The Pennsylvania State University
The Graduate School
College of Engineering

OPTIMAL CYBER-DEFENSE STRATEGIES FOR ADVANCED PERSISTENT THREATS: A GAME THEORETICAL ANALYSIS

A Thesis in
Computer Science and Engineering
by
Jeffrey R. Acquaviva

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

May 2017

The thesis of Jeffrey R. Acquaviva was reviewed and approved* by the following:

Tom LaPorta
William E. Leonhard Professor of Computer Science and Engineering
Thesis Co-Advisor

Mark Mahon
Professor of Computer Science and Engineering
Thesis Co-Advisor

Bruce T. Einfalt
Research Engineer, Penn State Applied Research Laboratory

Mahmut Kandemir
Professor of Computer Science and Engineering
Graduate Program Chair

*Signatures are on file in the Graduate School.

Abstract

This thesis introduces a novel model of Advanced Persistent Threats in a network, and uses game theory to compute the optimal defense strategies to counter these attacks. Performance of equilibrium strategies are compared to other intuitive but sub-optimal strategies as well as their best-responses. Parallels are drawn between the strategies computed by this novel model and the canonical security paradigms of Defense-in-Depth and Perimeter Defense. It is shown that defense-in-depth may not be optimal when the defender is budget constrained. Lastly, two use-cases are presented to demonstrate how this model can be used in real-world scenarios.

Table of Contents

List of Figures

List of Algorithms

List of Abbreviations

APT	Advanced Persistent Threat
ICS	Industrial Control System
IDS	Intrusion Detection System
SCADA	Supervisory Control and Data Acquisition

Chapter 1
Introduction

One of the most challenging tasks in managing a network is defending it against cyber-attack. Attackers have many different skill levels that range from novices to organizations with the financial support of an entire country. Insider attacks are especially dangerous since the defense policy of the organization is readily available to the attacker. Network managers, on the other hand, are severely limited in their ability to counter attacks. Financial budgets may restrict the type of equipment an administrator can purchase to make attack identification and recovery easier. Other budgets, like a user's tolerance of security policies, also exist. It may be unrealistic to ask a group of users to change their password every 30 days, or remember a new lock combination to enter a secure room. Therefore, it is necessary for a network administrator to find the optimal resource allocation that minimizes risk of attack as well as unnecessary burden to its users.

1.1 Real World Threats

Advanced cyber-attackers can use multiple stages and a diverse set of techniques to cover their tracks. Such attackers are called *Advanced Persistent Threats* (APTs) and can lie dormant in a network for months before being noticed. In 2013, Target Corporation was the subject of a cyber attack resulting in the loss of personal information and credit card numbers of 40 million customers [1]. It was later noted that attackers first compromised credentials of the HVAC company that serviced Target's stores [2]. Malicious code was then uploaded to point-of-sale systems where credit card numbers were forwarded to a control server set up to be accessed from the outside [3]. Target was alerted that a breach had occurred only when credit card processors noticed an increase in fraudulent credit card usage from cards that had also been used at Target Inc. [4].

If the concern for personal privacy is not enough, attacks on critical infrastructure pose a more serious threat to a country's safety. In 2010, the world saw how destructive and sophisticated cyber attacks could be with the release of the Stuxnet

virus. Stuxnet was a piece of malware that attempted to destroy nuclear enrichment centrifuges by operating the motors at destructive speeds. This attack took place in several stages. First, certificates were stolen to falsely verify malicious firmware. Then the virus navigated networks that had specific types of industrial equipment connected and established command and control servers to be updated once in place [5]. A year later, a small water utility in Illinois was attacked using similar industrial control intrusion methods [6]. Therefore it is important to consider security policies that generalize to both the physical and cyber domains, and that reduce an APT's success even when detection methods fail.

1.2 Game Theory

Game theory is a method to analyze these security threats that has gained significant traction in the research community. This method examines how players or agents might act when trying to optimize a utility function. These agents might be working together with profits shared among all participants. This case is called a *cooperative game* model. In contrast, a *non-cooperative game* contains participants who seek to optimize their individual utility functions regardless of the utilities of the other players involved.

The payoff structure is another aspect that defines the game model. If one player's increase in utility causes a decrease in utility for the other players by equal amounts, the game is called *zero sum* [7]. For example, in the classic game of rock-paper-scissors, a winner of the game requires that the other player loses by an equal amount. This is in contrast to a *general sum* game where one player doing well does not necessarily mean that other players must suffer [7]. Some games can be represented by repeatedly playing smaller games called *subgames* [7]. When the actions of one subgame dictate the payoffs and the next subgame played with some probability distribution, the game is called *stochastic* [7].

The actions of each player (and therefore their payoffs) are determined by the knowledge each player has of their opponent's strategy. If a player knows the strategy of the opponent, and the opponent knows that their strategy is known ... *ad infinitum*, each player's strategy is said to be *common knowledge*. Choosing an action based on common knowledge provides a defense against an opponent who plays optimally. That is to say, if an agent plays an equilibrium strategy, their

payoff will be no worse than when their opponent plays optimally. This does not mean that the equilibrium strategy will always provide the highest payoff, but rather that it represents the strategy for which the opponent could do no better when knowing their strategy.

In the security domain, a typical model includes two adversarial players: an attacker and a defender. The attacker may attempt to compromise or destroy a target under the defender's control. Therefore, the utility function for an attacker might represent the time spent controlling a resource, or the reward for having destroyed a target. Likewise, the defender might earn a reward for identifying or removing an attacker from inside the network, or for controlling resources. Game theory is useful because the utility functions allow success probabilities to be weighted by the rewards they bring the agent. An attacker action that succeeds with a high probability may not be that desired because it brings the attacker little reward. Game theory allows for a structured reasoning of the actions and rewards available to agents.

1.3 Problem Statement

The model presented in this thesis involves the adversarial interaction of two players: an attacker and a defender. The attacker is the player modeling an APT trying to exfiltrate some amount of data from a network by sequentially compromising the passwords, credentials, or services in the network. The attacker acts stealthily allowing their actions to remain unnoticed by the defender. The defender represents the player who owns and manages the network. They aim to minimize the amount of information the attacker can steal from the network and only has one action: reset. This action represents a generalization of any action that restores a password, credential, or service to a non-compromised state. It is justified by the fact that this is an action that any defender has at their disposal.

One remedy, albeit extreme, for removing malware from a compromised computer is to re-install the operating system. While more advanced malware exists where a simple re-install will not remove it, this thesis assumes that a reset-type action will always return the asset to a non-compromised state. Likewise, compromised passwords can be restored by changing them. Exfiltration of data can be terminated by resetting the outgoing communication path. In the physical domain,

a reset action might be changing the combination of the lock on a door.

There are two potential concerns when limiting the defender to the reset action. First, it promotes the possibility of a denial of service attack by the adversary. Resetting an operating system or other service will cause downtime and cost the defender availability. Second, it imposes a potential burden to users. A user may not tolerate a policy that requires them to change his password every 30 days. Therefore, the defender must be constrained by a budget that represents both the user tolerance and availability requirements. The budget considered in this thesis is an organization wide budget. The defender may mandate that some users reset their passwords more frequently than others according to the optimal defense policy as long as the total frequency of resets does not exceed some budget threshold.

The game model is a zero-sum game where the attacker gets a reward for exfiltrating data located at specific points in the network. This means that the defender loses this amount in reward for each unit of data successfully stolen by the attacker. The game takes place on a directed graph structure that represents the communication links and services of the network. Edges in the graph are the communication paths that the attacker attempts to compromise by compromising passwords or credentials. Nodes represent services, host computers, or entire departments that the attacker must move through to find the rewards in the network. The defender has the ability to reset the nodes in the network according to a certain rate that is constrained by their budget. Likewise, the attacker must choose which communication paths (passwords or credentials) to attempt to compromise as well as for how long. The longer the attacker attempts a compromise, the higher their probability of success. However this also increases the risk of being caught or having their time wasted by the defender resetting the corresponding service. The attacker will pay a penalty if one of the links they attempt to compromise or has already compromised leads to a node reset by the defender. The attacker will continue to attack the network until they successfully exfiltrate some data.

This thesis is organized as follows. In Chapter 2, related work is presented, and aspects that distinguish this thesis from current research is highlighted. Chapter 3 gives the formal problem statement and its analysis. This includes a description of the model used and actions considered for both the attacker and defender. In the analysis, a running example is introduced to analyze algorithms and model behavior. Chapter 4 introduces four simple networks on which the model is simulated and

its behavior analyzed. In Chapter 5, two use-cases are introduced and analyzed. Finally Chapter 6 concludes this thesis by summarizing the results. Future work is presented and applicability to an additional use-case is demonstrated.

Chapter 2
Related Work

The concepts explored in this thesis have been approached by others from three primary directions. Work that explores other game theoretical approaches for defense are first discussed. Next, ways an attacker can move through a network using attack graphs are presented. Lastly, related research from the Autonomic Computing domain that focuses on self-protection is given.

2.1 Game Theory

Game theory differs from the other approaches in that it considers multiple agents acting to maximize or minimize a utility function. In *co-operative games*, these agents work together and payoffs can be shared between all participants. However, most security games examine two adversarial agents: an attacker and a defender. A general survey of how game theory has been used for security can be found in [8].

2.1.1 FlipIt

The game FlipIt was introduced in [9] as an attacker-defender game where players fight for control of a single resource. The game allows each player to take an action that would flip control of the resource in their favor where it remains until their opponent acts. A player never knows when their opponent acts and only learns who controls the resource when they decide to act. Taking an action requires a player to pay a cost. The objective for each player is to maximize control over the resource while minimizing their cost. The authors analyze several different strategies and consider variations of the game where players have a fixed budget, or may pay an additional cost to see who controls the resource.

This work was then extended in [10] to the case of multiple resources where it was given the name FlipThem. The authors consider two cases. One where the attacker receives a reward when they control all resources, and another where they must only control one out of the many resources. The authors mention possible

extensions of their to complex scenarios by combining these two cases.

My work extends both of these models by considering the case where the attacker cannot attack any resource arbitrarily. Their choice of which resource to attack is a function of those resources currently under their control. Furthermore, each node can provide the attacker with a different amount of reward.

2.1.2 Stackelberg Games

Stackelberg games have traditionally been used to model leader-follower interactions of large corporations in an economic setting, but have grown quite popular for their use in the physical security domain. One player, typically the defender, must commit an amount of security resources to protect several targets. The other player, an attacker, gets to see the defender's commitment before they act. These games have been used in numerous physical security domains for creating schedules for guards at LAX Airport terminals [11], Federal Air Marshal for flights [12], and patrol routes for tourist ferries on the New York harbor [13]. A survey of these types of games can be found in [14]. Unlike the work presented in this thesis thesis, those models typically do not take place in continuous time, and attackers are not constrained to attack targets based on what had previously been targeted, i.e., there is no progression of attacks.

2.1.3 Advanced Persistent Threat Modeling

Lateral movement of APTs has been studied in [15]. The authors use a zero-sum game as the basis of a reasoning engine that responds to an attacker's malicious communication paths. It is assumed that the network contains an Intrusion Detection System (IDS) that can identify the currently compromised services in the network. In a similar manner to this thesis, they use a graph structure where nodes represent network services and edges represent communication paths. In contrast to this thesis, the defender selects nodes to disconnect to prevent the attacker from reaching their target nodes. The game continues for a predefined number of stage-games, where in each stage-game, the attacker can choose one node to hop and the defender can choose a number of services to disconnect. The game ends when one of three events occurs: the attacker reaches their target, it is impossible for the attacker to reach the target due to the services the defender, or the number of stage-games has been reached. Note here that disconnected services

are not brought back online, nor is there a service availability requirement for the defender. Therefore the attacker could easily trick the reasoning engine into a denial of service scenario.

The game proposed in [16] studies the use of Defense-in-Depth to counter APTs. The attacker's target is the center node surrounded by a number of other node layers that can be thought of as concentric circles. Each layer consists of a game where the attacker has the option to stay at the current layer or advance one layer further. On the other hand, the defender chooses a node to inspect. Inspecting a node will remove the attacker if compromised. The attacker's reward is a function of the number of layers they have penetrated.

2.2 Attack Graphs

Attack graphs aim to model the paths of vulnerable nodes an attacker might attempt to compromise in a network. In general, each graph consists of a set of possible states in which the attacker might exist. Each state is connected by a potential vulnerability the attacker could exploit. Attack graphs differ from game theory approaches in a few aspects. First, the paths are not typically weighted by their potential reward. Therefore, for each path, only the probability of success is calculated, not the likelihood of compromise.

The Predictive, Probabilistic Cyber Security Modeling Language (P2CySeMoL) [17] is a tool that aims to identify the most likely paths an attacker might attempt to compromise in enterprise networks. It contains a predefined set of asset types (e.g. operating system variants), attacks and defenses, as well as quantitative data on how these objects interact which was collected from a mixture of literature searches and interviews with domain experts. However, tuning of these parameters can be automated with the aid of network scanners. The tool was then evaluated by modeling two Swedish power utilities with the help of their respective owners. Some of the results of running the tool are summarized by the authors as follows: "Reconfigure the rule sets of two critical firewalls ... Train engineers regarding IT security awareness ... Disable USB autorun and web browsing for a few key computers" [17]. This tool primarily seeks to identify weaknesses in a network and suggest additional defenses to counter possible attacks. The defender, in this case, is not constrained. Therefore, this tool aims to identify the missing defenses, whereas

my model assumes that a defender cannot deploy all defenses all the time. Attacks are only modeled by their probability of success. There is no utility function that weights different paths by their expected rewards. Nor does this model consider how an attacker might respond when the suggested defenses are implemented.

Attack graphs can be extended to include defense countermeasures as in [18]. Normal attack graphs only look at options the attacker has and their paths of exploitation through a network. This work proposes coupling these graphs with defender actions for automated response. As with other attack graph based solutions, the probability of attacker paths are not weighted by the rewards they produce.

K-Zero Day Safety is a metric related to attack graphs [19]. The metric is used to compute the number of zero-day exploits that must be used before a given network architecture is compromised. Metrics like these are important in order to evaluate the effectiveness of defense methods. In this thesis, the Attacker's reward could be interpreted as a security metric where lower means more secure.

2.3 Autonomic Computing

In 2001, IBM noted that an increase in complexity would be a growing threat against creating and managing modern networks. Worried that humans would not be able to manage the complexity required by networks of the modern era, they proposed that networks themselves take on this burden. They coined the term *Autonomic Computing* to describe these next generation networks, taking inspiration from a human's autonomic nervous system whose responsibility is to manage key functions of the body. [20]

To better understand how such networks would be constructed, [21] expanded upon the original idea and described four properties that an autonomic system should include. Named the *self-** properties, they include self-configuration, self-optimization, self-healing, and self-protection. Network architects would be responsible for identifying the business level goals of the network, but the network itself should decide how to achieve these goals through the self-* properties.

As the self-healing and self-protection properties are most pertinent to the subject of this thesis, only work related to these properties will be presented. A good survey of papers can be found in [22]. Since autonomic computing attempts to achieve a more general goal than the creation of a robust or resilient system, both

game theory and attack graphs have been used in reasoning engines for autonomic architectures.

An intrusion response system is presented in [23] that relies on IDS support for attack detection. For every sequence of events produced by the IDS, a set of response actions must be identified that would successfully counter the potential attack. The response engine will then select the best action to deploy based on a cost-sensitive analysis. A probability threshold value is used to represent the point at which the sequence of events can be assumed to be malicious. Up until that threshold, the response could preemptively respond to the events if it decides the cost of deploying resources now is less than the cost of a successful attack later. Both the cost estimation of responses as well as the threshold of attack identification are automatically updated after every response to keep the system adaptable.

A trust-based system for attack detection in a multi-agent system is presented in [24]. Here, agents in the network can ask others, or be asked by others, for some computation. Malicious agents may also join the network, so all agents must be aware of who they communicate with. Since agents have the choice of whom they will communicate, providing agents with a trust mechanism allows for the identification of malicious agents by viewing the communication topology.

The system described in [25] is an architecture based on self-cleaning. By periodically cleaning the servers on the network, the time available for malware is significantly decreased. The focus of this research is on a cloud environment where several servers can be used to serve data to users. Cleansing a server inflicts downtime, therefore another server must be started to compensate for the loss of computing power. The number of redundant servers dictates the minimum amount of time each server must be running. When there is more redundancy, the period of time before a server goes down for cleaning will decrease. A central controller is used to determine where incoming traffic will be routed based on which servers are currently running.

In [26], a distributed multi-agent approach for self-protection is described. Each agent in the network has the ability to read and understand the reasoning of another agent. If one agent performs in a manner that is inconsistent with how another agent thinks it should be acting, the other agent will analyze the reasoning processes to ensure there was no compromise. This network of introspection creates

an emergent self-explanation network that can be used to protect against the actions of malicious agents. The authors introduce an ontology that can be used for this meta-level reasoning.

Chapter 3
Problem Statement

Cyber attacks can be modeled as games between attackers and defenders. An adversary attempts to intrude into a network while the system administrator or defender deploys resources to prevent these attacks. In the model presented in this thesis, the attacker attempts to compromise credentials for various nodes in a network. These nodes can represent a computer, a service, or an entire department that must authenticate a user before they are given access. The credentials that are susceptible to an adversary's attacks at any point in time are determined by the network structure as well as the credentials already compromised. This means that they cannot compromise any credential arbitrarily.

Unlike the attacker, who always knows which credentials have been compromised, the defender never knows how many or which credentials are currently in the attacker's possession. To counter the attack, the defender chooses a rate at which to periodically reset all credentials associated with a particular node without knowing where the attacker is.

However, not all attackers are after information kept inside a network. Some seek to prevent access to legitimate users. In addition, not all users have a high tolerance for security. Therefore, the defender must specify a budget that incorporates both the cost of downtime associated with a reset as well as the users' frustrations with needing to change credentials.

The interaction between the attacker and the defender can be modeled as a zero-sum repeated stochastic game. The attacker stochastically moves through different sub-games based on the credentials currently compromised, the credentials attempted, and the probability of success. Since the defender is unaware that the attacker is in the network, the game state remains unknown to them. The defender must find the optimal distribution of budget to minimize loss of data to the attacker.

On the other hand, the attacker must determine which credentials to attempt to compromise. If the defender resets a node while the attacker is attempting to compromise it, the attacker must pay an attempt penalty for wasted time.

Furthermore, the attacker will pay a possession penalty if the defender resets a node for which the attacker has compromised a credential.

The game only ends when the attacker successfully exfiltrates data from the network. This means the adversary will repeatedly attack the network until some data is retrieved.

This chapter is divided into several parts. First, the previous discussion is formalized, introducing the game model, each player's action space and knowledge, and the game's payoff structure. Second, an example network is introduced to demonstrate how concepts generalize to arbitrary networks. Lastly, algorithms are given and analyzed to calculate equilibrium strategies.

3.1 Model

The model takes as input a directed graph $G = (N, E)$ which represents the communication structure of the network. Each directed edge $e \in E$ defines a credential used by a communication link to access some network host, service, node $n \in N$. The set of currently compromised credentials identifies the attacker's current state. This state determines which credentials the attacker can attempt to compromise.

Each edge (i.e. credential) has an associated parameter, λ_X, that specifies its expected rate of compromise. Compromises are assumed to occur according to an exponential distribution. If an attacker chooses to attempt to compromise credential X, they should expect to spend an average of $1/\lambda_X$ time units before the compromise is successful. An attacker cannot attempt to compromise a credential arbitrarily; they must follow the structure defined by the network.

Alternatively, the defender controls nodes in the network and decides how frequently these nodes reset. Resets also occur according to an exponential distribution, and the sum of all reset rates must be below the defender's budget. For example, if the network has two resettable nodes and the defender's budget is 3, then the defender could choose to reset Node 1 at a rate of $\lambda_1 = 0.5$ and Node 2 at a rate of $\lambda_2 = 2.5$. This means that Node 1 will reset according to an exponential distribution with an expected value of 0.5 resets per time unit (i.e. one reset every two time units). Likewise, Node 2 will reset an average of 2.5 times per time unit.

In light of the previous discussion, a stochastic repeated game Γ can be con-

structed from the smaller sub-games that represent each attacker state. The game begins in the state where the attacker has not compromised any credentials.

3.1.1 Players

There are two players in this game. The attacker or adversary seeks to exfiltrate data from the network. Their values will be positive and represent the value gained from exfiltrated data. The defender or system administrator seeks to keep the network free of compromise and wants to minimize the data lost to the attacker.

The attacker always knows the state of the game. That is, they always know what credentials they have compromised and what credentials are available to compromise next. They are not budget constrained. They do not have to choose between compromising one of two credentials on the basis of resources. They can exert equal effort into compromising all possible credentials in a given state. This assumption makes sense in the context of APTs. Since most APTs have the backing of nation-states or nation-state level adversaries, it is reasonable to assume that resources are not an issue.

The attacker will continuously and repeatedly attempt to compromise the network only until they begin to exfiltrate data. Once the attacker starts to exfiltrate data, they do not continue to compromise other credentials. Their reward is the amount of data exfiltrated once exfiltration is possible.

Lastly, credentials are modeled by an exponential probability distribution with respect to time. This means that the longer the attacker attempts to compromise a credential, the higher his probability of success. It also increases the probability that they will be caught and will incur a penalty for attempting a compromise. Details of the different types of penalties will be presented in the following section.

On the other hand, the defender never knows the state of the game. They must determine optimal budget allocation to minimize loss of the network.

3.1.2 Payoffs

This game is analyzed from the perspective of the attacker, where a positive value indicates a positive reward for the attacker. The attacker is the value maximizing player and the defender is the value minimizing player. The defender aims to minimize the information gained by the attacker and therefore prefers negative values. The payoffs in this game can be partitioned according to whom they benefit

most: those benefiting the defender (negative rewards) and those benefiting the attacker (positive rewards).

The attacker will pay a penalty for possessing a node when it is reset. This makes sense from the attacker's perspective for two reason. (i) It represents wasted time. This was effort the attacker had exhausted, but is now useless. (ii) There may be a social cost for being caught inside the network. For example, there may be geopolitical ramifications if one country is found with valid credentials from another country's network. This also makes sense from the perspective of the defender since they aim to keep their network free of compromise. It is clearly undesirable for any part of the network to be compromised, so the defender should aim to keep compromise to a minimum.

The attacker must also pay a penalty for attempting to compromise a credential when a node is reset. From the attacker's perspective, this is because they wasted time and energy. Correspondingly, the defender would like to keep the attacker out. Changing a password after the attacker has exhausted resources could be viewed as a success for the defender. These penalties will be denoted as $att(X)$ and $pos(X)$ for attempting and possessing of credential X, respectively.

Lastly, the attacker receives reward for exfiltrating data from the network. Each node may have some amount of reward stored inside. How this reward is distributed is determined by the network layout and could also be considered part of the defender's strategy. However, this thesis does not examine optimal reward location strategies. This thesis assumes that the the reward distribution is fixed for a particular network.

Each node generates a reward for the attacker at a certain rate $R_N(t)$, where N denotes the node containing this reward and t is the amount of time the attacker has to exfiltrate the data. The only restriction on $R_N(t)$ is that $0 \leq \int_0^\infty R_N(t)dt < \infty$. That is the total reward stored in the node is non-negative and finite. However, this thesis will only consider rates that are monotone-decreasing. For example, the rate $100\ln(2)2^{-t}$ implies that after one time unit, the attacker has received 50 units of reward, and after two time units, the attacker has received a total of 75 units of reward. This continues asymptotically with the attacker receiving 100 units after exfilitrating data for $t = \infty$.

In order for the attacker to receive this generated reward, they must possess a set of credentials that creates a path from this particular node to the outside. Any

node that is reset along this path will end the exfiltration.

The attacker's value for a given pair of attacker-defender strategies can be derived from computing the expected value of the reward received for being in the state where no credentials have been compromised. If the attacker believes that this value is positive, then it makes sense to attempt to compromise the network. Otherwise, they gain nothing for playing this game. This means that the defender can discourage any rational attacker by having a sufficiently large budget, or ensuring that the attacker's penalties are large enough.

3.1.3 Justification

Note that this formulation is general enough to apply to both the cyber and physical domains. In the physical domain, it is not uncommon to have an employee 'badge in,' as well as 'badge out' of a space. Since these identification badges typically have a small microcomputer inside them, it is plausible that different codes are required to to enter and exit a location.

By modeling state as the set of compromised edges, rather than compromised nodes, the defender can easily see both the nodes that are likely to be compromised as well as the direction of the compromise. This can help the defender focus defense resources on those ingress and egress points.

Directed edges are important because communication is not always bidirectional. For example, a SQL server may be susceptible to query injection, but not all servers report the output of commands. The result of the injected command could instead be retrieved by a vulnerability that displays log files.

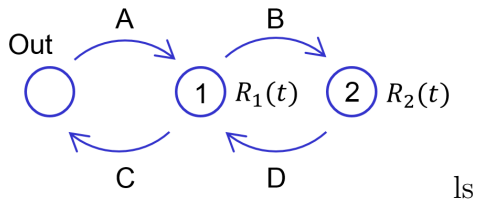
Using the exponential probability distribution for expected attacker rates and defender resets has several advantages. First, it has been shown in [27] that an exponential distribution is a reasonable approximation when attacks take less than 400 days. It was also shown in [9] that an exponential strategy performs moderately well against several other strategy classes in the FlipIt game model, even though it is not strictly dominant.

Lastly, the exponential distribution has the advantage of the memoryless property [28]. That is, the probability of success is independent of the amount of energy previously exerted: $P\{X > s + t | X > t\} = P\{X > s\}\ \forall s, t \geq 0$. This allows for a reduction in the number of states required. Without the memoryless property, the state would also need to represent the amount of time spent previously attempting

a compromise. Take, for example, a state where there are two potential credentials, A and B. Say the attacker attempts to compromise both but compromises A first. Since state transitions occur by events, the attacker is sent into state $\{A\}$. However, from this state, credential B could still be available to compromise. Since the attacker has already spent time t_A attempting to compromise credential B, this effort should be represented in the probability of success to compromise B in this new state, i.e. $P(B|t_A)$. The use of the exponential distribution is a simplifying assumption that allows the the same probability to be used since the likelihood of compromise for a particular credential does not change with effort exerted.

3.2 Analysis

To understand how all the pieces fit together, a running example is introduced. After each stage, the methodology is abstracted to the general case, with algorithms given when necessary. Figure 3.1 shows a network architecture with four edges and three nodes, called the 3n4e Network. Two of the nodes are defender controlled, with Node 1 contains a reward earned at a rate $R_1(t)$, and Node 2 contains a reward earned at a rate of $R_2(t)$.



ls

Figure 3.1: 3n4e Network Architecture

The attacker begins with control of the outside node `out` and can only access the network through compromise of credential A, which occurs with a rate of λ_A. After the successful compromise of edge A, the attacker then has the choice of compromising edge B or edge C. If they choose to compromise edge C and the reward stored at Node 1 is non-zero (i.e. $\int_0^t R_1(x)dx > 0$), then the game enters a terminal state, and attacker ends the game with the amount of information successfully exfiltrated for time t at rate $R_1(t)$ while Node 1 has not yet reset. When such a state is reached, the attacker does not continue to attempt to compromise credentials.

If instead, the defender resets Node 1 before the successful compromise of edge C then two things occur. The attacker pays a penalty for possessing edge A. Possession of credential A is lost, and the attacker is sent back to the state with no credentials compromised.

With the intuition based on the previous discussion, an attacker state graph can be generated. This graph shows the next possible states to which the attacker could advance. The attacker state graph for network 3n4e is shown in figure 3.2

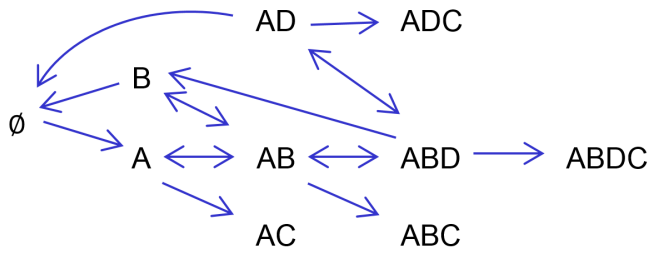


Figure 3.2: Attacker State Graph for 3n4e Network

The attacker state graph unsurprisingly represents the graph of all possible attacker states. Each node in this graph represents a set of compromised credentials, and edges correspond to events that occur during game play. Either a credential is successfully compromised (and the attacker advances to the state with that edge added), or a node is reset. Depending which credential is compromised, a node reset may or may not remove an edge from set of compromised credentials. For example, in the previous network, a Node 2 reset when the attacker is in state A will not cause a state change. At every state, the attacker must decide if they should continue. The attacker can cut their losses and quit early if they believe the cost of continuing outweighs any reward they might eventually receive. In this case, the attacker accepts the losses generated thus far from attempt and possession penalties, and the game ends.

In the general case, Algorithm 3.1 is used to generate attacker state graphs from various communication architectures. The algorithm builds the attacker state graph using a depth-first search approach, and maintaining two lists: states already explored, and states to explore next. Exploring a state means determining what edges can be compromised from this state, which state those edge-events will lead to, and likewise for nodes and node-events. Since terminal states signal the last state of game play, they do not need to be expanded.

Algorithm 3.1 Attacker State Graph Generation

Input: G ▷ Network communication graph G
Output: ASG ▷ Attacker State Graph ASG

```
1: procedure ASG_Generate(G)
2:     Initialize stack, a stack with a state with node out compromised
3:     Initialize a list of completed states, done
4:     while stack is not empty do
5:         s ← stack.pop()
6:         if s is a terminal state then
7:             Move it to done
8:         end if
9:         Create a list of possible edge events
10:        Create a list of possible node events
11:        for all edge events do
12:            Create a new state with this additional edge compromised
13:            if This state does not already exist in done then
14:                Add it to stack
15:            end if
16:        end for
17:        for all node events do
18:            Create a new state with this node and all incoming edges removed
19:            if This state does not already exist in done then
20:                Add it to stack
21:            end if
22:        end for
23:    end while
24: end procedure
```

3.3 Computing State Values

The game starts in the state where the attacker has control of node `out`, but no credentials have been compromised.The attacker must decide if this game is worth playing. If the expected value of being in this initial state is positive, then they would expect to win more then they lose, and they should play the game. However, if the expected value is negative, then it does not make sense to play the game, and a rational player should not attempt to compromise the network. Therefore, the attacker must compute the value of this state to determine if the game is worth playing.

Computation of this value for a particular pair of attacker-defender strategies

can be computed using conditional expectation and the value of all subsequent states. To do this, the set of all possible states are divided into two categories: transition states, where the attacker continues to attempt to compromise credentials, and terminal states, where the attacker generates reward and no credentials are further compromised. In the sequel, the term *event* will be used to describe one of three actions: the attacker successfully compromise of a credential, the attacker giving up a compromise attempt, or the defender resetting a node.

3.3.1 Transition States

To compute the value of a transition state, conditional expectation is used. The value for being in a particular state can be written as the product of the value of advancing to the next and the probability this event occurs, summed over all possible events that could occur in the network. For example, the value of being in the state where no credentials have been compromised, denoted $V_\emptyset$ is $P_\emptyset^A V_A + P_\emptyset^1(att(A) + V_\emptyset)$ where $P_\emptyset^A$ and $P_\emptyset^1$ are the probabilities that credential A is compromised and Node 1 is reset first respectively, $att(A)$ is the penalty for attempting to compromise credential A while it was reset, and V_A is the value of being in state $\{\mathrm{A}\}$. All transition states in the 3n4e network yield the following set of equations:

$$
\begin{aligned}
V_\emptyset &= P_\emptyset^A V_A + P_\emptyset^1(att(A) + V_\emptyset) \\
V_A &= P_A^B V_{AB} + P_A^C V_{AC} + P_A^1\left(pos(A) + V_\emptyset\right) + P_A^2\left(att(B) + V_A\right) \\
V_B &= P_B^A V_{AB} + P_B^1\left(att(A) + V_B\right) + P_B^2\left(pos(B) + V_\emptyset\right) \\
V_{AB} &= P_{AB}^C V_{ABC} + P_{AB}^D V_{ABD} + P_{AB}^1\left(att(D) + pos(A) + V_B\right) + P_{AB}^2\left(pos(B) + V_A\right) \\
V_{AD} &= P_{AD}^B V_{ABD} + P_{AD}^C V_{ACD} + P_{AD}^1\left(pos(A, D) + V_\emptyset\right) + P_{AD}^2\left(att(B) + V_{AD}\right) \\
V_{ABD} &= P_{ABD}^C V_{ABCD} + P_{ABD}^1\left(pos(A, D) + V_B\right) + P_{ABD}^2\left(pos(B) + V_{AD}\right)
\end{aligned}
$$

The probability of transition from the current state to the next state is a function of both the probability that this event occurs first, and the amount of time the attacker is willing to compromise a credential.

Take, for example, the initial state, $\emptyset$. In this state, one of three events could occur: the attacker successfully compromises credential $\mathcal{A}$, the defender resets Node 1, or the attacker gives up before either Node 1 is reset or the credential is compro-

mised. Call the amount of time the attacker is willing to spend on a compromise of credential $\mathcal{A}$ in state $\emptyset$, $t_\emptyset^A$. Let A denote the random variable representing the time at which credential $\mathcal{A}$ is compromised if the attacker spends sufficient time attempting to compromise. Likewise, let N_1 denote the random variable representing time at which Node 1 will reset. Then the probability that the attacker advances to state {A}, $P_\emptyset^A$, can be derived from the three independent events:

$$\begin{aligned}
P_\emptyset^A &= \Pr[\{A < t_\emptyset^A\}\{A < N_1\}\{N_1 < t_\emptyset^A\} \\
&\qquad \cup \{A < t_\emptyset^A\}\{A < N_1\}\{t_\emptyset^a < N_1\}|\{A = a\}] \Pr[A = a] \\
&= \Pr[\{a < t_\emptyset^A\}\{a < N_1\} \left(\{N_1 < t_\emptyset^A\} \cup \{t_\emptyset^A < N_1\}\right)] \Pr[A = a] \\
&= \Pr[\{a < t_\emptyset^A\}\{a < N_1\}] \Pr[A = a] \\
&\qquad = \int_0^{t_\emptyset^A} \lambda_A e^{-a(\lambda_A + \lambda_1)} \\
&\qquad = \left(1 - e^{-t_\emptyset^A(\lambda_A + \lambda_1)}\right) \frac{\lambda_A}{\lambda_A + \lambda_1}
\end{aligned}$$

Where the last equality holds because A and N_1 are exponential random variables that are only defined for $t \geq 0$. A similar equation can be constructed for $P_\emptyset^1$. Note that, in the derivation for $V_\emptyset$, $P_\emptyset^A + P_\emptyset^1$ may be less than one. If the attacker quits before either of these events occur, they receive zero reward. Since the value for this action is zero, the probability term has been dropped for legibility. However, discussion in the subsequent sections will demonstrate that quitting early is a sub-optimal strategy for the class of penalty functions considered, and all state transition probabilities discussed in the sequel will sum to unity.

Complexity of calculating the transition probabilities grows with the number of possible credentials. This is because the possible penalties incurred are determined by which credentials are currently compromised and which ones are attempted. A possession penalty will always be a possibility as long as the attacker attempts at least one credential. However, an attempt penalty is only a possibility if the attacker attempts a credential. To see this, look at state {A}. From this state, the attacker can attempt credential B, or credential C. Therefore P_A^B is determined in part by the strategy component t_A^B and likewise P_A^C, t_A^C. However, a Node 1 event will incur a possession penalty as long as at least one of t_A^B or t_A^C is large enough, and similarly, a Node 2 event will only incur an attempt penalty if the attacker

attempts credential B.

To understand how to compute the probability for a particular event the general case, imagine a timeline that describes the order in which each possible event would have occurred if the state did not change. These orderings include only compromise events X, and node events N_i. Define the *target event* as the event for which the probability is currently being calculated. Each target event has a set of continuation criteria that the attacker's strategy must satisfy in order for this event to occur first. For compromise events, the attacker must still be attempting to compromise the credential when event occurs (i.e $\{X < t_X\}$). The continuation criteria for node events are split into two cases. If the node reset would cause an attempt penalty, then the associated credential must still be attempted (i.e. $\{N_i < t_X\}$ for X an incoming edge to node i). This is the case for Node 1 when the attacker is in state $\emptyset$. An an example timeline for the Node 1 target in state {A} is given in Figure 3.3.

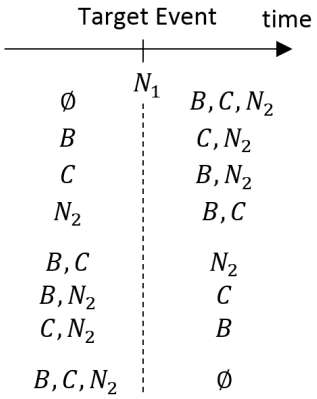


Figure 3.3: Sample Timeline for State {A} and Target Event N_1

This timeline shows all possible orderings of events with respect to the target event. The first row states that if no events occur (i.e., $\emptyset$), then the events B, C, N_2 must occur after the target event. Order within the sub-groups for a particular ordering does not matter. That is, if events B, C, N_2 occur after N_1, all combinations of events (e.g. $B < C < N_2,\ B < N_2 < C,\ C < B < N_2 \ldots$) must be considered and can be simplified by only mandating that these events occur after the target event N_1.

Therefore the set of all orderings is mutually disjoint and their probabilities can be calculated independently and summed. However, due to the continuation criteria, several ordering probabilities will be zero. The continuation criteria for N_1

to impose a possession penalty requires at least one of t^B or t^C occur after N_1. The ordering for which both events B and C occur before N_1 requires that $t^B < B$ and $t^C < C$ in order for N_1 to be possible, however the continuation criteria require that $N_1 < t^B$ or $N_1 < t^C$. This is an impossibility thus the probability of this ordering is zero.

With this methodology in mind, the probability P_A^1 is worked out in full. Note that the orderings where events (B, C), (B, N_2), and (B, C, N_2) occur before N_1 have zero probability and will be omitted for legibility.

$$
\begin{aligned}
P_A^1 = \Pr[&\{N_1 < B\}\{N_1 < C\}\{N_1 < N_2\}\left(\{N_1 < t_A^B\} \cup \{N_1 < t_A^C\}\right) \\
&\cup \{B < N_1\}\{t_A^B < B\}\{N_1 < C\}\{N_1 < N_2\}\{N_1 < t_A^C\}\left(\{N_1 < t_A^B\} \cup \{N_1 < t_A^C\}\right) \\
&\cup \{C < N_1\}\{t_A^C < C\}\{N_1 < B\}\{N_1 < N_2\}\{N_1 < t_A^B\}\left(\{N_1 < t_A^B\} \cup \{N_1 < t_A^C\}\right) \\
&\cup \{N_2 < N_1\}\{t_A^C < N_2\}\{N_1 < B\}\{N_1 < C\}\{N_1 < t_A^B\}\left(\{N_1 < t_A^B\} \cup \{N_1 < t_A^C\}\right) \\
&\cup \{C < N_1\}\{t_A^C < C\}\{N_2 < N_1\}\{t_A^C < N_2\}\{N_1 < B\}\{N_1 < t_A^B\}] \\
&= \int_0^{\max(t_A^B, t_A^C)} e^{-\lambda_B n_1} e^{-\lambda_C n_1} e^{\lambda_2 n_1} \lambda_1 e^{-\lambda_1 n_1} dn_1 \\
&+ \int_0^{\max(t_A^B, t_A^C)} \left(\int_{t_A^B}^{n_1} \lambda_B e^{\lambda_B b} db\right) e^{-\lambda_C n_1} e^{-\lambda_2 n_1} \lambda_1 e^{-\lambda_1 n_1} dn_1 \\
&+ \int_0^{\max(t_A^B, t_A^C)} \left(\int_{t_A^C}^{n_1} \lambda_C e^{\lambda_C c} dc\right) e^{-\lambda_B n_1} e^{-\lambda_2 n_1} \lambda_1 e^{-\lambda_1 n_1} dn_1 \\
&+ \int_0^{\max(t_A^B, t_A^C)} \left(\int_{t_A^C}^{n_1} \lambda_2 e^{\lambda_2 n_2} dn_2\right) e^{-\lambda_B n_1} e^{-\lambda_C n_1} \lambda_1 e^{-\lambda_1 n_1} dn_1 \\
&+ \int_0^{\max(t_A^B, t_A^C)} \left(\int_{t_A^C}^{n_1} \lambda_C e^{\lambda_C c} dc\right) \left(\int_{t_A^C}^{n_1} \lambda_2 e^{\lambda_2 n_2} dn_2\right) e^{-\lambda_B n_1} \lambda_1 e^{-\lambda_1 n_1} dn_1 \\
&= \frac{\lambda_1 \left(1 - e^{-\max(t_A^B, t_A^C)(\lambda_B + \lambda_C + \lambda_1 + \lambda_2)}\right)}{\lambda_B + \lambda_C + \lambda_1 + \lambda_2} \\
&+ \int_0^{\max(t_A^B, t_A^C)} \left(e^{-\lambda_B t_A^B} - e^{-\lambda_B n_1}\right) e^{-\lambda_C n_1} e^{-\lambda_2 n_1} \lambda_1 e^{-\lambda_1 n_1} dn_1 \\
&+ \int_0^{\max(t_A^B, t_A^C)} \left(e^{-\lambda_C t_A^C} - e^{-\lambda_C n_1}\right) e^{-\lambda_B n_1} e^{-\lambda_2 n_1} \lambda_1 e^{-\lambda_1 n_1} dn_1 \\
&+ \int_0^{\max(t_A^B, t_A^C)} \left(e^{-\lambda_2 t_A^C} - e^{-\lambda_2 n_1}\right) e^{-\lambda_B n_1} e^{-\lambda_C n_1} \lambda_1 e^{-\lambda_1 n_1} dn_1 \\
&+ \int_0^{\max(t_A^B, t_A^C)} \left(e^{-\lambda_C t_A^C} - e^{-\lambda_C n_1}\right) \left(e^{-\lambda_2 t_A^C} - e^{-\lambda_2 n_1}\right) e^{-\lambda_B n_1} \lambda_1 e^{-\lambda_1 n_1} dn_1
\end{aligned}
$$

In general, Algorithm 3.2 can be used to construct a function that will be integrated to compute the probability for the possession penalty when the target event is a node reset. Similar algorithms can be constructed for the other cases.

Algorithm 3.2 Possession Penalty Probability Generation

Input: A target event and all possible events in a state
Output: The probability that event leads to a possession penalty

1: Construct the powerset, $\mathcal{P}$ of all non-target events
2: **for all** $s \in \mathcal{P}$ **do**
3: **for all** events in s **do**
4: Let $\hat{t}$ be the variable of integration
5: Add the term $(CDF(u) - CDF(l))$
6: where CDF is the cumulative distribution function
7: **if** event is a compromise event **then**
8: $l = t_E$ for E this edge event ▷ the lower bound
9: $u = \max(\hat{t}, t_E)$ ▷ the upper bound
10: **else** ▷ Event is a node reset
11: $l =$ maximum of all incoming edges
12: $u = \max(\hat{t}, l)$
13: **end if**
14: **end for**
15: **for all** events in s^c **do**
16: Add the term $(1 - CDF(\hat{t}))$
17: **end for**
18: Add the term $PDF(\hat{t})$
19: where PDF is the probability density function.
20: Integrate the product of all terms added with respect to $\hat{t}$
21: **end for**
22: **return** The sum of all integrations performed

The algorithm presented in Algorithm 3.2 will calculate the probability for the general case. However, when probabilities are restricted to the exponential distribution, the equations are greatly simplified. This will be described in Section 3.3.3. The next section describes how to compute the value for a terminal state.

3.3.2 Terminal States

The value of a terminal state is derived from the reward nodes and credentials along the exfiltration path. There are four terminal states in the attacker state graph representation of the 3n4e network: $\{AC\}$, $\{ABC\}$, $\{ACD\}$, and $\{ABCD\}$.

State $\{AC\}$ only has one node compromised, and data is exfiltrated only along edge C. Therefore, data will be exfiltrated through this path as long as Node 1 has not yet reset. The expected value of this state is then:

$$V_{AC} = \int_0^{t_{AC}} e^{-t\lambda_1} R_1(t) dt$$

Where t_{AC} is the amount of time the attacker continues to exfiltrate data, and the term $e^{-t\lambda_1}$ is the probability that Node 1 has not yet reset. Since there are no more penalties for being caught at this point, this function is monotone-increasing and the optimal strategy for the attacker in this state is $t_{AC} = \infty$.

Correspondingly, the value for the state $\{ABCD\}$ is as follows.

$$V_{ABCD} = \int_0^{t_{ABCD}} e^{-t\lambda_1} R_1(t) + e^{-t(\lambda_1+\lambda_2)} R_2(t) dt$$

Since the reward stored in Node 2 must pass through Node 1, the amount that can be exfiltrated is dependent on probability that both Node 1 and Node 2 have not yet reset. However, as in the case of state $\{AC\}$, the reward stored in Node 1 only depends on Node 1.

In general, the reward for a node i in a terminal state can be written as:

$$\int_0^t (1 - F(t)) R_i(t) dt$$

Where the attacker continues exfiltrating for time t and F is the cumulative distribution function for the path (i.e. the probability exfiltration has stopped at time t due to a node reset).

It should be noted that a node can have multiple paths through which reward can be exfiltrated. Information will be exfiltrated as long as at least one path has not yet been reset. The cumulative distribution (CDF) of this event can be defined as follows. Denote the CDF of a node i reset by F_i. Let W_j be the event that path j, made up of of nodes $N_i, 0 \leq i \leq m_i$ is reset. Then $W_j = \min(N_i)\ \forall i \in 0 \ldots m_j$, and $CDF_W(t) = \Pr[W_j \leq t]$. Likewise, let Z denote the event that at lest one path has not yet been reset. Then $Z = \max(W_j)$, and $CDF_Z(t) = \Pr[Z \leq t]$. Therefore, for a node with M possible exfiltration paths can be expressed in terms of each

node's CDF as follows:

$$\Pr[Z \leq t] = \Pr\left[\bigcap_{j=0}^{j=M} \{W_j \leq t\}\right] = \Pr\left[\bigcap_{j=0}^{M} \bigcup_{i=0}^{m_j} \{N_i \leq t\}\right]$$

Note that the events $\{W_j \leq t\}$ are not independent because a node may be involved with more than one path. However, the events $\{N_i \leq t\}$ are independent since node resets are assumed to occur independently. This implies a naïve approach to computing $\Pr[Z \leq t]$ by repeated use of the principle of inclusion-exclusion. Recall that the principle of inclusion-exclusion states the probability of dependent events can be evaluated by adding and subtracting combinations of subsets of events. For the case of three events A_1 and A_2, the principle states:

$$\begin{aligned}\Pr[A_1 \cup A_2 \cup A_3] = \Pr[A_1] + \Pr[A_2] + \Pr[A_3] - \Pr[A_1 \cap A_2] \\ - \Pr[A_1 \cap A_3] - \Pr[A_2 \cap A_3] + \Pr[A_1 \cap A_2 \cap A_3]\end{aligned}$$

This expression can be rearranged to define the intersection of all events in terms of the intersection of a smaller number of events and the union of all events. The smaller subsets of intersection pairs can be computed by another rearrangement of the principle. However, this approach becomes intractable as the number of paths and nodes in each path increases.

Instead, note that probabilities of unions are easier to compute probabilities intersections. For example, consider a reward node has two paths, W_1 and W_2, where path W_1 depends on nodes N_1, N_2, N_3, and path W_2 depends on nodes N_1, N_4:

$$\Pr[Z \leq t] = \Pr[\max(W_1, W_2) \leq t] = \Pr[\max(\min(N_1, N_2, N_3), \min(N_1, N_4)) \leq t]$$

However:

$$\begin{aligned}\Pr[\min(W_1, W_2) \leq t] &= \Pr[\min(\min(N_1, N_2, N_3), \min(N_1, N_4)) \leq t] \\ &= \Pr[\min(N_1, N_2, N_3, N_4) \leq t] \\ &= 1 - \prod_{i=1}^{4} (1 - F_i(t))\end{aligned}$$

Where the last equality holds because all node resets are independent.

Therefore, a version of the principle of inclusion-exclusion that defines the probability of the intersection of events in terms of the probabilities of unions is used.

Lemma 3.3.1 (Principle of Inclusion-Exclusion for Intersections)**.**
For events $A_i, i \in \{1 \dots n\}$*:*

$$\Pr\left[\bigcap_{i=1}^{n} A_i\right] = \sum_{i=1}^{n} \Pr[A_i] - \sum_{i<j} \Pr[A_i \cup A_j] + \sum_{i<j<k} \Pr[A_i \cup A_j \cup A_k] - \dots + (-1)^{n+1} \Pr\left[\bigcup_{i=1}^{n} A_i\right]$$

Proof by Induction.

Base Case: $n = 2$

$$\Pr[A_1 \cap A_2] = \Pr[A_1] + \Pr[A_2] - \Pr[A_1 \cup A_2]$$

This is true by the normal definition of the principle of inclusion-exclusion.

Induction Step: Assume true for case n, show it also holds for $n + 1$. The inductive hypothesis states that:

$$\Pr\left[\bigcap_{i=1}^{n} A_i\right] = \sum_{i=1}^{n} \Pr[A_i] - \sum_{i<j} \Pr[A_i \cup A_j] + \dots + (-1)^{n+1} \Pr\left[\bigcup_{i=1}^{n} A_i\right]$$

Take the left hand side and add one more event, A_{n+1}:

$$\Pr\left[\left(\bigcap_{i=1}^{n} A_i\right) \cap A_{n+1}\right] = \Pr\left[\bigcap_{i=1}^{n} A_i\right] + \Pr[A_{n+1}] - \Pr\left[\left(\bigcap_{i=1}^{n} A_i\right) \cup A_{n+1}\right]$$

Where equality holds from the base case with $A_1 = (\bigcap A_i)$ and $A_2 = A_{n+1}$. Invoke the inductive hypothesis to expand $\bigcap A_i$ twice on the right hand side:

$$\Pr\left[\left(\bigcap_{i=1}^{n} A_i\right) \cap A_{n+1}\right] = \sum_{i=1}^{n+1} \Pr[A_i] - \sum_{i<j} \Pr[A_i \cup A_j] + \dots + (-1)^{n+2} \Pr\left[\bigcup_{i=1}^{n+1} A_i\right]$$

QED

With Lemma 3.3.1, Algorithm 3.3 can be used to construct the probability function to be integrated with the reward function. In a similar manner to Algorithm 3.2, this algorithm creates the cumulative distribution function by summing terms for each subset of a powerset. Given a set of exfiltration paths, Algorithm 3.3 iterates over the powerset of paths and adds each partial probability according to the principle of inclusion-exclusion.

Algorithm 3.3 Reward Node Exfiltration Probability

Input: The set of all paths from a given reward node to the outside
Output: The CDF of the maximum of all such paths

```
1: Compute the powerset of all paths excluding the empty set, 𝒫
2: Initialize l: a list of CDF functions; one for each s.
3: for all s ∈ 𝒫 do
4:     if s has even cardinality then
5:         Add the negative of the product of the CDF of all nodes in this path
6:     else
7:         Add the product of the CDF of all nodes in this path
8:     end if
9: end for
10: return The sum of all sign adjusted products in l
```

3.3.3 Optimality of the $0 - \infty$ Strategy

In this section, it is shown that the attacker's value for any state is maximized for either $t = 0$ or $t = \infty$. That is, if an attacker should choose to attempt a compromise, they should only stop when the compromise is successful or when the node resets.

This is easily seen in the terminal states. In these states, there is no penalty for exfiltrating more data. The value of these states are always monotone increasing with respect to t. Therefore, the value is always maximized for $t = \infty$. This means that the value for terminal states can be computed in advance.

For transition states, this thesis only considers constant penalty functions (e.g. $pos(X) = -0.5$). Note that the transition probability functions are increasing with respect to t; the more time the attacker spends attempting a compromise, the more likely they will either succeed or be caught. A attacker should only attempt a compromise if it leads to a state with positive reward value. An attacker can

choose states that have only non-negative reward values because they always have the option of quitting and receiving zero reward.

Mathematically, this can be seen by taking the derivative with respect to t to see where values are maximized. Recall that transition probabilities have the form:

$$\frac{\lambda_i}{\sum \lambda_j}\left(1 - \exp(-t \sum \lambda_j)\right)$$

The derivatives with respect to t will then have the form:

$$\beta \exp(-t \sum \lambda_j)$$

for some positive constant β. Note that the derivative is positive for $t \geq 0$. This means that the value for each state has extrema only at the bounds. The bounds for this case are 0 and ∞. Therefore, when the penalty functions are constant, the attacker strategy represents the states and credentials to attempt, not how long to attempt to compromise.

3.3.4 Computing the value for a particular strategy pair

As was previously discussed, the value for a particular pair of strategies is equivalent to the value of state $\emptyset$. The value of this state, as well as all other transition states, was computed using conditional probability. While this is mathematically true, this formulation leads to some issues computationally. For example, in the 3n4e network, the value of state $\emptyset$ depends on the value of state $\{A\}$, and the value of state $\{A\}$ depends on the value of state $\emptyset$. These recursive relationships exist manifold across the states. While it is theoretically feasible to symbolically define closed form solutions for sets of equations for arbitrary networks, it becomes rapidly intractable.

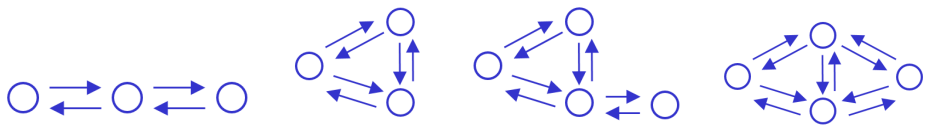
Therefore, each state value can be computed using a fixed-point iteration method. Each state value equation represents one row in a vector valued function $\vec{g}(v)$. The element $v_\emptyset$ at the fixed point $\vec{v^*} = \vec{g}(\vec{v^*})$ is the solution to the system of equations.

Chapter 4
Simulation and Analysis of Simple Networks

In this chapter, simple networks are introduced. From these networks, the performance of equilibrium strategies is compared to the performance of suboptimal strategies as well as the best response strategies for the sub-optimal strategies. Lastly, this chapter examines how the canonical security paradigms of Defense-in-Depth and Perimeter Defense come into play with respect to the equilibrium strategies.

4.1 Simple Networks

Four simple networks of increasing complexity are used to analyze how equilibrium strategies compare to sub-optimal strategies. These networks are listed in Figure 4.1. For each network, the outside attacker-controlled node is the left most node of the network. Attacker rates for each edge are chosen uniformly at random from integers in the interval [1,5], and a defender budget was fixed at 3. While these parameters were arbitrarily chosen, their impact will be studied in the following sections.



(a) 3n4e (b) 3n6e (c) 4n8e (d) 4n10e

Figure 4.1: Simple Communication Networks of Increasing Complexity

Each network is named by the total number of nodes and edges. As an example, the network 3n4e as three nodes and four edges. While there exist other network configurations to which these names apply, this thesis will only examine one particular instance of each class as demonstrated in Figure 4.1. Therefore, the names are unique in the context of this thesis.

4.2 Sub-Optimal Strategies

Several sub-optimal strategies were chosen for both the attacker and the defender. These strategies are used as a baseline to compare the performance of the equilibrium strategies. In addition, if a player knows their opponent plays with a sub-optimal strategy, they can optimize against that particular strategy. This class of strategies will be called *Best Response* (BR) strategies. The performance of the equilibrium strategies is also compared to that of this class of BR strategies.

By comparing the performance of sub-optimal, BR, and equilibrium strategies, the importance of player knowledge is demonstrated. This chapter shows how one player can improve their reward if they know and optimize against their opponent's strategy. For the attacker, the following sub-optimal strategies are considered:

All The attacker never quits. If a credential can be compromised, the attacker will attempt to do so.

Fast The attacker never quits. They only choose the one credential with the fastest expected compromise time to attempt. If this rate is tied with another, both are attempted.

Short The attacker compromises credentials along the shortest path to a particular reward node. Shortest path is defined by the fastest rate of compromise.

Marginal Reward (MR) Average reward per state is calculated. Only those links that lead to states with the highest reward are attempted. For example, if compromising a credential where three terminal states are possible with an average reward of 40 units each, but attempting another credential will lead to a state where two terminal states each with an average 50 units of reward, then the later credential will be chosen.

Weighted Marginal Reward (WMR) The same as Marginal Reward, only the rewards are weighted by the expected compromise weight.

Random An arbitrary but fixed strategy

These sub-optimal attacker strategies were selected for their somewhat intuitive appeal. That is, an attacker that has not performed any optimization calculations might think that these are intelligent strategy choices. The random strategy

was included to provide a baseline for sub-optimal strategies; if no thought went into a strategy, how might the rewards compare. The `fast` and `short` strategies represent two different opportunistic approaches. The `fast` strategy can be thought of a smash-and-grab approach where the attacker quickly attempts to grab as much information as they can readily get. On the other hand, the `short` strategy represents a calculated and targeted approach. The attacker has identified one node contains the reward they most value. They take a targeted approach to attack that specific node by compromising credentials in a shortest-path manner with respect to expected compromise times.

The strategies `MR` and `WMR` represent a more holistic approach. The attacker knows their capabilities and attempts to seek out those states for which their reward is maximized. In a similar manner, four sub-optimal strategies are considered for the defender:

Proportional The defender distributes their budget proportionally according to the total reward located at each node

MaxN The defender distributes their budget equally among all N-most nodes containing the valuable data

Equal The defender distributes their budget equally among all nodes

Random The defender arbitrarily distributes their budget

As was the case for the attacker strategies, a fixed but arbitrary strategy `random` provides a baseline to see how other sub-optimal strategies perform. It should be noted that the the `MaxN` strategy is the same as `equal` when N is all the defender resettable nodes.

4.3 Simulation

Simulations for the entire game are performed by simulating each state. The attacker moves through the states according to their strategy, collecting penalties until the game ends with them quitting or collecting their reward. The time until an event occurs is represented by a sampling of the exponential distribution. In each state, all possible events are ordered by the time at which they will occur. The

event occurring first will determine the next state, and this time will be subtracted from all remaining events.

To make this more concrete, consider a case where the attacker attempts to compromise two credentials, A and B. These credentials lead to two different nodes, N_1 and N_2 respectively. If this is the first time the attacker has attempted credentials A, and B, then new values, t_A and t_B, are sampled for each credential with parameters λ_A and λ_B respectively. These values represent the true amount of time required to compromise each credential. Likewise, N_1, and N_2 are sampled at rates λ_1, and λ_2 to determine the amount of time (t_1 and t_2) until these nodes reset. Say, for example, the ordering $t_A < t_1 < t_2 < t_B$ occurs. Then the simulator transitions into the state with credential A added, and remaining times are updated as follows: $t_1 \leftarrow t_1 - t_A$, $t_2 \leftarrow t_2 - t_A$, and $t_B \leftarrow t_B - t_A$. In this new state, N_1, N_2, and B are not sampled again because they still have time left. However, if this state exposes a new potential credential, C, that the attacker would like to attempt, then C will be sampled with parameter λ_C. In this state, all events are reordered, with t_1, t_2, t_B as their previously updated values and t_C a newly sampled variable.

Note that simulations do not take advantage of the simplifying assumption of the memoryless property. This allows the simulation engine to use any probability distribution for node resets or credential compromises. In addition, since time is recorded, the expected time of compromise can also be calculated. Therefore, it can be seen how the length of time an attacker is in the network corresponds to the average amount of reward they receive.

In addition to providing insights about the time-reward trade-off, simulation data also shows the distribution of rewards for a particular strategy pair. Because the game is stochastic, the value received for any instance of attack is a sampling of distribution of rewards defined by both players' strategies. For the results given in the following sections, the quartiles (as calculated from 10,000 simulation runs) of this distribution will be given for a particular strategy pair. Lastly, simulations are also used to verify the algorithms used to calculate the expected value as described in Chapter 3

In the rest of this section, simulation results and calculated values are presented in graphs like Figure 4.2. Attacker strategies identified by groups on the x-axis and defender strategies are identified by colors. For each pair of strategies, the bar shows the mean value as calculated with the algorithms as described in Chapter 3,

and the simulated quartiles are shown on the error lines.

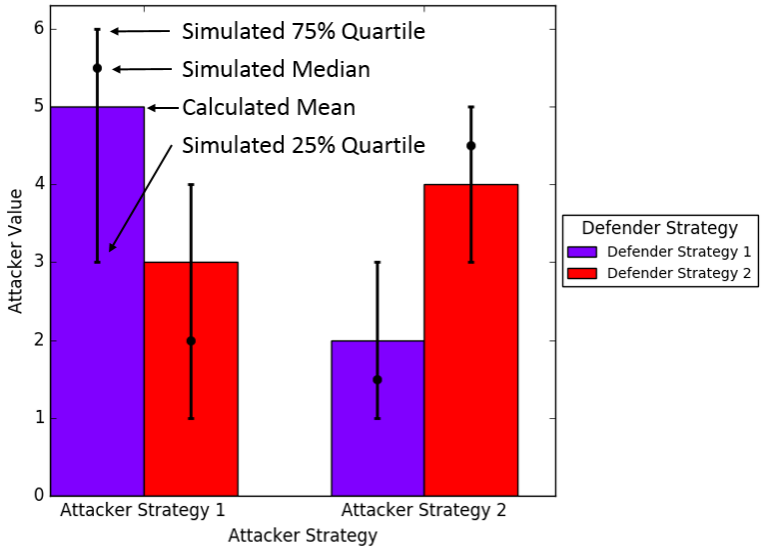
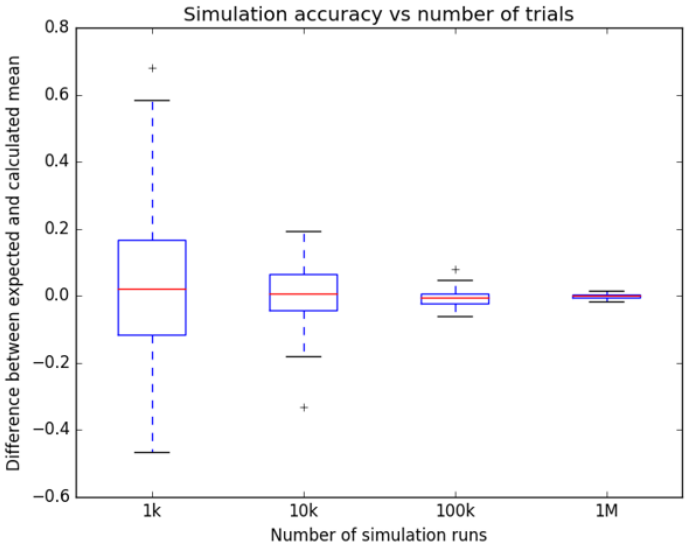


Figure 4.2: Sample Graph Demonstrating Simulated and Calculated Data

The decision to use 10,000 simulation runs for each strategy pair is justified by the tradeoff between simulation time and accuracy as demonstrated in Figure 4.3. The calculated and simulated means were computed for 49 pairs of strategies for several different number of simulation trials. The differences between the calculated and the simulated means are shown in Figure 4.3a. This plot demonstrates two important points. First, it verifies the correctness of the algorithms derived in Chapter 3. Second, it shows the diminishing returns in the number of simulation runs. As the number of simulation runs increases by a factor of 10, the spread of differences decreases only by about half.

The execution times shown in Figure 4.3b were gathered from an Intel Core Duo with a 2.66 GHz clock. Note that a tenfold increase in the number of simulation trials corresponds with a tenfold increase in execution time. This linear increase makes sense because simulations were performed sequentially. Therefore, using 10,000 trials to simulate a pair of strategies offers a good trade-off between accuracy and runtime.



(a) Accuracy as a Function of the Number of Trials

(b) Simulation Time as a Function of Number of Trials

Figure 4.3: Tradeoff between Accuracy and Time

4.3.1 3n4e Network

An instance of the 3n4e network is presented in Figure 4.4. In this network, $\lambda_A = 4$, $\lambda_B = 2$, $\lambda_C = 4$, $\lambda_D = 3$, $R_1(t) = 30\ln(2)2^{-t}$, and $R_2(t) = 40\ln(2)2^{-t}$. The defender budget is fixed at 3.

The first simulation run demonstrates how the equilibrium strategies perform

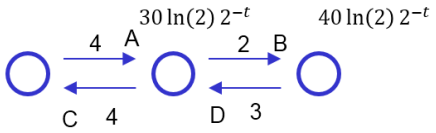


Figure 4.4: 3n4e Network Architecture Instance

compared to the defender's other best response strategies. Figure 4.5 shows the results after 10,000 simulation trials. The bar graph shows the calculated expected value. The error bars show the distribution of rewards for each of the runs. The ends of the bar show the lower and upper quartiles, and the dot is the median.

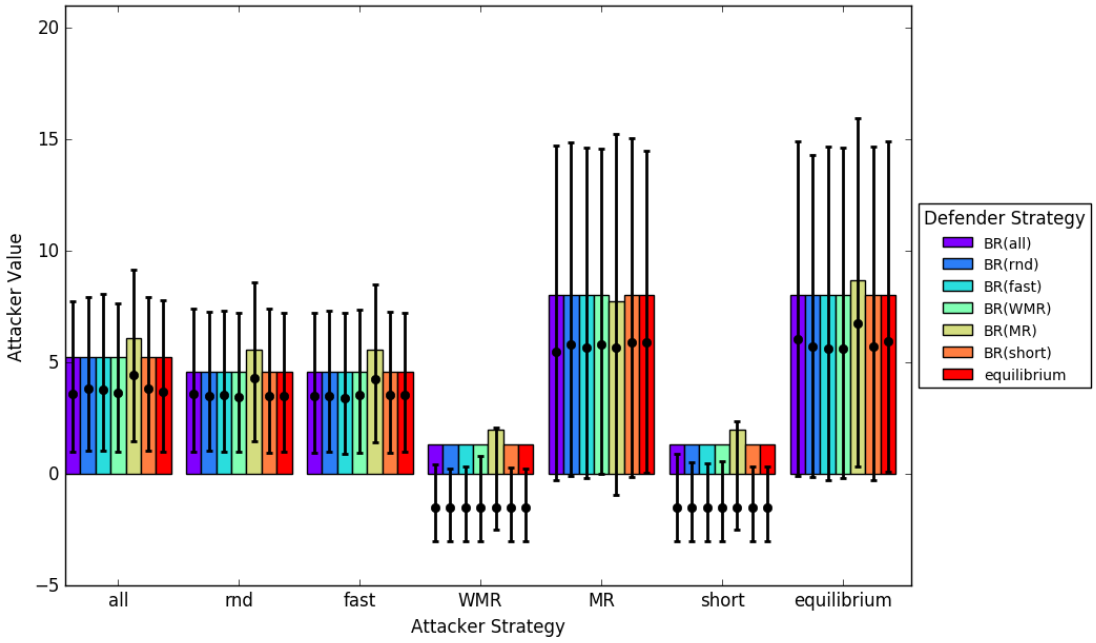


Figure 4.5: 10,000 Simulation Trials for Different Defender Best-Response Strategies

The colored bars in the figure demonstrate two important concepts. First, the defender's best response to any of the attacker's sub-optimal strategies is approximately the same as their equilibrium strategy. For a particular attacker strategy, there is not much variation in attacker value across the different defender strategies. Although the advantages for the defender playing `equilibrium` are not obvious in this particular network, it will become more apparent later in this section. This suggests that a defender who optimizes against any particular attacker

strategy, when the network is simple enough, might reasonably approximate the equilibrium strategy.

Second, the attacker's value for playing `MR` is about the same as the value received for playing the equilibrium strategy. Note that these two strategies dominate the other attacker strategies. Regardless of how the defender plays, the attacker receives the highest values when they play either `MR` or `equilibrium`. However, a rational attacker should play `equilibrium` because playing `MR` would potentially allow the defender to optimize against this strategy, as seen by the small dip in value for the yellow bar.

The quartile lines for each strategy pair show an interesting relationship between means and the medians. The medians are consistently lower for this network. In the extreme case, `WMR` and `short` have positive expected values, but negative medians. This means that 50% of attackers would earn a negative reward if they attempted to play the game with those strategies, but if they played enough times an attacker would come out ahead. It is interesting to note that an attacker who's rationality depends on the mean should play this game, but one who's rationality depends on the median should not. However, this discussion is left for future work as the computation of the median, in the general case, is significantly more complex and does not have guarantees for uniqueness [29].

The skewness of the attacker's value distribution for the pair of strategies, `WMR` and `BR(WMR)`, can be seen in Figure 4.6. On expectation, the high rewards that occur with low probability are able to make up for the numerous low rewards with high probability. This shows that the game has an element of high-risk, high-reward.

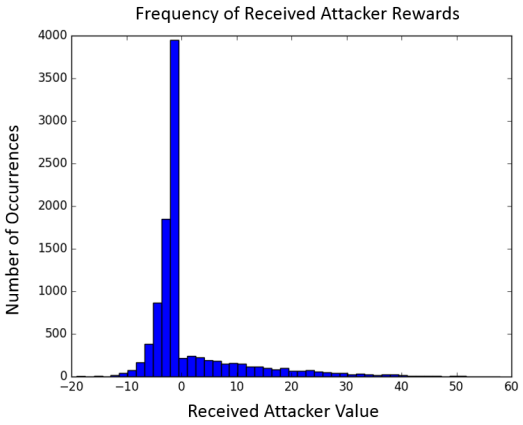


Figure 4.6: Distribution of Rewards for `WMR` Vs. `BR(WMR)` Strategy Pair

The previous figures showed the interaction between the a sub-optimal attacker and a defender who knows the attacker's strategy and plays their best response. Both the attacker's and defender's equilibrium strategies were given for comparison. On the other hand, Figure 4.7 shows how a baseline comparison between an attacker and defender who both play sub-optimal strategies with the equilibrium strategies again given for reference.

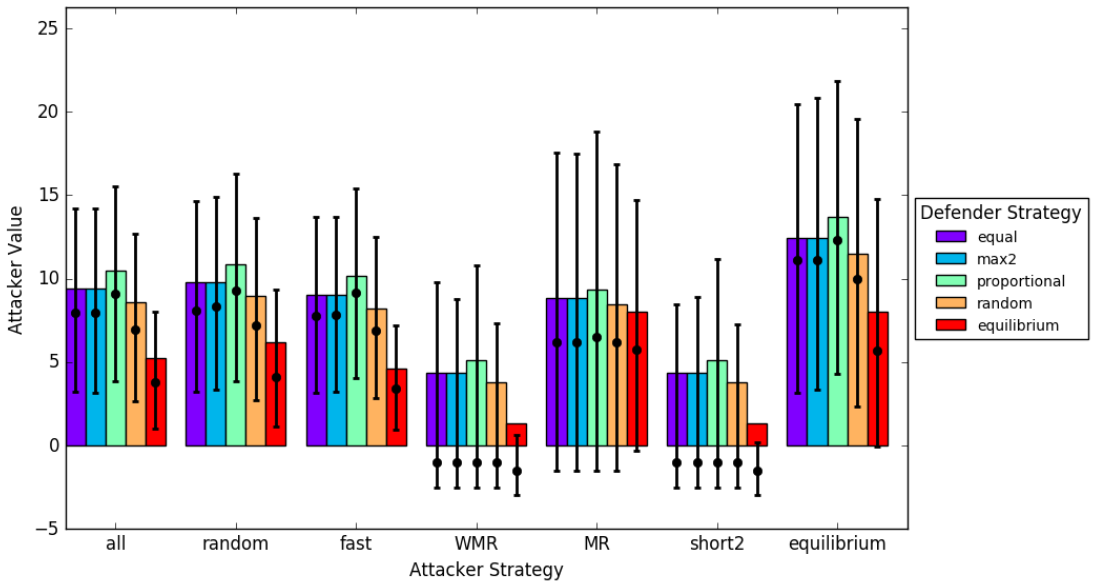


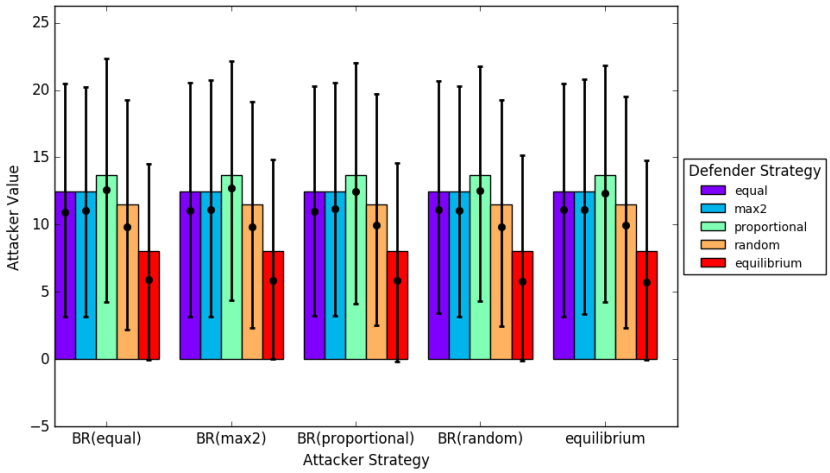
Figure 4.7: Performance of Sub-Optimal Strategies for Both Attacker and Defender

Some of the same trends previously seen in Figure 4.5 for a best-response defender also hold for a sub-optimal defender. The attacker's strategies of `all`, `random`, and `fast` all perform approximately the same. This is most likely because of the simplicity of the network. There are only a handful of states the attacker can decide to play, so for this network, these strategies are almost identical. The strategies of `WMR` and `short` are dominated by the other attacker strategies. However, the performance of `equilibrium` is more easily seen. If the defender plays sub-optimally, the attacker's strategy of `equilibrium` will produce significantly more reward.

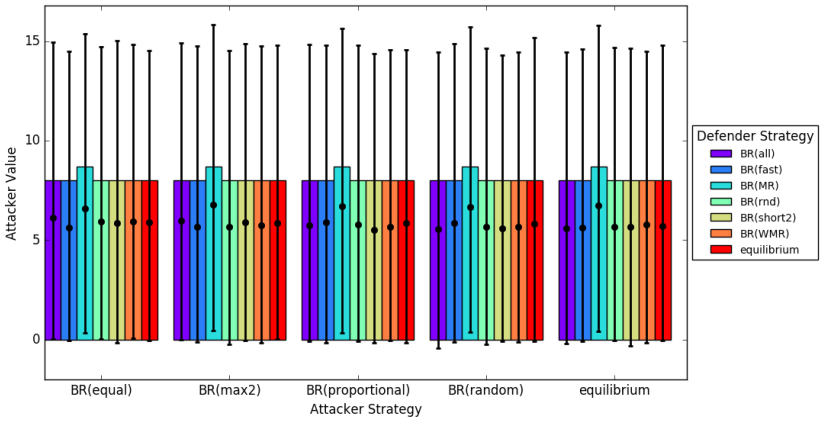
From the defender's perspective, their strategy of `equilibrium` performs the best regardless of the attacker strategy. That is, the red bar is the lowest of all attacker strategy groupings. The two strategies, `equal` and `max2`, are the same for

this graph. This is why Figure 4.7 shows these strategies yield the same value. It is interesting to note that `proportional` is the worst sub-optimal strategy. Although it might be counterintuitive that a strategy that resets according the value of the reward in the node is outperformed by one that resets all nodes equally, this phenomena is discussed later in Section 4.4.

Two final cases are shown in Figure 4.8. In 4.8a, the performance of a best-response attacker playing against a sub-optimal defender is given. In 4.8b, both players are best-response players who incorrectly anticipate their opponent's strategies.



(a) Best-Response Attacker Vs. Sub-Optimal Defender



(b) Best-Response Attacker and Defender

Figure 4.8: Best-Response Attacker Vs. Sub-Optimal and Best-Response Defenders

As seen in Figure 4.8a, there is not much variation performance of attacker strategies. This is because the best-response attacker strategies closely approximate the equilibrium strategy. However, this figure does demonstrate the importance of optimizing for the defender. Picking an intuitive, but sub-optimal, strategy will perform worse than a calculated equilibrium strategy.

In the extreme case, the best-response defender strategies approximate the equilibrium strategies and thus all pairs of strategies shown in Figure 4.8b are all the same (with the exception of `BR(MR)`). This explains why there is little variation in value for the different strategy pairs. However, as the complexity of the network grows, and as parameters change, this will no longer be the case.

4.3.2 3n6e Network

The 3n6e network with sample parameters is shown in Figure 4.9. As before, the defender's budget is fixed at 3, and there are a total of 70 units of reward that the attacker could receive. However, this network is more connected which gives the attacker more avenues for compromise and exfiltration.

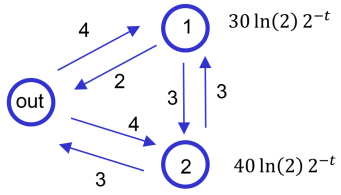


Figure 4.9: 3n6e Network

The next two figures show the interaction between two classes of players (i.e. sub-optimal and best-response) compared to the equilibrium strategies. Figure 4.10 shows that `equilibrium` is best for both sub-optimal attackers and defenders. This makes sense because the `equilibrium` strategy considers the best-response of the opponent whereas the sub-optimal strategies demonstrate strategies that might have intuitive appeal. However, the magnitude of difference between sub-optimal and equilibrium strategies might not be as large as expected. This makes sense for smaller networks since each player's strategy search space is significantly smaller. For example, `proportional` approximates `equilibrium` for the defender. This suggests a heuristic where an increase in network connectivity corresponds with an approximation of `equilibrium`.

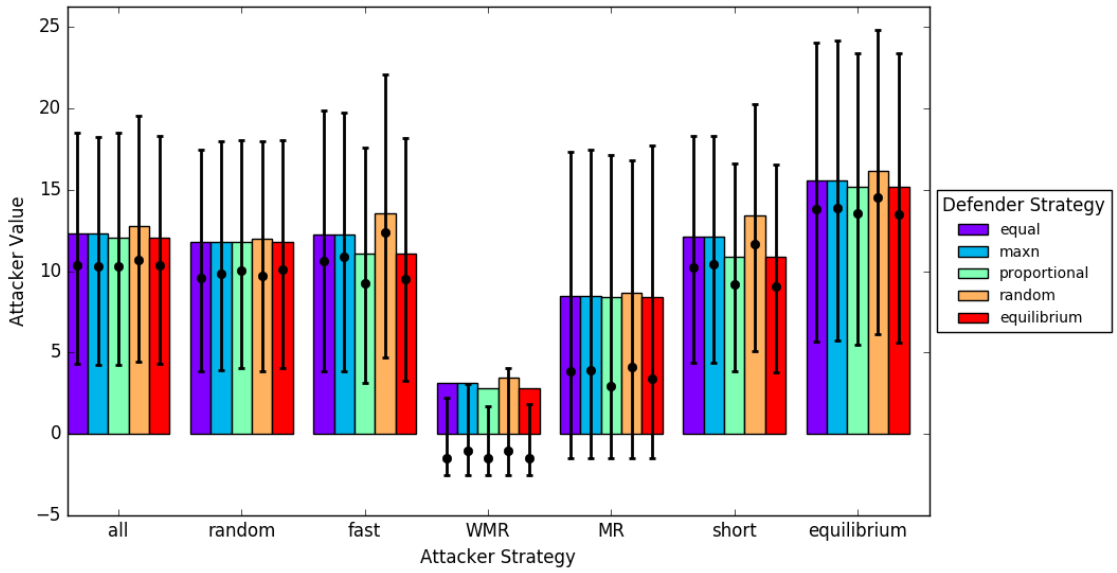


Figure 4.10: Sub-Optimal Attacker and Defender

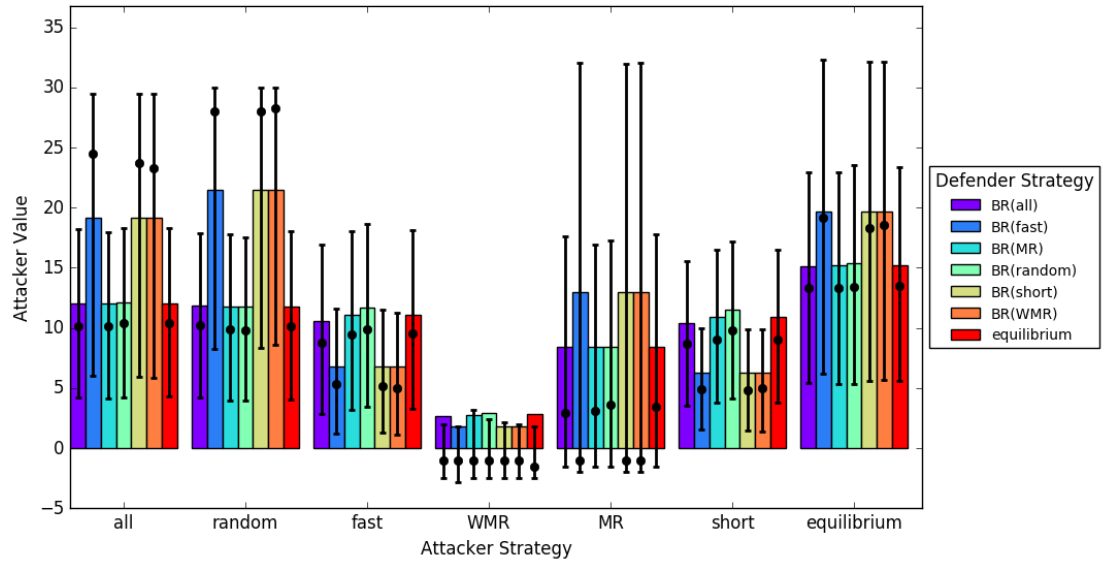


Figure 4.11: Sub-Optimal Attacker and Best-Response Defender

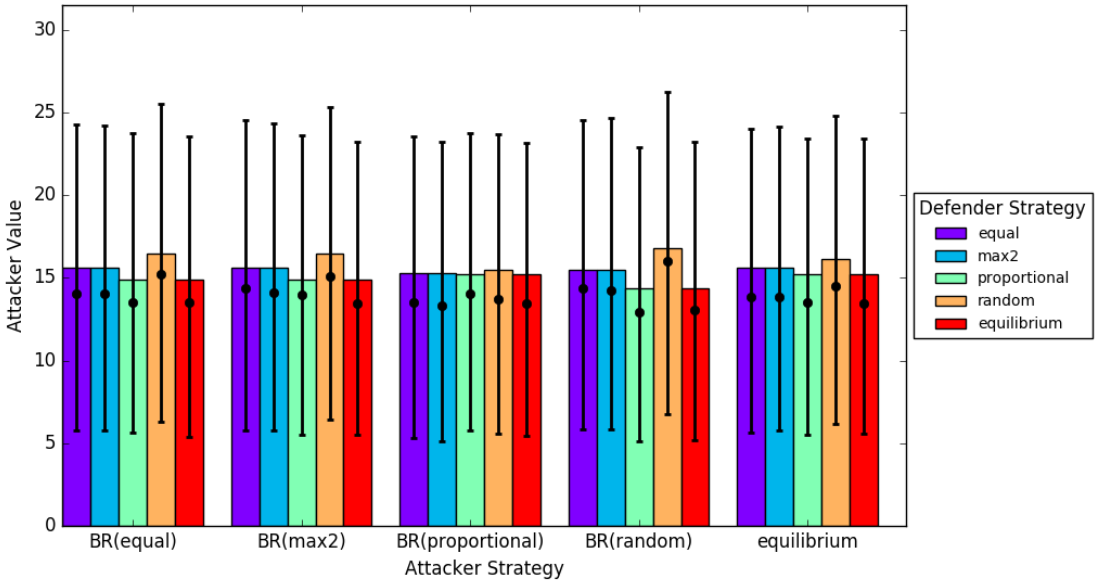
The attacker's `all` strategy consistently performs well regardless of the defender's strategy considering the little amount of computation required to calculate this strategy. This is be because the penalties are relatively low. Recall that there

are 70 total units to be earned in this network, but each penalty is only 0.5. The importance of modeling the penalties will be further discussed in Section 4.4.

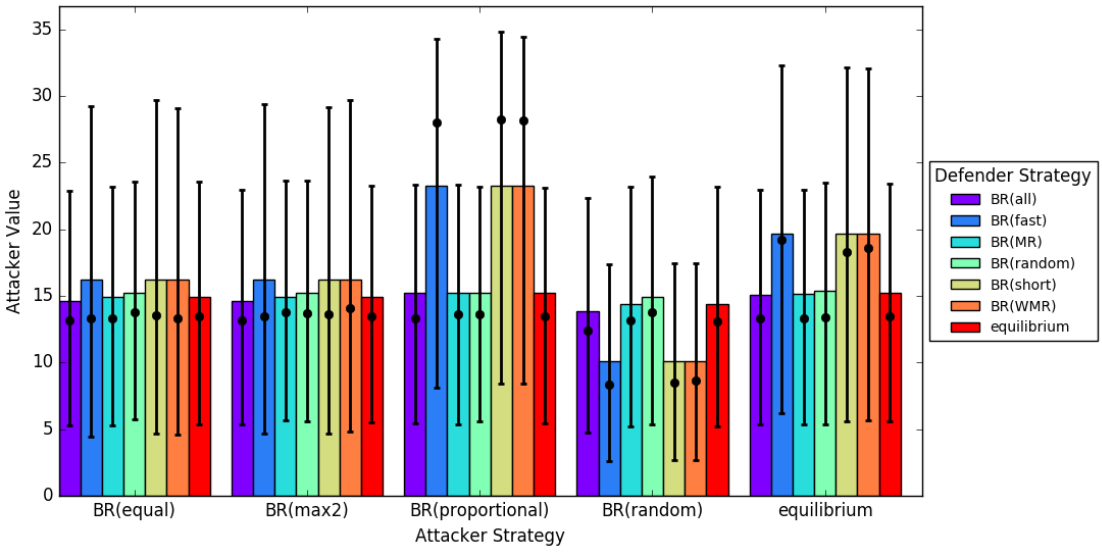
Figure 4.11 shows the best the defender can do when they know the attacker's strategy. This graph demonstrates how knowledge of the opponent's strategy can give a player an advantage. When the defender plays `equilibrium`, they can be certain that no other attacker strategy can yield a higher value. This strategy provides a bound on the worst case scenario. Looking at the `fast` attacker strategy, if the defender knows the attacker will play `fast`, the defender's best response yields a value around 7 units. However, if the defender is wrong and the attacker instead plays `equilibrium`, then the defender yields a value of around 20 units.

The sub-optimal and best-response attacker strategies are shown in Figure 4.12. Figure 4.12a shows the interaction between a best-response attacker and a sub-optimal defender. The reason for the lack of variation in reward for a particular attacker strategy is because the defender strategies are mostly the same. Both the network and the defender's action space are still quite small, so there is not much variation between sub-optimal defender strategies. Therefore the best-response to those strategies will be similar.

Figure 4.12b shows the interaction between a best-response attacker and best-



(a) Best-Response Attacker and Sub-Optimal Defender



(b) Best-Response Attacker and Defender

Figure 4.12: Simulation Results for Players on the 3n6e Network

response defender. This figure highlights the dangers of incorrectly guessing an opponent's strategy. A defender who anticipates an attacker that plays `fast`, `short`, or `random` will perform significantly worse against an attacker that prepares for a defender that plays `proportional`.

4.3.3 4n8e Network

Figure 4.13 shows a network with four nodes and eight edges. Unlike the previous two networks, this network has 100 units of reward. The attacker's rates in this network were chosen to generalize what might occur in a real network. A thorough discussion of parameter modeling is presented in Section 5.1. Since communication

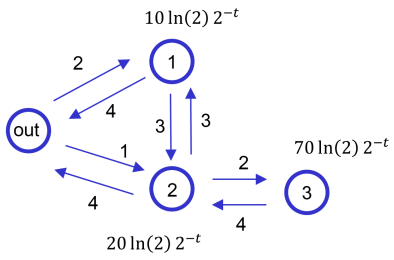
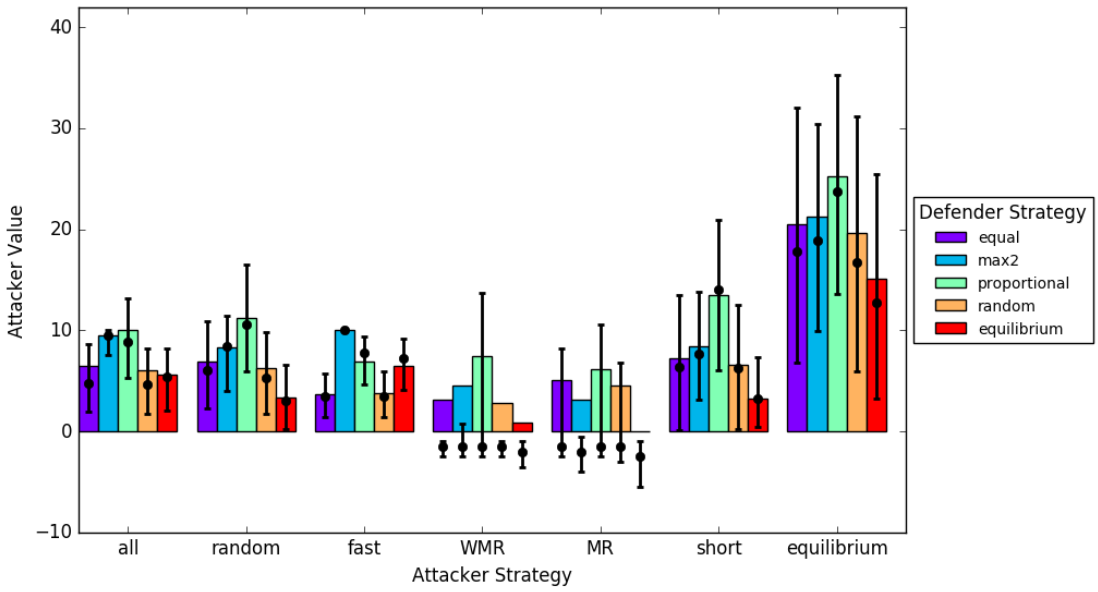


Figure 4.13: 4n8e Network

tends to be bidirectional, the difficulty for the attacker is establishing the incoming connection. The returning edge is compromised with relative ease. Node 1 can be compromised from the outside at a rate of 2 times per hour, while the reverse occurs twice as fast.

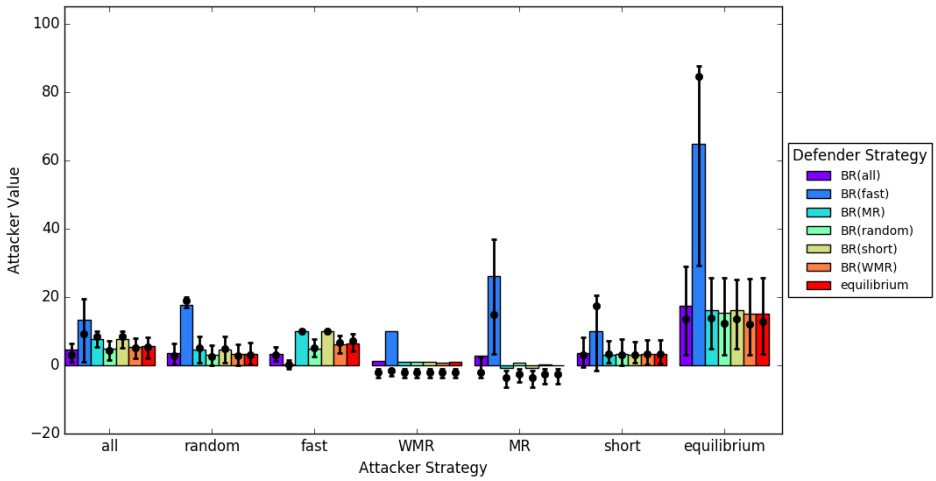
As a more concrete example, this network might represent a company with two departments and a secret project. Node 1 is the administrative staff that does not need access to the secret project room, Node 3. Node 2 represents the engineering department that needs access to the secret room. Assuming the engineering department is more security-minded than the administrative department, it is much more difficult to steal their credentials than an administrator's credentials. However, credentials are commonly checked only on entry rather than exit, so it is easier to leave an area than enter.

The four classes of interactions are given in the four figures of Figure 4.14. These graphs demonstrate that the defender's `equilibrium` strategy is almost as good as their best response strategies for a particular attacker strategy. That is, the equilibrium defense performs well (although not optimally) against sub-optimal and best-response attacker strategies, and gives an upper bound for how well an attacker could possibly do.

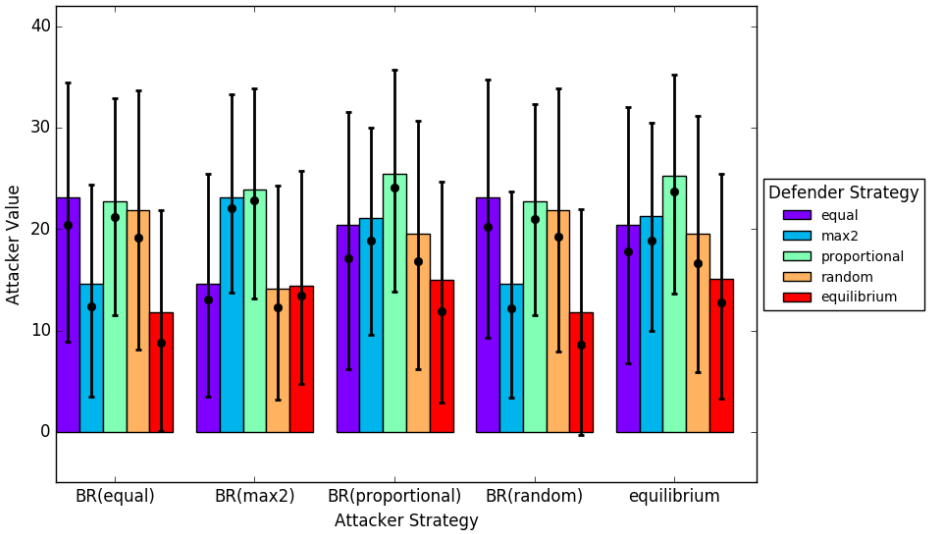


(a) Sub-Optimal Attacker and Defender

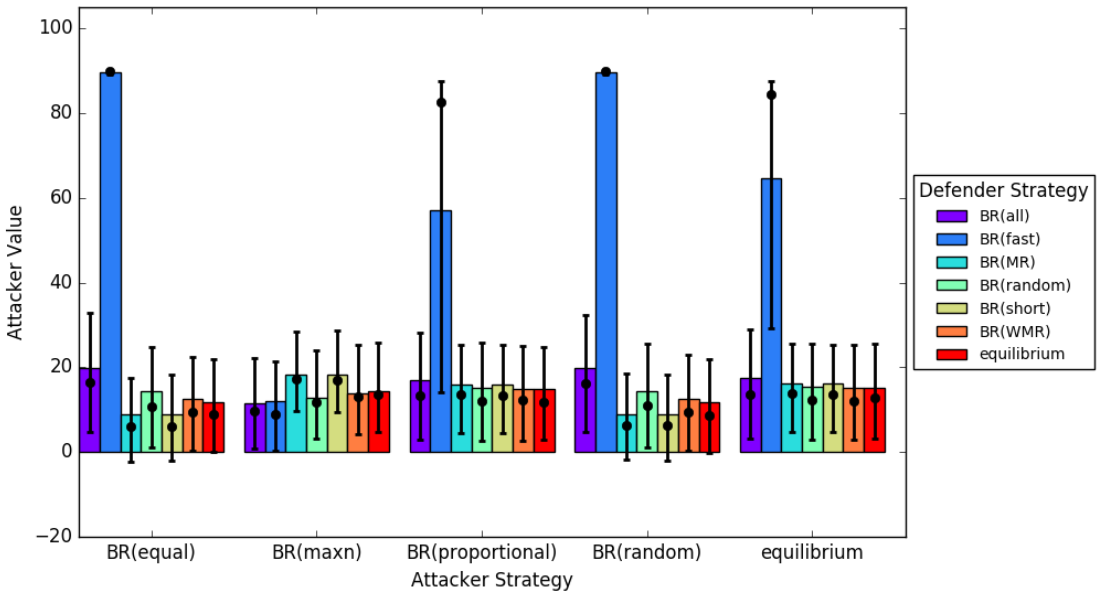
Note that an attacker who anticipates a defender that plays `proportional` yields a payoff close to that of an attacker who plays `equilibrium`. This means that the attacker's strategy of `BR(proportional)` is a good approximation of `equilibrium`. Therefore, an attacker who does not have time to compute the true equilibrium strategy might consider playing `BR(proportional)` or using this strategy as an initial guess to find the true equilibrium strategy.



(b) Sub-Optimal Attacker and Best-Response Defender



(c) Best-Response Attacker and Sub-Optimal Defender



(d) Best-Response Attacker and Defender

Figure 4.14: Simulation Results for Players on the 4n8e Network

Lastly, note how poorly the defender's strategy of `BR(fast)` performs. This makes sense because the attacker's strategy of `fast` goes after the reward in Node 1 since this credential is compromised the fastest. Therefore, the defender's best response to this strategy is to put the entire budget on Node 1. This strategy only performs well for the defender when the attacker plays `fast`. It leaves the rest of the network wide open for other attacker strategies, hence the high reward values in Figures 4.14b and 4.14d.

4.3.4 4n10e Network

The last network considered is shown in Figure 4.15. This network expands the previous case to include edges connecting Node 1 and Node 3. In the example introduced in the previous section, this means that the administrative department (Node 1) now has access to the secret room (Node 3). However, administrators rarely have a need to access the secret room, so any time they do, their credentials might be scrutinized more than an engineers. This is the reason for a lower compromise rate with these credentials.

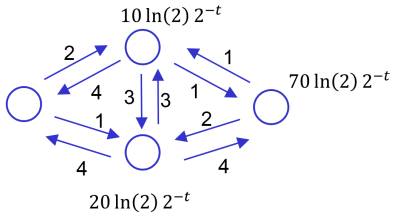
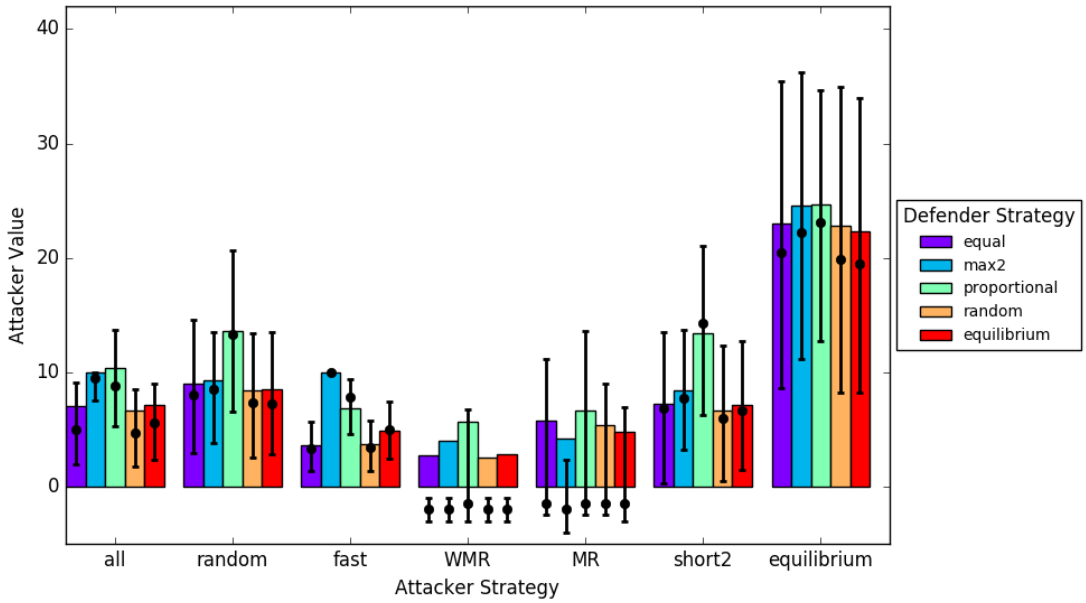


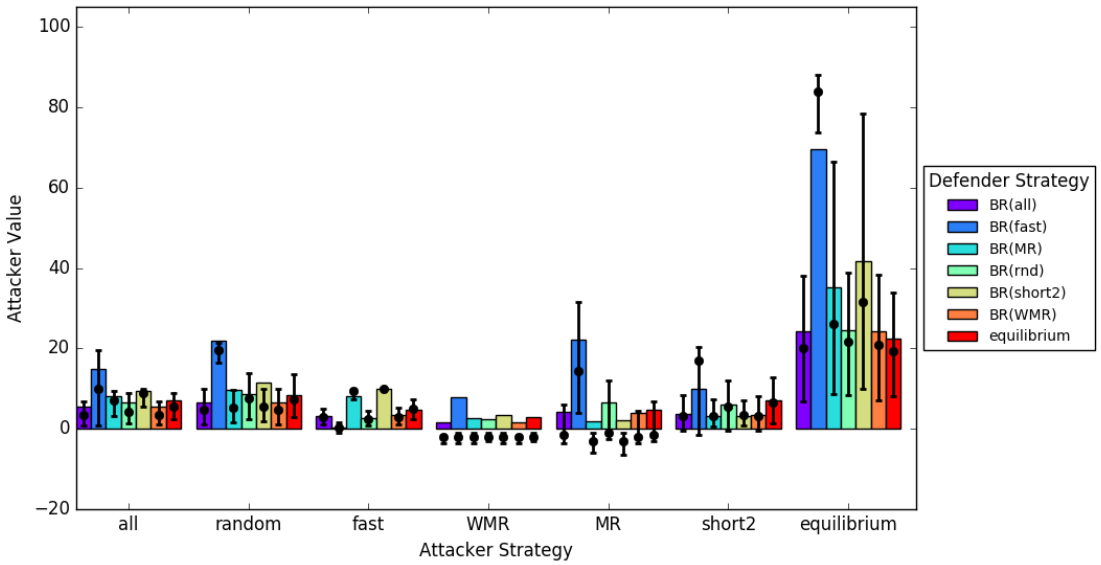
Figure 4.15: 4n10e Network

The four plots in Figure 4.16 show the simulation results between the two classes of players. Figures 4.16a and 4.16b demonstrate the importance of optimizing for the attacker. No other attacker strategy comes close to the reward received when the attacker plays `equilibrium`.

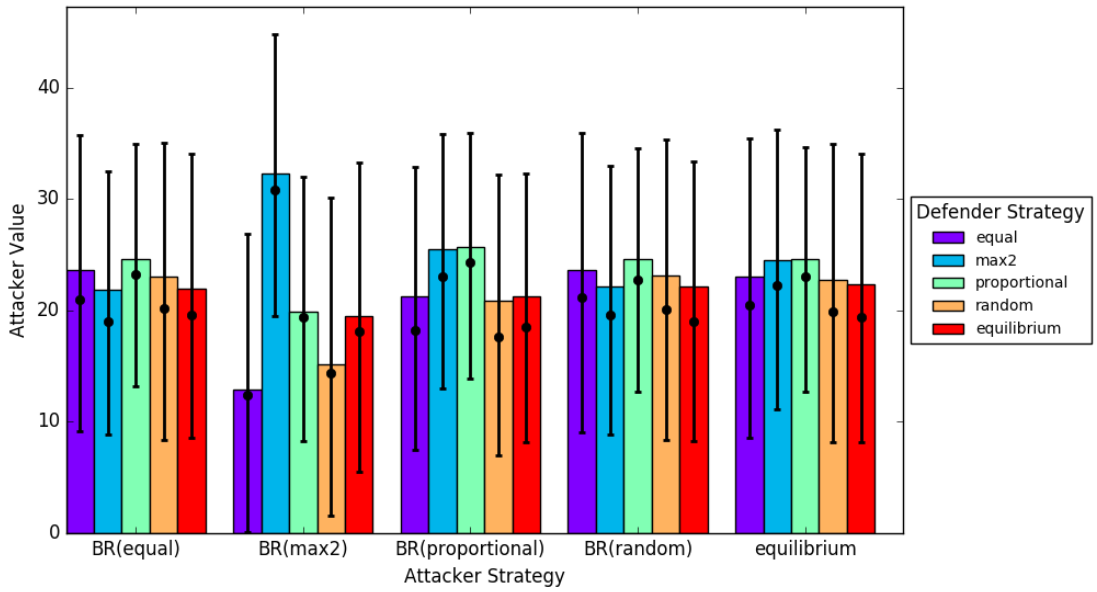
Note how the magnitude of the reward increases from the previous 4n8e network to this 4n10e network. The graphs that show a best-response attacker, Figures 4.16c and 4.16d, mostly receive above 20 units of reward. For the previous 4n8e network, these strategies were yielding under 20 units of reward. This hints at the importance of reducing the surface of attack and will be explored further in Section 4.4. The more ways that an attacker can compromise a network, the harder it is to defend.



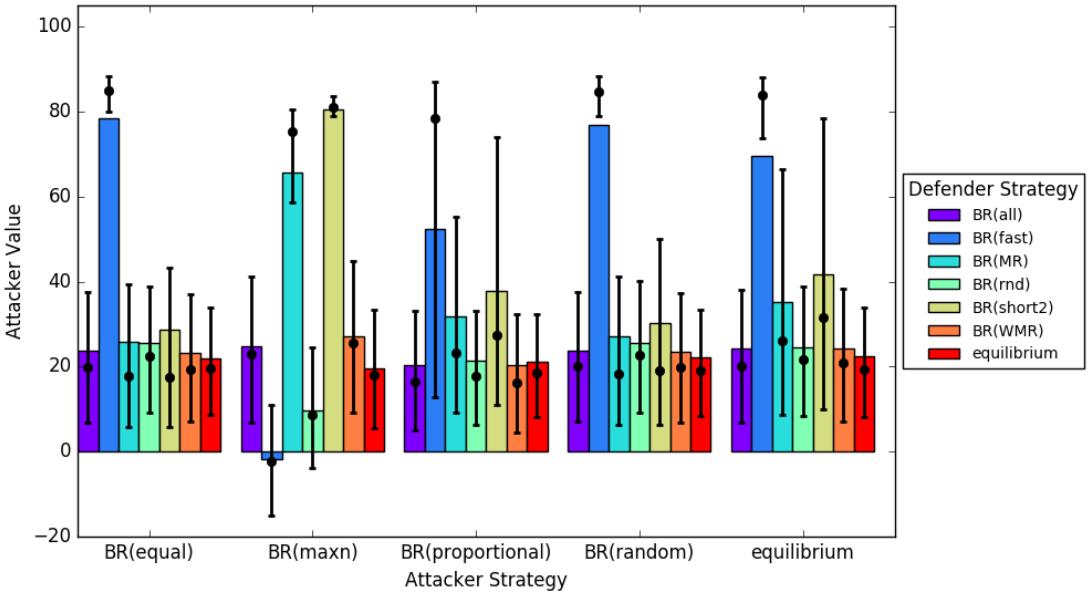
(a) Sub-Optimal Attacker and Defender



(b) Sub-Optimal Attacker and Best-Response Defender



(c) Best-Response Attacker and Sub-Optimal Defender



(d) Best-Response Attacker and Defender

Figure 4.16: Simulation Results for Players on the 4n10e Network

4.4 Defense In Depth

Defense-in-Depth is the security principle that a network should be defended in layers [30]. A network should be segmented and resources should be deployed at each segment to catch intrusions that make it through earlier defenses. It is interesting to note that not all equilibrium strategies demonstrate this security principle. In the context of this thesis, a strategy that demonstrates defense-in-depth should allocate a small portion of the budget to most nodes in the network. However, some equilibrium strategies do not recommend this strategy.

Take, for example, a modified version of the 3n4e network shown in Figure 4.17 where all reward is pushed to the 'back' of the network (i.e. the reward only resides in Node 2). A defense-in-depth strategy would suggest that some of the budget be

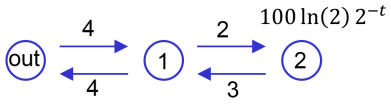


Figure 4.17: Modified 3n4e Network

allocated to both Node 1 and Node 2. Equilibrium strategies were then computed for various budgets to see which strategies demonstrate defense-in-depth. Since the defender's strategy for this network only has two nodes, it can be represented entirely by the amount of budget allocated for Node 2. Allocating nothing to Node 2 ($\lambda_2 = 0$) would imply allocating the entire budget to Node 1 since it is assumed the defender uses their entire budget ($\lambda_1 + \lambda_2 = Budget$). Figure 4.18 shows how the equilibrium strategy and attacker reward changes as the defender's budget grows.

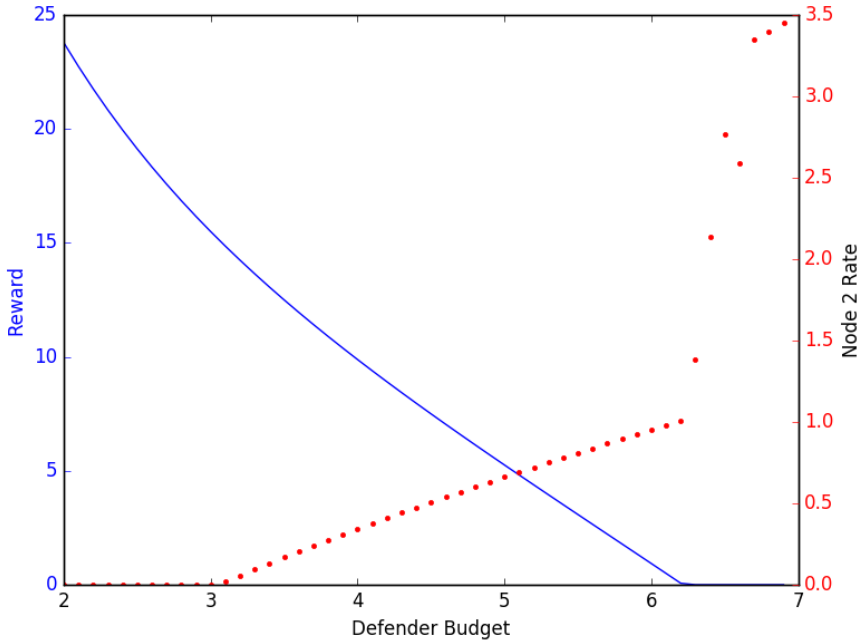


Figure 4.18: Attacker Reward and Node 2 Equilibrium Reset Rate Vs. Budgets

The blue line shows that the attacker's expected reward decreases as the budget grows. This is intuitive because a larger budget means the defender has more resources available to protect their network. The red line shows that for small budgets (i.e. $B < 3$), the defender puts their entire budget on Node 1 (i.e. $\lambda_2 = 0$). However, as the defender increases their budget from 3 to 6, the equilibrium strategy suggests putting more weight on Node 2. This suggests that defense-in-depth may not be an optimal strategy when the defender is severely budget constrained.

It should be noted that when the budget is large enough ($B > 6$ in this case), no rational attacker should attempt to compromise the network. The fact that the defender's equilibrium strategy jumps around in this region is an artifact from the

optimization solver. When the budget is large enough, there are many strategies that can dissuade an attacker from playing. The equilibrium solver only picks one of these strategies.

Furthermore, the defender's equilibrium strategy suggests that a perimeter defense is near optimal for this network. This is demonstrated in Figure 4.19 where three different defender strategies are plotted. The red line shows the performance of the pure perimeter-defense strategy where the defender assigns their entire budget to Node 1. The blue line shows the performance of allocating the entire budget to Node 2, and the dotted line shows the performance of the equilibrium strategy.

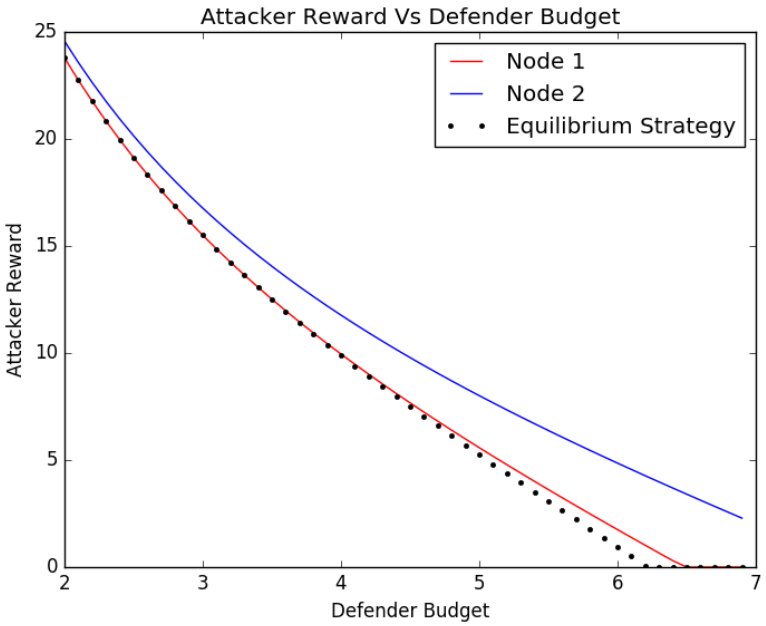


Figure 4.19: Performance of Different Defender Strategies for Various Budgets

Since the perimeter of this network consists only of Node 1, it makes sense that a perimeter-defense is near optimal. The attacker must go in and out through Node 1 in order to earn their reward.

However, as the defender's budget grows, the perimeter-defense can be improved by employing defense-in-depth. This can be seen in Figure 4.20 where the budget is fixed at 5. The x-axis shows the entire defender's action space with respect to Node 1. Each point on the x-axis is a reset rate of Node 1, the reset rate for Node 2 can be calculated by subtracting the Node 1 rate by 5 (the budget).

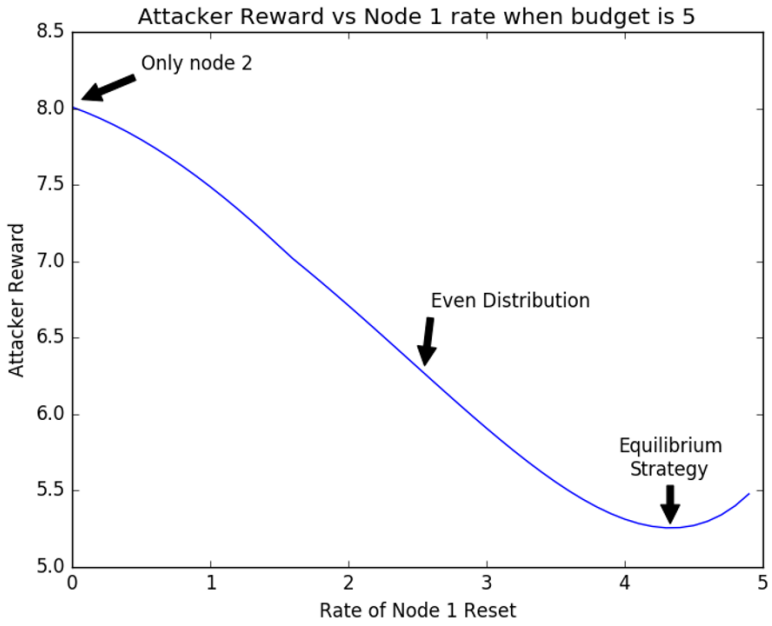


Figure 4.20: Defender's Action Space When Budget is Fixed at 5

This suggests that it is more important to have a secure perimeter than defense-in-depth. If the defender has a large enough budget, then they can start to consider deploying defense-in-depth.

Increasing the defender's budget is not the only change that can cause the defender's equilibrium strategy to demonstrate defense-in-depth. Adding an extra edge to the network can significantly change the equilibrium strategy. The previous 3n4e network is further modified with the addition of an outgoing edge from Node 2 to the outside as shown in Figure 4.21

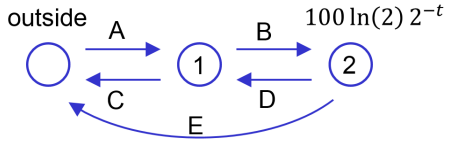


Figure 4.21: Modified 3n4e With Extra Credential E

The addition of this edge means that once the attacker has compromised Node 2, they can find a path back to the outside without going through the perimeter Node 1. This is justified by the fact that firewall rules may be more permissive for communication paths leaving the network than they are for incoming connections.

One example of this is DNS tunneling [31] where data is exfiltrated through a clever use of DNS queries.

To see how the addition of this edge affects the equilibrium strategy, the budget is fixed at 2 with all other parameters are kept the same as before and λ_E is increased from 0.001 to 0.12. The budget is fixed at 2 because the previous configuration did not demonstrate defense-in-depth at this budget level. Note that this value is an order of magnitude less than the other edges in the network. Figure 4.22 shows the defender's equilibrium strategy and corresponding attacker reward as a function of this new edge.

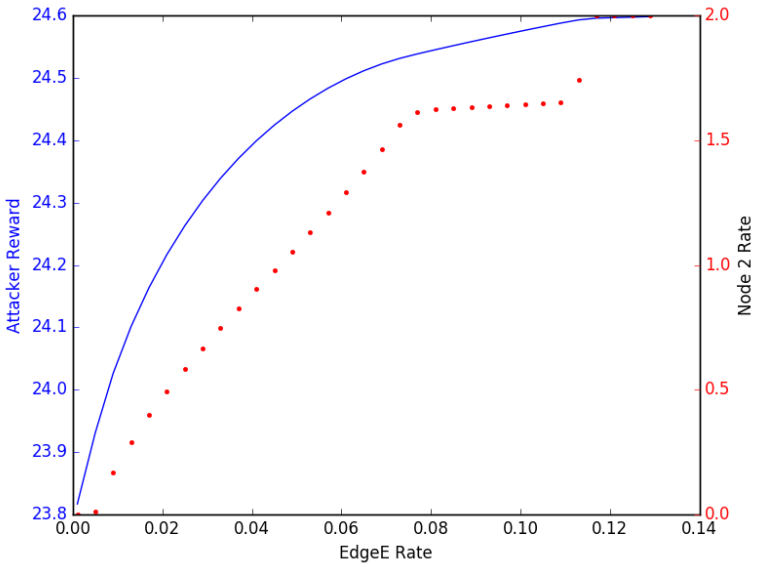


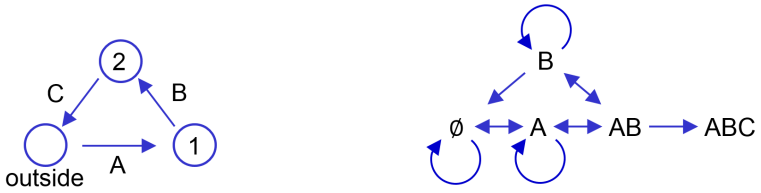
Figure 4.22: Equilibrium Strategy and Attacker Reward Vs. Edge E Rate

Even at such low compromise rates, there is a drastic change in equilibrium defender strategy. With an edge E compromise rate of 0.08, the equilibrium strategy suggests putting 75% of the budget on Node 2. Recall that the equilibrium strategy for previous configuration without edge E recommended allocating nothing to this node at this budget level.

This suggests two important ideas. First, adding an outgoing edge such as edge E increases the perimeter. Second, monitoring outgoing connections is just as important as restricting incoming connections.

These security principles can be better understood mathematically by looking at a smaller network. Consider the simple network and corresponding attacker

state graph presented in Figure 4.23.



(a) Simple Network Architecture (b) Attacker State Graph

Figure 4.23: Simple Network and State Graph to Study Defense-in-Depth

In this network, the attacker only has two strategies that could yield a positive reward. The first strategy is to play every state. The second strategy is to give up if caught in state $\{B\}$. That is, if the attacker is in state $\{AB\}$, and Node 1 is reset, the they must decide if it is worth it to continue or to quit.

Recall that the attacker will choose the strategy that will yield the highest reward. Therefore, the attacker will choose $\sigma \in \Sigma$ such that $v(\hat{\lambda}, \sigma)$ is maximized where Σ is the attacker's strategy space and $\hat{\lambda}$ is the defender's budget allocation. For this particular instance, $\Sigma = \{Continue_B, Stop_B\}$ where $Continue_B$ is the strategy where the attacker continues their compromise if caught in state $\{B\}$ and $Stop_B$ is the strategy where the attacker ceases compromise if caught in state $\{B\}$.

Alternatively, the defender must find a budget allocation $\hat{\lambda} = (\lambda_1, \lambda_2, \ldots \lambda_n)$ subject to $\sum_i \lambda_i \leq B$ for a budget B that minimizes the attacker's value. The defender's goal is to $\min f(\hat{\lambda})$ where $f(\hat{\lambda}) = \max v(\hat{\lambda}, \sigma)$. For this particular network, $f(\hat{\lambda}) = \max(v(\hat{\lambda}, Continue_B), v(\hat{\lambda}, Stop_B))$. Note that since $\lambda_1 + \lambda_2 = B$, the defender's strategy can written in terms of λ_1 for a fixed budget. Figure 4.24 shows both attacker strategies for all possible defender strategies when the budget is 3. All attacker parameters are fixed at $\lambda_A = \lambda_B = \lambda_C = 1$, and both nodes have the reward function $R_1(t) = R_2(t) = 50 \ln(2) 2^{-t}$.

This figure shows that $Continue_B$ is always better than $Stop_B$ for the attacker because $Continue_B$ will always yield a higher reward, and the smallest value the defender can achieve occurs at the boundary where $\lambda_1 = 0, \lambda_2 = 3$. For this choice of parameters, the defender shows neither defense-in-depth nor perimeter-defense because the entire budget is located only at Node 2.

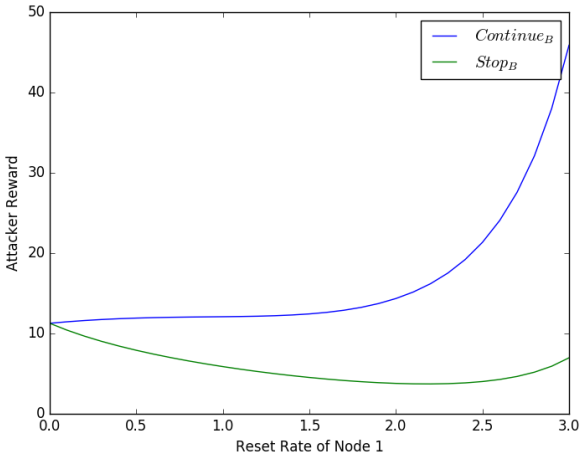


Figure 4.24: Strategy Space for Attacker and Defender on the Simple Network

The value for the strategy $Continue_B$ can be explicitly written as:

$$\frac{\begin{array}{l} Att(A)(\lambda_1^3\lambda_2 + \lambda_1^2\lambda_2^2 + \lambda_1^2\lambda_2\lambda_A + \lambda_1^2\lambda_2\lambda_B + \lambda_1^2\lambda_2\lambda_C + \lambda_1^2\lambda_A\lambda_B + \lambda_1^2\lambda_A\lambda_C + \lambda_1\lambda_2\lambda_B\lambda_C + \lambda_1\lambda_A\lambda_B\lambda_C) \\ + \text{Att(B)}(\lambda_1\lambda_2^2\lambda_A + \lambda_2^3\lambda_A + \lambda_2^2\lambda_A^2 + \lambda_2^2\lambda_A\lambda_C + \lambda_2\lambda_A^2\lambda_c) \\ + Pos(A)(\lambda_1^2\lambda_2\lambda_A + \lambda_1\lambda_2^2\lambda_A + \lambda_1\lambda_2\lambda_A^2 + \lambda_1\lambda_2\lambda_A\lambda_B + \lambda_1\lambda_2\lambda_A\lambda_C + \lambda_1\lambda_A^2\lambda_B + \lambda_1\lambda_A^2\lambda_C) \\ + Pos(B)(\lambda_1\lambda_2\lambda_A\lambda_B + \lambda_2^2\lambda_A\lambda_B + \lambda_2\lambda_A^2\lambda_B) \\ + V_{ABC}\lambda_2\lambda_A\lambda_B\lambda_C + V_{ABC}\lambda_A^2\lambda_B\lambda_C \end{array}}{\lambda_A\lambda_B\lambda_C(\lambda_2 + \lambda_A)}$$

Where $Att(A), Att(B), Pos(A), Pos(B)$ are the attempt and possession penalties for credentials A, and B respectively. Since all penalties are negative, the only positive term is that of the terminal state V_{ABC} which can be explicitly written as:

$$V_{ABC} = \int_0^\infty e^{-t(\lambda_1+\lambda_2)} R_1(t)dt + \int_0^\infty e^{-t\lambda_2} R_2(t)dt$$

Here, it can be seen that the only positive value in the expression for the attacker's strategy can be minimized by the defender by putting their entire budget on Node 2. Since $\lambda_1 + \lambda_2 = B$ and is fixed, the reward due to Node 1, $R_1(t)$ is constant. Therefore, the defender's best strategy is to allocate the entire budget to Node 2 to minimize the second term.

Because the penalties considered throughout these examples have been relatively small compared to the rewards stored in the nodes (0.5 vs 100), it makes sense that the defender should try to minimize the expected reward earned by the attacker.

This is also why the defender's equilibrium strategy changes drastically when the extra edge E was added to the modified 3n4e network discussed earlier. Adding

this link created another state whose value is determined only by the reset rate of Node 2. The reward for the attacker when exfiltrating data through Node 1 depends one the reset rate of both Node 1 and Node 2 is fixed since the budget is fixed. Therefore, It makes sense for the defender to allocate resources to Node 2 to minimize the amount of reward the defender could earn.

Chapter 5
Use Case

To demonstrate how the algorithms and strategies apply to realistic networks, two use-cases are presented. First a discussion is presented concerning realistic attacker rate parameters. Next, two use-case networks are introduced and modeled using realistic parameters, and results are analyzed. The first network, a simple enterprise network, compares the performance of the equilibrium strategies to two best-response strategies. The second network, a Navy shipboard network, provides a similar comparison, but goes further to examine the time until data exfiltration is possible. This chapter is then concluded with a summary of the results found for the two use-case networks and discussion of how the model might be extended to consider time-sensitive attackers.

5.1 Parameter Selection

The attackers modeled in this thesis are those that are actively attempting to compromise a network. This means the parameters need to model the expected time it takes to compromise a credential when the attacker is diligently working in a live environment. The time expended by the attacker off-line is not considered.

There has been some work on empirical measurements of a network's Mean Time to Compromise (MTTC) [32] and Time to First Compromise (TFC) [27]. This section gives a brief survey of those results and derives three attacker rate classes based on those existing empirical studies.

Virtual Machines (VMs) in a cloud environment have been shown to leak cryptographic keys in co-location attacks. In these attacks, a malicious VM uses a side channel attack to eavesdrop on the keys used by another VM residing on the same hardware [33]. The authors in [34] demonstrated that a 3072 bit El Gamal key could be extracted in 12-27 minutes. Another shared cache attack demonstrated that 150-600 encryptions are necessary to correctly identify a 128 bit AES key [35].

In addition to side channel attacks, phishing attacks have also been studied. In 2005, a university conducted a phishing experiment on its students [36]. They found

that 70% of all credentials stolen occurred in the first 12 hours. This corresponded to a total of 145 credentials during this period; therefore, an average of 5 minutes per credential was assumed.

A red-team experiment in 2010, called the Baltic Cyber Shield Exercise, analyzed the behavior of a red team (attackers) and blue team (defenders) in a mock live-fire environment [37]. The group of attackers consisted of 16 professional penetration testers and were asked to attack a typical critical national infrastructure architecture that included SCADA systems over the course of two days. They found that the average attack time was 2 hours with the fastest compromise occurring after 22 minutes and longest taking 7 hours.

Lastly, United States Department of Defense published the standard *Trusted Computer System Evaluation Criteria* (TSEC) in 1985 [38]. It was stated that covert channels that leak information up to 0.1 bits per second were acceptable. Furthermore, those channels that leak information at rates between 0.1 and 1 bits per second were also acceptable, but must be audited. Given a 2048 bit key, this means that it would be acceptable to leak a key every 34 minutes to 5.7 hours. The *Common Criteria* replaced the TCSEC standard in 2003, however, it did not provide recommendations on newer acceptable information leakage rates.

Based on these results, credentials were partitioned into three classes. Credentials that can be compromised in less than an hour are considered high compromise rates. Although the fastest compromise was a phishing attack, taking 5 minutes, a lower bound of 10 minutes is used. Therefore credentials with high compromise rates will take on values $\lambda \in [1, 6]$ and are used to describe those communication paths normally allowed by the network architecture. Medium compromise rate credentials will take on average 1 to 4 hours to compromise. They take on values $\lambda \in [1, 0.25]$, and describe the typical case with the 'normal' number of security vulnerabilities. Finally, low compromise rate credentials represent those credentials that have gone through some sort of hardening or provide access to security critical appliances like firewalls. They might also represent unknown or zero-day vulnerabilities present in the network. These rates take on values $\lambda \in [0.125, 0.25]$, meaning that it takes an attacker an average of 4-8 hours per credential.

5.2 Enterprise Network

This use-case network shows a simple enterprise scenario as was considered in [19, 39]. It consists of an attacker controlled host, a firewall, and two defender controlled hosts: an HTTP server, and a host that uses SSH to configure the HTTP server. This network, as the authors analyzed it, is presented in Figure 5.1.

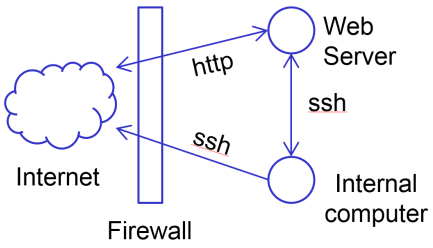


Figure 5.1: Enterprise Network Considered in Related Work

Note that the internal computer is protected from the rest of the Internet by the firewall. The Internet should not be able to access the internal computer directly. However, this can be converted into a more expressive authentication network as presented in Figure 5.2.

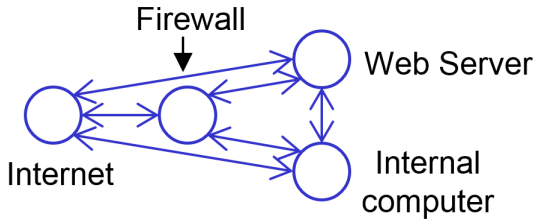


Figure 5.2: Full Enterprise Authentication Network

Here, the security appliance (a firewall) becomes an attackable node in the network. If an attacker has access to an account in the firewall itself, it is likely this account can communicate with both the web server and internal computer. Furthermore, there is an additional directed edge connecting the Internet to the internal computer. This is justified by two possible scenarios: the firewall could be misconfigured, or there could be a zero day vulnerability that bypasses the firewall completely.

5.2.1 Results

Equilibrium strategies were evaluated on this enterprise network using several sets of parameters. Four different attacker rate parameter profiles were considered. First, all credentials are sampled from the low compromise rate class as described in the previous section. This models a hardened system where compromise of nodes requires discovering unknown vulnerabilities or finding zero-day exploits. Likewise, two other profiles consider the case where all credentials are sampled from the medium and high rate classes. In addition, a mixed-rate case is considered where each edge in the graph is given a compromise-rate class based on its role in the network. This case aims to more closely model a realistic scenario. Figure 5.3 shows the rate classes assigned to each credential. The abbreviation 'med' is used to describe the medium compromise-rate class.

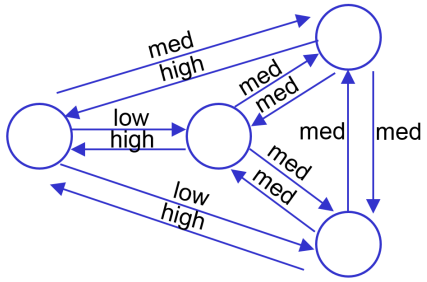


Figure 5.3: Mixed-Rate Attacker Profile

The firewall in Figures 5.2 and 5.3 is a security appliance. It is assumed that some amount of hardening went into its configuration to make it more difficult to compromise. Therefore, compromises coming from the outside and attacking the firewall directly are successful at low rates. In a similar manner, the internal computer is protected by the firewall. Zero-day vulnerabilities and misconfigurations in the firewall may exist that allow the attacker to bypass the firewall and access the internal computer directly. However, this should not typically happen and the compromise rate is assumed to be low through this edge. The web server, on the other hand, has a larger attack surface, making it easier to compromise than either the firewall or the internal computer from the outside internet. This means that the incoming edge from the outside to this web server node should be compromised at a medium rate. All of these nodes have edges that lead to the outside at high rates. This makes sense for several reasons. First, communication is typically

bidirectional, so establishing an SSH session will also provide an avenue for data exfiltration if the attacker chooses this path. Second, data leaving the network is not monitored as strictly as it is coming into the network. Once inside the network, the 'normal' number of vulnerabilities exist. Therefore, all credentials used for communication between the firewall, web server, and internal computer all have medium compromise rates.

Five different budgets are used to demonstrate the performance of the three strategy pairs. Budgets of 0.375, 1, 3, 6, and 9 are chosen because this allows the defender to reset each node once every 8 and 4 hours as well as 1, 2, and 3 times an hour respectively.

The equilibrium strategies are compared against two other strategy pairs. The strategy pair where the attacker plays `equilibrium` and the defender plays `BR(all)` describes the interaction between an equilibrium attacker and a defender who anticipates an attacker that plays `all`. The strategy pair where the attacker plays `BR(proportional)` and the defender plays `equilibrium` describes the interaction between an attacker who anticipates a defender that plays `proportional` and an equilibrium defender. These two strategies were chosen based on their performance demonstrated in Chapter 4 as well as their intuitive appeal.

Figure 5.4 shows the performance of these three strategy pairs for all four attacker profiles when the budget is fixed at 0.375. In the figure, the solid blue line shows the value awarded to the attacker when both players play their equilibrium strategy. The dotted orange line, denoted `Eq - BR(all)`, shows the performance of the strategy pair where the attacker plays `equilibrium` and the defender plays `BR(all)`. Likewise, the dashed gray line shows the performance of the strategy pair where the attacker plays `BR(proportional)` and the defender plays `equilibrium`.

It is expected that the strategy pair `Eq - BR(all)` yields a higher reward to the attacker than the `Eq - Eq` strategy pair since the defender's best-response to an attacker that plays `equilibrium` is `equilibrium`, not `BR(all)`. However, Figure 5.4 shows that when the compromise rates are mixed, the pair `Eq - BR(all)` yields a lower value to the attacker making it appear that the `BR(all)` strategy is better for the defender than `equilibrium` because of a lower attacker value. Despite this lower attacker value, `BR(all)` is not an optimal strategy for the defender because they are at risk of the attacker learning their strategy and optimizing against it. If the attacker that knows that the defender plays `BR(all)`, they can optimize against

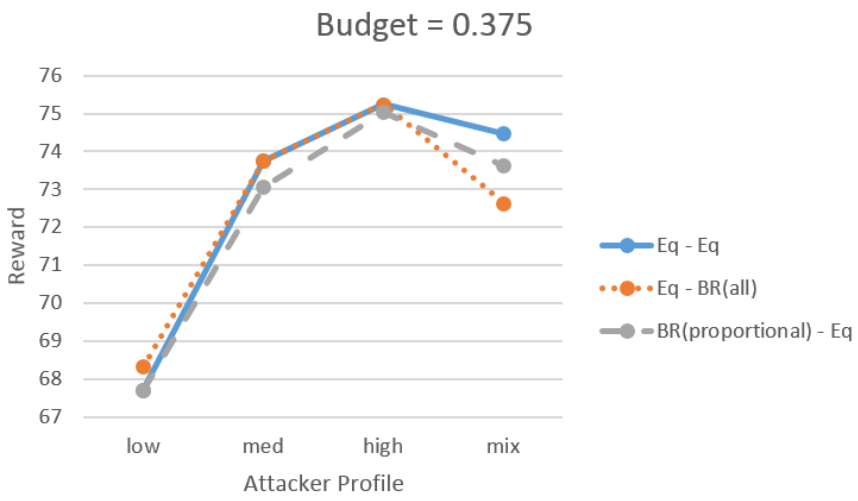


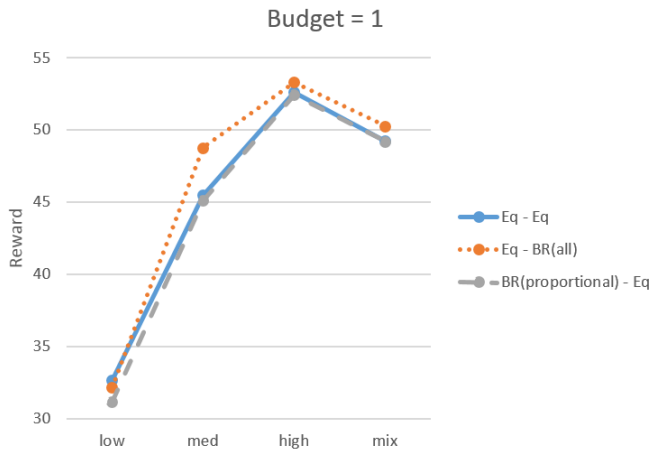
Figure 5.4: Performance of Strategies for a Budget of 0.375

this strategy and receive a value of 75.683 (not shown), which yields a higher value to the attacker (which is worse for the defender) than the `Eq - Eq` strategy pair.

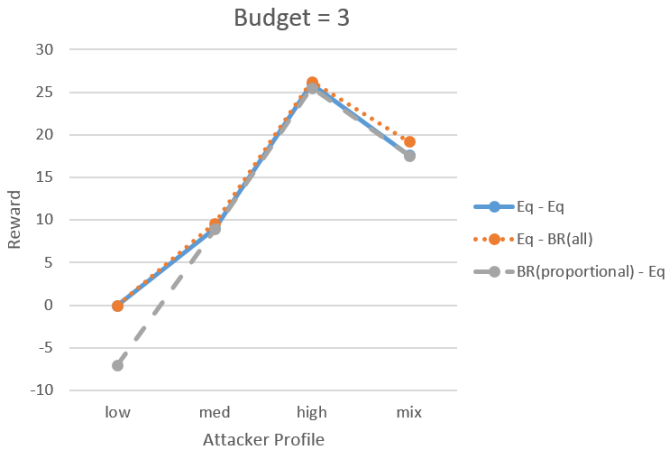
A similar argument explains why the `BR(proportional) - Eq` strategy pair should yield a lower value to the attacker than the `Eq - Eq` strategy pair. The attacker's strategy of `BR(proportional)` is not a best response to an equilibrium defender, meaning that the attacker should generally expect to earn lower rewards than those earned when playing `equilibrium`.

It can therefore be seen that the strategy pair `Eq - Eq` defines bounds on the worst case scenario for both players. Any attacker that deviates from playing `equilibrium` will receive a lower reward when the defender either plays `equilibrium` or learns the attacker's strategy and optimizes against it. Likewise, any defender that deviates from playing `equilibrium` will yield a higher reward to the attacker when facing an attacker that plays `equilibrium` or an attacker that learns and optimizes against the defender's strategy.

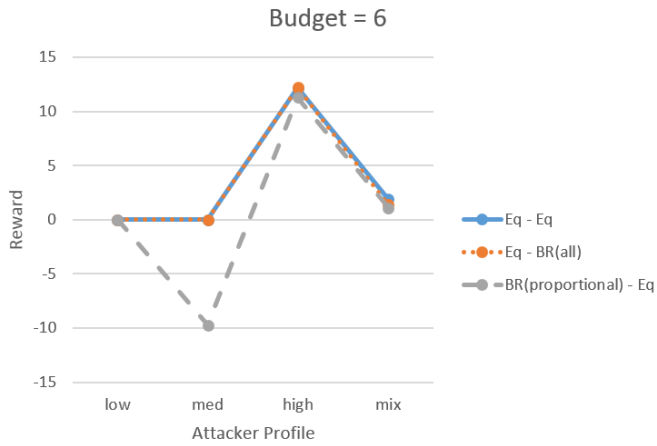
Figure 5.5 shows similar results for budgets of 1, 3, 6, and 9. Note that the magnitude of the reward yielded to the attacker significantly decreases as the budget grows. Tripling the budget from 1 to 3 decreases the reward from 33 units, when the compromise rates are low, to completely dissuading the attacker from even trying to compromise. A budget of 6 is enough to prevent attacks for low and medium rates and provides marginal benefits for the mixed-rate scenario.



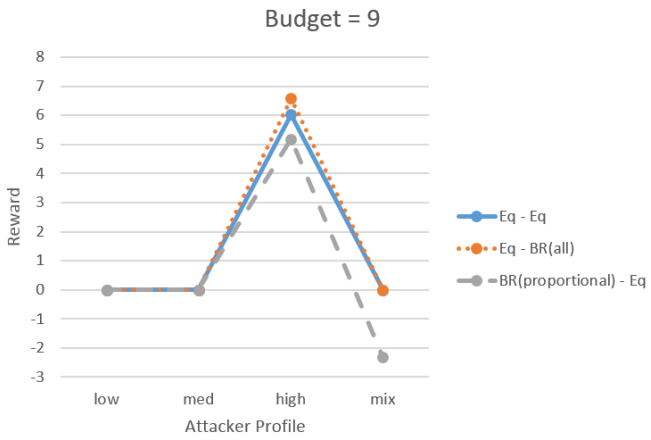
(a) Performance of Strategies for a Budget of 1



(b) Performance of Strategies for a Budget of 3



(c) Performance of Strategies for a Budget of 6



(d) Performance of Strategies for a Budget of 9

Figure 5.5: Enterprise Network Performance for Budgets 1, 3, 6, and 9

Recall that a rational attacker should not attempt to compromise a network when their expected payoff is non-positive. This is why the two strategy pairs where the attacker plays `equilibrium` (solid blue and dotted orange lines) never receive a value less than zero. An attacker who expects to lose would rather not play. However, when the attacker plays `BR(proportional)`, they receive a negative value when playing against a defender that plays `equilibrium`. This makes sense since the attacker was anticipating that the defender plays a different strategy. On the other hand, `equilibrium` sets a lower bound on what the attacker could possibly receive, thus defending themselves against incorrectly guessing their opponent's strategy.

5.3 Navy Shipboard System

In 2013, a news article reported that one of the Navy's newest ships, runs an IP network that is partitioned into different subnets [40]. This network is shown in Figure 5.6. Each subnet contains a collection of systems related to one of five specific tasks: external communications, central command, sensors, ship control, and weapons control.

There are several different ways to represent this network. One way is to represent each subnet as a node in the network. The authentication structure follows from the arrangement of the subnets and is shown in Figure 5.7. Note that

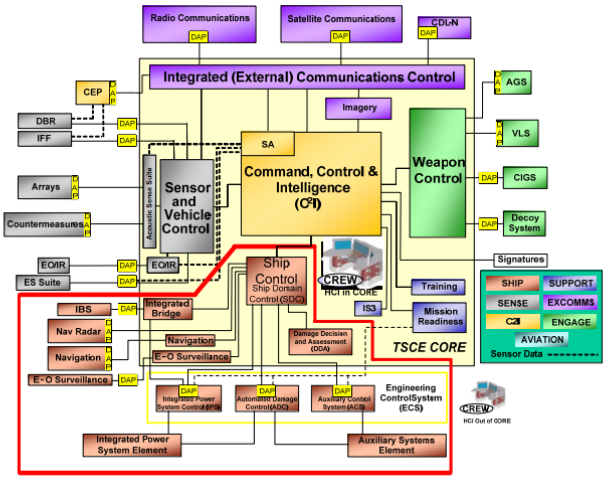


Figure 5.6: Shipboard Communication Network

the outside node has an edge connecting to the weapons control node. This is to represent the possibility of a zero day attack on one of the components of this subsystem. Both the sensor and the ship control nodes can also be compromised from the outside by similar means as well as physical attacks that might result in a vulnerable state. For example, an attacker may deploy a physical object that sends fake or malformed GPS data to a control sensor resulting in a buffer overflow in that sensor's software.

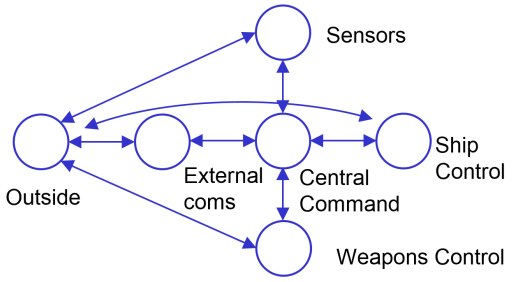


Figure 5.7: Navy Shipboard Network

However, these extreme attacks will have probabilities of success different than those described for a traditional enterprise network. Therefore, it is important to find attacker parameters that better model these types of complex attacks. Domain specific knowledge is required to understand the potential vulnerabilities that might exist in these networks as well as the likelihood that these vulnerabilities will be

exploited. For demonstration purposes Figure 5.8 shows the sample parameters that will be used for simulations.

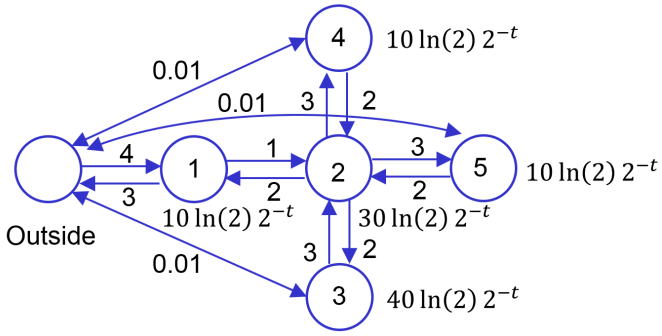


Figure 5.8: Shipboard Network with Parameters

In this configuration, the External Coms, Sensors, and Ship Control all have 10 units of reward. The Weapons Control subsystem has 40 units, and the Central Command (Node 2) subnet has 30 units of reward. Once inside the network, most compromise rates are similar. It should be easier to compromise each subsystem from Central Command than visa versa, therefore the rates leaving Node 2 are higher than those incoming. External Coms (Node 1) has many possible avenues of attack, therefore the probability of compromise from the outside is slightly higher coming into this node, at a rate of 4. Finally, zero-day exploits are modeled with rates of 0.01 coming into and leaving from the perimeter nodes: Sensors, Weapons Control, and Ship Control subsystems.

5.3.1 Results

As before, the three strategy pairs of `Eq - Eq`, `BR(proportional) - Eq`, and `Eq - BR(all)` are analyzed. Figure 5.9 shows simulation results of 10,000 runs comparing the reward awarded to the attacker for each strategy pair fixing the defender's budget at 3 units.

As before, the similarity in performance for these three strategy pairs can be seen. Note that when the attacker anticipates a defender who plays `proportional`, the spread of rewards is larger. This means that it is slightly more risky for the attacker to play this strategy. Although the average value for this distribution of simulated rewards is about the same when they play `equilibrium`, there is a possibility that both higher and lower rewards can be earned. When the defender

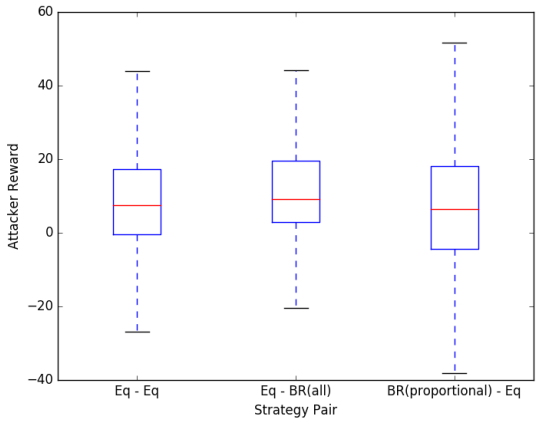


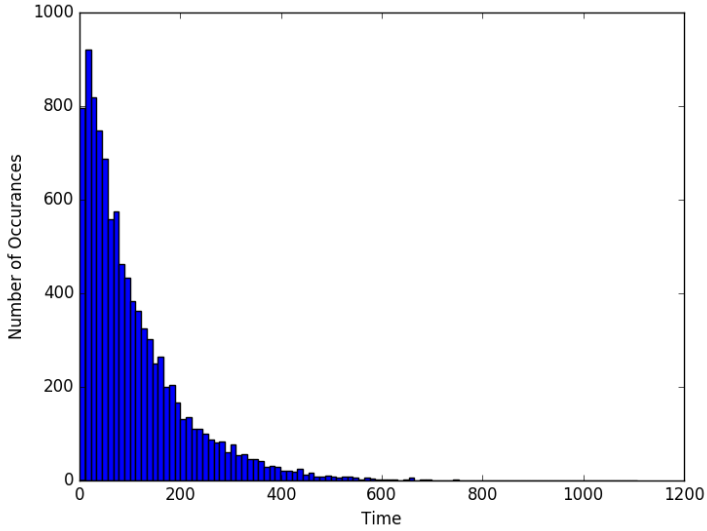
Figure 5.9: Distribution of Simulated Rewards for Different Strategy Pairs

anticipates an attacker who plays `all`, the distribution of rewards has a more narrow spread, but the average values are slightly higher. For this strategy pair, the defender can be more confident that the attacker will receive a lower reward, but on average, the reward is slightly greater than if the defender were to play `equilibrium`.

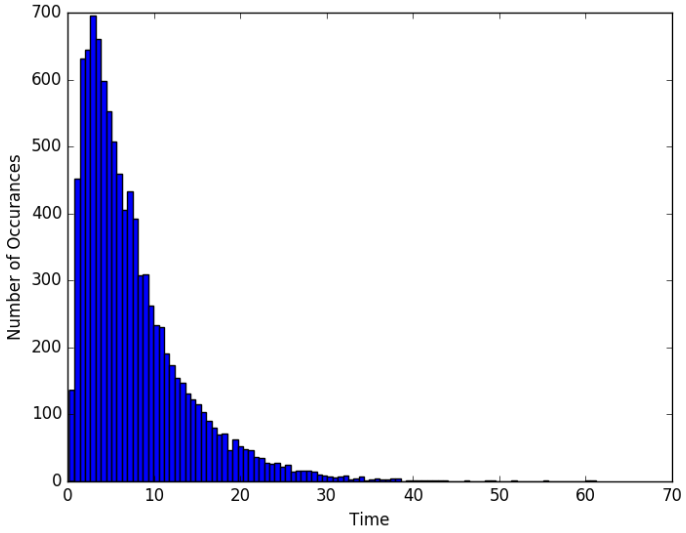
The time until exfiltration was also recorded. This is the amount of time the attacker spends in the network before exfiltration is possible and the attacker starts generating reward. Figure 5.10 shows the distribution of these times for the three strategy pairs on the same 10,000 trials. Each plot shows a histogram of the number of trials that spent a given amount of time before exfiltration was possible.

It is interesting to note that Figure 5.10b, depicting the distribution of times for the `BR(proportional) - Eq` strategy pair, is an order of magnitude faster than either of the other two strategy pairs. This suggests that the notion of time should play a larger role than originally hypothesized. If two strategies yield similar rewards with one producing the reward an order of magnitude faster, then this faster strategy should be chosen. The model can easily be extend to include a discount factor to take in the time dimension, but this is left as future work.

From these graphs, it can be seen that the equilibrium strategies considered do not account for the speed at which the reward is earned. Although the equilibrium strategies still yield the best value for each player, there may be better strategies when a player is time constrained. In a shipboard system, where the network itself is mobile, time becomes a critical factor for compromise.



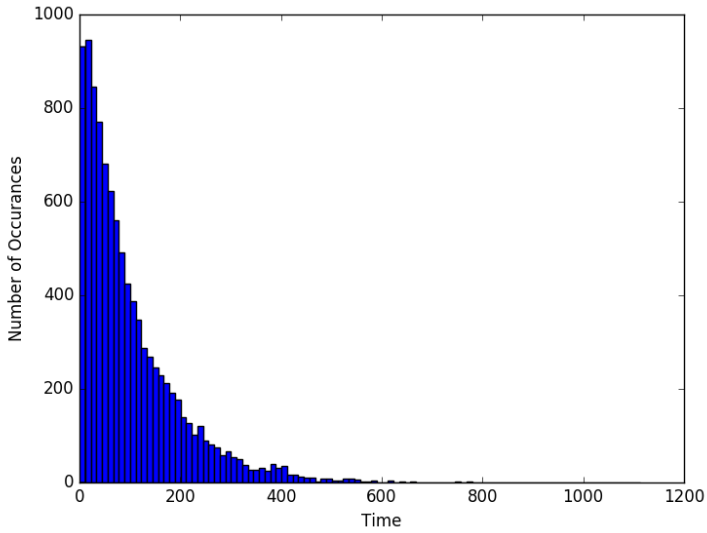
(a) Distribution of Times for Equilibrium vs Equilibrium



(b) Distribution of Times for BR(proportional) vs Equilibrium

5.3.2 Results Summary

This chapter has shown how equilibrium strategies perform compared to two best-response strategies for the attacker and defender on two realistic use-case networks: a simple enterprise network, and a Navy shipboard system. It was shown that the `BR(proportional)` strategy is a good approximation of the equilibrium



(c) Distribution of Times for Equilibrium vs BR(all)

Figure 5.10: Distribution of Times Until Exfiltration

strategy for the attacker. This means that if the attacker does not have the time or computation power to find the true equilibrium strategy, it might be worth playing `BR(proportional)` instead. Likewise, the strategy `BR(all)` is a good approximation of the defender's equilibrium strategy.

The Navy shipboard network demonstrated that time-to-compromise is a factor that should be considered when evaluating strategies. If a network is only physically present for a certain amount of time due to mobility, an attacker strategy that results in a complete compromise but takes three times as long is not feasible. One way to resolve this issue is to discount rewards that are earned later in time. After every time unit, the rewards are worth a fraction of what they were before. The magnitude of the discount factor determines the attacker's patience. The higher the discount factor, the more the attacker is willing to wait for rewards that are earned later in time.

Chapter 6
Conclusion

In this thesis, a novel model for calculating the optimal defense strategy was proposed. The performance of equilibrium strategies was compared to other best-response strategies as well as intuitive but sub-optimal strategies for several simple networks. These simple networks demonstrated that the equilibrium strategies provide a bound for the worst case scenario for both the attacker and defender. If it is too computationally expensive to compute the equilibrium strategy, a defender that anticipates an attacker who plays `all` performs reasonably well. Likewise, an attacker that anticipates a defender who plays `proportional` also performs reasonably well.

Next, the relationship between the equilibrium strategies and the canonical security paradigms of defense-in-depth and perimeter-defense was explored. It was shown that defending the perimeter seems to be the most important. However, when the budget allows, adding defense-in-depth can improve the strategy. This means that when the defender is budget constrained, it may not make sense to deploy a defense-in-depth strategy.

Furthermore, a defender who has a sufficiently high budget can completely dissuade an attacker from attempting to intrude into a network. Increasing the budget increases the probability the attacker will incur penalties for intruding into the network. It is further hypothesized that this can occur if the defender has a mechanism for enforcing higher penalties for the attacker. For example, if the defender has a sufficient geopolitical power, they might be able to impose penalties in the form of heavy economic sanctions, rather than just bad press.

Lastly, two real-world use-cases were presented. These use-cases were introduced with a discussion of appropriate parameters for modeling attacker compromise rates. In this discussion, three classes of rates were described: those that can be compromised between 6 and 1 times per hour (high rates), those that can be compromised once every 1-4 hours (medium rates), and those that take 4-8 hours to compromise (low rates). These rates were combined to model different attacker capabilities called attacker profiles. Creating an attacker profile for a specific

network remains an open question, but an intuitive approach was given to describe how a profile might be constructed.

The enterprise use-case analyzed four profiles at five different defender budget levels. Again, it was shown that a sufficiently high budget can deter an attacker from attempting to compromise a network. In addition, the best-response strategies proved to perform well compared to the equilibrium strategies.

The Navy shipboard system demonstrated that time is a crucial factor when defining optimality. The same three strategy pairs were simulated on this network, and performance of these strategies with respect to attacker reward was found to similar to that of the enterprise network. However, the `BR(proportional) - Eq` strategy pair yielded reward to the attacker at times that were an order of magnitude faster than that of the other two strategy pairs. To account for this discrepancy, a discussion of the use of discount factors was presented.

6.1 Additional Use Case

To show applicability to a wider range of use cases, this section describes how the model can be applied to a SCADA system. There are many ways to model these systems, but only one example is presented.

SCADA system architectures tend to mirror architectures of enterprise networks more so than a shipboard system. Since most SCADA systems exist as a subnet of a corporate network, SCADA systems are susceptible to many of the same types of vulnerabilities that enterprise networks are. The authors in [41] describe a simple SCADA architecture arranged in a bus configuration. This means that all nodes behind the firewall can communicate with each other. Figure 6.1 shows a sample network representing the SCADA architecture style.

A more expressive architecture can be created by adding links that bypass the firewall; however, domain specific knowledge is required to validate which vulnerabilities could be exploited in this scenario.

For these networks, the question may not be determining the optimal allocation of defense resources. Instead, it may make sense to consider what defense budget is required to deter attack. The model presented in this thesis provides a framework to answer this question.

One of the shortcomings of the algorithms presented with the model in this

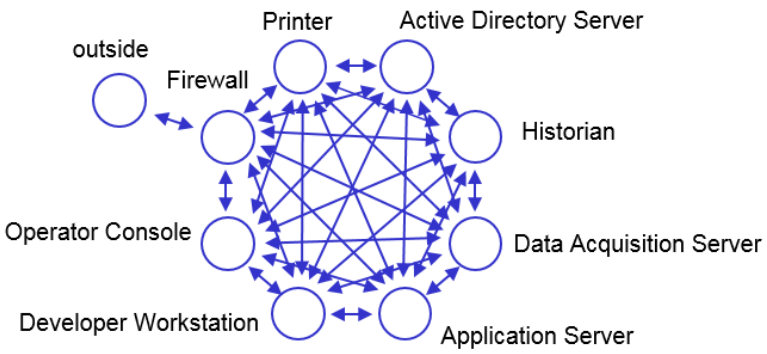


Figure 6.1: Simple SCADA Network

thesis is time complexity. The algorithms scale exponentially with the number of edges. This means that the larger networks quickly become intractable. However, the equilibrium strategy rarely visits all states in the attacker state graph. This means that heuristics may exist to shorten the search space when finding the attacker's equilibrium strategy.

6.2 Future Work

There are several places where this work could be expanded. This thesis made several assumptions that could eventually be relaxed to provide a more expressive model.

First, probability distributions were assumed to be exponential. While this distribution has been shown to be sufficient to model attacker rates [27], there may exist better distributions for the defender strategy. For example, the FlipIt model [9] demonstrated that a delayed exponential strategy outperforms the pure exponential strategy. This comes at the cost of more expressive states and more complex probability calculations.

Second, attacker penalty functions could be expanded to non-constant functions. This would increase complexity of the optimal attacker strategy because each credential would have an optimal compromise time associated with it. The attacker no longer can consider the discrete action space of `attempt` or `not attempt`, but instead would have a continuum of strategies $t \in [0, \infty)$ for every t associated with each credential.

Third, a Bayesian Stackelberg approach could be used to model the non-zero

sum game version where attackers of different types value the data stored at nodes differently. To do this, the defender must optimize over a probability distribution of these different attacker types.

Lastly, this game could be formulated as a competitive partially observable Markov decision process. Each time the attacker is caught attempting or in possession of a credential, the defender could be alerted to the attacker's current estimated position. This would allow the defender to adapt their strategy. With this formulation, it might make sense to consider an additional attacker strategy parameter: intensity. The attacker could to choose the intensity of an attack at the risk of becoming more observable to the defender.

Bibliography

[1] Target (2014), "Target Provides Update on Data Breach and Financial Performance," Accessed: 2017-01-27.
URL https://corporate.target.com/press/releases/2014/01/target-provides-update-on-data-breach-and-financia

[2] Krebs, B. (2014), "Email Attack on Vendor Set Up Breach at Target," Accessed: 2017-01-27.
URL https://krebsonsecurity.com/2014/02/email-attack-on-vendor-set-up-breach-at-target

[3] Krebs, B. (2014), "A First Look at the Target Intrusion, Malware," Accessed: 2017-01-27.
URL http://krebsonsecurity.com/2014/01/a-first-look-at-the-target-intrusion-malware/

[4] Finkle, J. and D. Skariachan (2013), "Target cyber breach hits 40 million payment cards at holiday peak," Accessed: 2017-01-27.
URL http://www.reuters.com/article/us-target-breach-idUSBRE9BH1GX20131219

[5] Falliere, N., L. O. Murchu, and E. Chien (2011) "W32. stuxnet dossier," *White paper, Symantec Corp., Security Response*, **5**, p. 6.

[6] St. John, J. (2011), "Reports Claim First-Ever Cyber Attack on US Utility," Accessed: 2017-01-27.
URL https://www.greentechmedia.com/articles/read/hackers-claim-first-u.s.-utility-cyberattack

[7] Fudenberg, D. and J. Tirole (1991) *Game Theory.*, The MIT Press.
URL http://ezaccess.libraries.psu.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=11352&site=ehost-live&scope=site

[8] Roy, S., C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya, and Q. Wu (2010) "A survey of game theory as applied to network security," in *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*, IEEE, pp. 1–10.

[9] Van Dijk, M., A. Juels, A. Oprea, and R. L. Rivest (2013) "FlipIt: The game of "stealthy takeover"," *Journal of Cryptology*, **26**(4), pp. 655–713.

[10] Laszka, A., G. Horvath, M. Felegyhazi, and L. Buttyán (2014) "FlipThem: Modeling targeted attacks with FlipIt for multiple resources," in *International Conference on Decision and Game Theory for Security*, Springer, pp. 175–194.

[11] Pita, J., M. Jain, J. Marecki, F. Ordóñez, C. Portway, M. Tambe, C. Western, P. Paruchuri, and S. Kraus (2008) "Deployed ARMOR protection: the application of a game theoretic model for security at the Los Angeles International Airport," in *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: industrial track*, International Foundation for Autonomous Agents and Multiagent Systems, pp. 125–132.

[12] Tsai, J., C. Kiekintveld, F. Ordonez, M. Tambe, and S. Rathi (2009) "IRIS-a tool for strategic security allocation in transportation networks," *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009)*.

[13] Fang, F., A. X. Jiang, and M. Tambe (2013) "Optimal Patrol Strategy for Protecting Moving Targets with Multiple Mobile Resources," in *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '13, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 957–964.
URL `http://dl.acm.org/citation.cfm?id=2484920.2485072`

[14] Kar, D., F. Fang, F. Delle Fave, N. Sintov, and M. Tambe (2015) ""A Game of Thrones": When Human Behavior Models Compete in Repeated Stackelberg Security Games," in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '15, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp. 1381–1390.
URL `http://dl.acm.org/citation.cfm?id=2772879.2773329`

[15] Noureddine, M. A., A. Fawaz, W. H. Sanders, and T. Başar (2016) "A Game-Theoretic Approach to Respond to Attacker Lateral Movement," in *International Conference on Decision and Game Theory for Security*, Springer, pp. 294–313.

[16] Rass, S. and Q. Zhu (2016) "GADAPT: A Sequential Game-Theoretic Framework for Designing Defense-in-Depth Strategies Against Advanced Persistent Threats," in *International Conference on Decision and Game Theory for Security*, Springer, pp. 314–326.

[17] Holm, H., K. Shahzad, M. Buschle, and M. Ekstedt (2015) "P2 CySeMoL: Predictive, Probabilistic Cyber Security Modeling Language," *IEEE Transactions on Dependable and Secure Computing*, **12**(6), pp. 626–639.

[18] Roy, A., D. S. Kim, and K. S. Trivedi (2012) "Attack countermeasure trees (ACT): towards unifying the constructs of attack and defense trees," *Security and Communication Networks*, **5**(8), pp. 929–943.

[19] Wang, L., S. Jajodia, A. Singhal, and S. Noel (2010) *k-Zero Day Safety: Measuring the Security Risk of Networks against Unknown Attacks*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 573–587.

[20] Horn, P. (2001) "Autonomic computing: IBM's Perspective on the State of Information Technology," .

[21] Kephart, J. O. and D. M. Chess (2003) "The vision of autonomic computing," *Computer*, **36**(1), pp. 41–50.

[22] Yuan, E., N. Esfahani, and S. Malek (2014) "A Systematic Survey of Self-Protecting Software Systems," *ACM Trans. Auton. Adapt. Syst.*, **8**(4), pp. 17:1–17:41.
URL `http://doi.acm.org/10.1145/2555611`

[23] Stakhanova, N., S. Basu, and J. Wong (2007) "A Cost-Sensitive Model for Preemptive Intrusion Response Systems." in *AINA*, vol. 7, pp. 428–435.

[24] Kantert, J., H. Scharf, S. Edenhofer, S. Tomforde, J. HÃĎhner, and C. MÃijller-Schloer (2014) "A Graph Analysis Approach to Detect Attacks in Multi-agent Systems at Runtime," in *2014 IEEE Eighth International Conference on Self-Adaptive and Self-Organizing Systems*, pp. 80–89.

[25] Huang, Y., D. Arsenault, and A. Sood (2006) "Closing cluster attack windows through server redundancy and rotations," in *Cluster Computing and the Grid, 2006. CCGRID 06. Sixth IEEE International Symposium on*, vol. 2, pp. 12 pp.–21.

[26] Kennedy, C. M. (2008) "Distributed Meta-Management for Self-Protection and Self-Explanation," *SCHOOL OF COMPUTER SCIENCE RESEARCH REPORTS-UNIVERSITY OF BIRMINGHAM CSR*, **3**.

[27] Holm, H. (2014) "A Large-Scale Study of the Time Required to Compromise a Computer System," *IEEE Transactions on Dependable and Secure Computing*, **11**(1), pp. 2–15.

[28] Ross, S. (2014) "The Exponential Distribution and the Poisson Process," in *Introduction to Probability Models (Eleventh Edition)* (S. Ross, ed.), eleventh edition ed., Academic Press, Boston, pp. 277 – 356.
URL http://www.sciencedirect.com/science/article/pii/B9780124079489000050

[29] CRAMÉR, H. (1999) "Sampling Distributions." in *Mathematical Methods of Statistics (PMS-9)*, Princeton University Press, pp. 341–415.
URL http://www.jstor.org/stable/j.ctt1bpm9r4.12

[30] McHugh, J., A. Christie, and J. Allen (2000) "Defending yourself: The role of intrusion detection systems," *IEEE software*, **17**(5), pp. 42–51.

[31] Farnham, G. and A. Atlasis (2013) "Detecting DNS tunneling," *InfoSec Reading Room*.

[32] Leversage, D. J. and E. J. Byres (2008) "Estimating a system's mean time-to-compromise," *IEEE Security & Privacy*, **6**(1).

[33] Inci, M. S., B. Gülmezoglu, G. I. Apecechea, T. Eisenbarth, and B. Sunar (2015) "Seriously, get off my cloud! Cross-VM RSA Key Recovery in a Public Cloud." *IACR Cryptology ePrint Archive*, **2015**, p. 898.

[34] Liu, F., Y. Yarom, Q. Ge, G. Heiser, and R. B. Lee (2015) "Last-Level Cache Side-Channel Attacks are Practical," in *2015 IEEE Symposium on Security and Privacy*, pp. 605–622.

[35] Irazoqui, G., T. Eisenbarth, and B. Sunar (2015) "S$A: A Shared Cache Attack That Works across Cores and Defies VM Sandboxing – and Its Application to AES," in *2015 IEEE Symposium on Security and Privacy*, pp. 591–604.

[36] Jagatic, T. N., N. A. Johnson, M. Jakobsson, and F. Menczer (2007) "Social Phishing," *Commun. ACM*, **50**(10), pp. 94–100.
URL http://doi.acm.org/10.1145/1290958.1290968

[37] Holm, H., M. Ekstedt, and D. Andersson (2012) "Empirical Analysis of System-Level Vulnerability Metrics through Actual Attacks," *IEEE Transactions on Dependable and Secure Computing*, **9**(6), pp. 825–837.

[38] United States Department of Defense (1985) *Trusted Computer System Evaluation Criteria*, *Tech. rep.*

[39] Nzoukou, W., L. Wang, S. Jajodia, and A. Singhal (2013) "A Unified Framework for Measuring a Network's Mean Time-to-Compromise," in *2013 IEEE 32nd International Symposium on Reliable Distributed Systems*, pp. 215–224.

[40] Gallagher, S. (2013) "The Navy's newest warship is powered by Linux," Accessed: 2017-03-02.
URL https://arstechnica.com/information-technology/2013/10/the-navys-newest-warship-is-powered-by-linux/

[41] McQueen, M. A., W. F. Boyer, M. A. Flynn, and G. A. Beitel (2006) "Quantitative Cyber Risk Reduction Estimation Methodology for a Small SCADA Control System," in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)*, vol. 9, pp. 226–226.