2017-09

Multi-armed bandit models of network intrusion in the cyber domain

Kronzilber, Dor

Monterey, California: Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

MULTI-ARMED BANDIT MODELS OF NETWORK INTRUSION IN THE CYBER DOMAIN

by

Dor Kronzilber

September 2017

Thesis Co-Advisors:	Roberto Szechtman
	Ruriko Yoshida
Second Reader:	Moshe Kress

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704–0188

1. AGENCY USE ONLY *(Leave Blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 2017	Master's Thesis 12-01-2016 to 09-01-2017

4. TITLE AND SUBTITLE

MULTI-ARMED BANDIT MODELS OF NETWORK INTRUSION IN THE CYBER DOMAIN

5. FUNDING NUMBERS

6. AUTHOR(S)

Dor Kronzilber

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: N/A.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

We model attacks against computer networks in the cyber domain from the attacker's point of view. We consider an attacker with limited resources and time, whose goal is to maximize the expected reward earned by exploiting infected computers, while considering the risks. A computer network is represented as a graph consisting of computers or routers, where each computer has unknown expected reward and the routers connect sub-networks of computers. At time zero the attacker starts from an infected computer, called the "home computer," while all the other computers in the network are not infected. In any given period, the attacker can try to earn a reward by exploiting the subset of infected computers, or can choose to expand by infecting adjacent computers and routers, which does not accrue any reward. However, each infected computer must be connected through other infected computers all the way to the "home computer" for the attacker to be able to exploit it (but this connectivity may be lost when attacks are detected). For the linear network model, which is a worst-case scenario from the attacker point of view, we find that the optimal number of nodes to attempt to infect is of the order square root of the time when the network is sufficiently large. Also, we determine a critical relationship between the attacker's probability to infect a new node and the probability of detection. When this critical condition is met, the attacker should not try to infect any additional nodes.

14. SUBJECT TERMS

multi-armed bandit, cyber intrusion, computer network, advanced persistent threat

15. NUMBER OF PAGES 75

16. PRICE CODE

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

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

MULTI-ARMED BANDIT MODELS OF NETWORK INTRUSION IN THE CYBER DOMAIN

Dor Kronzilber
Major, Israel Defense Forces
B.Sc., Hebrew University of Jerusalem, 2008.
M.Sc., Tel Aviv University, 2016.

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 2017

Approved by: Roberto Szechtman
Thesis Co-Advisor

Ruriko Yoshida
Thesis Co-Advisor

Moshe Kress
Second Reader

Dr. Patricia Jacobs
Chair, Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

We model attacks against computer networks in the cyber domain from the attacker's point of view. We consider an attacker with limited resources and time, whose goal is to maximize the expected reward earned by exploiting infected computers, while considering the risks. A computer network is represented as a graph consisting of computers or routers, where each computer has unknown expected reward and the routers connect sub-networks of computers. At time zero the attacker starts from an infected computer, called the "home computer," while all the other computers in the network are not infected. In any given period, the attacker can try to earn a reward by exploiting the subset of infected computers, or can choose to expand by infecting adjacent computers and routers, which does not accrue any reward. However, each infected computer must be connected through other infected computers all the way to the "home computer" for the attacker to be able to exploit it (but this connectivity may be lost when attacks are detected). For the linear network model, which is a worst-case scenario from the attacker point of view, we find that the optimal number of nodes to attempt to infect is of the order square root of the time when the network is sufficiently large. Also, we determine a critical relationship between the attacker's probability to infect a new node and the probability of detection. When this critical condition is met, the attacker should not try to infect any additional nodes.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

THIS PAGE INTENTIONALLY LEFT BLANK

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

Executive Summary

This research deals with modeling attacks against computer networks in the cyber domain. This problem received considerable attention recently, as computer networks became ubiquitous in government and industry, and as cyber attacks became more common [1]. As a consequence, considerable investments are being made in developing defensive and attack capabilities, the latter mostly driven by government agencies [2] and criminal syndicates.

This work offers a framework for an attacker to plan his or her operation given a limited prior knowledge on the target network. As a result, this thesis gives the attacker a set of rules of thumb to guide his decision making towards a policy that is close to optimal. We present qualitative operational insights that apply in generality.

We study in detail the linear network configuration, where computers are linked in a single chain. A linear network is a worst-case scenario from the attacker's point of view because it is hard to expand within the network, and detection of an attack eliminates access to the rest of the network. We suggest a policy for an attacker with limited time (to exploit or expand) and information on the network: infect a precise number of computers before any exploitation takes place, stop infecting, and exploit for the rest of the time. We analytically derive the optimal number of computers to infect, which depends on the attacker's time horizon. Also, we find theoretical guarantees for the performance of this policy, and verify them by simulation.

We compare our policy to an oracle who knows the real values of all the computers in the network in advance. This allows the oracle to achieve an optimal expected reward by solving a dynamic program. The measure of performance for the recommended policy is the gap between the oracle's expected reward and that of the proposed policy by the end of the time horizon. We find that the optimality gap decreases to zero at a rate slightly larger than one over square root of the time horizon. When detections are not possible, the attacker should be indifferent to the value of the probability of successful attack (i.e., probability of infecting a node when attacking it), and in general should not invest his resources in that manner.

When introducing risk of attack detection, we find a critical relationship between attack

capability and detection probability, for which the attacker should decide to avoid any attempt to attack at all. In general, under the suggested model, the attacker should invest in improving the attack capabilities, rather than on detection avoidance. This result also yields a tool for the decision maker to capture the trade-off between investments in attack and evasion capabilities

Last, we offer preliminary work for more advanced and realistic models, leaving the details for further research.

References

[1] "NATO: Changing gear on cyber defense," 2016. Available:http://www.nato.int/docu/review/2016/Also-in-2016/cyber-defense-nato-security-role/EN/index.htm

[2] “U.S. intelligence community budget,” 2017. Available: https://www.dni.gov/index.php/what-we-do/ic-budget

Acknowledgments

I would like to thank my advisors for guiding me through the experience of theoretic research. It was an exciting journey thanks to your advice, devotion and enthusiasm. You opened my eyes to the front line of academic research in operations research.

You made me feel as an equal research partner and a friend rather than a student. For that I am greatly thankful.

I would like to thank my second reader for his comments and keeping me on the target.

More important, I am grateful for all of them for the extraordinary hospitality, which made my family feel at home when living in a foreign country.

I also thank my wife, Topaz, who supported my research along the year and made it possible.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

This chapter provides the background and the motivation for our problem. The novel aspects of our work and contribution are presented as well.

1.1 Background and Motivation

This research deals with modeling attacks against computer networks in the cyber domain. This problem has received considerable attention recently, as computer networks became ubiquitous in government and industry and as cyber attacks became more common [1]. As a result, considerable investments are being made in developing defensive and attack capabilities, the latter mostly driven by government agencies [2] and criminal syndicates.

In the recent years, cyber warfare (either offensive or defensive in nature) became common for different government agencies. This work offers a framework for an attacker to plan his operation given a limited prior knowledge on the target network. (For convenience, for the rest of this work we refer to the attacker as a male.) As a result, this will give the attacker a set of rules of thumb to guide his decision making toward a policy that is close to optimal. We aim to get qualitative operational insights that apply in generality.

From a broader perspective, this research is applicable to any sequential decision problem in a multi-armed bandit (MAB) setting with similar characteristics of risk, stochastic outcomes, cost for expanding, and a graph structure.

1.2 Model and Approach

The attacker starts from a single *infected* node under his control; for example, a secretary's computer with a password known to the attacker. This node is the attacker's "home node." At each turn, the attacker can gather intelligence from this node or try to expand to an adjacent node. Only infected nodes that are connected to the "home node" can be exploited and yield reward. Our meaning of reward is generic, and depends on the attacker's individual utility. Since our original motivation for the model is intelligence collection with respect

to a request for information, we often talk about an item being relevant (in which case the reward is one), or irrelevant (with a zero reward).

We assume that the attacker has some prior knowledge on a part of the network structure (topology and some characteristic parameters, such as probability of being detected while attacking a node), and also a prior distribution for the value of the intelligence assets on the nodes. That is, there is a prior for the probability μ that a node yields a 1 (meaning, relevant item) or a 0 (meaning, irrelevant item). Specifically, we assume a Beta prior for the collection of parameters μ_i for each node i; this assumption makes the model tractable because the Beta distribution is conjugate for the Bernoulli distribution.

A risk for the attacker is to be detected while trying to infect a new node; the risk of detection while exploiting an infected node typically is very small, and assumed to be zero. If detected, the source node from where the attack was launched and all its edges are removed from the graph. Given that, the attacker tries to maximize the expected number of relevant items collected from the network nodes, taking into account the risk of his actions, over some finite time horizon determined by the attacker.

We use an MAB framework for a policy that allows learning, and incorporate a methodology to consider the opportunities and risks associated with expanding the network under the control of the attacker.

1.3 Novelty

The novelty of our approach is in applying an MAB framework to a network whose arms need to be acquired: in a given period, the attacker can choose between exploring/exploiting the part of the network under his control (and thus earning a random reward) or expanding the infected sub-network amenable to exploration/exploitation. Another novelty is that we force the attacker to have connectivity back to his entrance node in order to retrieve information. These two problem conditions make the model much more realistic, and under some reasonable assumptions it is analytically tractable. From a more general perspective, in relation to the exploration and exploitation phases classical in MAB settings, we introduce a novel phase: infection.

1.4 Research Questions

The main operational issues we address in this work are:

- What are the characteristics of a good policy for the attacker? Can an optimal policy be found?
- How far should an attacker expand into the network? What parameters might affect the attacker's behavior?
- What is the value of information about the network for the attacker?
- How sensitive should the attacker be to the risk of detection and his capabilities to infect new computers?

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Background and Literature Review

In this chapter, we give the reader a brief overview of the theoretic background to this thesis and guidance to relevant sources of information.

2.1 Multi-armed Bandit Problem

The multi-armed bandit problem is a model for a gambler standing in front of several slot machines, who needs to decide on the sequence of arms to be played (i.e., a sequential sampling policy). Each slot machine generates prizes from a distribution that is unknown to the gambler. For each coin played, the gambler chooses a specific machine to play. The gambler needs to decide at each turn which machine to play in order to maximize the expected sum of rewards earned through the sequence of trials. Without any prior knowledge on the arms' distributions, the gambler needs to balance "exploiting" the best machine so far and "exploring" new machines that might be better. The idea is that without exploring every arm, and just focusing on the arm that appears to be the best, the gambler can get stuck in a bad arm due to a sequence of prizes that favors the sub optimal arm. Essentially, the MAB is a learning problem through sequential trials. The key in the MAB setting is to balance the exploration of the bad arms against the exploitation of the arm that appears to be the best.

2.1.1 Formulation

Next, we discuss the main aspects of the classical MAB problem. Let us denote:

- K as the number of bandits (slot machines). The bandits can be seen as a set of distributions. Each of the distributions has an expected value for the reward given to the gambler.
- $\mu_1, \mu_2, \ldots \mu_K$ as the distributions' means (unknown to the gambler, and finite).
- n as the time horizon; that is, the number of turns played (equivalent to the gambler's number of coins).

- $R(n)$ as the regret, which is the difference between the reward obtained by always playing the best arm, and the sum of the collected rewards by the gambler. Letting $\mu^* = \max\{\mu_k\}$ and $r(t)$ be the reward collected at time t, the regret by time n is given by $R(n) = n\mu^* - \sum_{t=1}^{n} r(t)$.
- $E[R(n)] = n\mu^* - E\left[\sum_{t=1}^{n} \mu_{I_t}\right]$ as the expected regret by time n, where I_t is the arm played at time t, and the expectation is with respect to the distribution of rewards up to time t and, possibly, of the arms selected before time t.

A special case is when the rewards are binary, 1 or 0, and reward distributions are Bernoulli with parameter μ_k (hidden to the gambler). For a reward distribution F with support over $(0, 1)$ with mean $\tilde{\mu}$, we can define a Bernoulli random variable X with parameter $\tilde{\mu}$ (and hence with the same mean since $E(X) = P(X = 1) = \tilde{\mu}$). It follows that the Bernoulli MAB has the same expected performance as the MAB for rewards drawn from F, so the Bernoulli reward setting includes as a special case any distribution with support over $[0, 1]$.

2.1.2 Lower Bound

Lai and Robbins [3], and later Burnetas and Katehakis [4], established the lower bound on the number of sub optimal draws for any sampling policy to be

$$\liminf_{n\to\infty} E[N_n(j)] \geq \frac{1}{KL(\mu_k.\mu^*)} \log(n), \tag{2.1}$$

where $N_n(j)$ is the number draws for the sub optimal arm j after n turns, and KL is the Kullback-Liebler divergence between probability distributions of arm k and the best arm under consideration. This result means that the best expected regret achievable for a gambler is of order $\log n / KL(\mu_k.\mu^*)$ plus other terms that are sub-logarithmic (i.e., $o(\log n)$).

Setting the stage for Chapter 3, in this thesis we deal with Bernoulli distributions with parameter μ_j for arm j, and μ^* is the best available parameter (i.e., $\mu^* > \mu_j$ for all j, and there is a single best arm). As the distributions are Bernoulli, the only two possible outcomes are 0 and 1, and the KL divergence is:

$$KL(\mu_j; \mu^*) = \mu_j \log \frac{\mu_j}{\mu^*} + (1 - \mu_j) \log \frac{1 - \mu_j}{1 - \mu^*}. \tag{2.2}$$

The KL divergence in Equation (2.2) equals zero for $\mu_j = \mu^*$, and is non-negative and

convex when viewed as a function of μ_j. Also, it approaches infinity as μ_j goes to zero or one. We use these facts later in Chapter 3.

As a closing remark, the lower bound for the expected regret in Equation (2.1) is very important because it can be used as a baseline to evaluate candidate policies; that is, a policy is considered good if its expected reward is close to the right-hand side in Equation (2.1).

2.2 Literature Review for Multi-armed Bandits

MAB problems have been extensively researched in the last several decades; see [5] and [6], and references therein. The research in this thesis is using some findings and notions from the following papers.

Auer, Cesa-Bianchi and Fischer [7] suggest policies for finite-time analysis of MAB problems based on upper confidence bounds (UCB). As in the MAB, the measure of performance is the expected regret (i.e., the expected difference in cumulative rewards between the suggested policy and the optimal course of action if the parameters were known). They suggest several different policies for K-armed problem, where the K machines have an arbitrary reward distribution with support in $[0, 1]$, and for the Gaussian reward case. They prove an upper bound on the expected regret that matches (up to the leading term) the best possible lower bound shown in Equation (2.1). Their policy for random rewards with support in $[0, 1]$, labeled *UCB1* by the authors, is shown next.

1. **Initialization:** Play each machine once.

2. **Loop:**

   For each machine calculate

$$R_j = \bar{x}_j + \sqrt{\frac{2\log n}{n_j}},$$

   where

   n - overall number of plays done so far.

   $\bar{x}_j$ - the average reward obtained from machine j.

   n_j - the number of times machine j has been played.

3. **Pull:** machine j that with the biggest R_j.

4. **Return** to step 2.

The authors of the paper prove an upper bound for the expected regret under this policy

$$E[R_n] \leq \left[8 \sum_{i:\mu_i<\mu^*} \left(\frac{\log n}{\Delta_i}\right)\right] + \left(1 + \frac{\pi^2}{3}\right)\left(\sum_{j=1}^{K} \Delta_j\right), \tag{2.3}$$

where $\Delta_j = \mu^* - \mu_j$ is the difference between the expected reward of the best arm to the j'th arm.

Two items are of notice: First, the expected regret achieves the logarithmic growth mentioned earlier, albeit with a coefficient that is larger than one over the KL divergence; second, the upper bound is a function only of the expected regret values $(\mu_1, \ldots, \mu_K)$ of the K different machines. We will use the second fact in our work.
Several other researchers take the same *upper confidence bound* of the UCB1 algorithm, and improve the constant and additive coefficients that appear in Equation (2.3). However, none of these approaches result in a $\log n/KL(\mu_j, \mu^*)$ finite time convergence.

Kaufman, Cappe and Garivier [8] further investigate the ideas presented in [7], by applying a Bayesian framework using a prior distribution for the reward of the K machines. Specifically, they consider Bernoulli distributed rewards in a Bayesian setting where

each arm has a Beta prior for the Bernoulli parameter μ_k.
In their algorithm, at time t the gambler plays the arm with largest $1 - 1/t$ posterior quantile,

$$\max_{k=1,\ldots,K} \left\{ Q(1 - 1/t; \alpha_{k,t}, \beta_{k,t}) \right\}.$$

For an arm k the value of $Q(1 - 1/t; \alpha_{k,t}, \beta_{k,t})$ is the value of x that solves

$$1 - 1/t = F_{\text{Beta}}(x; \alpha_{k,t}, \beta_{k,t}), \tag{2.4}$$

where $F_{\text{Beta}}(x; \alpha_{k,t}, \beta_{k,t})$ is the Beta cumulative distribution function with parameters $(\alpha_{k,t}, \beta_{k,t})$. Equation (2.4) is solved numerically.

This approach performs very well in relation to UCB methods in terms of expected regret; see the numerical section in [8]. Also, it is especially tractable because the posterior for μ_k is Beta distributed with parameters $\alpha_t = \alpha_0 + I_1(t)$, and $\beta_t = \beta_0 + I_0(t)$, where $I_1(t)$ and $I_0(t)$ are the number of collected prizes up to time t that were equal to 1 and 0 respectively, and Beta(α_0, β_0) is the initial prior. That is, the Beta distribution is a conjugate prior for the Bernoulli distribution.

Why is the term $1/t$ in the posterior quantile? Motivated by the lower bound on the expected regret in Equation (2.1), we want the probability of sampling from a suboptimal node to be of order $1/t$ by stage t, so that the expected number of pulls from that arm is of order $\log n$ over a time horizon n. Sampling from the arm with largest $1 - 1/t$ posterior quantile leads to P(sampling a sub optimal arm) of order $1/t$; see [8] for the details.

The algorithm is presented next.

1. **Require:** n (horizon), Π^0 (initial prior), c (parameters of the quantile).
2. **For** $t = 1$ **to** n **do:**
3. **For** each machine j calculate:

$$q_j(t) = Q\left(1 - \frac{1}{t(\log n)^c}; \alpha_{j,t-1}, \beta_{j,t-1}\right).$$

4. **Draw** a sample from arm j^* with the biggest $q_j(t)$,

$$X_{j^*} \sim \text{Ber}(\mu_{j^*}).$$

5. **Update**

$$\alpha_{j^*,t} = \alpha_{j^*,t-1} + X,$$

$$\beta_{j^*,t} = \beta_{j^*,t-1} + 1 - X.$$

In their paper, the horizon-dependent term $(\log n)^c$ is said to be an artifact of the theoretical analysis that guarantees finite-time logarithmic regret bound for $c \geq 5$. It is emphasized that in simulations, the choice of $c = 0$ is proved to be the most satisfying.

For a binary reward bandit problem, they proved an upper bound for the maximum number of draws for each sub optimal machine for binary reward bandit problem (which can be easily converted to an upper bound on the expected regret). For any $\epsilon > 0$ and $c \geq 5$, the authors show that

$$E[R_n] \leq \sum_{k=1, \mu_k \neq \mu^*}^{K} \frac{\mu^* - \mu_k}{KL(\mu_k, \mu^*)}(1 + \epsilon)\log(n) + o_{\epsilon,c}(\log(n), \tag{2.5}$$

where $KL(\mu_j, \mu^*)$ is the Kullback-Leibler divergence (see Section 2.1.2), and $o_{\epsilon,c}(\log(n))$ means a term that is dominated by $\log(n)$ asymptotically for any given ϵ and c.

By using the lower bound of sub optimal draws given by Lai and Robbins in [3] (see 2.1.2), the leading term in this result is the same as the lower bound and hence UCB Bayes is leading-order optimal.

Wang, Audibert and Munos [9] suggest a formulation for the so-called infinitely many-armed bandits problem (i.e., the number of arms is larger than the possible number of experiments), where the expected reward for each arm is random. They suggest an optimal number of arms (K) to be randomly drawn from the set of arms, and then apply UCB-V exploitation policy on this set. (UCB-V is an extension on UCB1 that includes a variance estimator, rather than using a worst case bound on the variance). The expected regret is affected by the number of arms K in two opposite ways:

- Term I: The expected regret caused by not having the best arm in the set of K randomly chosen arms. This term is a function of chosen number of arms K, the parameters of the distribution, and a linear term corresponding to the time horizon. This term decreases as the player chooses a bigger set of arms (bigger K).

- Term II: The expected regret caused by exploring sub optimal arms among the K arms drawn. This term is the cost of learning until finding the best arm among the chosen K arms, and depends on the specific sampling algorithm (UCB-V in this paper), the time horizon, and the number of arms. This term increases linearly with the number of arms.

Wang et al. employ UCB-V on a set of K arms policy as exploitation method:

1. At time t, play an arm in the set maximizing $B_{k,T_k,(t-1),t}$ where:
   - $B_{k,T_k,(t-1),t} \triangleq \bar{X}_{k,s} + \sqrt{\frac{2V_{k,s}\mathcal{E}_t}{s}} + \frac{3\mathcal{E}}{s}$.
   - Number of times arm k was chosen by the policy during the first t plays: $T_k(t)$.
   - The mean reward estimator: $\bar{X}_{k,s} \triangleq \frac{1}{s}\sum_{j=1}^{s} X_{k,j}$.
   - The variance estimator: $V_{k,s} \triangleq \frac{1}{s}\sum_{j=1}^{s}(X_{k,j} - \bar{X}_{k,s})^2$.
   - $\mathcal{E}_t$ is a non-decreasing sequence of nonnegative real numbers (typically $\log t$).
2. Set $t = t + 1$ and go back to 1.

The authors prove an upper bound for the regret as a function of the number of arms K,

$$E[R_n] \leq C\left\{(\log K)nK^{-1/\beta} + K(\log n)E\left[(\frac{V(\Delta)}{\Delta} + 1) \wedge (n\Delta)\right]\right\},$$

where n is the time horizon, $\Delta = \mu^* - \mu$ is the difference between the expected reward

of an arbitrary arm (μ) and the best one available (μ^*), C is a positive constant, and β is a parameter that depends on the prior distribution for the rewards.

The authors determine an optimal (in terms of expected regret) number of arms to draw, and then apply an exploitation policy (UCB-V) on this set. In Chapter 3, we extend these ideas to include the acquisition of arms (i.e., infection of arms via Bernoulli trials) in a linear network setting, with the exploration of arms done using Bayesian UCB [8].

2.3 Literature Review for Cyber Models

In this section we comment on several recent papers devoted to cyber security. The literature on this area is vast and dynamic, so we make no claim of completeness. The three papers below not only deal with "cyber," but also employ MAB techniques.

Liu and Zhao [10] consider a large-scale cyber network, with components that are either in a normal or abnormal state, evolving over time according to an arbitrary stochastic process. The defender can select a fixed number of components to probe and fix if they are in the abnormal state. The authors describe and analyze a policy that minimizes the infinite horizon cost incurred by the components in abnormal states. The methodological technique is rooted in restless MAB.

Zheng, Shroff, and Mohapatra [11] model a defender again so-called advanced persistent threats that pose two types of cost: security updates, and cost of being attacked. The authors develop upper confidence bound techniques that produce low regret in relation to the policy that assumes knowledge of the attack times distribution.

Qian, Zhang, Krishnamachari, and Tambe [12] study the problem faced by a defender of a number of assets subject to attack. The issue is that the defender cannot protect all the assets simultaneously. In their model, the attack intensity in the unprotected assets changes according to a Markov process with unknown transition matrix, while the attack intensity follows a Markov process with different transition matrix for the assets protected by the defender. The issue for the defender is to learn the transition matrices over time, so as to optimally allocate the finite resources at his disposal. The authors use restless MAB ideas to formulate and solve the problem.

Elderman, Pater, Thie, Drugan, and Wiering [13] examine an attacker-defender stochas-

tic zero-sum game, where the goal of the attacker is to reach a specific node located in a known place in the network. The attacker and defender allocate resources along the intermediate nodes that lie between the starting node and the node targeted by the attacker. The authors discuss several strategies for both the attacker and defender, rooted in approximate dynamic programming, that are suitable when the game is played repeatedly.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:
Problem Formulation and Methodology

In this chapter we present the problem formulation, our approach to the problem, and the theory behind it. We consider a conscious attacker in the cyber domain whose goal is to maximize the intelligence yield from a computer network, given that he has access to only a single computer – the "home node." The attacker can infect new computers and network devices (routers) for exploring the network, and can exploit each infected computer to yield intelligence. The intelligence yield from each infected computer is random, with a distribution not known to the attacker. Routers don't provide any intelligence, but give some information about the expected intelligence value contained in the computers connected to it.

Our baseline is a multi-armed bandit (MAB) model. We adapt the basic MAB framework to our setting by introducing uncertainties, risks, graph structure, network devices, and the fact that in order to exploit an infected computer the attacker has to have a path from the home node.

This chapter is organized as follows. In Section 1 we provide background for linear networks comprised uniquely of computers. Section 2 deals with the best-case scenario – a perfect attacker who knows the expected reward in each computer. In Section 3 we discuss the model when the attacker has a uniform prior for the reward distribution parameters. In Section 4 we analyze the case when failed attacks may be detected. Section 5 extends the linear network case to include network devices (routers). In the last section we analyze the case when the prior on the reward distribution parameters is not uniform.

3.1 Background

We study a linear network comprising M nodes – meaning M nodes connected in a chain topology. All of the nodes are computers with expected reward $\mu_1, \ldots, \mu_M$ which are generated from a Beta distribution when the network is created. We assume that the attacker knows the parameters of the generating Beta distribution, because it symbolizes intelligence about his target network. As discussed in Chapter 2, when the rewards have

a Bernoulli distribution we end up with a Beta posterior for the reward probabilities, thus making the problem analytically tractable.

The attacker always starts with an infected computer. This is the "home node", which enables the attacker access to the network and allows sending the gathered intelligence back to the attacker's headquarters. A linear network is illustrated in Figure 3.1. In the figure, the attacker has only one infected node (in red) – the "home node."

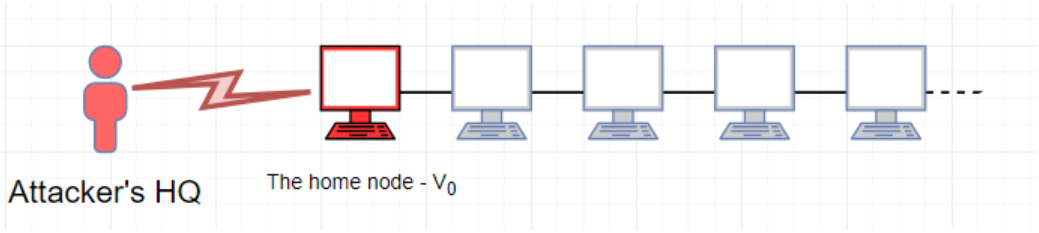

Figure 3.1. Linear Network Illustration.

We consider two special cases:

- The prior parameters of the Beta distribution are $\alpha_0 = \beta_0 = 1$, meaning that μ_i's have a uniform distribution $U(0, 1)$.
- At least one of the prior parameters is not equal to one, so the prior distribution for the μ_i's is no longer $U(0, 1)$.

Several Beta probability density functions for different parameter combinations are shown in Figure 3.2.

As it turns out, the two special cases above result in different optimal attack policies, so we study them separately in later sections. In the rest of this section we explore the aspects they share in common.

The attacker wishes to maximize the expected total reward derived from sampling the computers under his control, over a discrete time horizon n. The linear network case is relatively easy to analyze because the subnetwork connected to the home node is linear, implying that in any epoch the attacker has two main alternatives:

- **Explore or exploit:**
  The attacker draws a single random reward – a zero or one, drawn from a Bernoulli

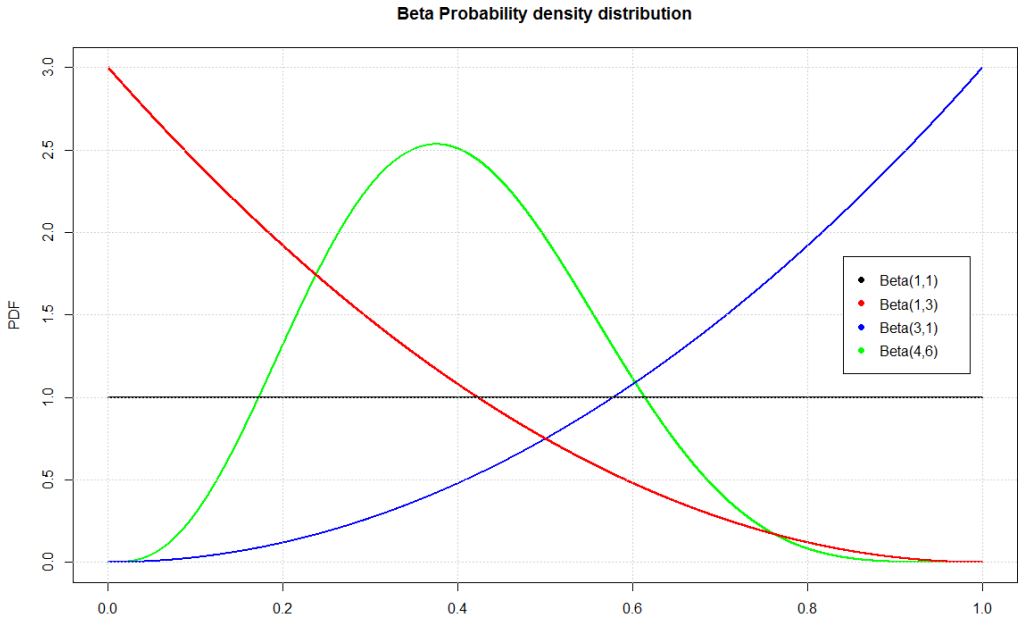

Figure 3.2. Beta Distribution with Various Parameters.

distribution with unknown parameter μ – from one of the computers already infected. By *exploring* we mean that the attacker attempts to cast a wider net and gain information about computers that have not been sufficiently sampled so far. The exact meaning of "sufficiently" depends on the specific algorithm but it typically is of order log(time). Conversely, *exploiting* refers to sampling from one of the computers that appear to be the best so far (i.e., that have the largest posterior mean). The attacker evaluates each computer i via the posterior distribution for its expected reward

$$\mu_i \sim \text{Beta}(\alpha_0 + \#\text{ of 1's so far}, \beta_0 + \#\text{ of 0's so far}).$$

As the number of observations for a computer gets large, the posterior distribution becomes approximately Gaussian, centered at the posterior mean

$$\frac{\alpha_0 + \#\text{ of 1's so far}}{\alpha_0 + \beta_0 + \#\text{ of observations so far}},$$

with a variance of order one over the number of observations so far. For example, when $(\alpha_0 = 40, \beta_0 = 40)$ the maximum difference with respect to a Gaussian probability

density function with matching mean and variance is 0.0017. Hence, as the number of observations gets large, the attacker gains a degree of certainty about the reward he can expect to obtain by sampling from each computer.

- **Attack:**

  The attacker attempts to infect a computer connected to the infected subnetwork, so that it can be explored/exploited in the following periods. Due to the linear network structure, there is only one uninfected computer connected to the infected subnetwork (see Figure 3.1), the one connected to the right-most infected computer. Attacks succeed with a constant probability s, and are independent of each other. Failed attacks are detected with a constant probability d (those probabilities are assumed to be the same for all the computers in the network). If a failed attack is detected, then the attacker loses access to the computer from which he attacked, and to the uninfected part of the network. In the linear network case, this means that,

  - ⋆ The attacker ends up with control over the original infected subnetwork minus the rightmost infected computer (from where the attack was launched).
  - ⋆ The attacker only can explore/exploit the computers that remain under his control for the duration of the operation. Infecting other computers is no longer possible.

  In particular, when $d = 0$ it follows that the number of trials to infect a new computer has a Geometric distribution with parameter s (known to the attacker), so its expectation is $1/s$. Intuitively, when the attack probability is low in relation to the remaining time horizon, the attacker is better off staying put, and explores/exploits the computers under his control.

A judicious policy balances exploration, exploitation, *and infection*; the latter is the main methodological contribution of our work. The main operational questions we wish to answer are:

1. How far out should the attacker infect computers in the network?
2. What is the expected performance of a smart attacker in relation to an attacker who knows all the μ_i values? This is a measure of the learning that takes place.
3. What are the effects of the successful attack probability s and of the detection probability d?

3.2 Perfect Attacker

A perfect attacker has full and precise information about the network, meaning that the values of all computers $\mu_1, \ldots, \mu_M$ are known in advance, and there is no need to do any exploration (no learning). At each turn, the attacker decides whether to stay and exploit the computers he already has, or try to infect the next computer in the network (which involves uncertainty regarding the success of infection and possible detection). An illustration of a linear network from the perfect attacker's point of view is shown in Figure 3.3.

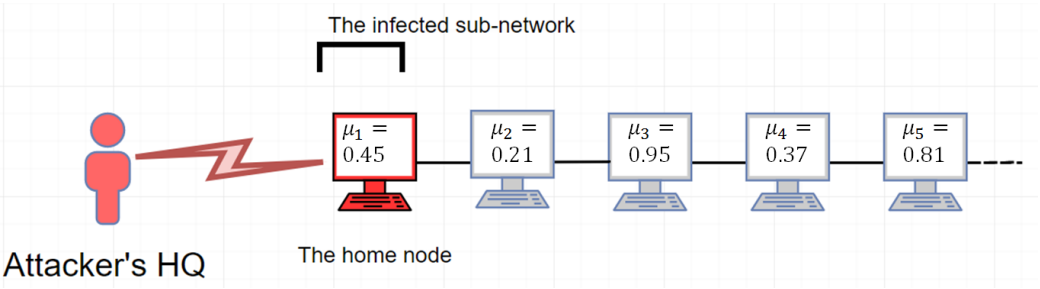

Figure 3.3. Linear Network for a Perfect Attacker.

Knowing the values of the computers, a perfect attacker can act optimally by solving a dynamic program to handle the randomness of each attack success or failure. Let $V^*_{(n,K)}$ be optimal value function from the attacker's point of view given n remaining turns and K infected computers; that is, $V^*_{(n,K)}$ is the maximum expected reward that the attacker might hope to obtain by time n. The problem can be formulated recursively as

$$V^*_{(n,K)} = \max \left\{ \underbrace{\mu^*_{1:K} + V^*_{(n-1,K)}}_{\text{stay}}, \right.$$
$$\left. \underbrace{sV^*_{(n-1,K+1)} + (1-s)(1-d)V^*_{(n-1,K)} + (1-s)d(T-1)\mu^*_{1:K-1}}_{\text{expand}} \right\}, \quad (3.1)$$

where $\mu^*_{1:K}$ is the best computer in the subnetwork consisting of the first K computers,

$$\mu^*_{1:K} = \max_{j=1,\ldots,K} \mu_j. \tag{3.2}$$

The attacker's dilemma is between staying and exploiting the current best computer for the next n turns, or try to infect the next one – including the possibility of a failed attack (a wasted turn) and the possibility of detection (the right-most infected computer v_K will be removed and so will his the access to the rest of the network).

An optimal decision can be made by solving the full dynamic program. The stopping step is when there is only one turn to go – then it is obvious that a "stay" decision is superior (because the attacker does not have enough turns to infect a new computer and explore it). An efficient implementation of the above dynamic programming approach is possible by storing the previous results instead of recalculating them.

3.2.1 Properties

We summarize the analysis of the perfect attacker scenario next. Later in this chapter we compare our suggested policy for attackers who do not know the rewards parameters $\mu_1, \ldots, \mu_M$ to the perfect attacker.

Proposition 3.2.1 *$V^*_{(n,K)}$ is non-decreasing with respect to n.*

Proof By the definition of $V^*_{(n,K)}$ and the max operator:

$$V^*_{(n,K)} \geq \mu^*_{1:K} + V^*_{(n-1,K)},$$

since $\mu^*_{1:K} \geq 0 \quad \forall x$. ∎

Proposition 3.2.2 *$V^*_{(n,K)}$ is a threshold policy – once decided to stay, the attacker will not attack any more. The dynamic programming recursion is,*

$$V^*_{(n,K)} = \max\left\{\underbrace{n\mu^*_{1:K}}_{stay}, \underbrace{sV^*_{(n-1,K+1)} + (1-s)(1-d)V^*_{(n-1,K)} + (1-s)d(n-1)\mu^*_{1:K-1}}_{expand}\right\}. \quad (3.3)$$

Proof Let assume that at n turns to go and at location K the attacker decided to exploit and not attack. Hence,

$$V^*_{(n,K)} = \mu^*_{1:K} + V^*_{(n-1,K)} \geq sV^*_{(n-1,K+1)} + (1-s)(1-d)V^*_{(n-1,K)} + (1-s)d(n-1)\mu^*_{1:K-1}$$

implies

$$\mu^*_{1:K} \geq sV^*_{(n-1,K+1)} + (sd-s-d)V^*_{(n-1,K)} + (1-s)d(n-1)\mu^*_{1:K-1}.$$

Now at the next turn, where there are $T-1$ turns to go,

$$\mu^*_{1:K} + V^*_{(n-2,K)} \geq sV^*_{(n-1,K+1)} + (sd-s-d)V^*_{(n-1,K)} + (1-s)d(n-1)\mu^*_{1:K-1} + V^*_{(n-2,K)}.$$

We already showed that $V^*_{(n,K)}$ is non-decreasing with respect to n, so $V^*_{(n-1,K)} \geq V^*_{(n-2,K)}$, and

$$\mu^*_{1:K} + V^*_{(n-2,K)} \geq sV^*_{(n-2,K+1)} + (sd-s-d)V^*_{(n-2,K)} + (1-s)d(n-2)\mu^*_{1:K-1} + V^*_{(n-2,K)}.$$

Therefore,

$$\mu^*_{1:K} + V^*_{(n-2,K)} \geq sV^*_{(n-2,K+1)} + (1-s)(1-d)V^*_{(n-2,K)} + (1-s)d(n-2)\mu^*_{1:K-1}.$$

The attacker will choose again to stay and not attack. ∎

Proposition 3.2.3 *$V^*_{(n,K)}$ is non-decreasing with respect to K.*

Proof We will prove separately for each case of the max argument.

If $V^*_{(n,K)} = n\mu^*_{1:K}$, then using the fact that $\mu^*_{1:K}$ is a non decreasing function of K,

$$V^*_{(n,K+1)} \geq n\mu^*_{1:K+1} \geq n\mu^*_{1:K} = V^*_{(n,K)}.$$

In case $V^*_{(n,K)} = sV^*_{(nT-1,K+1)} + (1-s)(1-d)V^*_{(n-1,K)} + (1-s)d(n-1)\mu^*_{1:K-1}$, we prove the statement by contradiction. Let's assume that $V^*_{(n,K)}$ is a decreasing function in K. Hence,

$$V^*_{(n,K)} \leq sV^*_{(n-1,K)} + (1-s)(1-d)V^*_{(n-1,K)} + (1-s)d(n-1)\mu^*_{1:K-1}.$$

By Proposition 3.2.2, $V^*_{(n,K)}$ is non decreasing in n,

$$V^*_{(n,K)} \leq sV^*_{(n,K)} + (1-s)(1-d)V^*_{(n,K)} + (1-s)d(n-1)\mu^*_{1:K-1},$$

so that

$$V^*_{(n,K)} \leq V^*_{(n,K)}(1-d+sd) + (1-s)d(n-1)\mu^*_{1:K-1}.$$

Hence $V^*_{(n,K)} \leq (n-1)\mu^*_{1:K-1}$. But, from our assumption,

$$V^*_{(n,K)} \geq T\mu^*_{1:K} > (n-1)\mu^*_{1:K-1}.$$

The last inequality is greater and not greater-equal because $\mu^*_{1:K} > 0, \forall K$. We have contradiction. ∎

Proposition 3.2.4 *Once the attacker infects the computer with the best value he will decide to stay.*

Proof Without loss of generality, we assume the $\mu_K = \mu^*_{1:M}$ and the attacker infected all the computers up to computer v_K. Now we prove that the attacker will always decide to stay.

By Proposition 3.2.3, $V^*_{(n,K)} \leq V^*_{(n,M)}$. Clearly, after infecting the entire network (all M computers), the only possibility is to "stay" and exploit. Hence:

$$V^*_{(n,K)} \leq V^*_{(n,M)} = n\mu^*_{1:M} = n\mu^*_{1:K}.$$

From the other side, by definition $V^*_{(n,K)} \geq n\mu^*_{1:K}$. So $V^*_{(n,K)} = n\mu_{1:K}$, meaning that it's

optimal for the attacker to "stay." Thus, by Proposition 3.2.2, the attacker will not try to attack any more. ∎

From the operational standpoint, the main take away from these results is that the optimal policy is threshold-type: The attacker attempts to expand without any exploitations and, once the expansion phase ceases, the attacker exploits the best arm under his control for the duration of the operation.

The perfect attacker model presents a best-case scenario for the attacker, because he knows the real values of all the computers in the network and does not need to do any learning for finding the best one. In the next sections we study the more realistic case when an attacker only has a prior for the reward parameters $\mu_1, \ldots, \mu_M$, and develop algorithms whose performance approaches that of the perfect attacker. Why should a wise attacker invest in perfect information when systematic learning is good enough?

3.3 Uniform Prior Scenario

In this scenario we consider an attacker who controls the first K computers by time t, and whose prior distribution for the expected reward of each computer is $U(0,1) \stackrel{D}{=} \text{Beta}(\alpha_0, \beta_0)$, with $\alpha_0 = \beta_0 = 1$. As discussed above, the attacker can explore/exploit an infected computer, or attack the next uninfected computer in the chain. Typically, the number of computers in the network M is larger than the time horizon n; so, for all practical purposes, the attacker views the network size as infinite. The case when $M \leq n$ is a special scenario of our analysis below, and is discussed at the end of Section 3.3.2. We initially assume that attacks cannot be detected (i.e., $d = 0$); this assumption is relaxed in Section 3.4.

3.3.1 Exploration and Exploitation of Computers

Let K be the number of infected computers out of M computers in the network. In the explore/exploit case, the attacker samples from the computer with largest $1 - 1/t$ posterior quantile, $Q(1 - 1/t; \alpha_{v,t}, \beta_{v,t})$; see Equation (2.4).

As discussed in Chapter 2, the main result of the Bayesian UCB approach is that an upper bound (c.f., Equation (2.5)) on its expected regret is of the same order as the lower bound

for the Bernoulli reward setting (recall Equation (2.1)),

$$\log n \sum_{i=1, \mu_i \neq \mu^*_{1:K}}^{K} \frac{\mu^*_{1:K} - \mu_i}{KL(\mu_i, \mu^*_{1:K})}, \tag{3.4}$$

where $\mu^*_{1:K}$ is defined in Equation (3.2), and $KL(x, p)$ is the KL divergence for Bernoulli random variables with parameters p and x (c.f., Equation (2.2)). Hence, Bayesian UCB is considered "optimal," as its expected regret is sandwiched between two bounds with the same leading order.

Since each μ_i is $U(0, 1)$ distributed, stopping attempts to infect other computers and exploring/exploiting leads to an expected regret of order

$$E\left[\sum_{i=1, \mu_i \neq \mu^*_{1:K}}^{K} \frac{\mu^*_{1:K} - \mu_i}{KL(\mu_i, \mu^*_{1:K})}\right] \log n,$$

where the expectation is with respect to K independent standard uniform distributions (on the section $(0, 1)$).

In order to study the expectation above, without loss of generality, we write $\mu_{(1)} \geq \mu_{(2)} \geq \cdots \geq \mu_{(K)}$ for the ordered statistics of the μ_i values. Then, we have

$$\sum_{i=2}^{K} E\left[\frac{\mu_{(1)} - \mu_{(i)}}{\mu_{(i)} \log(\mu_{(i)}/\mu_{(1)}) + (1 - \mu_{(i)}) \log(\frac{1-\mu_{(i)}}{1-\mu_{(1)}})}\right] \leq .5 \sum_{i=2}^{K} E\left[\frac{1}{\mu_{(1)} - \mu_{(i)}}\right], \tag{3.5}$$

by Pinkser's inequality (essentially, it is a quadratic lower bound on the KL function). The random variable $\mu_{(1)} - \mu_{(i)} \stackrel{D}{=} \text{Beta}(i - 1, K - i + 2)$; this can be seen by putting $K + 1$ uniforms on a circular rope of length 1, and choosing one of them as the baseline (call it $\mu_{(K)}$ and set it to 0). Then,

$$\mu_{(1)} - \mu_{(i)} \stackrel{D}{=} \mu_{(i-1)} - \mu_{(K)} \stackrel{D}{=} \mu_{(i-1)} - 0 \stackrel{D}{=} \text{Beta}(i - 1, K - i + 2),$$

where the last equality is a well-known result that relates a Beta random variable to the

order statistics of $U(0,1)$ samples; see, for example [14]. Therefore, Equation (3.5) equals

$$.5\sum_{i=2}^{K} E\left[\frac{1}{\text{Beta}(i-1,K-i+2)}\right] = .5\sum_{i=2}^{K}\frac{K}{i-2} \approx .5K\log K,$$

the first equality holds since one over a Beta random variable also is a Beta with inverted parameters (this can be shown by direct integration), and the $\approx$ is justified by approximating the sum by $\int 1/x$. The division by zero in the last approximation can be resolved by a finer analysis of Pinkser's inequality (see [15]). While numerical results via Monte Carlo simulation (discussed in the next chapter) suggest that the upper bound is conservative, the conservatism does not have a significant impact on the attacker's optimal policy because the approximation does not affect the leading order term (this also is illustrated in Chapter 4).

In conclusion, the expected regret due to not attempting to infect any more computers when K computers already are infected, and $n-t$ turns remain, is bounded above by

$$.5K\log K\log(n-t). \tag{3.6}$$

The expression in Equation (3.6) is equivalent to a bound on the price in performance the attacker expects to pay for finding the best computer out of the K he has already infected. If he knew a priori the best computer out of computers $1,\ldots,K$, then this price would be zero, because there is no need for a learning process and the attacker will always exploit the best computer within the K infected computers.

As we will see momentarily, the attacker attempts to infect arms non-stop up to a point, and from there he just exploits/explores the acquired arms. Hence, the Bayesian UCB algorithm is run over the set of "unexplored" infected arms, all of them with a uniformly distributed prior for the Bernoulli parameter μ.

3.3.2 Infecting Computers

Next we study the *infection* aspect of the attacker's problem. The benefit of infecting more computers is that it increases the chance of acquiring a computer with high value. The negative side of the coin is that, due to the uncertain success/fail nature of the attacks, the

attacker has less time to explore/exploit the computers he already controls.

To study the problem we consider an attacker who wishes to minimize his cumulative expected regret, given by

$$E[R_n] = \sum_{t=1}^{n} (\text{best possible reward at } t - E[\text{reward at } t]), \tag{3.7}$$

where we choose the best possible reward at time t, to be the maximum of M uniforms. As mentioned before, we work in the scenario where the length of the network $M \geq n$, so the attacker can infect at most n computers. Hence, we choose the best possible reward to be the maximum of n uniforms. Regarding the second term in Equation (3.7), the expected reward of the attacker's action at time t is 0 if he decides to attack a new computer.

If the attacker attempts to infect K computers one after the other, then Equation (3.7) is bounded above by

$$\underbrace{\frac{K-1}{s}}_{\text{expected time to acquire } K-1 \text{ computers}} + \underbrace{\left(n - \frac{K-1}{s}\right)\left(E[\mu^*_{1:n}] - E[\mu^*_{1:K}]\right)}_{\text{expected regret of ending up with the best of } K \text{ computers}} \tag{3.8}$$

$$+ \underbrace{.5K \log K E\Big[\log(n - (K-1) - \text{Negative Binomial}(K-1; 1-s))\Big]}_{\text{bound on expected regret of finding the best computer out of } K \text{ computers}}.$$

We compute the expected rewards of the best computer out of K computer $E[\mu^*_{1:K}]$ next,

$$E[\mu^*_{1:K}] = \int_0^1 P(\mu^*_{1:K} > x)dx = \int_0^1 (1 - F^K_{U(0,1)}(x))dx = 1 - \int_0^1 x^K dx$$

$$= 1 - \frac{1}{K+1}, \tag{3.9}$$

where $F_{U(0,1)}$ is the cdf of a $U(0,1)$ random variable. Hence, an upper bound on the expected regret is

$$E[R_n] \leq \frac{K-1}{s} + \left(n - \frac{K-1}{s}\right)\left(\frac{1}{K+1} - \frac{1}{n+1}\right) + .5K \log K \log n \tag{3.10}$$

$$= \frac{K-1}{s}\left(1+\frac{1}{n+1}-\frac{1}{K+1}\right)+n\left(\frac{1}{K+1}-\frac{1}{n+1}\right)+.5K\log K\log n$$

$$\leq \frac{K}{s}+\frac{n}{K}+.5K\log K\log n$$

$$\leq \frac{n}{K}+K\max\left(\log K\log n,\frac{2}{s}\right).$$

Therefore, for $K \geq 3 > \exp(1)$ and $\log n \geq 2/s$ (both realistic since $K \geq 1$ at time 0 and typically $s > 1/2$), the attacker solves the optimization problem

$$\min_{1\leq K\leq n} g(K,n), \tag{3.11}$$

where

$$g(K,n) = \frac{n}{K}+K\log K\log n.$$

Rather surprisingly, the attack success probability plays no role in how deep the attacker should go into the network. The operational implication of this insight is immediate: the attacker should not invest significant resources in increasing the probability of successful attacks (as long as $\log n \geq 2/s$). As we will see later in this chapter, this conclusion is no longer valid when failed attacks are detected with positive probability.

The optimization problem in Equation (3.11) can be solved numerically for any n given. In more generality, we have the bounds

$$\frac{n}{K}+K\log n \leq g(K,n) \leq \frac{n}{K}+CK^{1+\gamma}\log n, \tag{3.12}$$

for $\gamma > 0$ arbitrarily small and $C > 0$ finite, since $CK^{\gamma} > \log K$ for any $K \geq 1$. Optimizing the bounds in Equation (3.12) leads to

$$K^*(n) = O\left(\frac{n}{\log n}\right)^{1/2}. \tag{3.13}$$

With this value of the number of computers to infect, the expected regret (c.f., Equation (3.7)) becomes

$$E[R(n)] = O\left(n\log n\right)^{1/2}. \tag{3.14}$$

From this analysis, we conclude that, for $n \geq e^{2/s}$:

- The number of computers to infect grows roughly like the square root of the time horizon; see Equation (3.13).
- The expected regret grows like the square root of time times the logarithm of time; see Equation (3.14).

These insights are supported by the numerical examples in the next chapter.

To close this section we consider the scenario where the total number of computers in the network M is smaller than the time horizon n. In this case, the attacker infects every computer in the network as the time horizon n gets large (specifically, larger than $(M/\log M)^2$). In particular, the second term in Equation (3.8) disappears, and the expected regret is of order $M \log M \log n$. Hence, when the computer network is sufficiently small in relation to the time horizon, the attacker achieves the best possible regret rate (recall Equation (2.1) in Chapter 2).

3.3.3 Bayes UCB-I Algorithm

The analysis in Section 3.3.2 was done under the premise that the attacker tries to infect one computer after another, leaving the exploration/exploitation of the computers acquired till the end. In this subsection we explain why such threshold policy is optimal for the attacker and offer the Bayes UCB-I algorithm (I for "infect").

Suppose the attacker controls K computers and that n turns remain. If the attacker tries to infect one more computer, then (c.f., Equation (3.10)) his expected regret would be upper bounded by

$$\text{expand}(K, n) = \left(n - \frac{1}{s}\right)\left(\frac{1}{K+1} - \frac{1}{n+1}\right) + \frac{1}{s} + .5(K+1)\log(K+1)\log n. \tag{3.15}$$

On the other hand, if the attacker explores/exploits for the remaining n turns, then his expected regret is

$$\text{stay}(K, n) = \frac{n}{K} + .5K \log K \log n. \tag{3.16}$$

If $\text{expand}(K, n) \geq \text{stay}(K, n)$, so that the attacker is better off staying, then it follows from

a contradiction argument that expand$(K,t) \geq$ stay(K,t) for all $t = 1,\ldots,n-1$. This means that the attacker "stays" forever. On the other hand, if expand$(K,n) <$ stay(K,n), so that expanding out to the next node is preferable, then an attack takes place, the value of expand$(\tilde{K}, n-1)$ is updated depending to the results (i.e., $\tilde{K} = K$ or $K+1$), and the situation is re-evaluated. These ideas are summarized in the algorithm below.

Bayes UCB-I Algorithm

1. **Require:** n (horizon), $K = 1$ infected computers, uniform initial prior, $s > 0$ probability of successful attack
2. *Stay* = False
3. **While** $n > 0$
4. **Evaluate** $E[R(n,K)]$ by Equation (3.15–3.16).
5. **If** decision is *Expand*:
6. **Attack**
7. **If** success : update $K = K+1; n = n-1$
8. **If** failed : update $n = n-1$
9. **Else** decision is *Stay*
10. **Execute** Bayes-UCB algorithm on the K infected computers until the end of remaining time

An attacker with incomplete information who follows the Bayes UCB-I algorithm is henceforth called a "rational attacker."

The rational attacker will reach an optimal K value (given the time horizon) because the expression in Equation (3.16) is convex in K. It can be shown by taking the second derivative:

$$\frac{\partial}{\partial K}\text{expand}(K,n)) = \frac{-n}{K^2} + \log K \log n + \log n,$$

and

$$\frac{\partial^2}{\partial^2 K}\text{expand}(K,n)) = \frac{2n}{K^3} + \frac{\log n}{K}.$$

One can see that the second derivative exists and is positive for any $n > 1$, hence this function is convex. See Figures 3.4 and 3.5 for an illustration.

This theoretic analysis indicates that a rational attacker attempts to expand non-stop and, when he stops expanding, spends the remaining time turns exploring/exploiting from the infected computers.

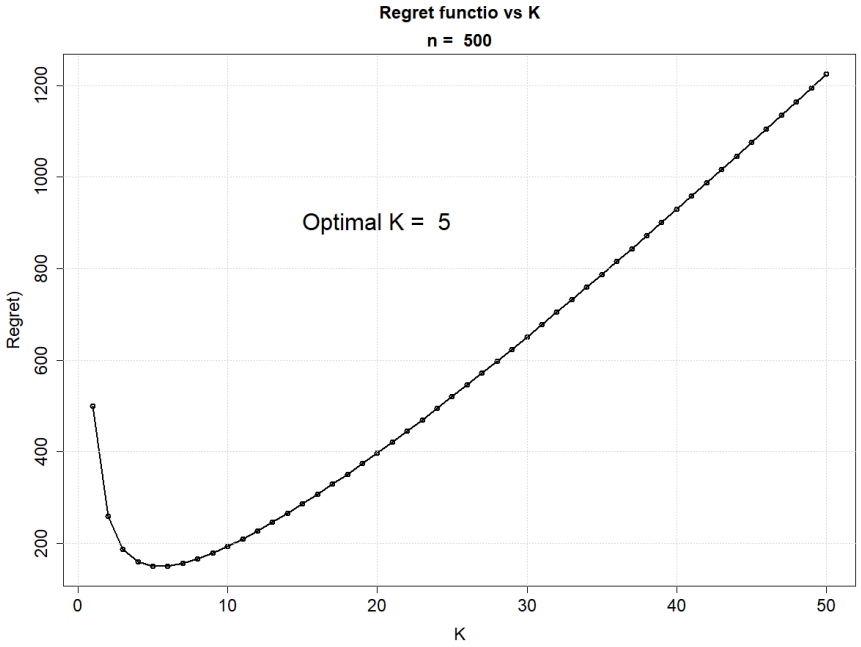

Figure 3.4. Regret vs. K Infected Computers for Time Horizon 500

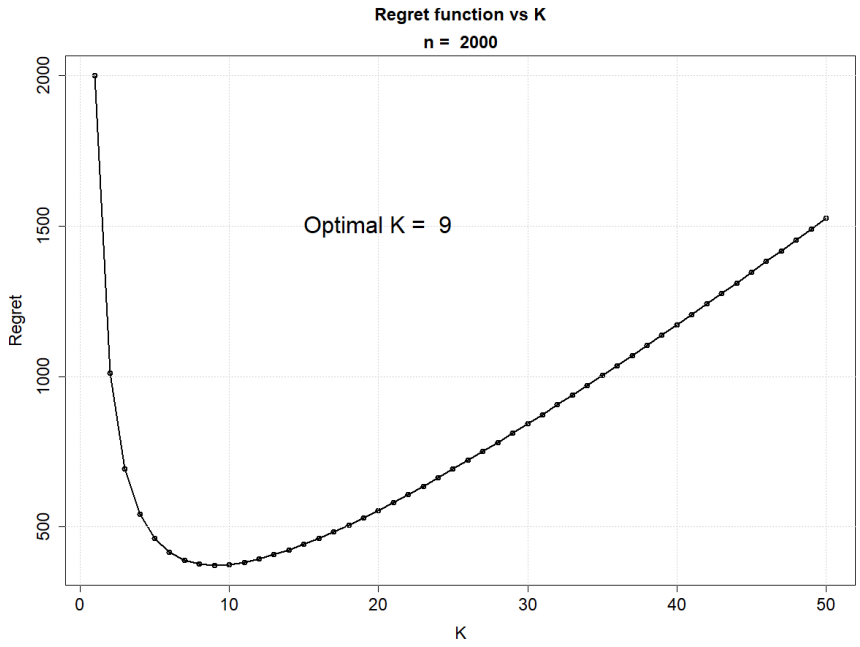

Figure 3.5. Regret vs. K Infected Computers for Time Horizon 2000

3.3.4 Performance Analysis

In this section, we compare the performance of a rational attacker who follows the Bayes UCB-I algorithm, against that of the perfect attacker, who knows the exact μ_i values in the network.

Recall from Equation (3.13) that the best number of computers to infect for a rational attacker is

$$K^*(n) = O\left(\sqrt{\frac{n}{\log n}}\right).$$

The expected regret for a perfect attacker emanates from two sources: the time spent infecting computers, and the expected regret of ending up with the best of K computers (vice the best of n computers). The perfect attacker does not need to do any learning, so he avoids the extra $\log n$ regret term caused by the learning process. On average (over many instances of networks), a perfect attacker infects up to K^*_{perfect} computers to minimize his expected regret. The expected regret for the perfect attacker as a function of K and n is

upper bounded by

$$E[R_{\text{perfect}}(n, K, s)] \leq \frac{K-1}{s} + \left(n - \frac{K-1}{s}\right)\left(\frac{1}{K+1} - \frac{1}{n+1}\right). \tag{3.17}$$

Numerical experiments (shown in Chapter 4) indicate that the upper bound is tight, so we optimize in K the expression above to obtain the best number of computers for the perfect attacker to infect, leading to

$$K^*_{\text{perfect}} = O(\sqrt{ns}), \tag{3.18}$$

so that

$$E[R_{\text{perfect}}(n, K^*_{\text{perfect}}, s)] = O\left(\sqrt{\frac{n}{s}}\right). \tag{3.19}$$

We define the optimality gap as the gap between a rational attacker's (Bayes UCB-I algorithm) and a perfect attacker's end reward:

$$\begin{aligned}\text{Gap}(n, s) &= \frac{E[\text{Reward}_{\text{perfect}}] - E[\text{Reward}_{\text{rational}}]}{E[\text{Reward}_{\text{perfect}}]} \\ &\approx \frac{\left(nE[\mu^*_{1:n}] - \sqrt{n/s}\right) - \left(nE[\mu^*_{1:n}] - \sqrt{n\log^3 n}\right)}{nE[\mu^*_{1:n}] - \sqrt{n/s}},\end{aligned}$$

where (recall Equation (3.9)) $E[\mu^*_{1:n}] = 1 - 1/(n+1)$. Therefore, as the time horizon n gets large we get:

$$\text{Gap}(n, s) = O\left(\sqrt{s\frac{\log^3 n}{n}}\right). \tag{3.20}$$

In summary, our analysis suggests that,

- The rational attacker's infection depth is proportional to $\frac{\sqrt{n}}{\log n}$ where n is the time horizon, without any dependency on the probability of attack (for big n).
- The perfect attacker infects computers to a depth of order $\sqrt{sn}$ into the network.

- The optimality gap decays to 0 as n increases at a rate of order $O\left(\sqrt{\frac{\log^3 n}{n}}\right)$.
- For big n, the rational attacker can expect a very small optimality gap without dependency on the probability of attack.

The practical implication is important for the attacker: If the time horizon is sufficiently large, there is no need to invest any intelligence resources to learn where is the highest value computer.

3.4 Model with Detection of Attacks

In this section we consider the case where a detection may occur after a failed attack with probability d (constant and the same for all the computers in the network). When a detection occurs, the origin of the attack (the right-most infected computer) gets removed from the network. Thus the attacker does not have access to any new computers.

An attacker who tries to attack node v_{k+1} from the right-most infected node v_k with n turns remaining, faces one of the following three outcomes:

- Success, with probability s: the attacker infects node v_{k+1}, losing one turn.
- Detection, with probability $(1-s)d$: the attacker loses computer v_k and remains only with the set $v_1, v_2, ..., v_{k-1}$ for the remaining $n-1$ turns.
- Nothing, with probability $(1-s)(1-d)$: the attacker just loses one turn.

In this section we analyze the effect of detections on the behavior and performance of a perfect attacker. We leave the analysis of the $d > 0$ scenario corresponding to the rational attacker for future work.

3.4.1 Perfect Attacker

On average, the perfect attacker stops expanding when the expected regret from expanding equals that of staying, so we solve for

$$E[R_p(n,K)] = sE[R_p(n-1,K+1)] + (1-s)(1-d)E[R_p(n-1,K)] + d(1-s)R_p(n-1,K-1),$$

where $E[R_p(n, K)]$ is defined in Equation (3.17) as $R_{\text{perfect}}(n, K)$). After simplifications we get the following:

$$\frac{n}{K(K+1)(K+2)}\Big[K(s-d(1-s))+sd-2d\Big]=1. \tag{3.21}$$

The term $sd - 2d$ can be neglected when $sd - 2d \ll K(s - d(1 - s))$; this can be made rigorous by controlling the approximation error in a neighborhood of the exact root in Equation (3.21). In the worst case $K = 1$, and then the inequality is equivalent to $s + d > 0$, which is always true. Thus, we ignore it and solve:

$$K^*_{\text{perfect}} = \sqrt{n(s - d(1 - s))}. \tag{3.22}$$

For $d = 0$, this result coincides with Equation (3.13). For $d > 0$, there is a discount factor of $d(1 - s)$ inside the square root when comparing to the original expression in Equation (3.13).

More interesting, a critical value of d with respect to s exists, for which there is no feasible solution. Meaning, for a given s value, there exists a level of risk d_c that the attacker will prefer to stay at the home node and will not infect any computer at all.

$$d_c = \frac{s}{1-s} \tag{3.23}$$

Plugging Equation (3.22) into $E[R_{\text{perfect}}(n, K)]$:

$$E[R_{\text{perfect}}] = \sqrt{n}\frac{(2s - d(1 - s))}{s\sqrt{s - d(1 - s)}}. \tag{3.24}$$

The result coincides with the former result as well when $d = 0$ (see Equation (3.19)). We expect the regret to increase rapidly as d gets closer to the critical value d_c. In Figures 3.6 and 3.7 we show the trends in Equation (3.22) for different values of probability of a successful attack.

One can see that the factor to the regret is infinite when d approach d_c. At this value, the attacker will tend to stay at the home node.

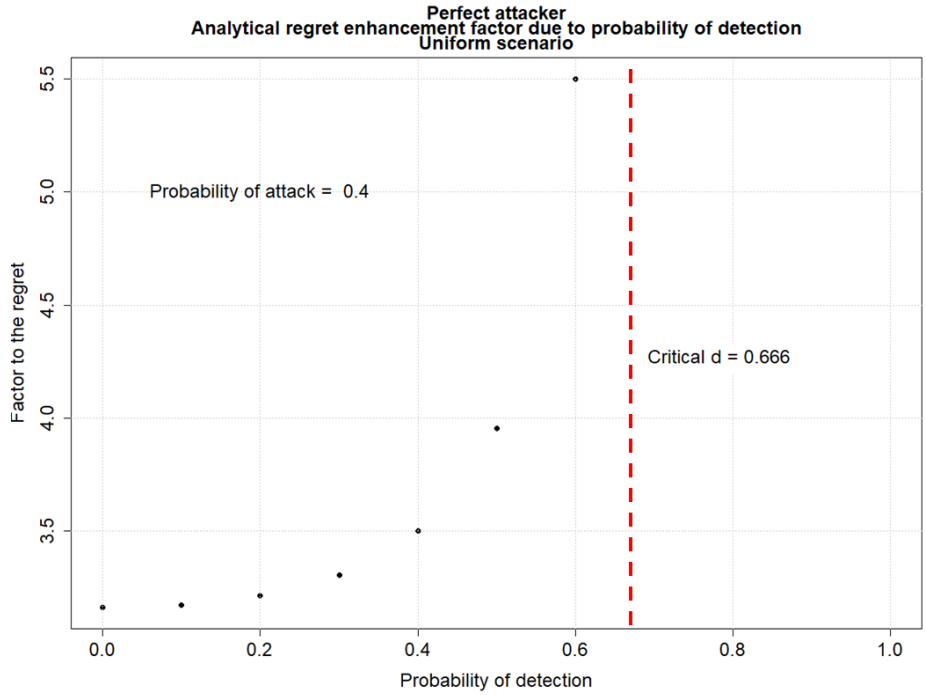

Figure 3.6. Enhancement Factor to the Regret vs. Probability of Detection, with $s = 0.4$.

3.5 Linear Network with Routers

In this section, we study network devices (routers), that don't contain any intelligence but can provide a value (for the attacker) by revealing information on the computers connected to them. The idea is that if the attacker acquires a router and explores it, he can "sniff" the traffic being transmitted by the computers within its subnetwork. To put things into our model framework, exploring a router yields no rewards, but allows the attacker to update the posterior distribution for the μ parameters of the computers that lie in its subnetwork. Hence, exploring a router might be desirable because it could shorten the exploration phase versus the alternative, which is to try to infect and then explore the computers in its subnetwork.

To keep the router model as simple as possible, we consider a star topology where the router lies at the center of the star, and M computers are connected to it (but the computers are disconnected with each other). We assume that at time zero the attacker controls the router but none of the computers are infected. Also, the attacker has a $U(0,1)$ prior for the expected Bernoulli rewards of each computer, and each exploration of the router

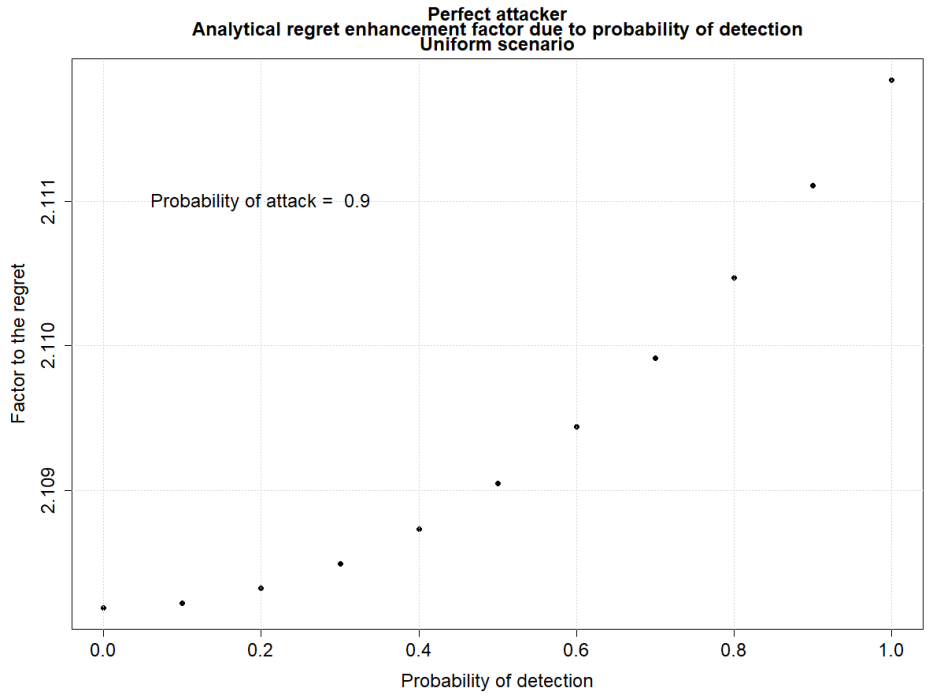

Figure 3.7. Enhancement Factor to the Regret vs. Probability of Detection, with $s = 0.9$.

yields ℓ (with $0 < \ell < \infty$) samples out of each computer connected to it (representing the information "packets" generated by each computer's network traffic). Typically, $\ell \ll 1$, which complicates the posterior updates. To overcome this issue without loss of generality, we set the time scale so that each turn yields $1/\ell$ (integer) samples out of each computer that could be individually explored. The key operational question for the attacker is: What are the characteristics of a good policy (in sense of total reward collected) to explore the router, and to attack, explore, and exploit the computers in its subnetwork?

Suppose initially that all the computers connected to a router are infected. From the proof of the Bayesian UCB [8] algorithm, in a first phase the attacker samples suboptimal computer k at most $\log n/KL(\mu_k, \mu^*)$ times. In a second phase, the probability that a suboptimal computer has the largest index is very small, because there are enough samples drawn from each computer in the first phase, so the chance that a bad computer has larger index than the best one is low. Hence, after the initial $\sum_{k=1,\mu_k \neq \mu^*}^{M} \log n/KL(\mu_k, \mu^*)$ turns, bad computers are sampled a few more times (up to logarithmic in n; see [8]).

Essentially, the first $\log n/KL(\mu_k, \mu^*)$ samples for each suboptimal computer k are wasted, meaning that the Bayesian UCB algorithm behaves as if each computer was a router (i.e., update the posterior without earning). The difference is that the router gives ℓ samples of each computer in its subnetwork each time it is sampled. Hence, the idea is that the router also should be sampled for $\sum_{k=1,\mu_k\neq\mu^*}^{M} \log n/KL(\mu_k, \mu^*)$ turns. Then, the regret accrued over the first phase becomes $\ell^{-1} \log n \sum_k \Delta_k/KL(\mu_k, \mu^*)$, which is smaller than $\log n \sum_k \Delta_k/KL(\mu_k, \mu^*)$ for ℓ sufficiently large. In other words, the router collapses all the bad computers into roughly $1/\ell$ computers.

From the implementation standpoint, there is the issue that $KL(\mu_k, \mu^*)$ is unknown (even μ^* is unknown). To overcome it, we know from the developments in Section 3.3.1 that

$$\frac{1}{KL(\mu_{(2)}, \mu_{(1)})} \leq \frac{1}{2(\mu_{(1)} - \mu_{(2)})^2},$$

and that $\mu_{(1)} - \mu_{(2)}$ is Beta distributed with parameters $(1, M + 1)$. From here we can find (numerically) any τ-quantile of the random variable $.5/(\mu_{(1)} - \mu_{(2)})^2$. By choosing the quantile large (i.e., τ close to 1), then we end up with a bound that holds with high probability. Intuitively, the quantile should be such that the probability of underestimating $.5/(\mu_{(1)} - \mu_{(2)})^2$ is of the same order of one over time, so that the contribution to the expected regret of the unlucky instances is constant.

These ideas suggest the following algorithm.

Algorithm with all computers already infected:

1. Sample the router the first $\ell^{-1} c \log n$ times, where c is the $1 - 1/n$ quantile of $.5/(\mu_{(1)} - \mu_{(2)})^2$.
2. Find the computer with the largest average reward, and sample only from it for the remaining time horizon.

The ideas above apply if all the computers connected to a router are infected. If none of them are infected, and failed attacks can be detected ($d > 0$), then the algorithm would be as follows.

Algorithm with detections:

1. Sample the router the first $\ell^{-1} c \log n$ times, where c is the $1 - 1/n$ quantile of $.5/(\mu_{(1)} - \mu_{(2)})^2$.
2. Find the computer with the largest average reward.
3. Try to infect it and sample only from it for the remaining time horizon.

A benefit of this algorithm is that only one computer needs to be infected, so less time is wasted infecting computers and the risk of detection is lessened.

3.6 Non-uniform Prior Scenario

In this section, we consider the more realistic scenario where the attacker has a Beta(α, β) prior for the expected rewards of the computers in the network. We follow the developments of Sections 3.3.1 and 3.3.2, and our goal is to find the number of computers to attack that minimize expected regret, in terms of the time horizon n. For simplicity we assume that the number of computers $M < n$, but the $M \geq n$ case easily follows along the lines in the last paragraph of Section 3.3.2.

Our starting point is Equation (3.8), where three terms make up a bound on the expected regret. The first term captures the expected number of turns that it takes the attacker to infect $K - 1$ extra computers, and remains unchanged in the non-uniform setting. The second term corresponds to the expected regret accrued by not infecting the best computer in the network,

$$\left(n - \frac{K-1}{s}\right)(E[\mu^*_{1:n}] - E[\mu^*_{1:K}]).$$

The $(n - (K-1)/s)$ part remains unchanged, while the $E[\mu^*_{1:n}] - E[\mu^*_{1:K}]$ term is affected by the non-uniform prior assumption. Classical results from extreme value theory (see p. 137 of [16]) indicate that a suitably scaled maximum of K IID Beta(α, β) random variables converges in distribution to a Weibull distribution, as K becomes large. The specific scaling leads to

$$E[\mu^*_{1:n}] - E[\mu^*_{1:K}] = O\left(\left(\frac{\beta B(\alpha, \beta)}{K}\right)^{1/\beta}\right), \tag{3.25}$$

where

$$B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha + \beta)} \tag{3.26}$$

is the incomplete beta function (and $\Gamma(\cdot)$ is the Gamma function). Monte Carlo simulations

indicate that the right-hand side in Equation (3.25) is a close upper bound for $E[\mu^*_{1:n}] - E[\mu^*_{1:K}]$ for all reasonable combinations of parameters α, β, n, and K.

The last term in Equation (3.8) captures the effect of finding out the best computer out of those infected by the attacker. From Equations (3.5) and (3.8), we now have

$$.5 \log n \sum_{i=2}^{K} E\left[\frac{1}{\mu_{(1)} - \mu_{(i)}}\right], \tag{3.27}$$

where $\mu_{(i)}$ is the i'th order statistic out of K IID Beta(α, β) random variables. The Monte Carlo simulations we ran suggest that $.5K \log K \log n$ remains a conservative upper bound for Equation (3.27) (even more so than for a uniform prior).

As in Equation (3.12), the attacker finds the value of K that solves

$$\min\left\{n\left(\frac{\beta B(\alpha, \beta)}{K}\right)^{1/\beta} + .5K \log n\right\}, \tag{3.28}$$

where the $\log K$ term is dropped from the optimization, due to a sandwich argument as in the discussion surrounding Equation (3.12). Optimizing Equation (3.28) leads to the optimal number of computers to infect being of order

$$K^*(n) = O\left(B(\alpha, \beta)\left(\frac{n}{\log n}\right)^{\frac{\beta}{1+\beta}}\right). \tag{3.29}$$

Several conclusions follow from Equation (3.29),

- The uniform case, $\beta = 1$ gives the same result as in Equation (3.13).
- As β increases (i.e., the probability that a computer has low expected reward gets larger), the attacker has to spread further out in the network; the converse is true in the other direction.
- The expected regret, obtained by substituting $K^*(n)$ for K in Equation (3.29), is of order

  $$E[R(n)] = O\left(n^{\frac{\beta}{1+\beta}}(\log n)^{1+\beta}\right),$$

  meaning that the attacker does not perform well for β large.

- In particular, the expected regret approaches linear growth with the time horizon n as β increases, so that learning does *not* take place. This is due to a catch-22 situation faced by the attacker: stopping the attack too soon may result in a bad set of computers, while expanding deeper into the network does not leave time to explore/exploit the computers acquired.
- Last, we mention the effect of the parameter α, which is manifested only through the Beta function in Equation (3.29). From Equation (3.26) we see that $B(\alpha, \beta)$ is largest when $\alpha \approx \beta$ and $\alpha + \beta$ is large; that is, when the prior is approximately Normal with variance of order $(\alpha + \beta)^{-1}$ (i.e., small). Conversely, $B(\alpha, \beta) \to 1$ as α and β become dissimilar. The operational implication is that the attacker has to spread deeper into the network in the first scenario, and less so in the second scenario (for the same value of β).

Since most computers do not contain valuable intelligence, computer networks within our model framework would tend to have high β parameter values; for example, for $\alpha = 3$ and $\beta = 10$, only 1% of the computers have a μ value larger than .54. In light of this, the bullet points above suggest that a smart attacker would find it challenging to extract intelligence from the most valuable computers. We leave for future work the optimality gap comparison with the perfect attacker.

CHAPTER 4:
Analysis and Results

In this chapter, we present numerical illustrations of the policies suggested in Chapter 3, and verify analytic results. In particular, we compare the rational attacker, who has a uniform prior on the expected rewards $\mu_1, \ldots, \mu_M$, to a perfect attacker who has full and accurate information about these parameters.

We focus on two metrics:

Infection index – K – how deep into the network each of the policies infect (perfect attacker and the rational attacker applying the Bayes UCB-I algorithm). This is useful to verify the rules of thumb presented in Chapter 3.

Optimality gap – Gap – comparing the average end reward of rational attacker to a perfect attacker's end reward. This gives the bottom line for how good our algorithm is and what is the equivalent cost of perfect information.

We verify analytic results for those metrics as a function of the time horizon (n), probability of attack (s), and probability of detection (d). Each replication of the Monte Carlo simulation consists of the following steps.

1. Randomly generate the $\mu_1, \ldots, \mu_M$ values from M IID $U(0, 1)$ random variates.
2. Generate IID $S \sim \text{Ber}(s)$ random variates, and apply them to
   (a) a perfect attacker who solves the dynamic program in Equation (3.1),
   (b) a rational attacker who follows the Bayes UCB-I algorithm.

For each setting, the pseudo-code above is repeated 200 times. In the model with detections, the second step above includes the generation of IID Bernoulli(d) random variates corresponding to the detections.

This chapter follows the structure of Chapter 3. We start with numerical examples of linear networks with a uniform prior, followed by the perfect attacker, rational attacker, and concluding with the model with detections. Numerical examples for the linear network with routers and non-uniform prior are left for future work.

4.1 Perfect Attacker

As defined in Chapter 3, we use the term "perfect attacker" for an attacker who knows the exact values of the rewards parameters $\mu_1, \ldots, \mu_M$ of all computers in the network, and can execute the optimal policy suggested in Chapter 3 (i.e., solving a dynamic program).

In this subsection, we demonstrate the behavior of a perfect attacker and the effect of different parameters on his policy. We are interested in how far into the network a perfect attacker is expected to infect.

We define our nominal base line case as:

$$\{s = 0.9, d = 0, \alpha_0 = 1, \beta_0 = 1\}.$$

The characteristics of this baseline case are as follows.

- High success probability ($s = 0.9$): the attacker can infect easily new computers.
- No risk ($d = 0$): there is no risk of being detected.
- Infinite network; we assume an effectively infinite network, meaning that the time horizon is not sufficient to infect the entire network: $M \geq n$. In hindsight, the perfect attacker will not tend to infect the entire network, because on average, the best cost-effective computer (in sense of μ value versus how deep it is in the network) will not be at the end of the network. It is sufficient to assume $M = 200$ for any time horizon $n < 10,000$ (significant in reducing running time).
- Uniformly distributed μ values: $\mu_i \sim \text{Beta}(\alpha_0 = 1, \beta_0 = 1)$ for $i = 1, \ldots, M$.

4.1.1 Mean Infection Index

For example, for specific time horizon of $n = 200$, a histogram of final infection index (which is how deep the attacker infects) is shown in Figure 4.1. The histogram is based on the result of playing the perfect attacker on 200 randomly generated networks. The estimated infection index was approximately 9 computers (shown as a red vertical line in Figure 4.1).

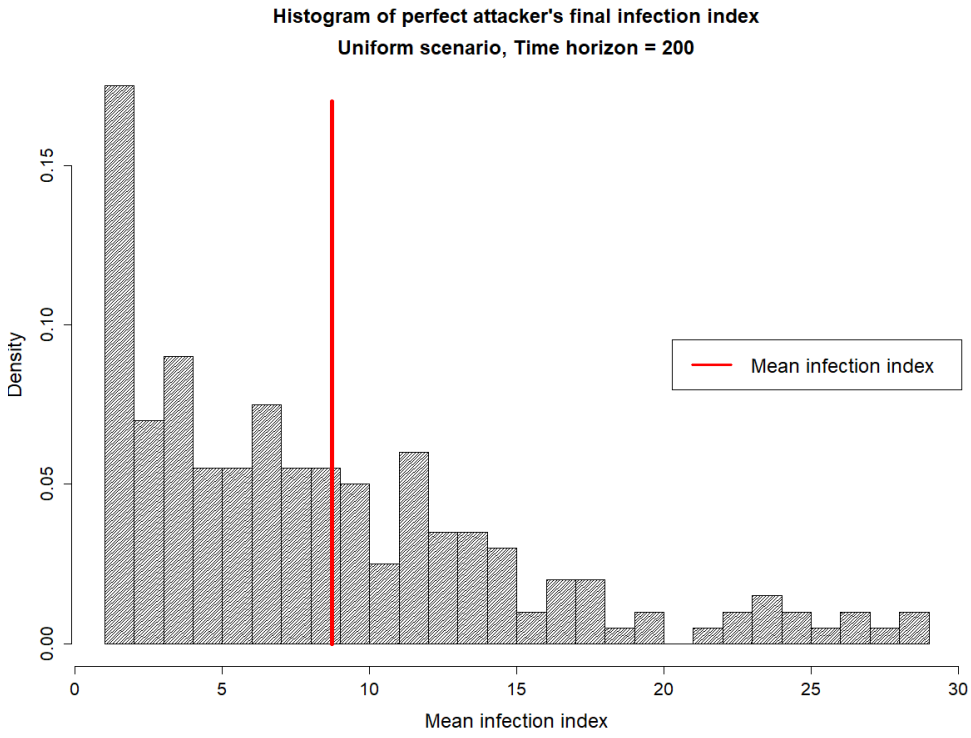

Figure 4.1. Histogram of Infection Index – Perfect Attacker with $n = 200$.

We repeated this experiment for different values of time horizon (i.e., playing a perfect attacker on 200 randomly generated networks on each value of time horizon). Figure 4.2 shows the estimate and 95% confidence band for the mean infection index K as a function of time horizon.

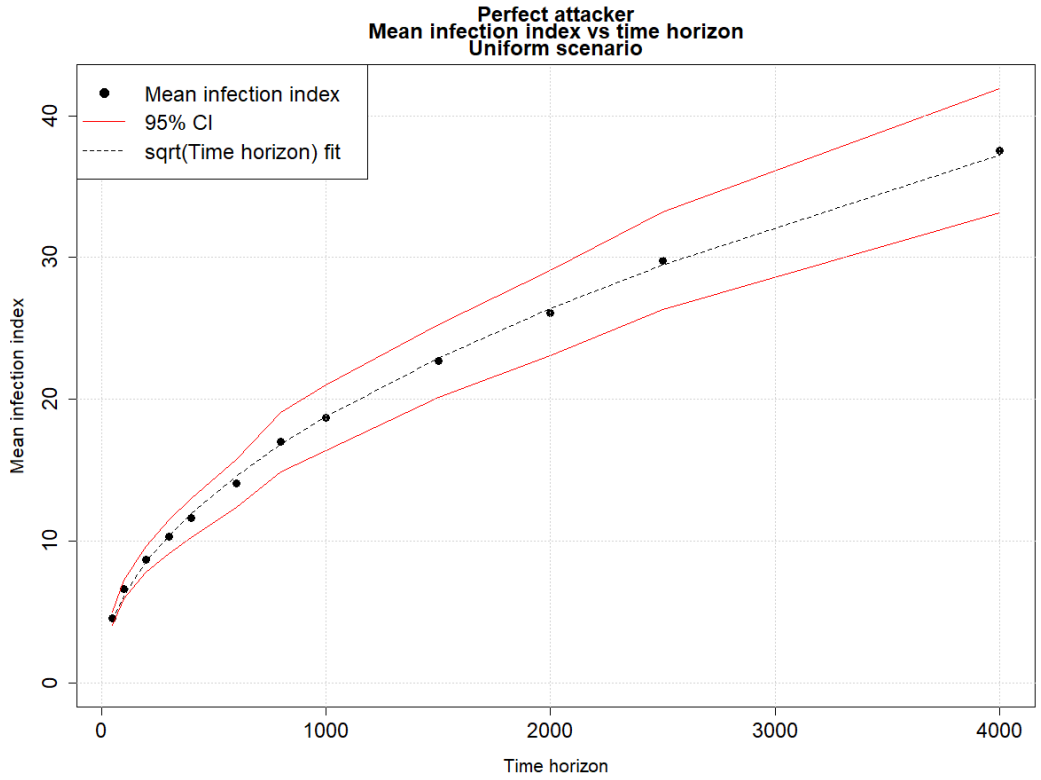

Figure 4.2. Mean Infection Index vs. Time Horizon for Perfect Attacker.

The result in Figure 4.2 fits the analytic result in Equation (3.18) from Chapter 3: the perfect attacker's infection index is proportional to square root of the time horizon.

For a constant $A = 0.584$, the fit $f(n) = A\sqrt{n}$ is showed in Figure 4.2 as a black dashed line.

Conclusion: For a linear network with $U(0, 1)$ distributed μ values and without possible detections, a perfect attacker will infect on average proportionally to the square root of the time horizon.

4.1.2 Effect of Attack Capability

Here, we examine the effect of the probability of successful attack s. As shown in Chapter 3 (see results from Equation (3.17)), we expect the mean infection index to be a function of the square root of s for a fixed time horizon. Each data point in Figure 4.3 is the estimated

infection index achieved by a perfect attacker playing 200 randomly generated networks. The result in Figure 4.3 matches the expected dependence in $\sqrt{n}$.

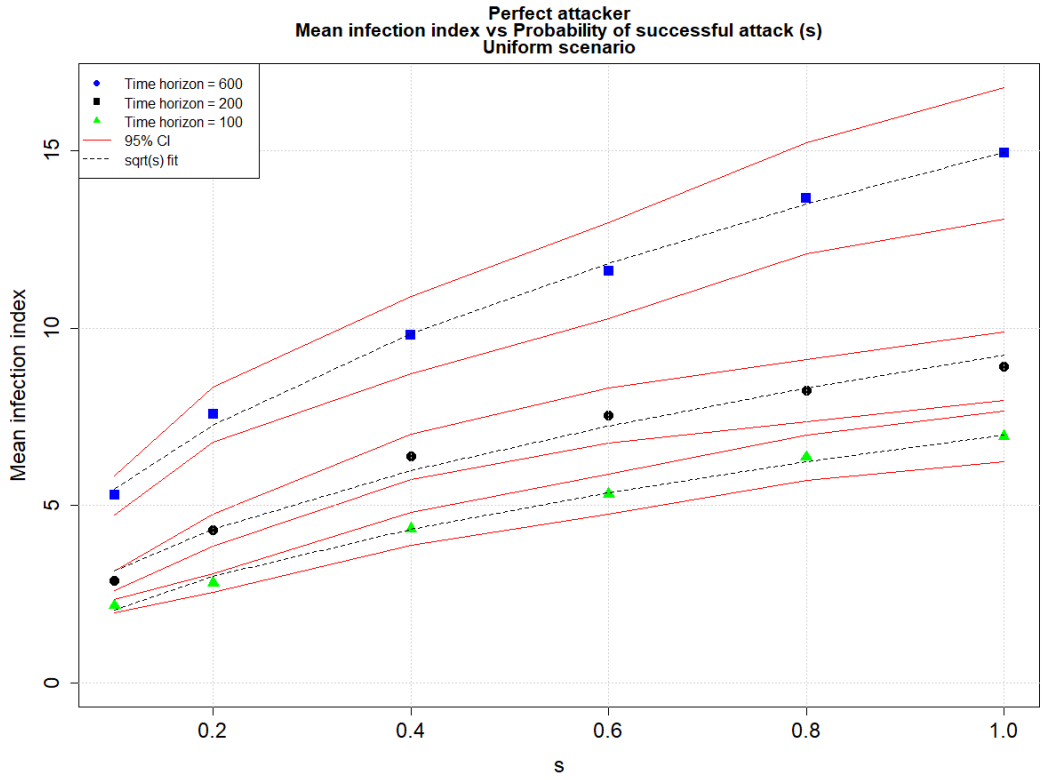

Figure 4.3. Mean Infection Index vs. Probability of Attack s.

For constants

$$A_{600} = 13.91; B_{600} = 1.05$$

$$A_{200} = 8.89; B_{200} = 0.34$$

$$A_{100} = 7.24; B_{100} = -0.25$$

we get the fits $f(s) = A\sqrt{(s)} + B$ as shown in Figure 4.3 — as expected, the infection index depends on $\sqrt{s}$. We verified that the estimated infection index for a perfect attacker behaves as $O(\sqrt{sn})$. Hence, result for $(n = 600, s = 0.2)$ should be the same as $(n = 200, s = 0.6)$ or $(n = 100, s = 1)$. This fact can be seen in Figure 4.3.

Conclusion: For a linear network with uniform distributed μ values and without possi-

ble detections, a higher probability of attack will cause a perfect attacker to infect more computers, proportionally to the square root s.

4.2 Rational Attacker

In this section we show the performance of the Bayes UCB-I algorithm, and compare it to a perfect attacker as a measure of performance.

4.2.1 Mean Infection Index

According to the theory from Chapter 3, the Bayes UCB-I algorithm's infection index is expected to be of order of $K^* = O(\sqrt{n/\log n})$ (see Equation (3.13)), while the perfect attacker's infection index is expected to be $K^*_{\text{perfect}} = O(\sqrt{n})$ (see Equation (3.18)).

In Figure 4.4, one can see both trends.

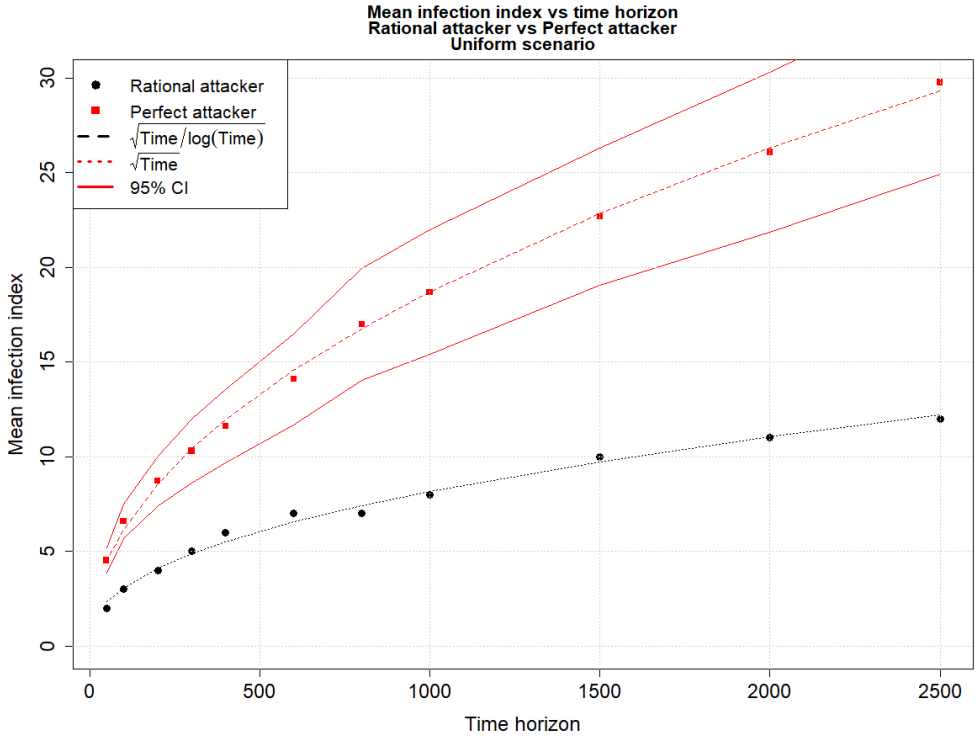

Figure 4.4. Mean Infection Index vs. Time Horizon for UCB-I Algorithm vs. a Perfect Attacker.

For constants

$$A_{\text{perfect}} = 0.579; \quad B_{\text{perfect}} = 0.384$$

$$A_{\text{rational}} = 0.689; \quad B_{\text{rational}} = -0.136$$

we get the fits $f_{\text{perfect}}(n) = A\sqrt{(n)} + B$ and $f_{\text{rational}} = A\sqrt{\frac{n}{\log n}} + B$ as shown in Figure 4.4 — as expected, the rational attacker's infection index depends on $\sqrt{\frac{n}{\log n}}$.

In Figure 4.4 there are no confidence interval lines for the rational attacker. This is because the standard deviation of the final infection index was 0. The rational attacker always stopped at the same number of infected computers. Because there are no detections in this section (i.e., $d = 0$), the attacker can always reach his desired number of infected computers if time allows.

4.2.2 Optimality Gap

From the theory in Chapter 3 (see Equation (3.20)), the optimality gap is:

$$\text{Gap}(n, s) = O\left(\sqrt{s\frac{\log^3 n}{n}}\right).$$

The optimality gap as a function of the time horizon is shown in Figure 4.5 (for a nominal case of $s = 0.9$, as defined earlier).

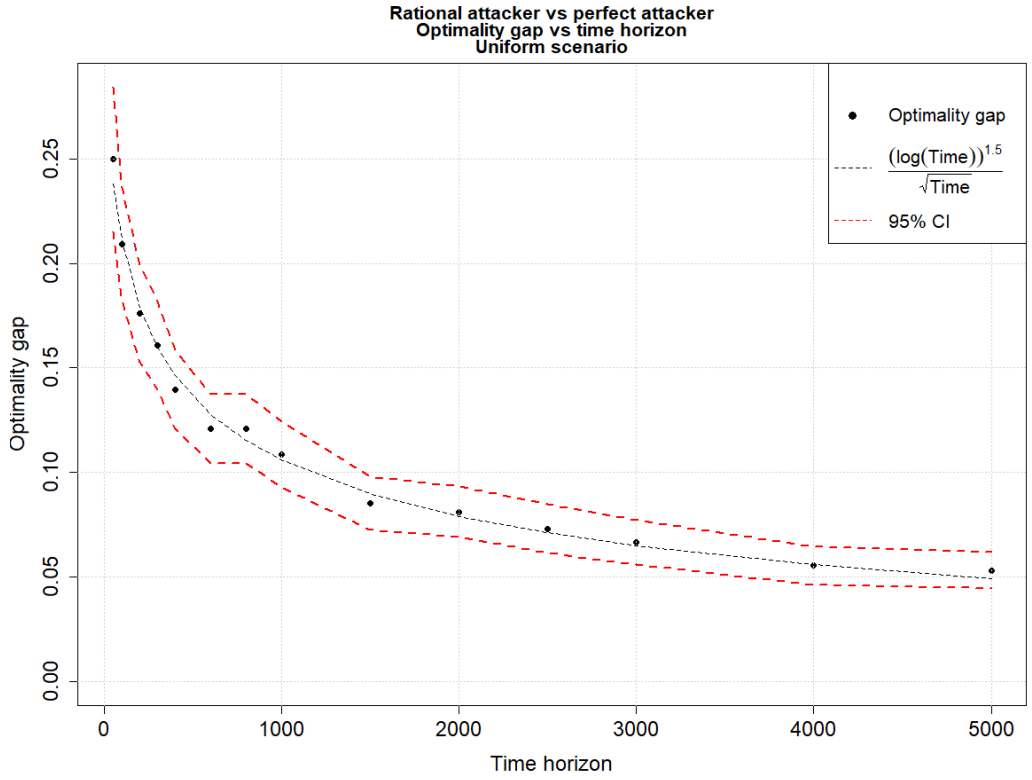

Figure 4.5. Optimality Gap vs. Time Horizon for $s = 0.9$.

For a constant $A = 0.254$, we get the fit $A\sqrt{\frac{\log^3 n}{n}}$ in Figure 4.5, verifying the expected behavior given by Equation (3.20).

Conclusion: For a linear network with uniform distributed values $\mu_1, \ldots, \mu_M$, an attacker who follows the Bayes UCB-I algorithm will do as well as 90% of a perfect attacker's performance, given a time horizon bigger than 1000.

4.2.3 Effect of Attack Capability

Similar to the section about a perfect attacker, we show the effect of the parameter s on the expected infection index for a rational attacker, and then we show the optimality gap as a function of s. According to Equation (3.13) from Chapter 3,

$$K^* \approx \sqrt{\frac{n}{\log n}}.$$

Hence, a rational attacker's infection index is indifferent to the probability of successful attack s given large n and s not extremely small. This finding is shown in Figure 4.6.

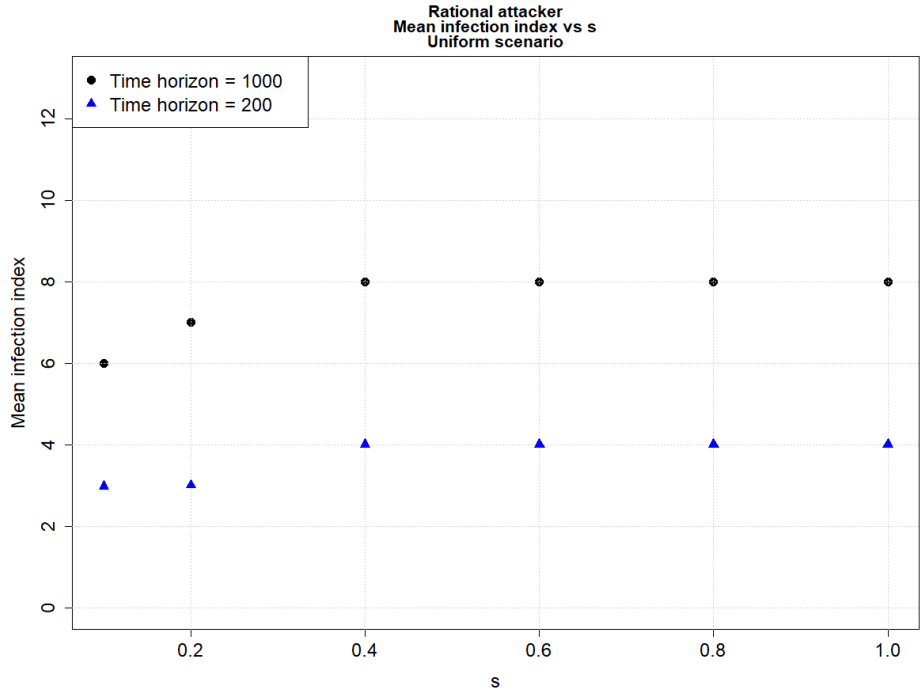

Figure 4.6. Rational Attacker – Infection Index vs. s.

There are no confidence intervals in Figure 4.6, because the standard deviation of the final infection index is 0. As explained before, without detections, the rational attacker will seek to infect the same number of computers (the optimal number).

From Equation (3.20), the optimality gap is proportional to

$$\sqrt{\frac{s \log^3 n}{n}}.$$

Hence, for small values of time horizon, we expect to notice a significant difference with respect to s values, but for large n, we expect to see a near constant gap with respect to different values of s. Results that verify Equation (3.20) are shown in Figure 4.7.

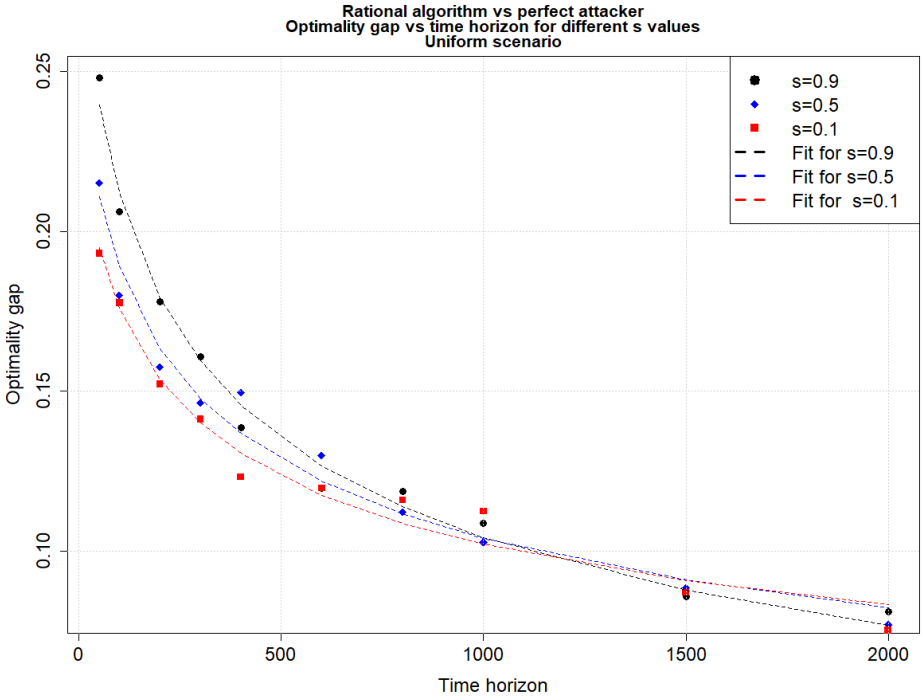

Figure 4.7. Optimality Gap vs. Time Horizon for Different s Values.

For the reader's convenience, the confidence interval lines in Figure 4.7 were omitted (the behavior is the same as in Figure 4.5).

As expected, the gap is significantly bigger for $s = 0.9$ than for $s = 0.1$ when the time horizon is small (less than 500). For $n > 1000$, the optimality gap is almost constant with respect to different s values.

These observations match the theoretical derivation for the effect of s values that are not particularly small (such as $s = 0.1$). For small s values, the lower order terms (see the derivation of Equation (3.20)) cannot be neglected and hence the deviation from our expected values.

Conclusion: For a linear network with uniform distributed values $\mu_1, \ldots, \mu_M$ and without possible detections, a rational attacker is indifferent to changes in the probability of successful attack, when the time horizon is bigger than 1000. For smaller time horizons, the rational attacker can expect a bigger optimality gap by a factor of $\sqrt{s}$.

4.3 Effect of Detections on the Perfect Attacker

From Equation (3.22), we expect on average that the perfect attacker's infection index will be a function of s and d. For a given s value, we expect the infection index to decrease when d increases, up to the critical value d_c, where the attacker should not infect any computer.

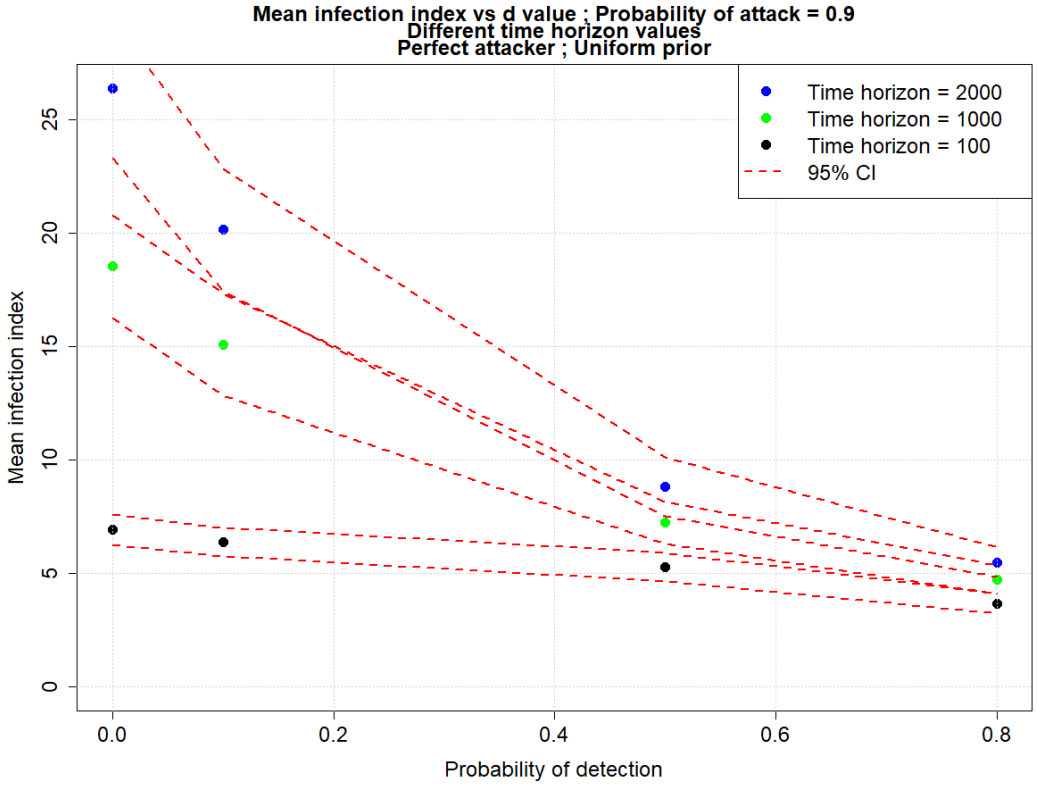

Figure 4.8. Mean Infection Index vs. Probability of Detection d, when $s = 0.9$.

In Figure 4.8, one can see the decreasing trend as the probability of detection increases.

As shown in Equation (3.23), there is a critical d value for each s value that causes the attacker to avoid any infection. The observed critical d for each s value is shown in Figure 4.9. A fit according to the theory was added.

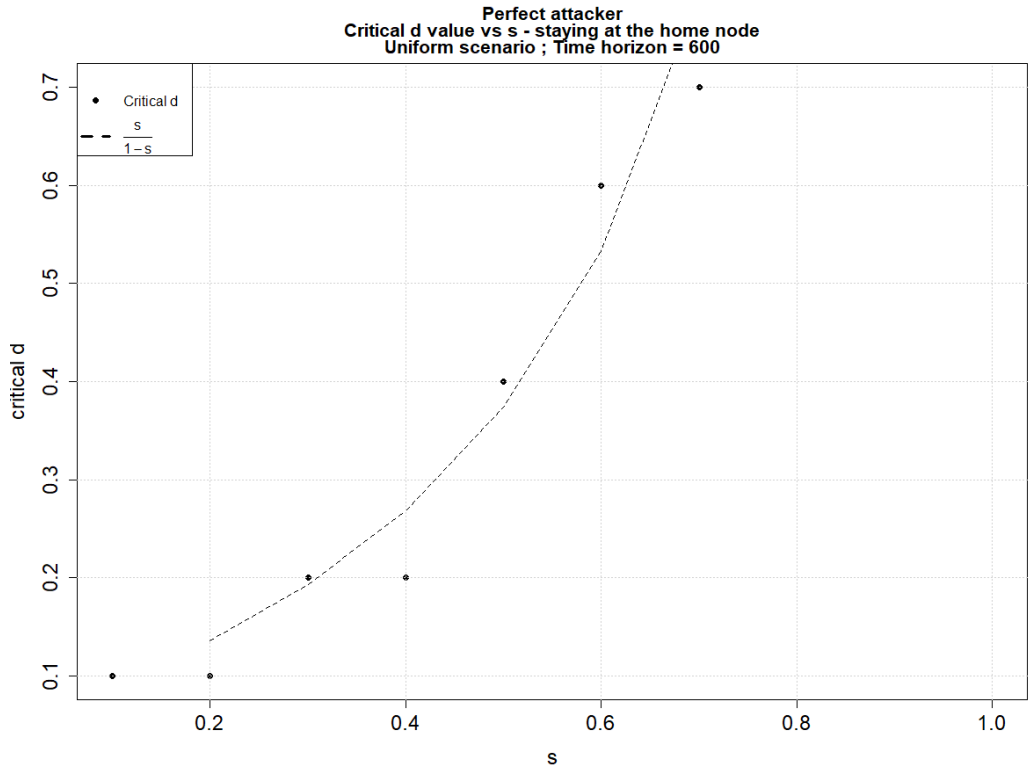

Figure 4.9. Mean Infection Index vs. Probability of Detection d, when $s = 0.9$.

We conclude that our analytical result is a very good approximation and can be used as a rule of thumb.

Conclusions:

- For a linear network with uniform distributed μ values, for a given s value, the perfect attacker can determine the maximum risk to operate as

$$d_c = \frac{s}{1-s} d.$$

  Above this value, the attacker should not attack at all.
- If a perfect attacker can achieve a probability of successful attack bigger than 0.7, than he should consider the option of infecting for any probability of detection.

According to our method, one can create equal-reward contour plots. This can be done by partial derivatives of the regret with respect to s and d. We show in Figure 4.10 the exact contour lines for time horizon 600. The shapes of the lines should be the same for any time horizon, just the values on them will differ.

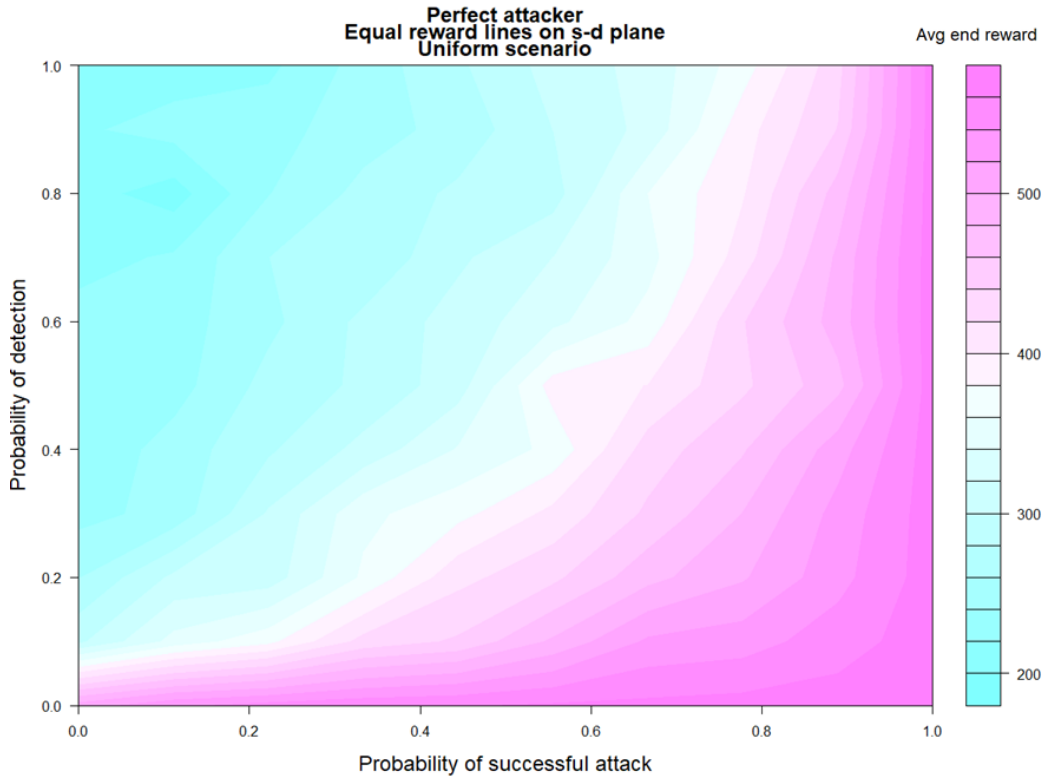

Figure 4.10. Mean Infection Index vs. Probability of Detection d, when $s = 0.9$.

From Figure 4.10 the reader can notice the relative low expected reward from the low-success high-detection scenario. One can notice the shape of the critical d value as a divider between the high expected reward region and the low expected reward region. In general, the attacker can use this result to quantify the trade-off between attack and detection-avoidance capabilities.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5:
Conclusions

In this chapter we present a summary of our main results. These were derived analytically in Chapter 3 and verified numerically in Chapter 4.

The following conclusions apply to the worst-case scenario for an attacker – a linear network – because it makes it difficult to spread deep into the network, and a single detection of an attack is enough to divide the network permanently, thus preventing the attacker from acquiring new computers.

We defined the perfect attacker as an oracle who knows in advance the probability that each individual computer contains relevant intelligence. This attacker can execute an optimal policy by solving a dynamic program. We also recommend a policy for attackers who have a uniform prior for the network parameters, without knowing their exact values. These *rational* attackers should infect computers sequentially, without gathering intelligence contained within the infected nodes, followed by a second phase devoted to exploring the infected computers.

When attacks cannot be detected, the oracle infects computers in proportion to the square root of the available time. On the other hand, the rational attacker with incomplete information should infect up to time over log of time computers. A surprising result is that, while the oracle tends to infect deeper into the network as his probability of successful attack increases, the limited-information attacker is indifferent to the probability of a successful attack. We found that the difference in total expected reward between the oracle and the attacker with incomplete information quickly decreases to zero, without dependence in the probability of successful attack. This result means that if there is no risk of detection and the time horizon is large, an attacker should not invest resources in increasing his attack capabilities. In other words, the penalty for not knowing the exact parameter values is small.

When attacks can be detected, we determined a critical relationship between the probability of successful attack and the probability of detection for which the attacker avoids any attempt to expand. We found that, once the probability of successful attacks is at least

0.7, the oracle should always operate in the network for any given probability of detection. Furthermore, we showed how to balance investments in attack and stealth capabilities, in order to maximize expected performance.

Last, we did a first pass on extending the basic model to include network devices (routers) and non-uniform priors on the network parameters (without detections). Our analysis suggests that the attacker should explore the routers up to log of time and, thereafter, infect the computer with the largest posterior mean, among those that lie within the router's subnetwork. When the prior on the network parameters is Beta with arbitrary parameters, we found that the attacker should spread further into the network as the average intelligence of the computers decreases. Numerical experimentation of these two aspects of the problem is left for future work.

List of References

[1] "NATO: Changing gear on cyber defense," 2016. Available: http://www.nato.int/docu/review/2016/Also-in-2016/cyber-defense-nato-security-role/EN/index.htm

[2] "U.S. intelligence community budget," 2017. Available: https://www.dni.gov/index.php/what-we-do/ic-budget

[3] T. L. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[4] A. N. Burnetas and M. N. Katehakis, "Optimal adaptive policies for sequential allocation problems," *Advances in Applied Mathematics*, vol. 17, no. 2, pp. 122–142, 1996.

[5] J. Gittins, K. Glazebrook, and R. Weber, *Multi-armed Bandit Allocation Indices*. John Wiley & Sons, 2011.

[6] S. Bubeck, N. Cesa-Bianchi *et al.*, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *Foundations and Trends® in Machine Learning*, vol. 5, no. 1, pp. 1–122, 2012.

[7] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, no. 2-3, pp. 235–256, 2002.

[8] E. Kaufmann, O. Cappé, and A. Garivier, "On Bayesian upper confidence bounds for bandit problems," in *Artificial Intelligence and Statistics*, 2012, pp. 592–600.

[9] Y. Wang, J.-Y. Audibert, and R. Munos, "Algorithms for infinitely many-armed bandits," in *Advances in Neural Information Processing Systems*, 2009, pp. 1729–1736.

[10] K. Liu and Q. Zhao, "Intrusion detection in resource-constrained cyber networks: A restless multi-armed bandit approach," *submitted to IEEE/ACM Transactions on Networking. Avialable at http://arxiv. org/abs*, vol. 1112.

[11] Z. Zheng, N. B. Shroff, and P. Mohapatra, "When to reset your keys: Optimal timing of security updates via learning," *CoRR*, vol. abs/1612.00108, 2016. Available: http://arxiv.org/abs/1612.00108

[12] Y. Qian, C. Zhang, B. Krishnamachari, and M. Tambe, "Restless poachers: Handling exploration-exploitation tradeoffs in security domains," in *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems* (AAMAS '16). Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2016, pp. 123–131. Available: http://dl.acm.org/citation.cfm?id=2936924.2936946

[13] R. Elderman, L. J. Pater, A. S. Thie, M. M. Drugan, and M. Wiering, "Adversarial reinforcement learning in a cyber security simulation." in *ICAART (2)*, 2017, pp. 559–566.

[14] H. A. David and H. N. Nagaraja, *Order Statistics*. Third Edition. New York: Wiley, pp. 459-465, 2003.

[15] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Hoboken, NJ: John Wiley & Sons, 2012.

[16] P. Embrechts, C. Klüppelberg, and T. Mikosch, *Modelling Extremal Events: For Insurance and Finance*. Berlin Heidelberg: Springer Science & Business Media, 2013, vol. 33.

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California