baseline, indicating that the additional machines did impose additional costs on the attacker. It was also the highest (worst) score for the attacker. The average number of attacker steps for this game were also up to 29.27, with a 73% completion rate.

The third scenario was distributed denial of service and had an average defender score of -3908.91 and an average attacker score of -1668.26, for a net defender benefit of -2240.65, a small improvement on the baseline. This indicates that the attacker can be more successful with this kind of decoy attack. The average completion rate was 90% and an average of 25.23 steps were taken by the attacker, both exceeding baseline numbers.

The way to assess the performance of the defender is to subtract the attacker score from the defender score. The differences can then be compared to those of the baseline game. For example, in the Baseline game, the defender average was -3843.54 and the attacker average was -1551.27, a difference of 2292.27. In the second scenario, the defender average was -3309.61 and the attacker average was -1739.53, a difference of 1570.09. So, the defender did better in the second scenario, which shows that the deception added in that scenario was effective.

For all of the scenarios, the most frequent attacker action was "discover local network" which was used on average 21.25 times in the four games (baseline and three scenarios). The least selected attacker option was "critical service corruption" which the attacker never chose in the 400 total games. The most common defender action was "audit logon events" which was selected 70 times. The least frequent defender action was "reimage machine" which was chosen only twice.

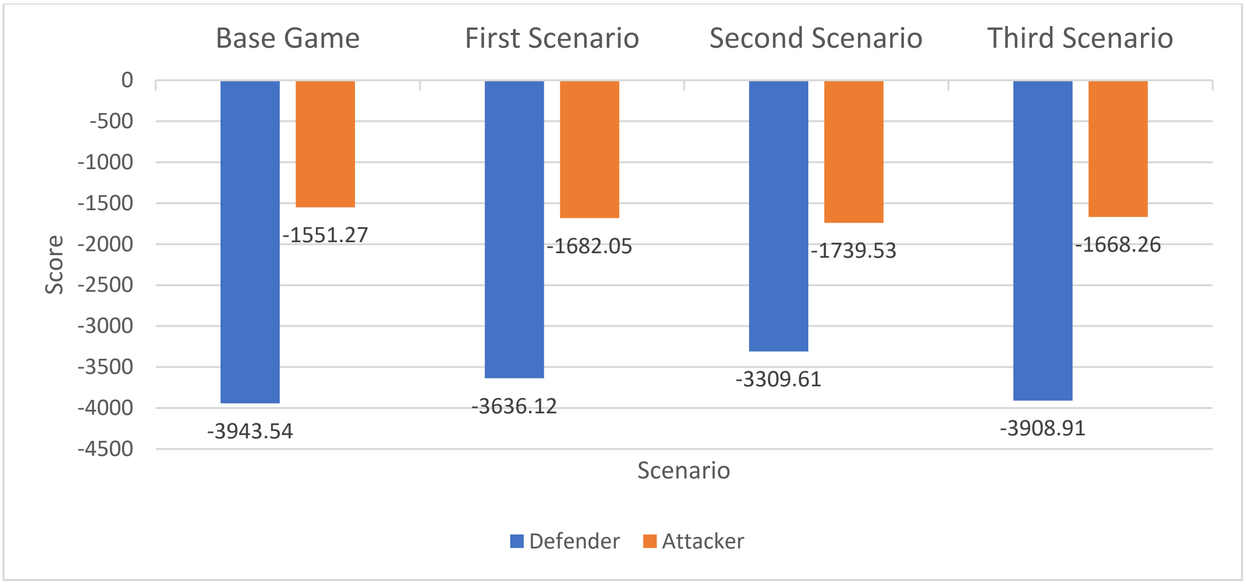


Figure 6. Defender and Attacker Scores in the Scenario

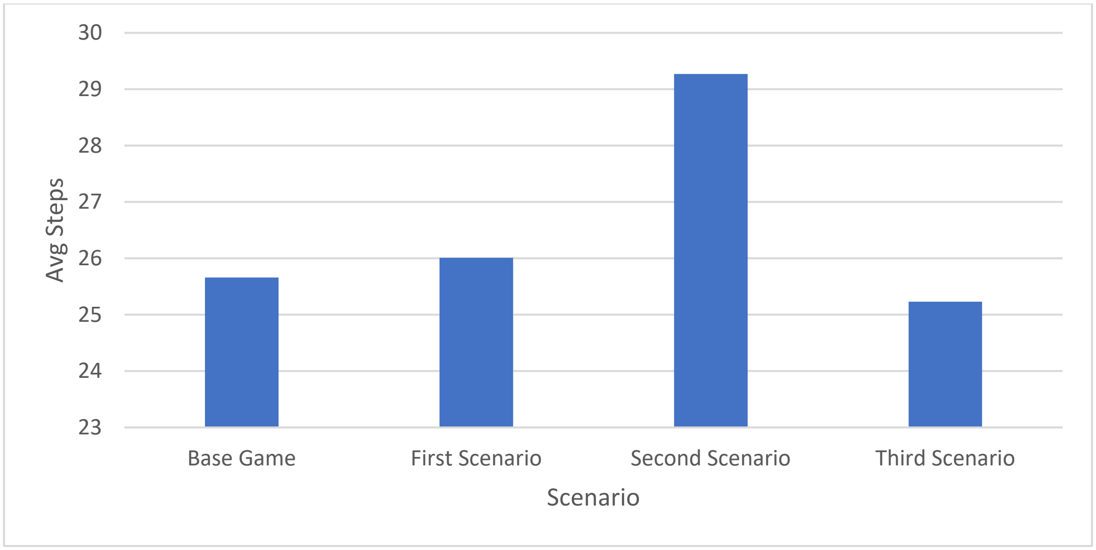


Figure 7. Average Steps Taken by Attacker

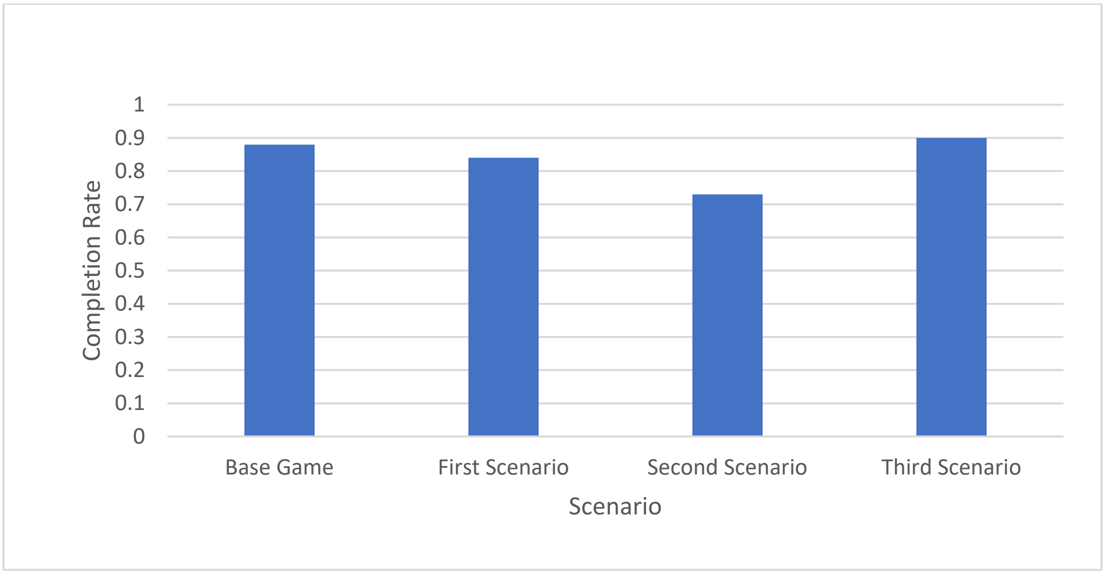


Figure 8. Completion Rate by Attacker

V. CONCLUSIONS AND FUTURE WORK

Defense through cyber deception can level the cyber battlefield by altering an enemy’s perception of reality through delays and disinformation, forcing them to reveal attack methods, and providing defenders with the attributions needed to discover the adversary’s strategy. Discouraging and delaying actions can enhance deceptions by providing time for forensics teams to analyze, identify, and handle attack vectors that could harm operational and support systems. Deception and game theory tools can help in decision making and support military operations in the cyber domain.

The results of this thesis suggested that deception in cyberspace can be systematically planned to be cost-effective for defenders. A lower defender score means the defender spent more time defending. The defender deceptive techniques increased the average amount of time the attacker used to meet their objectives. Since defenders used deception to increase the overall attack time, they were allotted more time in return to detect and respond to the attack. The success rate for the attacker meeting objectives decreased when the defender used deception. The attacker and defender scores are all based on the baseline game. Thus, since the defender score was closer to 0 in the scenarios and the attacker score was farther away from the baseline it means that deception was effective.

Deception can help defenders at a low cost, while increasing costs for the attacker, significantly limiting the attacker’s ability to complete their objectives. Further research into game theory and deception could identify more ways to improve cyber defense. For instance, we could consider a limited budget an attacker might have for defensive actions.

Deception techniques can be introduced in network services and protocols, to severely degrade an attacker’s methods of exploitation and attack. Forcing changes in the attacker’s actions or behavior can also highlight and expose their activities, enabling their detection, inference of their intent, and remediation for actions already taken. These effects can provide good deterrence from attack.

The CCAT tool can be extended with topologies created for air, land, sea, and space terrains. The topologies could represent actual terrains, making it realistic for actual operators. Understanding how the attacker and defender agents react in these environments will enhance defensive operations. Scenario topology can include mobile devices and Supervisory Control and Data Acquisition (SCADA) systems for future experiments. Scenarios can also include more spoofed IP addresses and delayed attacks; spoofed IP addresses can be challenging to recognize.

Encrypted data could be included in the games to test how far attackers are willing to go to steal secrets. Also, surveys can be sent to operators to evaluate deception, and game theory tools such as ours will enhance operations. Their feedback could be helpful because they execute real-world operations and could judge how these tools will enhance learning and decision-making.

Other second-order deceptions are possible that the CCAT could include at low cost. Having a defender sending many messages to the attacker would require no change to the current environment. Increasing the number of files within machines to confuse attackers would be easy to do without changing the current environment, as it would just require a random generator to name the files in a realistic way.

If the CCAT tool could successfully simulate multiple layers of deception, it would enable analyzing how an attacker would react when met with a set of challenges. Other attacker options can be explored if information resources are withheld deceptively within the CCAT. Implementation of these methods would help analyze the potential costs of the attacker versus the cost of the defender.

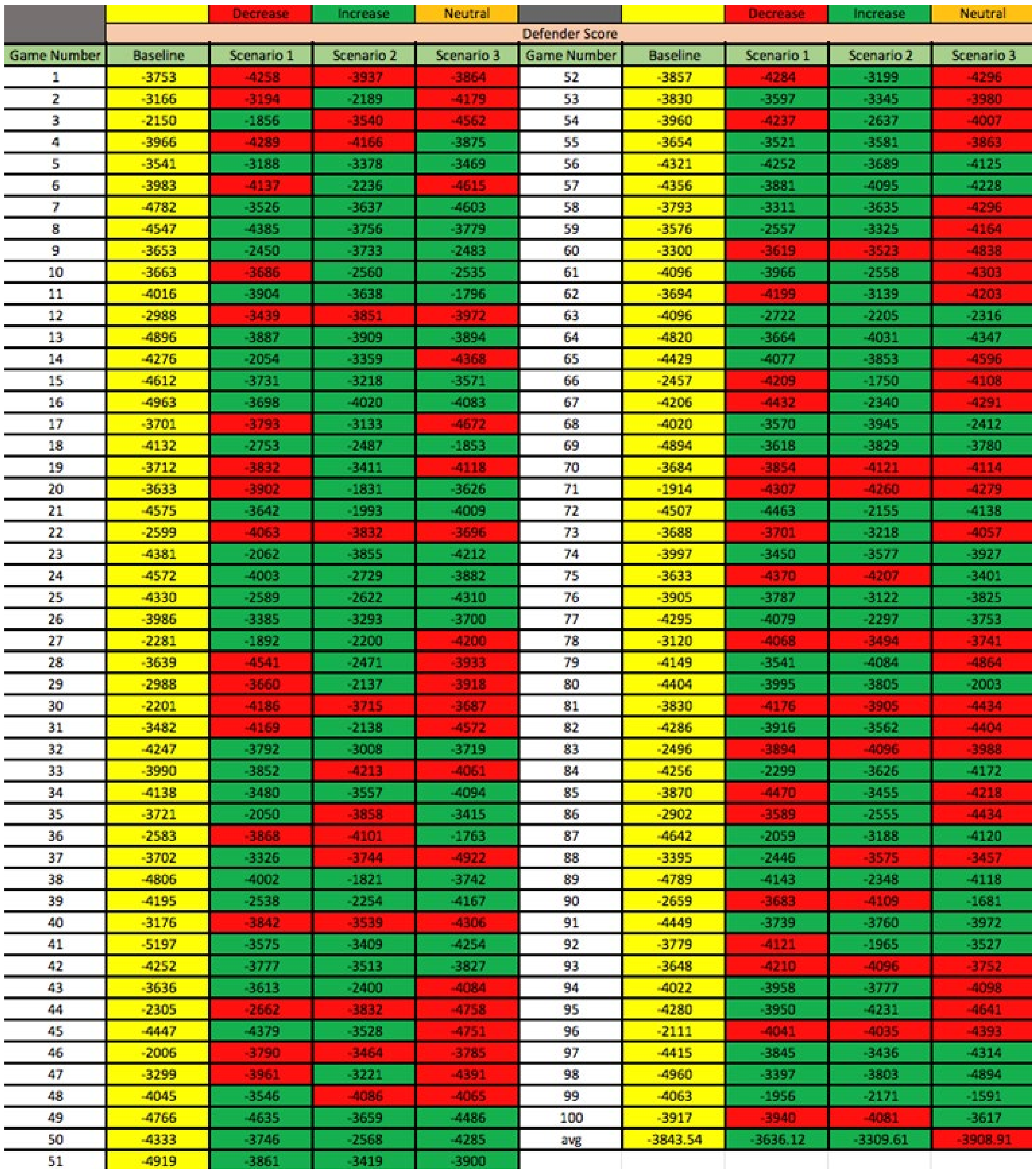
APPENDIX. EXPERIMENTAL DATA

The tables in the Appendix show the results of the 100-game evaluation period for each experiment.

A. OVERALL SCORES

Tables 8 and 9 show overall scores for the experiments as the sum of all costs and rewards for both attacker and defender agents. Red indicates runs in which the player did worse than the baseline, and green indicates runs in which they did better than the baseline. Run numbers are given in the white columns.

Table 8. Overall Defender Scores



		Decrease	Increase	Neutral
	Defender Score			
Game Number	Baseline	Scenario 1	Scenario 2	Scenario 3
1	-3753	-4258	-3937	-3864
2	-3166	-3194	-2189	-4179
3	-2150	-1856	-3540	-4562
4	-3966	-4289	-4166	-3875
5	-3541	-3188	-3378	-3469
6	-3983	-4137	-2236	-4615
7	-4782	-3526	-3637	-4603
8	-4547	-4385	-3756	-3779
9	-3653	-2450	-3733	-2483
10	-3663	-3686	-2560	-2535
11	-4016	-3904	-3638	-1796
12	-2988	-3439	-3851	-3972
13	-4896	-3887	-3909	-3894
14	-4276	-2054	-3359	-4368
15	-4612	-3731	-3218	-3571
16	-4963	-3698	-4020	-4083
17	-3701	-3793	-3133	-4672
18	-4132	-2753	-2487	-1853
19	-3712	-3832	-3411	-4118
20	-3633	-3902	-1831	-3626
21	-4575	-3642	-1993	-4009
22	-2599	-4063	-3832	-3696
23	-4381	-2062	-3855	-4212
24	-4572	-4003	-2729	-3882
25	-4330	-2589	-2622	-4310
26	-3986	-3385	-3293	-3700
27	-2281	-1892	-2200	-4200
28	-3639	-4541	-2471	-3933
29	-2988	-3660	-2137	-3918
30	-2201	-4186	-3715	-3687
31	-3482	-4169	-2138	-4572
32	-4247	-3792	-3008	-3719
33	-3990	-3852	-4213	-4061
34	-4138	-3480	-3557	-4094
35	-3721	-2050	-3858	-3415
36	-2583	-3868	-4101	-1763
37	-3702	-3326	-3744	-4922
38	-4806	-4002	-1821	-3742
39	-4195	-2538	-2254	-4167
40	-3176	-3842	-3539	-4306
41	-5197	-3575	-3409	-4254
42	-4252	-3777	-3513	-3827
43	-3636	-3613	-2400	-4084
44	-2305	-2662	-3832	-4758
45	-4447	-4379	-3528	-4751
46	-2006	-3790	-3464	-3785
47	-3299	-3961	-3221	-4391
48	-4045	-3546	-4086	-4065
49	-4766	-4635	-3659	-4486
50	-4333	-3746	-2568	-4285
51	-4919	-3861	-3419	-3900
52	-3857	-4284	-3199	-4296
53	-3830	-3597	-3345	-3980
54	-3960	-4237	-2637	-4007
55	-3654	-3521	-3581	-3863
56	-4321	-4252	-3689	-4125
57	-4356	-3881	-4095	-4228
58	-3793	-3311	-3635	-4296
59	-3576	-2557	-3325	-4164
60	-3300	-3619	-3523	-4838
61	-4096	-3966	-2558	-4303
62	-3694	-4199	-3139	-4203
63	-4096	-2722	-2205	-2316
64	-4820	-3664	-4031	-4347
65	-4429	-4077	-3853	-4596
66	-2457	-4209	-1750	-4108
67	-4206	-4432	-2340	-4291
68	-4020	-3570	-3945	-2412
69	-4894	-3618	-3829	-3780
70	-3684	-3854	-4121	-4114
71	-1914	-4307	-4260	-4279
72	-4507	-4463	-2155	-4138
73	-3688	-3701	-3218	-4057
74	-3997	-3450	-3577	-3927
75	-3633	-4370	-4207	-3401
76	-3905	-3787	-3122	-3825
77	-4295	-4079	-2297	-3753
78	-3120	-4068	-3494	-3741
79	-4149	-3541	-4084	-4864
80	-4404	-3995	-3805	-2003
81	-3830	-4176	-3905	-4434
82	-4286	-3916	-3562	-4404
83	-2496	-3894	-4096	-3988
84	-4256	-2299	-3626	-4172
85	-3870	-4470	-3455	-4218
86	-2902	-3589	-2555	-4434
87	-4642	-2059	-3188	-4120
88	-3395	-2446	-3575	-3457
89	-4789	-4143	-2348	-4118
90	-2659	-3683	-4109	-1681
91	-4449	-3739	-3760	-3972
92	-3779	-4121	-1965	-3527
93	-3648	-4210	-4096	-3752
94	-4022	-3958	-3777	-4098
95	-4280	-3950	-4231	-4641
96	-2111	-4041	-4035	-4393
97	-4415	-3845	-3436	-4314
98	-4960	-3397	-3803	-4894
99	-4063	-1956	-2171	-1591
100	-3917	-3940	-4081	-3617
avg	-3843.54	-3636.12	-3309.61	-3908.91

Table 9. Overall Attacker Scores

		Decrease	Increase	Neutral			Decrease	Increase	Neutral
	Attacker Score								
Game Number	Baseline	Scenario 1	Scenario 2	Scenario 3	Game Number	Baseline	Scenario 1	Scenario 2	Scenario 3
1	-1516	-1961	-2792	-1737	52	-822	-878	-2707	-1839
2	-678	-1333	-3310	-2114	53	-2230	-955	-709	-485
3	-2840	-3376	-1856	-2992	54	-1431	-1393	-2588	-1496
4	-1869	-621	-1368	-1188	55	-262	-219	-103	-1751
5	-1809	-1399	-714	-2558	56	-1884	-1974	-896	-1462
6	-286	-1938	-2346	-2781	57	-741	-1217	-1771	-867
7	-2602	-1881	-1733	-1713	58	-706	-1708	-2750	-2703
8	-745	-299	-2238	-2323	59	-1852	-3353	-13	-2375
9	-437	-2836	-175	-1652	60	-1332	-26	-50	-1750
10	-28	-2477	-2912	-1717	61	-2096	-3245	-2941	-2266
11	-3207	-2446	-1394	-1666	62	-1344	-2181	-719	-2166
12	-1537	-447	-2802	-1574	63	-1651	-3064	-2513	-1632
13	-823	-2530	-1432	-1653	64	-1230	-3636	-1277	-2455
14	-607	-2737	-1628	-3105	65	-502	-438	-392	-786
15	-649	-2329	-171	-2962	66	-4360	-2347	-3135	-1200
16	-2225	-297	-416	-2600	67	-1839	-2082	-2959	-1702
17	-991	-2628	-2857	-1976	68	-1805	-305	-196	-2330
18	-2167	-3062	-2829	-586	69	-213	-1644	-240	-1789
19	-732	-2822	-208	-1501	70	-133	-4081	-1263	-2595
20	-1657	-922	-2723	-1314	71	-3546	-2997	-1297	-2278
21	-1299	-2112	-3147	-2484	72	-2292	-94	-3176	-1266
22	-3284	-1466	-267	-1021	73	-1355	-1380	-2374	-2742
23	-1394	-3636	-3518	-1337	74	-1998	-696	-1857	-2014
24	-2902	-3004	-2196	-3165	75	-1458	-2204	-944	-800
25	-3142	-3013	-2318	-2341	76	-2093	-1165	-755	-734
26	-831	-387	-1415	-1790	77	-1218	-2214	-2412	-2109
27	-3072	-3215	-2249	-2829	78	-1608	-151	-2524	-1972
28	-3166	-837	-3224	-1391	79	-2285	-3461	-3371	-1029
29	-2455	-1792	-2835	-1766	80	-2169	-1646	-1773	-1871
30	-3301	-619	-390	-638	81	-238	-776	-2252	14
31	-72	-784	-3297	-2156	82	-38	-1389	-1278	-984
32	-1590	-815	-732	-1626	83	-2532	-916	-605	-1343
33	-1219	-416	-3036	-263	84	-2599	-2852	-131	-901
34	-1669	-1198	-2235	-2213	85	-640	-146	-432	-2627
35	-1112	-3857	-2305	-154	86	-1860	-1517	-2235	-3431
36	-3684	-55	-3433	-2795	87	-1833	-3801	-2552	-2755
37	-1529	-660	-1617	-3146	88	-2559	-2897	-559	-1440
38	-1375	-252	-3037	-1490	89	-2656	-921	-2621	-718
39	-1802	-3161	-2246	-1926	90	-3937	-1968	-2584	-2027
40	-118	-217	-1389	-120	91	-92	-3039	-1371	-1538
41	-80	-402	-654	-2502	92	-728	-1011	-2996	-1435
42	-801	-1262	-772	-1757	93	-573	-1615	-973	-404
43	-889	-1116	-2973	-1543	94	-528	-1030	-283	-1318
44	-2850	-3399	-2042	-698	95	-952	-725	-107	-2420
45	-332	-760	-1685	-987	96	-3760	-2052	-1313	-74
46	-2893	-2069	-1346	-526	97	-1801	-3064	-698	-1494
47	-120	-661	-1887	-1355	98	-199	-1513	-1138	-903
48	-1018	-489	-641	-1418	99	-1715	-3142	-3138	-360
49	-920	-405	-1065	-722	100	-51	-1319	-2011	-1794
50	-2556	-216	-2372	-2266	avg	-1551.27	-1682.05	-1739.53	-1668.26
51	-501	-1142	-744	-273					

B. COUNTS OF ATTACKER AND DEFENDER ACTIONS TAKEN

Tables 10 and 11 show the counts of the actions taken over all games for both the attacker and defender agents while showing all available actions. Here red means a decrease compared to the baseline runs, green mean an increase, and orange means no change. This data shows significant changes in strategy in the three experiments compared to the baseline.

Table 10. Counts of Attacker Actions Taken in All Test Runs

			Baseline	Decrease	Increase	Neutral
Attacker Action	Cost	Reward	Base	10 Extra	100 Extra	DDOS
Connect to Network	5	-	1	1	1	1
Map Network	5	-	9	12	13	8
Spear-phish Link	10	-	4	5	7	5
Spear-phish Attachment	20	-	3	5	4	5
Exploit Public-Facing Mail	15	-	5	2	1	5
Web Crawl	10	-	5	5	3	4
Exploit Public-Facing App	15	-	7	3	4	6
Discover Local Network	10	-	19	20	27	19
Discover User	10	-	12	24	24	14
Discover Processes	20	-	12	12	14	11
Search Local Data Host	10	-	10	18	26	15
Exfiltrate Local Data	20	-	6	8	14	7
Get User Hash	25	-	7	8	8	7
Get Admin Hash	35	-	4	3	3	5
Process Injection	35	-	3	6	4	5
Get Info	20	-	9	9	11	9
Pass the Hash	30	-	6	6	10	9
Exec WinRm Lateral Movement	20	-	6	7	7	6
Discover Hosts	15	-	7	13	13	7
Search Local Data File Server	10	-	12	7	4	9
Search Local Data Critical Server	20	-	6	7	2	5
Exfiltrate Data Encrypted	35	100-300	1	1	1	1
Exfiltrate C2	25	100-300	1	1	0	1
Critical Service Destruction	25	900	1	1	1	1
Critical Service Corruption	75	1200	0	0	0	0
DDOS Scenario Actions						
Decoy DDOS (Dest./Corrup)	50	1050	0	0	0	4

Table 11. Counts of Defender Actions Taken in All Test Runs

Defender Action	Cost	Linked Action				
Audit Logon Events	15	Exec WinRm Lateral Move, Pass the Hash	16	19	21	14
Monitor Network Traffic	25	Map Network, Exfiltrate Data, Spearphishing	13	13	22	17
Check Firewall Alerts	15	Map Network, Exfiltrate Data, Spearphishing, Web Crawl	12	17	17	12
Audit HIPS Logs	25	Critical Service Destruction/Corruption	14	17	14	12
File Monitoring	20	Search Local Data	14	11	11	8
Process Monitoring	30	Process Injection, Get Hashes	15	12	11	9
Isolate Machine	60	Exfiltrate Data	3	0	0	0
Block IP Address	20	Map Network, Exfiltrate Data, Spearphishing	6	7	6	4
Enforce Robots.txt	20	Web Crawl	8	5	5	4
Disable Email Links	25	Spearphishing Link	7	9	6	3
Disable Admin Account	50	Pass the Hash	3	4	1	2
Disable User Account	50	Exec WinRm Lateral Move, Pass the Hash, Search Local Data, Get Hashes	6	4	2	3
Reset Password	20	Get Hashes	9	8	4	7
Re-Image Machine	90	Process Injection	0	0	0	2
DDOS Scenario Actions						
Terminate excess network connections	50	DDOS Attack	0	0	0	16

C. ATTACKER GAME STEPS

Table 12 shows the number of game steps taken to complete each round of the game. Again, red means a decrease compared to the baseline runs, green mean an increase, orange means no change, and "DNF" means the run did not finish in the 45 steps allowed.

Table 12. Number of Attacker Steps Taken per Game to Complete Each Run.

		Decrease	Increase	Neutral			Decrease	Increase	Neutral
	Attacker Timesteps								
Game Number	Baseline	Scenario 1	Scenario 2	Scenario 3	Game Number	Baseline	Scenario 1	Scenario 2	Scenario 3
1	24	32	40	25	52	19	17	40	30
2	20	32	DNF	30	53	35	18	22	10
3	) NOT FINISH (D	DNF	34	15	54	34	32	DNF	35
4	32	23	9	33	55	14	18	19	23
5	30	22	11	44	56	37	34	24	23
6	20	31	DNF	18	57	25	22	37	8
7	34	32	32	33	58	22	33	39	25
8	12	13	37	11	59	36	DNF	19	23
9	17	DNF	19	DNF	60	28	19	18	32
10	20	29	DNF	DNF	61	35	42	DNF	32
11	44	33	36	DNF	62	29	33	12	17
12	28	21	44	33	63	26	DNF	DNF	DNF
13	20	30	30	35	64	25	45	31	21
14	11	DNF	35	33	65	15	22	16	15
15	22	34	19	31	66	DNF	35	DNF	16
16	32	17	23	21	67	31	33	DNF	22
17	21	41	45	33	68	25	13	25	DNF
18	33	DNF	DNF	DNF	69	20	31	23	23
19	26	36	13	26	70	14	45	39	24
20	28	27	DNF	17	71	DNF	36	30	15
21	23	39	DNF	19	72	35	11	DNF	36
22	DNF	25	20	34	73	24	28	45	37
23	30	DNF	45	24	74	33	21	34	23
24	33	42	DNF	18	75	30	33	25	12
25	44	DNF	DNF	25	76	28	25	24	27
26	22	18	31	20	77	24	32	DNF	18
27	DNF	DNF	DNF	23	78	29	16	36	26
28	36	17	DNF	32	79	32	39	45	12
29	34	32	DNF	16	80	38	32	38	DNF
30	DNF	18	21	21	81	18	25	38	31
31	14	12	DNF	42	82	14	26	32	28
32	35	24	24	26	83	DNF	22	24	24
33	27	12	44	22	84	42	DNF	19	24
34	29	27	39	23	85	21	13	28	32
35	21	DNF	35	17	86	30	28	DNF	19
36	DNF	11	43	DNF	87	26	DNF	38	23
37	28	17	39	30	88	36	DNF	29	21
38	24	8	DNF	33	89	36	25	DNF	32
39	26	DNF	DNF	20	90	DNF	31	40	DNF
40	17	13	36	29	91	16	38	34	11
41	17	15	28	33	92	22	27	DNF	30
42	21	22	26	35	93	16	23	27	31
43	25	28	DNF	32	94	18	26	18	30
44	DNF	DNF	42	8	95	21	22	25	24
45	14	13	28	21	96	DNF	27	32	29
46	DNF	29	33	10	97	26	42	14	28
47	15	23	30	36	98	17	33	28	31
48	25	22	12	26	99	30	DNF	DNF	DNF
49	19	20	9	45	100	17	32	34	23
50	37	15	DNF	33	avg	25.66	26.01	29.27	25.23
51	19	25	24	24		88	84	73	90

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Amin, M. A. R. A., Shetty, S., Njilla, L. L., Tosh, D. K., & Kamhoua, C. A. (2020). Dynamic cyber deception using partially observable Monte-Carlo planning framework. In *Modeling and Design of Secure Internet of Things* (pp. 331–355). IEEE. https://doi.org/10.1002/9781119593386.ch14

Anitha, A., Paul, G., & Kumari, S. (2016). Cyber defense using artificial intelligence. *International Journal of Pharmacy and Technology, 8*(4), 25352–25357.

Balbix. (2021, February 09). Using artificial intelligence in cybersecurity. Retrieved from Balbix.com: https://www.balbix.com/insights/artifical intelligence in cybersecurity

Bai, T., Bian, H., Daya, A. A., Salahuddin, M. A., Limam, N., & Boutaba, R. (2019). A machine learning approach for RDP-based lateral movement detection. *2019 IEEE 44th Conference on Local Computer Networks (LCN),* 242–245. https://doi.org/10.1109/LCN44214.2019.8990853

Bruggen, D., Ferguson-Walter, K., Major, M., Fugate, S., & Gutzwiller, R. (2019). The world of CTF is not enough data: Lessons learned from a cyber deception experiment. *IEEE Workshop on Human Aspect of Cyber Security (HACS)*, 1–8. https://doi.org/ 10.1109/CIC48465.2019.00048

Changwook-Park, & Kim, Y. (2019). Deception tree model for cyber operation. *2019 International Conference on Platform Technology and Service (Platon)*, 1–4. https://doi.org/10.1109/PlatCon.2019.8669410

Chong, W., & Koh, C. K. R. (2018). *Learning cyberattack patterns with active honeypots* [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. https://calhoun.nps.edu/bitstream/handle/10945/60377/18Sep_Chong_Koh.pdf?sequence=1&isAllowed=y

Ciancioso, R., Budhwa, D., & Hayajneh, T. (2017). A framework for zero-day exploit detection and containment. *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*, 663–668. https://doi.org/10.1109/DASC-PICom-DataCom-CyberSciTec.2017.116

Feng, M., & Xu, H. (2017). Deep reinforcement learning-based optimal defense for cyber-physical system in presence of unknown cyberattack. *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, 1–8. https://doi.org/ 10.1109/SSCI.2017.8285298

Gartzke, E., & Lindsay, J. R. (2015). Weaving tangled webs: Offense, defense, and deception in cyberspace. *Security Studies*, 24(2), 316–348. https://doi.org/10.1080/09636412.2015.1038188

Goh, H. (2007). *Intrusion deception in defense of computer systems*. [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. http://hdl.handle.net/10945/3534

Green, J. (2020). *The fifth masquerade: An integration experiment of military deception theory and the emergent cyber domain* [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. http://hdl.handle.net/10945/66078

Kaloudi, N., & Li, J. (2020). The AI-based cyber threat landscape: A survey. *ACM Computing Surveys,* 53(1), 20:1-20:34. https://doi.org/10.1145/3372823

Karuna, P., Purohit, H., Jajodia, R., & Ganesan, R. (2021). Fake document generation for cyber deception by manipulating text comprehensibility. *2021 IEEE System Journal,* 15:835-845. https://doi.org/10.1109/JSYST.2020.2980177

Major, M., Fugate, S., Mauger, J., & Ferguson-Walter, K. (2019). Creating cyber deception games. *2019 IEEE first international conference on cognitive machine intelligence (CogMI)*, Los Angeles, CA, USA 102–111. https://doi: 10.1109/CogMI48466.2019.00023

Matthew, A. (2020). Automation in cyber-deception evaluation with deep learning. Researchgate https://www.researchgate.net/publication/340061861_Automation_in_Cyber-Deception_Evaluation_with_Deep_Learning

Najada, H. A., Mahgoub, I., & Mohammed, I. (2018). Cyber intrusion prediction and taxonomy system using deep learning and distributed big data processing. 2018 IEEE Symposium Series on Computational Intelligence (SSCI), 631–638. https://doi.org/10.1109/SSCI.2018.8628685

Nguyen, T. T., & Reddi, V. J. (2020). Deep reinforcement learning for cybersecurity. ArXiv:1906.05799 [Cs, Stat]. http://arxiv.org/abs/1906.05799

Park, Y., & Stolfo, S. J. (2012). Software decoys for insider threat. *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, 93–94. https://doi.org/10.1145/2414456.2414511

Parks, R. C., & Duggan, D. P. (2011). Principles of Cyberwarfare. *2011 IEEE Security and Privacy Magazine, 9(5):30-35.* https://www.researchgate.net/pulication/224259524_Principles_of_Cyberwarfare

Randhawa, S., Turnbull, B., Yuen, J., & Dean, J. (2018). Mission-centric automated cyber red teaming. *Proceedings of the 13th International Conference on Availability, Reliability and Security*, 1–11. https://doi.org/10.1145/3230833.3234688

Rieck, K. (2011). Computer security and machine learning: Worst enemies or best friends? *2011 First SysSec Workshop*, 107–110. https://doi.org/10.1109/SysSec.2011.16

Ross, D. (2019, March 08). Game theory. Retrieved February 09, 2021, from https://plato.stanford.edu/entries/game-theory/

Roberts, E.. (February, 2021). Strategies of play. Retrieved from https://cs.stanford.edu/people/eroberts/courses/soco/projects/1998-99/game-theory/Minimax.html

Rowe, N. C., & Rothstein, H. S. (2004). Two taxonomies of deception for attacks on information systems. *Journal of Information Warfare*, 3(2), 27–39.

Rowe, N. C., & Rrushi, J. (2016). Introduction to cyberdeception. Berlin: Springer International Publishing.

Shang, T., Dong, H., & Yu, Y. (2019). Research on deep reinforcement learning exploration strategy in wargame deduction. *Proceedings of the 2019 2nd International Conference on Information Systems and Computer Aided Education (ICISCAE),* 622–625.

Sun, J., Liu, S., & Sun, K. (2019). A scalable high fidelity decoy framework against sophisticated cyberattacks. *Proceedings of the 6th ACM Workshop on Moving Target Defense,* 37–46. https://doi.org/10.1145/3338468.3356826

Swarup, S., & Rezazadegan, R. (2019). Generating an agent taxonomy using topological data analysis. *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, 2204–2205.

Swiatocha, T. L. (2018). Attack graphs for modeling and simulating sophisticated cyberattacks [Master's thesis, Monterey, CA; Naval Postgraduate School]. NPS Archive: Calhoun. https://calhoun.nps.edu/handle/10945/59599

Trassare, S., Beverly, R., & Alderson, D. (2013). A technique for network topology deception. *Proceedings of the Military Communications Conference,* https://ieeexplore.ieee.org/document/6735885

Trifonov, R., Manolov, S., Tsochev, G., & Pavlova, G. (2020). Recommendations concerning the choice of artificial intelligence methods for increasing of cyber-security. *Proceedings of the 21st International Conference on Computer Systems and Technologies '20*, 51–55. https://doi.org/10.1145/3407982.3407986

Veith, E. M., Fischer, L., Tröschel, M., & Nieße, A. (2019). Analyzing cyber-physical systems from the perspective of artificial intelligence. *Proceedings of the 2019 International Conference on Artificial Intelligence, Robotics and Control*, 85–95. https://doi.org/10.1145/3388218.3388222

Wilkens, F., Haas, S., Kaaser, D., Kling, P., & Fischer, M. (2019). Towards efficient reconstruction of attacker lateral movement. *Proceedings of the 14th International Conference on Availability, Reliability and Security*, 1–9. https://doi.org/10.1145/3339252.3339254

Yahyaoui, A., & Rowe, N. C. (2015). Testing simple deceptive honeypot tools (I. V. Ternovskiy & P. Chin, Eds.; p. 945803). https://doi.org/10.1117/12.2179793

Zhang, X. (2009). A note on dynamic zero-sum games. *Proceedings from 2009 Chinese control and decision conference.* 4758–4760. https://doi.org/10.1109/CCDC.2009.5194849

Zheng, J., & Castañón, A. (2012). Dynamic network interdiction games with imperfect information and deception. *Proceedings from 2012 IEEE 51st IEEE conference on decision and control (CDC),* 7758–7763. https://doi.org/10.1109/CDC.2012.6425974

Zhu, M., Hu, Z., & Liu, P. (2014). Reinforcement learning algorithms for adaptive cyber defense against Heartbleed. In Proceedings of the First ACM Workshop on Moving Target Defense (pp. 51–58). New York, NY: ACM. doi:doi.org/10.1145/2663474.2663481

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California