Essays in Applied and Computational Game Theory

A Dissertation
SUBMITTED TO THE FACULTY OF THE
UNIVERSITY OF MINNESOTA
BY

Taylor Jay Canann

IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Advisor: Jan Werner

JUNE 2019

Acknowledgements

I am extremely grateful to my patient advisor, Jan Werner, for all of his help and guidance in completing this dissertation.

I would also like to thank to Brennan Platt, Richard Evans, Kerk Phillips, the BYU MCL workshops, David Rahman, Aldo Rustichini, the Minnesota Workshop on Mathematical Economics, Brad Greenwood, Robert Mrkonich, Samuel Kaplan and Ryne Belliston for very helpful comments, advice, and direction.

Dedication

I dedicate this dissertation to my wonderful wife, Cait, and my children, Austin, Taylor June, Anderson, and Walter, for their love, devotion, and understanding through this entire process.

Abstract

This dissertation considers computational and applied aspects of cooperative and non-cooperative game theory. The first chapter discusses a novel applied game theory approach within the field of vulnerability disclosure policy. I introduce a three-player game between software vendors, software users, and a hacker in which software vendors attempt to protect software users by releasing updates, i.e. disclosing a vulnerability, and the hacker is attempting to exploit vulnerabilities in the software package to attack the software users. The software users must determine whether the protection offered by the update outweighs the cost of installing the update. Following the model set up, I describe why low-type software users, software users that do not get much value out of the software and are thus not very damaged by an attack, prefer Non-Disclosure, and Disclosure can only be an optimal policy in cases when the cost to the hacker of searching for a zero-day vulnerability is small.

Many economic problems are inherently non-linear, so in the second chapter we introduce the MGBA, the Modular Gröbner Basis Approach, which is a solution technique from Algebraic Geometry that can be used to "triangularize" polynomial systems. The MGBA is a computational tool that overcomes the typical computational problems of intermediate coefficient swell and solving for lucky primes that can limit the ability to compute Gröbner bases. The Gröbner basis is an all-solution computational technique that can be applied to many fields in economics. This chapter focuses on applying the MGBA to Bertrand games with multiple equilibria and a manifold approach to solving dynamic programming problems.

Advances in computational power and techniques have greatly benefited both economic theory, in allowing economists to solve more realistic models, and data analysis, such as machine learning. However, the field of cooperative game theory has fallen behind. Therefore, in the final chapter, I introduce the compression value, a computationally efficient approximation technique for the non-transferable utility (NTU) Shapley value. This algorithm gives a reasonable approximation of the NTU Shapley value if the initial guess of Pareto weights is near the actual solution.

Contents

List of Figures

Chapter 1

Chapter 2

Chapter 3

Introduction

Game theory is a very versatile tool, and, within this dissertation, I will be exploiting this versatility by using game theory in the fields of cyber-security and computational economics. In chapter one, I discuss the implications on optimal vulnerability disclosure policies of a game between hackers, software users, and software vendors. Following this, I examine a novel computational technique, the Modular Groebner Basis Approach (MGBA) in chapter two. The MGBA provides an algorithm to solve both static and dynamic games. I conclude with chapter three by introducing a computationally efficient solution technique for NTU games.

On May 7, 2018, Baltimore was hit by a debilitating ransomware attack. Was Baltimore targeted? No. This is believed to be a crime of opportunity, meaning the hackers scanned a large number of online systems for known vulnerabilities. These known vulnerabilities are known as N-Day vulnerabilities, i.e. vulnerabilities that have been known to software users and vendors for some days. When a vulnerability is found, software vendors release an update to protect users from being exploited by the newly found vulnerability. However, this type of protection policy, called Disclosure policy, requires software users to update their machines in order to not be vulnerable to attack. Given that most users do not automatically or immediately update their machines, see 1.1, Disclosure reveals the holes in the software to hackers and thus decreases the cost hackers face when searching for a vulnerability. As a result of Disclosure policy and Baltimore's failure to keep their servers updated, it is estimated that the city of Baltimore has had to pay $18 million to repair their systems.

Thus, whether software vulnerabilities should be disclosed, and, if so, what type of vulnerabilities are the biggest threats, are still open and pressing questions in cybersecurity. In the first chapter I analyze whether vulnerabilities should be disclosed by examining the welfare effects of both the network externalities of a set of workers and hacker behavior on vulnerability disclosure policy.

In order to describe the best type of disclosure policy, I build a model of a heterogeneous IoT network, which is made up of an interconnected set of software users, that are attempting to defend themselves against a profit-maximizing hacker. Within my model, there are three decisions to be made: (i) The strategy of attack to be played by the hacker, (ii) The optimal disclosure policy, and (iii) The updating decision made by the software user. I formulate welfare maximizing policies to decrease a hacker's efforts in infiltrating networks and increase the software users' utility.

The optimal policy is dependent both on the distribution of software users on the network and how costly finding a previously unknown vulnerability, i.e. a Zero-Day vulnerability, is for the hacker. Software users that do not expect to bear the majority of the burden of an attack, known as low-type users, do not want vulnerabilities to be disclosed, i.e. a Non-Disclosure policy, since they will not update their machines, deeming it too costly. Thus, if there is a large enough contingent of low-type users, Non-Disclosure is the optimal policy.

Also, if the cost of searching for a Zero-Day exploit is high, then the hacker is not willing to expend the energy searching for a Zero-Day, and Non-Disclosure is an optimal policy since there are no vulnerabilities available to the hacker. Therefore, the only case in which Disclosure can be an optimal policy is when search costs are low and there are enough users that desire to update their machines.

In the second chapter, we introduce the Modular Gröbner Basis Approach, the MGBA, which is a computational tool that can be used to overcome the difficulties of solving for all equilibria of non-linear systems. This problem is especially rampant when solving economics problems where there is strategic interaction. The problem of multiple equilibria is not new to economic theory or applied theory. For example, Maghsudi and Hossain (2016) setup a multi-agent, multi-armed bandit game in order to design the next generation wireless networks to move toward new networking paradigms that are able to efficiently support resource-demanding applications such as personalized mobile services. In many cases, they find that there exist multiple equilibria, and the problem then turns to guiding the agents to the most efficient equilibrium. In order to do this, all equilibria must be solved for, then the determination of the most efficient equilibrium is possible.

There have been many attempts to solve these problems, but these other methods require stringent simplifications of the models and do not allow economists to solve for all equilibria of the complex models. We build off of Arnold (2003), and have developed the MGBA, which can solve for all equilibria in non-linear economic models via Groebner bases. We discuss the application of the MGBA to Bertrand pricing games as well as an application of a manifold approach to dynamic programming.

In the third chapter of this dissertation, I introduce a new solution technique for NTU games, the compression value, an algorithm to solve for the compression value, and I discuss how the compression value can be used as an approximation for the NTU Shapley value. The NTU Shapley value is a solution concept from cooperative game theory introduced in Shapley (1969) that states that "an outcome is acceptable as a value of a game only if there exist scaling factors for the individual (cardinal) utilities under which the outcome is both equitable and efficient". The compression value is a solution technique that does not require modifications or simplifications of the original game. This chapter presents a step toward a general algorithm to solve for the NTU Shapley value for a given game.

Computational power and techniques have drastically increased over the last couple of decades, and economics has greatly benefited from this increased accessibility. However, the field of cooperative game theory has not taken full advantage of these computational advancements. The compression value is a linear scaling of the Shapley value of the TU representation of the original NTU game. This solution technique satisfies a reasonable set of properties for an NTU solution technique.

1 Toward a Theory of Vulnerability Disclosure Policy: A Hacker's Game

1.1 Introduction

Every piece of software, no matter what care is taken by a software vendor, is riddled with vulnerabilities, which leaves users open to attack by hackers. To protect users, software vendors release patches to address these found vulnerabilities, but this is a double-edged sword. Releasing updates, a.k.a. vulnerability disclosure, may in fact increase the vulnerability of current users, in particular, those who chose not immediately install the updates. In other words, as new versions of a software package are released, via an update, then the holes, or vulnerabilities, in the old software version are made explicitly clear for hackers. These types of hacks have been gaining in prevalence over the last couple of years.

The first set of attacks I want to discuss were all propagated via slight deviations of the EternalBlue exploit. In May of 2017 the WannaCry attacks[1] infected over 300,000 systems in 150 countries and the approximate estimated cost that these attacks is $4 billion. One month later, in June, the NotPetya attacks, another major global attack that primarily targeted Ukrainian systems[2], began. The approximated costs of the NotPetya attacks were even larger than the WannaCry attacks and have been estimated at around $10 billion. Following the NotPetya attacks, the Retefe banking Trojan began leveraging the EternalBlue exploit in September. Finally, in August of 2018 the Taiwan Semiconductor Manufacturing Company, an Apple chip supplier, was hit by a new variant of the WannaCry attack that cost the company approximately $170 million. The problem was not that Windows is an inherently flawed system, but

[1] Ransomware attacks that targeted Windows systems demanding payment in Bitcoin.

[2] Approximately 80% of the attacks were in Ukraine.

instead that these attacks could have been avoided if users/firms had only updated. In March of 2017, Microsoft patched this vulnerability in their monthly, second Tuesday, update.

Another major attack that received global notoriety was the Equifax hack that compromised 145.5 million American accounts. This exploit attacked Apache Struts[3] and spread through Equifax's systems between May and July of 2017. This data breach is estimated to have cost $439 million, and, yet again, this hack was not inevitable; Apache released an update for this vulnerability on March 7, 2017.

These are just a couple examples of what are called N-Day vulnerabilities[4]. N-Day exploits have been on the rise, and have gained a significant amount of notoriety. The other type of vulnerability analyzed in this paper is known as a Zero-Day attack. A Zero-Day attack is an attack that exploits a previously unknown vulnerability. Many of the largest hacks over the last couple of years have been N-Day exploits, and, since the users of the software did not update their machines, hackers were able to easily exploit these vulnerabilities in the software.

According to Symantec[5], "The use of zero days continues to fall out of favor. In fact, only 27 percent of the 140 targeted attack groups that Symantec tracks have been known to use zero-day vulnerabilities at any point in the past." Since hacker behavior is shifting away from Zero-Days and toward the exploitation of N-Days, policy makers and software vendors should also think about what type of changes, if any, should be made to disclosure policies. By "disclosure policy", I mean how often, if ever, should the vendor release updates and thus disclose the location of a vulnerability found by the vendor.

I build a model of a network, made up of an interconnected set of workers, that is attempting to defend itself against a profit-maximizing hacker. Within the model,

[3] A software published by the Apache Software Foundation.

[4] Known vulnerabilities.

[5] See Sym (2018) and Sym (2016).

there are three decisions to be made, (i) The strategy of attack to be played by the hacker[6], (ii) The optimal disclosure policy is determined by the vendor/social planner, and (iii) The workers must decide whether to update their machines if a vulnerability is disclosed.

To motivate this approach the set of workers can be thought of as a Mobile Ad Hoc Network (MANET). A MANET is a collection of wireless mobile hosts forming a temporary network without the aid of any centralized administration or standard support services. Mobile nodes have a limited communication range and are thus connected only to the devices located within some given radius of the node[7]. The model in this paper is a static game[8] between the network of workers and a single hacker.The encryption of packets on a MANET are hard to secure, so we assume that the vulnerabilities in the MANET are in the encryption packages.

The demand for MANETs has been expanding rapidly as of late. MANETs have many military applications, such as communications networks since MANETs are able to quickly re-route communications as the military units change their locations. However, these networks are not only used by the military, but there is an ever growing demand by the average households to use MANETs. The first example of these networks are Vehicular Ad-hoc Network (VANET), in which vehicles and roadside equipment communicate with each other. This is a very relevant field as self-driving cars are starting to hit the roads and we need to understand the potential risks associated with different types of vulnerabilities and different types of disclosure policy regimes.

[6] A lot of the literature, e.g. see Hong and Neilson (2018), model hacker behavior as similar to a Becker model (Becker (1968)), but this approach assumes that (i) Law enforcement can easily track and find a hacker and (ii) that hackers can easily be prosecuted. These assumptions, however, are not realistic. For example, the WannaCry and NotPetya attacks were launched by, as far as we know, North Korean and Russian hackers, respectively. It is very difficult to extradite and prosecute foreign hackers.

[7] Therefore, we can use a random geometric network to model a MANET at any point in time.

[8] MANET are typically dynamic with nodes entering and leaving the network as well as changing their set of neighbors over time, but this would require a dynamic model, and that is the aim of future research

Another example that needs particular attention is that of Smart Phone Adhoc Networks (SPANs). These are a specific example of Internet of Things (IoT) networks that use Bluetooth and/or Wi-Fi to create P2P networks. As the number of devices that are available to be linked to these SPANs grows, so will the amount of vulnerabilities across these networks. The amount of information available to hackers will also increase if they are able to exploit the network structure within an attack. Thus, one of the questions that needs to be answered within this field, is how software producers should think about releasing updates[9].

In addition to the contribution of modeling the set of workers as a network, and thus MANET disclosure policies can be analyzed, this is the first model to attempt to incorporate hacker decisions into the discussion on vulnerability disclosure analysis. The hacker must decide whether to search for a Zero-Day vulnerability, exploit the N-Day disclosed by the vendor, or do not hack, i.e. exit the game, while attacking the network. In doing so, they maximize an expected profit[10] function. This approach allows for a better understanding of disclosure policy, because any optimal disclosure policy should be dependent on the attack strategy of the hacker.

1) Networks matter 2) Theorems 3) Given the fall in Zero-Days and increase in N-Days, then should shift toward Non-Disclosure

Now to outline the major findings of the paper. The first contribution of this paper is to introduce a formal game between a hacker and a network of software users in order to inform optimal disclosure policies. This is important since the hacker's action has a large impact optimal workers' decisions, and thus should impact the policy maker's choice. For example, as the hacker spends less time searching for Zero-Days and more time exploiting the known vulnerabilities, then workers are going to

[9] Since the vendor's actions are not interesting until a dynamic model is developed, I am assuming that there is only one software vendor. This assumption will need to be relaxed within future research.

[10] The logic could be extended to utility, but then the intuition behind the weight parameters would be changed.

increase their willingness to update since even though updating is costly, updating will protect the worker from being attacked.

Additionally, I find that as the cost of searching for Zero-Days increases, as in the data discussed in Section 1.2, the hacker will tend toward exploiting the vulnerability in the released update instead of searching for a Zero-Day. Therefore, the policy that maximizes the workers' utility is to decrease disclosure. This is not a new idea, but a formalization of previous arguments, e.g. see Rescorla (2005). This analysis can also be used to analyze how effective different disclosure policies are in protecting users in a MANET due to the generality of the distribution of software users within the model. Probably the most important implication is that for any cost of searching for a Zero-Day, Non-Disclosure can be the optimal[11] policy.

Starting in January of 2020, Microsoft will no longer support Windows 7, unless the users enroll in "Extended Support". This new type of disclosure policy is discussed in Section 1.5.1. That section also contains the final result of the paper, which is that Microsoft's new policy increases the cost of exploiting the disclosed vulnerability, and, even though the policy increases the cost of updating, causes the software uses to receive higher payoffs. This new approach to disclosure policy increases overall welfare relative to the policy of disclosing all vulnerabilities.

The sections of the paper are as follows: Section 1.2 is the literature review, followed by an introduction to the model as well as the first main contribution: A discussion of optimal policy when the hacker are decision making agents in Section 1.3. Following the baseline models is a discussion of both policy implications of the baseline models and a newly proposed policy by Microsoft[12] in Section 1.5. Finally, we conclude in Section 1.6.

[11] The policy that maximizes the welfare of all workers.

[12] For Windows 7.

1.2 Literature Review

In a seminal paper in the field of vulnerability disclosure, Rescorla (2005) asked if finding vulnerabilities is optimal for social welfare. Since then, vulnerability disclosure policy has been greatly debated in the literature. There have been many attempts, both empirical and theoretical, to understand the underlying factors influencing the key decision makers in the game.

The model outlined in this paper explores the decisions made by both the network of workers and a hacker given a policy regime followed by the vendor. The interaction between vendors and firms was first modeled by Arora et al. (2008), in which they find that vendors will always want to delay the release of patches, but this action is not socially optimal. However, Arora et al. (2008) do not pose an answer to whether a vendor should engage in disclosing vulnerabilities, which is the main focus of this paper.

One of the first papers on the economic modeling of hacker behavior was developed in Png et al. (2006), where they attempt to estimate the effects of the fixed costs of hacking on the incentives of a profit maximizing hacker. We introduce this style of hacker modeling into the vulnerability disclosure debate. The hacker behaving as a profit maximizer allows for an investigation into the problem of under investment in cyber-defense by individual workers. The network framework is an extension of the work in Choi et al. (2010)[13], where they focus on the welfare effects of disclosure policy for a representative set of workers with the vendor facing a monopolistically competitive market. Their analysis does not take the hacker's decision or the possibility of different distributions of workers into account while solving for optimal policies. That approach is especially problematic when attempting to model diverse network configurations such as MANETs.

[13] I follow the notation in Choi et al. (2010) rather closely so as to maintain a constant notational scheme within the vulnerability disclosure literature.

Others have examined how attack propensity changes under different disclosure regimes (e.g. Arora et al. (2006)), and have found that releasing patches tends to increase the number of attacks. This model also predicts that attacks may increase with disclosure[14], but this is due to the fact that workers will desire more disclosure as the average desirability[15] of each worker to the hacker increases, the cost of searching for Zero-Days increases[16], or the cost of updating decreases. Therefore, this model is able to give a causal relationship between attack propensity and disclosure regimes which strengthen the story behind these correlations.

There is also a subset of the literature that focuses on what types of vulnerabilities are/should be disclosed[17]. This paper does not contribute to this literature since every disclosed vulnerability and every found Zero-Day can inflict the same amount of damage. It would be very interesting to model both the hacker and the vendor drawing vulnerabilities from specific distributions, but that should be done in a dynamic game so as to truly capture the vendor's optimal decision making process.

Our model also makes contributions in the growing applied information security literature. Optimal network defense[18] is a growing field in which they use game theoretical models to discuss how to defend a network from attack. Adding in vulnerability disclosure via the software vendor and having the network players making decisions would allow for a framework to analyze a wider range of games with strategic attack. Additionally, allowing the attacker to choose from a set of potential attack strategies, such as searching for Zero-Days or attacking with some portfolio of N-Days, with defenders able to dynamically update would create fascinating dynamic strategies.

Lastly, the model contributes to the ever growing MANET literature. The litera-

[14] Except under the new policy proposed by Microsoft discussed in Section 1.5.1.

[15] Worker desirability can be though of as network centrality.

[16] Implying that, as the vendor discloses more vulnerabilities, the hacker will choose to exploit the N-Days instead of paying the large cost of searching for Zero-Days, and thus increase the number of hacks, e.g. see Pon (2016).

[17] See K.C. (2012) for a full literature review.

[18] Dziubinski and Goyal (2017), Cerdeiro et al. (2017), and Goyal and Vigier (2014)

ture has mainly focused on the types of vulnerabilities and attacks (such as sinkhole or eavesdropping) and methods to defend against those attacks (such as key management or intrusion detection systems)[19]. This is the only paper that attempts to discuss vulnerability disclosure policy in MANETS instead of focusing on the technicalities associated with specific attack or defense methods[20].

1.3 Static Game

The actors within this static model are the hacker and the software users (called workers). The software vendor follows a welfare maximizing disclosure policy, and thus determines the rules of the game. Hackers maximize their profits by choosing a hacking strategy of exploiting either a Zero-Day, the patch released by the software vendor, i.e. an N-Day attack, or he can exit the game. Lastly, the workers must decide whether or not to update their machines if a vulnerability is disclosed, i.e. an update is released[21]).

Software is assumed to be produced by a single vendor that is only concerned with maximizing worker welfare, similar to a social planner. The vendor is unable to detect all vulnerabilities before selling the software, but the vendor, under a Disclosure policy, will attempt to find these vulnerabilities ex post[22]. The probability that the vendor is able to find a vulnerability is is exogenously given as $\alpha \in (0,1)$.

Let $I = \{1,\dots,m\}$ be a set of interconnected workers within a firm, where each worker has an associated weight parameter[23], $\theta_i \in (0,1)$. The set of workers can be described by $\theta \equiv \{\theta_1,\dots,\theta_m\}$. As each worker i uses the software produced by the

[19] See Goyal et al. (2010), Lalar (2014), and Kalambe and Apte (2017)

[20] Future research will attempt to incorporate a more sophisticated set of attacks and defenses, but this work is laying a groundwork for future modeling.

[21] The notion of disclosure timing being weighed against updating intensity will be the focus of a dynamic model which will be examined in future research.

[22] This can either be thought of as individual vendors searching for vulnerabilities by themselves or as bounty systems such as Microsoft's Bounty System (E.g. See: Ozment (2004), Coyne and Leeson (2005), Laszka et al. (2016), and Kuehn and Mueller (2016))

[23] This weight parameter can be thought of as the network centrality of the worker, or as how desirable the worker's information is to the hacker.

vendor, they receive a value of $\theta_i v$, for some constant $v>0$.

Due to the existence of hackers and the inability of vendors to solve all vulnerabilities ex ante, each worker is vulnerable to an attack. To allow for heterogeneity of damages across workers; the damage, $D>0$, done to worker i is scaled by their associated weight parameter, $\theta_i D$. Meaning that the hacker is only able to extract[24] as much information as is available to worker i. To make purchasing the software worthwhile to every worker, the damage done by a hacker exploiting a vulnerability must be less than the value added by the software ($D<v$).

The vendor does not usually charge the workers to install the updates[25], but the updates are still costly in terms of opportunity costs, i.e. the time to install the update. Updates often require workers to stop working or even shutdown their machines, we call this cost $c_u>0$. For simplicity, this cost is assumed to be a fixed cost to be paid if the worker decides to update[26]. To model the fact that some people do not update under any policy[27] I make the following assumption.

Assumption 1.1. *Let* $\theta_1 \leq \theta_2 \leq \cdots \leq \theta_m$ *and* $\theta_1 < \frac{c_u}{v+D} < \theta_m$.

A single hacker attempts to exploit vulnerabilities to maximize profits via gaining access to the network of workers. The hacker must maximize profits dependent on both the chosen policy and the workers' optimal updating decision. The hacker has two types of exploitation available to them, he is able to hack via a known vulnerability, an N-Day exploit, or by a previously unknown vulnerability, a Zero-Day attack. The information available to the hacker consists of the distribution of

[24] This could also be thought of a direct transfer from the worker to the hacker.

[25] In Section 1.5.1, I analyze the impacts of charging for updates.

[26] As in Choi et al. (2010), "While there are considerable differences among consumers regarding the potential damage of an attack, the cost of installing an update is likely fairly uniform among consumers because it typically involves shutting the system down and restarting it, as well as possibly conducting some tests before installing the updates. This cost is likely to be more uniform across users than the potential damage." This results of this model are also robust to an increasing cost of updating, by types, at a decreasing rate.

[27] Which can be observed by the examples given in Section **??** and in papers such as Ion et al. (2015).

worker weights, θ, the strategies available to the workers[28], and the probability of a successful Zero-Day attack. This probability is dependent on whether a patch has been released. If the vendor releases an update, then the probability of a successful search is $\widehat{\delta}$, whereas, when no update is released, then δ is the probability that the hacker is successful in his search.

Hacking, however, is not cost-less. A constant hacking cost, or opportunity cost, of searching for a Zero-Day, $c_s > 0$ is imposed. If the hacker decides to exploit a known vulnerability, meaning to attack vulnerability that was just patched by the vendor[29], then the hacker's cost of hacking is assumed to be zero[30]. This is to account for the relative ease of reverse engineering an update to find the vulnerability in the code. To say this another way, hackers are able to pull apart the code in an update to find the vulnerability, and attack any non-updating worker without having to pay the search cost c_s. If the vendor releases an update, then the hacker's probability of finding a Zero-Day decreases from δ to $\widehat{\delta}$, where $\widehat{\delta} < \delta$.

Under a Non-Disclosure regime, the hacker is only able to search for Zero-Day exploits, *Search* or S, or exit the game, *Exit* or X; while the worker makes no decision under this regime[31]. If policy dictates a Disclosure regime is optimal, then the hacker can still search for Zero-Day exploits, *Search* or S, or exit the game, *Exit* or X, as in the Non-Disclosure regime, but he can also choose to exploit the updated vulnerability, *Exploit* or E, on all machines that have not had the patch installed. Given a disclosed vulnerability, the workers are able to update their software, *Update* or u, or they are allowed to choose to not update, $Not\ Update$ or nu.

The remainder of the section is broken down according to the two following regimes: (i) The software vendor does not release updates, i.e. the vendor adheres

[28] If the software is of poor quality, then the hacker does not have to work as hard to gain access to the network. In other words, successful hacks increase as vendors produce software that contains more vulnerabilities.

[29] I.e. an N-Day attack.

[30] This assumption is relaxed in Section 1.5.1.

[31] Again, this is relaxed in as an extension in Section 1.5.1.

to a Non-Disclosure policy, (ii) The software vendor releases updates when a vulnerability is found, i.e. the vendor adheres to a Disclosure policy. Following this, I will determine which policy maximizes worker welfare[32].

1.3.1 Non-Disclosure Regime

If the vendor chooses not to release updates, or is forced to withhold this information, then everyone is living under a Non-Disclosure regime. In this case the worker does not make any decisions, they just use the software to gain, at most, $v\theta_i$. Therefore, the description of the payoffs of worker i can be found in the following figure, Figure 1.1.

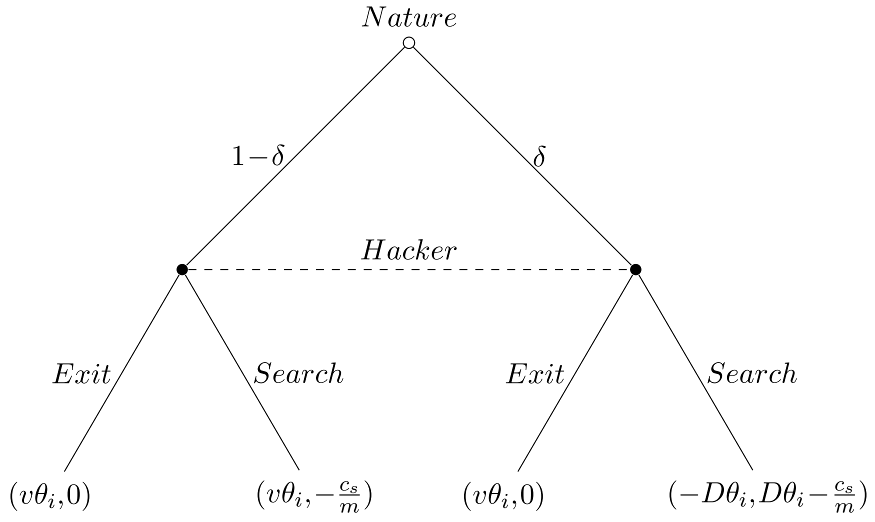


Figure 1.1: Non-Disclosure Game for Worker i

If the hacker decides to leave the game, *Exit*, then the worker always receives the full value of the software, i.e. $v\theta_i$. Otherwise, the hacker is searching for a Zero-Day exploit, i.e. *Search*, and payoffs are reliant on the probability of a successful search δ. In other words, the utility payoff of worker i, U^i_{nd}:$\{Search,\ Exit\}\times\theta_i\rightarrow\mathbb{R}$, maps from the hacker's action and worker i's type into the real numbers.

[32] See Definition 1.1 for the formal definition of welfare.

The next step is to set up the expected payoffs of the hackers under both *Search* or *Exit*. I will start with the trivial case: *Exit*. If the hacker chooses to exit the game, then the hacker's profits are trivially

$$\Pi_X^{ND}(\theta)=0.$$

Whereas, when the hacker chooses to search for a zero-day vulnerability, their only other available strategy under a Non-Disclosure regime, he receives

$$\Pi_S^{ND}(\theta)=\delta(n)\left[D\sum_i \theta_i\right]-c_s.$$

Zero-day exploits are very costly for the hacker to find, but the payoff of finding one is large, i.e. the ability to extract all information from the network.

Under Non-Disclosure, the only actions played are by the hacker, and thus the optimal strategy of the hacker can be split into three cases, that of low search costs, high search costs, and the knife-edge case of equal costs and revenues[33].

High Search Costs

Now that the cost of searching for a Zero-Day is greater than the expected profits, i.e. $c_s>\delta D\sum_{i\in I}\theta_i$, then under Non-Disclosure, the unique Nash equilibrium is to exit the game, $A_{nd}^{*}=(X)$.

1.3.2 Low Search Costs

Lastly, when the cost of searching for a Zero-Day is exceeded by the expected profits of Searching, i.e. $c_s<\delta D\sum_{i\in I}\theta_i$, then under a Non-Disclosure regime the Nash Equilibrium is that hacker will search for Zero-Days, $A_{nd}^{*}=(S)$.

[33] In some sense, this is a zero-profit condition for the hackers. This can be found in Appendix A.1.

1.3.3 Disclosure Regime

In the second case, the vendor chooses to, or is forced to, release updates every time they find a vulnerability. The workers then must choose whether to update, and thus endogenously define the two sets Γ_{nu} and Γ_u as the set of workers that do not update and the set of workers that do update, respectively. If worker i installs the update, $i \in \Gamma_u$, then she protects her device from the known vulnerability, but her machine is still vulnerable to attack since she still has a positive probability of being attacked via a Zero-Day exploit. However, when a worker decides not to install the released update, $i \in \Gamma_{nu}$, she increases her probability of being hacked via the attack strategy *Exploit*, but does not have to pay the opportunity cost, c_u. Hence, the game facing worker i is given in Figure 1.2.

Now there are two stages within the game, the first being the possible release of updates by the vendor, which happen with probability α, followed by the game between the hacker and the workers. When the vendor is unable to find a vulnerability, the $1-\alpha$ branch, the game is identical to that of the Non-Disclosure regime where the hacker must choose between searching for Zero-Days and exiting the game, and as in Figure 1.1. The hacker's action set under the Non-Disclosure branch of the Disclosure game is $A_d^{1-\alpha}$. Recall, the worker does not make any decision when no vulnerability is found[34].

When the vendor finds a vulnerability and releases an update, then both the hacker and the worker must choose their actions, A_d^{α} and A^i, respectively. When a worker chooses to update, she protects her machine from N-Day exploits, but is still vulnerable to Zero-Days. However, due to the costly nature of updating, some workers may still choose not to update[35], and leave their computers open to both Zero-Day and N-Day hacks.

[34] See Figure 1.1

[35] e.g. Ion et al. (2015)

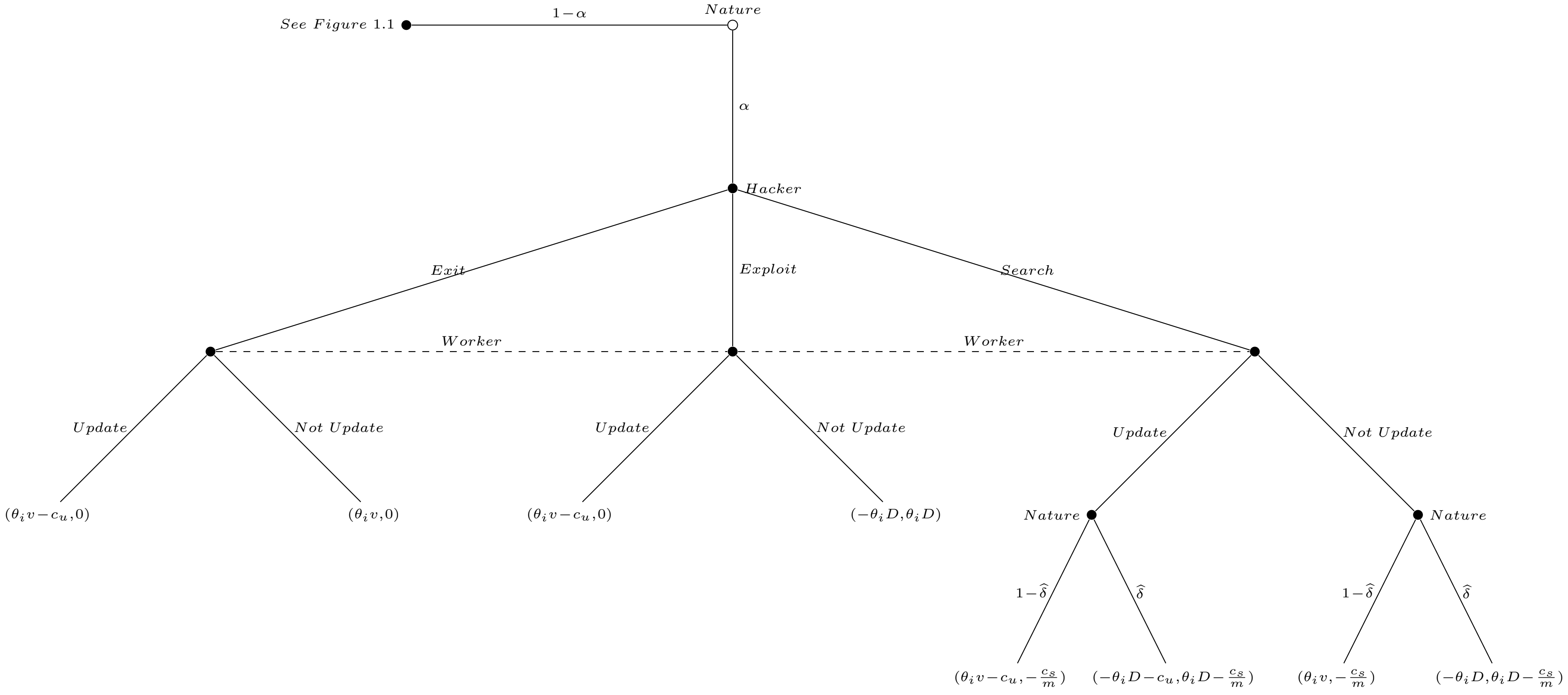


Figure 1.2: Disclosure Policy Game for Worker i

In Figure 1.2, the payoffs are laid out for the workers and the payoff of the hacker as the payoff from attacking that specific worker. The utility of worker i, U_d^i:$\{Search,\ Exploit,\ Exit\}\times\{Search,\ Exit\}\times\{Update,\ Not\ Update\}\times\theta_i$, is now dependent on the actions of the hacker when a vulnerability is both found and when one is not found, worker i's action, and the worker i's type.

Now I present the payoff to the hacker under a Disclosure regime in the form[36]: $(A_d^{\alpha},A_d^{1-\alpha})\in\{Search,\ Exploit,\ Exit\}\times\{Search,\ Exit\}$.

When the hacker chooses ($Exploit$, $Search$), he receives the expected payoff of

$$\Pi^D_{(E,S)}(\theta,\{\Gamma_u,\Gamma_{nu}\})=\alpha\left[D\sum_{i\in\Gamma_{nu}}\theta_i\right]+(1-\alpha)\left[\delta D\sum_i\theta_i-c_s\right]$$

The payoff is equivalent to the sum of damages done to of all the workers that do not update, if an update is released, as well as, the probability of successfully finding a Zero-Day times the damages done to all members of the network less the cost of searching for a Zero-Day when the vendor is unable to find a vulnerability.

The hacker could also choose to $Exit$ when the vendor is unable to find the vulnerability, yielding

$$\Pi^D_{(E,X)}(\theta,\{\Gamma_u,\Gamma_{nu}\})=\alpha\left[D\sum_{i\in\Gamma_{nu}}\theta_i\right]$$

However, if the hacker chooses to search for a Zero-Day when the vendor finds a vulnerability, and he chooses to $Search$ or $Exit$ when the vendor when no update is released, respectively. Therefore, his expected payoffs are

$$\Pi^D_{(S,S)}(\theta,\{\Gamma_u,\Gamma_{nu}\})=\left[\alpha\widehat{\delta}+(1-\alpha)\delta\right]D\sum_i\theta_i-c_s$$

[36] (action when a vulnerability is found and released (α), action when a vulnerability is not found $(1-\alpha)$). Each action is denoted as follows: $Search$ as S, $Exploit$ as E, and $Exit$ as X.

$$\Pi^D_{(S,X)}(\theta,\{\Gamma_u,\Gamma_{nu}\})=\alpha\left[\widehat{\delta}D\sum_i\theta_i-c_s\right]$$

Note that when the vendor releases an update, the probability of a successful hack decreases to $\widehat{\delta}$.

Lastly, the hacker could choose to leave the game when the vendor releases the update while either choosing *Search* or *Exit* when no update is released.

$$\Pi^D_{(X,S)}(\theta,\{\Gamma_u,\Gamma_{nu}\})=(1-\alpha)\left[\delta D\sum_i\theta_i-c_s\right]$$

$$\Pi^D_{(X,X)}(\theta,\{\Gamma_u,\Gamma_{nu}\})=0$$

Equilibrium

There are three main drivers of the Nash equilibria under Disclosure[37]:

(a) Do there exist any workers that choose not to update when an update is released[38]?

(b) Under the "Non-Disclosure" branch of the game, does the cost of finding a Zero-Day exceed the expected profits of searching? I.e.

$$c_s \lesseqgtr \delta D\sum_{i\in I}\theta_i. \tag{1.1}$$

(c) Under the "Disclosure" branch of the game, does the cost of finding a Zero-Day exceed the expected profits of searching? I.e.

$$c_s \lesseqgtr \widehat{\delta}D\sum_{i\in I}\theta_i. \tag{1.2}$$

Under Disclosure, we must solve for the actions of both the workers and the hacker.

[37] For a closed form solution with a continuum of workers, see Appendix A.2

[38] Via Assumption 1.1, these workers exist.

Each worker i must choose an action when the vendor is able to find a vulnerability, the α branch of Figure 1.2. The hacker is able to make decision under both a found vulnerability, α, and when no vulnerabilities are found, $1-\alpha$.

1.3.3.1 High Search Costs

The first case to examine is when the cost of searching exceeds the expected profits on the Non-Disclosure branch of the tree, i.e. $c_s > \delta D \sum_i \theta_i$. Similar to the Non-Disclosure case when there are high search costs in the Disclosure game and when no vulnerability is found, the $1-\alpha$ branch of the game, $A_d^{1-\alpha *} = (X)$ is the equilibrium of the sub-game.

Since the search costs are so high for the hacker, and as long as there exists at least one worker that does not update[39], then $A_d^{\alpha *} = (E)$ is the only strategy to survive elimination of strictly dominant strategies for the hacker, and is thus the only strategy in the best response for the hacker. Given the hacker strategy (E), the best response of worker i is to not update, i.e. $i \in \Gamma_{nu}^*$, if $\theta_i < \frac{c_u}{v+D}$. Otherwise, for worker j such that $\theta_j > \frac{c_u}{v+D}$, updating is optimal[40], $j \in \Gamma_u^*$.

Therefore, the Nash equilibrium of the Disclosure game is

$$((A_d^{\alpha *}, A_d^{(1-\alpha)*}), (A_i^*)_{i \in I}) = ((E, X), (nu)_{i \in \Gamma_{nu}^*}, (u)_{j \in \Gamma_u^*}) \tag{1.3}$$

Where $\Gamma_{nu}^* = \left\{ i \in I | \theta_i < \frac{c_u}{v+D} \right\}$ and $\Gamma_u^* = \left\{ j \in I | \theta_j > \frac{c_u}{v+D} \right\}$.

1.3.3.2 Medium Search Costs

The next case is when searching is profitable on the Non-Disclosure branch but not on the Disclosure branch since $\delta > \widehat{\delta}$, i.e. $\widehat{\delta} D \sum_{i \in I} \theta_i \leq c_s < \delta D \sum_{i \in I} \theta_i$. On the "Non-Disclosure" branch of the tree, the cost of searching is still exceeded by the expected

[39] Recall that this is assumed to happen.

[40] If $\theta_i = \frac{c_u}{v+D}$, then any mixture $p_j \in [0,1]$ of *Update* and *Not Update* are all equivalent to the worker.

profits of searching, and thus $A_d^{(1-\alpha)*}{=}(S)$ is his best response. However, when the vendor finds a vulnerability[41], then the expected profits of searching for a Zero-Day are surpassed by the cost of searching for a Zero-Day, then the action of (S) on the α branch yields a strictly lower payoff then exiting, $A_d^{(1-\alpha)*}{=}(X)$. Since there always exist workers that do not update, then the best action for the hacker is $A_d^{\alpha*}{=}(E)$.

Then, notice that all workers such that $\theta_i{<}\frac{c_u}{v+D}$ will be in Γ^*_{nu}, and all workers[42] $\theta_j{>}\frac{c_u}{v+D}$ will be in Γ^*_u. Therefore, the Nash equilibrium is

$$((A_d^{\alpha*},A_d^{(1-\alpha)*}),(A_i^*)_{i\in I}){=}((E,S),(nu)_{i\in\Gamma^*_{nu}},(u)_{j\in\Gamma^*_u}) \tag{1.4}$$

Where $\Gamma^*_{nu}{=}\left\{i{\in}I|\theta_i{<}\frac{c_u}{v+D}\right\}$ and $\Gamma^*_u{=}\left\{j{\in}I|\theta_j{>}\frac{c_u}{v+D}\right\}$.

1.3.3.3 Low Search Costs

The final case is to determine what happens when searching yields positive profits, i.e. $c_s{<}\hat{\delta}D\sum_i\theta_i$. Since both the hacker and the workers know whether an update has been released, then the solution can be split into the Non-Disclosure and the Disclosure sub-games. With probability $1-\alpha$, no update is released and we obtain the same solution as in the Non-Disclosure game in Section 1.3.1. Then the equilibrium of that sub-game is, as above, $A_d^{(1-\alpha)*}{=}(S)$.

Next is to determine the best response of both workers and the hacker when an update is released. The first thing to notice is that (X) is never a best response since exiting gives a payoff of zero while (S) and (E) both yield positive expected payoffs. Now I will set up the workers' best response, then determine the hacker's best response, followed by an analysis of the Nash equilibrium.

Given the hacker strategy (E), not updating, $i{\in}\Gamma^*_{nu}$, is the worker i's best response so long as $\theta_i{<}\frac{c_u}{v+D}$. However, when $\theta_j{>}\frac{cu}{v+D}$, then worker j's best response is $j{\in}\Gamma^*_u$.

[41] The α branch of Figure 1.2.

[42] As with the above cases, if there exists a worker k such that $\theta_k{=}\frac{c_u}{v+D}$, then worker k will mix with any probability $p_k{\in}[0,1]$ in the Nash Equilibrium.

Whenever the hacker plays (S), updating will not protect the worker from a hack, and thus, $i \in \Gamma_{nu}^*$ is the best response for all $i \in I$.

Allowing for the hacker to use mixed-strategies introduces the probability $\rho \in (0,1)$, where ρ is the probability that the hacker chooses (E) and $(1-\rho)$ gives (S). Then, notice that as the hacker increases the probability of searching for Zero-Days, the set of workers that will want to update decreases. Using the expected payoffs of the workers given ρ, then any worker i's best response is to not update[43], i.e. $i \in \Gamma_{nu}^*$ when $\theta_i < \frac{c_u}{\rho(v+D)}$. For all workers j such that $\theta_j > \frac{c_u}{\rho(v+D)}$, updating is their optimal action, i.e. $j \in \Gamma_u^*$. For any worker k such that $\theta_k = \frac{c_u}{\rho(v+D)}$, the worker is indifferent between updating and not updating, and will mix with any probability $p_k \in [0,1]$, where p_k is the probability of choosing update.

Now to examine the best response of the hacker on the Disclosure branch of the game given the workers' strategies. If all of the workers update, i.e. $\Gamma_u = I$, then the best response is for the hacker to search, $A_d^{\alpha *} = (S)$. Similarly, the worker strategy is $\Gamma_{nu} = I$, then $A_d^{\alpha *} = (E)$ is the only strategy in the best response for the hacker.

Define $\Omega \equiv \left\{ j \in I | \theta_j \geq \frac{c_u}{v+D} \right\}$ as the set of high-type workers that will update with positive probability if the hacker chooses (E). For some $k \in \Omega$, define $\Gamma_{nu}^k = \{ i \in I | \theta_i < \theta_k \}$ and $\Gamma_u^k = \{ j \in I | \theta_j > \theta_k \}$. Given a worker strategy of $(\Gamma_{nu}^k, (p_k(u), (1-p_k)(nu)), \Gamma_u^k)$, for some mixed strategy $p_k \in [0,1]$ for worker k, then the hacker's expected payoff of mixing with $\rho \in [0,1]$ between exploiting and searching is

$$\rho \left[D \sum_{i \in \Gamma_{nu}^k} \theta_i + (1-p_k) D \theta_k \right] + (1-\rho) \left[\widehat{\delta} D \sum_{i \in I} \theta_i - c_s \right] \tag{1.5}$$

For all $\rho \in [0,1]$, if

$$c_s > \widehat{\delta} D \sum_{i \in I} \theta_i - D \sum_{i \in \Gamma_{nu}^k} \theta_i - \big(1-p_k\big) D \theta_k \tag{1.6}$$

[43] Notice that for any $\rho \in [0, \frac{c_u}{\theta_m(v+D)})$, (nu) is the best response for all workers.

then $\rho^*=1$ is the best response for the hacker given the workers' strategy.

However, if for every value $\rho\in[0,1]$,

$$c_s<\widehat{\delta}D\sum_{i\in I}\theta_i-D\sum_{i\in\Gamma_{nu}^k}\theta_i-\big(1-p_k\big)D\theta_k \tag{1.7}$$

then the hacker will send ρ^* to zero.

The last case is if there exists a $p_k\in[0,1]$ such that Inequality 1.6 holds with equality, i.e.

$$c_s=\widehat{\delta}D\sum_{i\in I}\theta_i-D\sum_{i\in\Gamma_{nu}^k}\theta_i-\big(1-p_k\big)D\theta_k \tag{1.8}$$

then any $\rho^*\in[0,1]$ is the hacker's best response to the workers' strategy of $(\Gamma_{nu}^k,(p_k(u),(1-p_k)(nu)),\Gamma_u^k)$.

Now to solve for the Nash equilibrium.

Theorem 1.1. *Let $k_{min}\in\Omega$ be the minimal worker in Ω. If Inequality 1.6 holds for $p_{k_{min}}=1$, then the Nash Equilibrium is*

$$((A_d^{\alpha*},A_d^{(1-\alpha)*}),(A_i^*)_{i\in I})=((E,S),(nu)_{i\in\Gamma_{nu}^*},(u)_{j\in\Gamma_u^*}) \tag{1.9}$$

Where $\Gamma_{nu}^*=\big\{i\in I|\theta_i<\frac{c_u}{v+D}\big\}$ and $\Gamma_u^*=\big\{i\in I|\theta_i>\frac{c_u}{v+D}\big\}$.

Otherwise, there exists a pivotal worker[44] *$k^*\in\Omega$ and a mixed strategy for worker k^*, $p_{k^*}^*\in[0,1]$, such that Equation 1.8 holds, and the Nash equilibrium is*

$$((A_d^{\alpha*},A_d^{(1-\alpha)*}),(A_i^*)_{i\in I})=((\rho^*(E,S),(1-\rho^*)(S,S)),(nu)_{i\in\Gamma_{nu}^{k*}},(p_{k^*}^*(u),(1-p_{k^*}^*)(nu)),(u)_{j\in\Gamma_u^{k*}}) \tag{1.10}$$

Where $\rho^*=\frac{c_u}{\theta_{k^*}(v+D)}$, $\Gamma_{nu}^{k*}=\{i\in I|\theta_i<\theta_{k^*}\}$, and $\Gamma_u^{k*}a=\{i\in I|\theta_i>\theta_{k^*}\}$.

Proof. Under Non-Disclosure, the hacker will search for Zero-Days, (S). Then, recall that, that if for all workers $k\in\Omega$ and all $p_k\in[0,1]$ for each k such that Inequality 1.6

[44] I.e. a worker that is indifferent between updating and not in the Nash equilibrium.

holds, then the hacker's best response is to always exploit, (E), when a vulnerability is disclosed. Next, given the hacker strategy of (E), then all workers $i{\in}I$ such that $\theta_i{<}\frac{c_u}{v+D}$ will not update, i.e. $i{\in}\Gamma^*_{nu}$. If $\theta_j{>}\frac{c_u}{v+D}$, then $j{\in}\Gamma^*_u$ is worker j's best response. Therefore, $((E,S),(nu)_{i\in\Gamma^*_{nu}},(u)_{j\in\Gamma^*_u})$ is in the best response of the hacker and all of the workers, and is thus a Nash equilibrium.

However, given a worker $k^*{\in}\Omega$ and a $p_{k^*}{\in}[0,1]$ such that Equation 1.8 holds, then the hacker's best response is to mix, with any $\rho{\in}[0,1]$, between E and S. Next, given the hacker strategy of $(\rho(E),(1{-}\rho)(S))$, where $\rho{=}\frac{c_u}{\theta_{k^*}(v+D)}$ and $k^*{\in}\Omega$, then all workers $i{\in}I$ such that $\theta_i{<}\frac{c_u}{\rho(v+D)}$ will not update, i.e. $i{\in}\Gamma^{k*}_{nu}$. If $\theta_j{>}\frac{c_u}{\rho(v+D)}$, then $j{\in}\Gamma^{k*}_u$ is worker j's best response. Lastly, notice that, given $\rho{=}\frac{c_u}{\theta_{k^*}(v+D)}$, worker k^* is indifferent between updating and not updating. Then worker k^*'s best response is to mix between (u) and (nu) with any probability $p_{k^*}{\in}[0,1]$, which includes $p^*_{k^*}$. Therefore, $((\rho^*(E,S),(1{-}\rho^*)(S,S)),(nu)_{i\in\Gamma^{k*}_{nu}},(p^*_{k^*}(u),(1{-}p^*_{k^*})(nu)),(u)_{j\in\Gamma^{k*}_u})$ is in the best response of the hacker and all of the workers, and is thus a Nash equilibrium. □

MANET Example

Let there be 3 worker nodes on the MANET such that the weights of each worker are $\theta{=}\left\{\frac{1}{4},\frac{1}{2},\frac{3}{4}\right\}$. Assume that the cost of updating is one hour[45], $c_u{=}1$. Each worker has a valuation of the network $v{=}2$ and damage parameter $D{=}1$. Observe that $\frac{c_u}{v+D}{=}\frac{1}{3}$, yielding $\Omega{\equiv}\{2,3\}$, or, in other words, worker 1 will never update, while workers 2 and 3 are sometimes willing to update.

Now to setup the hacker problem. If the hacker attempts to find a Zero-Day when a vulnerability has not been disclosed, then he is successful half the time, i.e. $\delta{=}\frac{1}{2}$. However, if the vendor is able to find a vulnerability and release an update, then the hacker's probability of success falls to one-third, $\widehat{\delta}{=}\frac{1}{3}$. Lastly, assume that the cost of finding a Zero-Day is $c_s{=}\frac{1}{8}$.

[45] This is a reasonable assumption as an average time spent installing the update since a minor update could take minutes, but major updates could take hours.

The first step is to determine which cost scenario this parameterization falls under. By these assumed values, $c_s < \delta D \sum_{i\in I} \theta_i$ since $\frac{1}{8} < \frac{3}{4}$. The next step is to determine which part of Theorem 1.1 is satisfied. For $k=2$, notice that Equation 1.8 holds when $p_k^* = \frac{3}{4}$. Solving for the optimal mixed strategy of the hacker, $\rho^* = \frac{c_u}{\theta_2(v+D)} = \frac{2}{3}$. Giving a Nash equilibrium of

$$\left(\left(\frac{2}{3}(E,S),\frac{1}{3}(S,S)\right),(nu)_{i=1},\left(\frac{3}{4}(u),\frac{1}{4}(nu)\right)_{k=2},(u)_{j=3}\right)$$

The final check is to see if this is the only equilibrium of the game. For the other case of $k=3$, the only solution to Equation 1.8 is $p_k = \frac{3}{2} > 1$. Therefore, the solution is unique.

1.4 Welfare Analysis

The "Optimal Disclosure Policy" must first be defined followed by solving for the optimal policy for each of the different search cost scenarios found in Section 1.3.

Definition 1.1. *The optimal policy $\Psi^* \in \{Disclosure,\ Non-Disclosure\}$ is chosen such that:*

$$\Psi^* = argmax_{\psi\in\{d,\ nd\}} \left\{ \sum_{i\in I} U_d(A_d^{\alpha *}, A_d^{(1-\alpha)*}, A_i^*, \theta_i), \sum_{i\in I} U_{nd}(A_{nd}^*, \theta_i) \right\} \tag{1.11}$$

Where $((A_d^{\alpha *}, A_d^{(1-\alpha)*}), (A_i^*)_{i\in I})$ and (A_{nd}^*) are the Nash equilibria under Disclosure and Non-Disclosure, respectively.

The optimal policy is to either force the Disclosure or Non-Disclosure regime in order to maximize the sum of worker utility. In the following sections, I examine the the optimal policies under the Nash equilibria listed above for each of the four cases: High Search Cost, Knife-Edge[46], Medium Search Cost, and Low Search Cost.

[46] In Appendix A.1.

1.4.1 High Search Costs

Under High Search Costs, recall that in the Nash equilibrium[47] the hacker chooses to exploit the N-Day under Disclosure and to exit the game under Non-Disclosure. Under Disclosure, all low-type workers, the workers in Γ^*_{nu}, are hacked if a vulnerability is found; while all other workers must pay the cost of updating, which is assumed to be strictly greater than zero. Under Non-Disclosure, the hacker exits the game, and all workers obtain $\theta_i v$. Defining $\xi^* = |\Gamma^*_u|$, i.e. the number of workers that update under a Disclosure policy, then the optimal policy can be solved for via the following theorem.

Theorem 1.2. *If $c_s > \delta D \sum_{i \in I} \theta_i$, then Non-Disclosure is the optimal policy.*

Proof. Notice that

$$\begin{aligned} \sum_{i \in I} U_d(A_d^{\alpha *}, A_d^{(1-\alpha)*}, A_i^*, \theta_i) &= v \sum_{j \in \Gamma^*_u} \theta_j - \xi^* c_u - D \sum_{i \in \Gamma^*_{nu}} \theta_i \\ &< v \sum_{j \in \Gamma^*_u} \theta_j + v \sum_{i \in \Gamma^*_{nu}} \theta_i \\ &= \sum_{i \in I} U_{nd}(A^*_{nd}, \theta_i) \end{aligned} \tag{1.12}$$

Therefore, $\Psi^* = \{Non-Disclosure\}$. □

Hence, when the hacker faces high search costs, Non-Disclosure is the optimal policy.

1.4.2 Medium Search Costs

In this case[48], under a Non-Disclosure regime the hacker searches for a Zero-Day. However, under Disclosure, the hacker chooses to exploit the released vulnerability.

[47] $((A_d^{\alpha *}, A_d^{(1-\alpha)*}), (A_i^*)_{i \in I}) = ((E, X), (nu)_{i \in \Gamma^*_{nu}}, (u)_{j \in \Gamma^*_u})$

[48] $\widehat{\delta} D \sum_{i \in I} \theta_i \leq c_s < \delta D \sum_{i \in I} \theta_i$

Then comparing the sum of the utilities, under Non-Disclosure the workers receive

$$\sum_{i\in I} U_{nd}(A_{nd}^*,\theta_i)=(1-\delta)\left(v\sum_{i\in I}\theta_i\right)-\delta\left(D\sum_{i\in I}\theta_i\right) \tag{1.13}$$

Then the welfare under the Disclosure regime is

$$\sum_{i\in I} U_d(A_d^{\alpha*},A_d^{(1-\alpha)*},A_i^*,\theta_i)=\alpha\left[v\sum_{j\in\Gamma_u^*}\theta_j-D\sum_{i\in\Gamma_{nu}^*}\theta_i\right]+(1-\alpha)\left[v\sum_{i\in I}\theta_i\right]-\xi^*c_u \tag{1.14}$$

Therefore, solving for the optimal policy is dependent on

$$\sum_{i\in\Gamma_{nu}^*}\theta_i+\xi^*\frac{c_u}{v+D}\lesseqgtr\delta\sum_{i\in I}\theta_i \tag{1.15}$$

Where[49] the left-hand side is the sum of the hacked low-type workers plus the value paid by high-type workers to update their machines. While, the right-hand side are the expected losses to all workers because of the ability of the hacker to successfully find a Zero-Day and hack all of their machines.

Thus, the optimal policy under medium search costs is as follows.

Theorem 1.3. *If $\widehat{\delta}D\sum_{i\in I}\theta_i\leq c_s<\delta D\sum_{i\in I}\theta_i$ then there exist three cases under Inequality 1.15,*

1. *If $\sum_{i\in\Gamma_{nu}^*}\theta_i+\xi^*\frac{c_u}{v+D}<\delta\sum_{i\in I}\theta_i$, then Disclosure is the optimal policy.*

2. *If $\sum_{i\in\Gamma_{nu}^*}\theta_i+\xi^*\frac{c_u}{v+D}>\delta\sum_{i\in I}\theta_i$, then Non-Disclosure is the optimal policy.*

3. *If $\sum_{i\in\Gamma_{nu}^*}\theta_i+\xi^*\frac{c_u}{v+D}=\delta\sum_{i\in I}\theta_i$, then both Non-Disclosure and Disclosure are optimal policies.*

[49] Notice that, given arbitrary Pareto weights, λ_i for all $i\in I$, Disclosure is the optimal policy if

$$\sum_{i\in\Gamma_{nu}^*}\lambda_i\theta_i+\frac{c_u}{v+D}\sum_{i\in\Gamma_u^*}\lambda_i<\delta\sum_{i\in I}\lambda_i\theta_i$$

Proof. Notice that the welfare of the workers can be calculated via Equations 1.13 and 1.14. In case 1, given $\sum_{i\in\Gamma^*_{nu}}\theta_i+\xi^*\frac{c_u}{v+D}<\delta\sum_{i\in I}\theta_i$,

$$\sum_{i\in I}U_d(A_d^{\alpha*},A_d^{(1-\alpha)*},A_i^*,\theta_i)>\sum_{i\in I}U_{nd}(A^*_{nd},\theta_i) \tag{1.16}$$

Therefore, $\Psi^*=\{Disclosure\}$.

The other cases trivially follow. □

In other words, so long as the expected losses from a Zero-Day exceed the cost of the low type workers being hacked since they did not update and the cost of updating for all ξ^* of the high type workers, then Disclosure is the optimal policy.

1.4.3 Low Search Costs

Now[50] the vendor, or policy maker that forces the vendor to follow a specific policy, must decide which policy maximizes the sum of the utility of the workers. Recall that the Nash equilibrium of the Non-Disclosure game is $A_d^{(1-\alpha)*}=(S)$, while the Nash equilibria of the Disclosure game take the form of mixing between (E) and (S) for the hacker while the workers split into $(\Gamma^{k*}_{nu},(p^*_k(u),(1-p^*_k)(nu)),\Gamma^{k*}_u)$. I will begin by analyzing the optimal policy for all low-type workers, followed by the optimal policy for all high-type workers. To conclude the section I will then combine these results to find the optimal policy.

For all workers $i\in\Gamma^{k*}_{nu}$, then we are able to analyze which policy they would prefer by solving

$$-\delta D\sum_{i\in\Gamma^{k*}_{nu}}\theta_i+(1-\delta)v\sum_{i\in\Gamma^{k*}_{nu}}\theta_i\lesseqgtr-\rho^*D\sum_{i\in\Gamma^{k*}_{nu}}+(1-\rho^*)\left[-\widehat{\delta}D\sum_{i\in\Gamma^{k*}_{nu}}\theta_i+(1-\widehat{\delta})v\sum_{i\in\Gamma^{k*}_{nu}}\theta_i\right] \tag{1.17}$$

[50] $c_s<\hat{\delta}D\sum_{i\in I}\theta_i$

First, notice that if $\rho^*=1$, then $\Gamma^*_{nu}=\left\{i\in I|\theta_i<\frac{c_u}{v+D}\right\}$ and Equation 1.17 becomes

$$-\delta D\sum_{i\in\Gamma^*_{nu}}\theta_i+(1-\delta)v\sum_{i\in\Gamma^*_{nu}}\theta_i\lesseqgtr -D\sum_{i\in\Gamma^*_{nu}}\theta_i \tag{1.18}$$

Since $\delta<1$, then Non-Disclosure is always optimal for workers that do not update.

Next, for any $\rho^*\in[0,1)$, I analyze the effects of search on the workers' welfare. Notice that

$$-\delta D\sum_{i\in\Gamma^{k*}_{nu}}\theta_i+(1-\delta)v\sum_{i\in\Gamma^{k*}_{nu}}\theta_i<-\widehat{\delta}D\sum_{i\in\Gamma^{k*}_{nu}}\theta_i+(1-\widehat{\delta})v\sum_{i\in\Gamma^{k*}_{nu}}\theta_i \tag{1.19}$$

since $\delta>\widehat{\delta}$ is strictly increasing. This could indicate that Disclosure may be welfare improving, since there are fewer vulnerabilities for hackers to find. However, since $\rho>0$, then the right-hand side decreases due to the fact that the hacker may choose to exploit the released vulnerability instead of searching for a Zero-Day.

Disclosure is the optimal policy for all workers that do not update so long as

$$\begin{aligned}\rho^*&<\frac{\left[-\widehat{\delta}D+(1-\widehat{\delta})v\right]-[-\delta D+(1-\delta)v]}{(1-\widehat{\delta})(v+D)}\\ \iff\rho^*&<\frac{\delta-\widehat{\delta}}{1-\widehat{\delta}}\end{aligned} \tag{1.20}$$

Notice that both the left-hand side and the right-hand side are strictly positive. Thus, the workers that do not update, workers in Γ^{k*}_{nu}, will sometimes want Disclosure to be the chosen policy.

Workers[51] $j\in\Gamma_u^{k*}$, then face the welfare decision of

$$-\delta D\sum_{j\in\Gamma_u^{k*}}\theta_j+(1-\delta)v\sum_{j\in\Gamma_u^{k*}}\theta_j\lesseqgtr\rho^*\left(v\sum_{j\in\Gamma_u^{k*}}\theta_j-\xi c_u\right)$$
$$+(1-\rho^*)\left[-\widehat{\delta}D\sum_{j\in\Gamma_u^{k*}}\theta_j+(1-\widehat{\delta})v\sum_{j\in\Gamma_u^{k*}}\theta_j-\xi^*c_u\right] \quad (1.21)$$

As with the low-type workers, the first analysis to be done is when $\rho^*=1$ and $\Gamma_u^*=\left\{j\in I|\theta_j>\frac{c_u}{v+D}\right\}$. Accordingly, Equation 1.21 can now be written as

$$-\delta D\sum_{j\in\Gamma_u^*}\theta_j+(1-\delta)v\sum_{j\in\Gamma_u^*}\theta_j\lesseqgtr v\sum_{j\in\Gamma_u^*}\theta_j-\xi^*c_u \quad (1.22)$$

Disclosure is then optimal as long as $\delta\sum_{j\in\Gamma_u^*}\theta_j>\frac{\xi^*c_u}{v+D}$, i.e. the expected losses of a search for Zero-Days exceeds the cost of installing updates.

For any $\rho^*\in[0,1)$, such that $\Gamma_u^{k*}=\left\{j\in I|\theta_j>\frac{c_u}{\rho^*(v+D)}\right\}$, Disclosure is optimal when

$$-\delta D\sum_{j\in\Gamma_u^{k*}}\theta_j+(1-\delta)v\sum_{j\in\Gamma_u^{k*}}\theta_j<-\widehat{\delta}D\sum_{j\in\Gamma_u^{k*}}\theta_j+(1-\widehat{\delta})v\sum_{j\in\Gamma_u^{k*}}\theta_j$$

Leading to the notion that, again, Disclosure might be the optimal choice for these workers. On the grounds that $\rho^*>0$ and $c_u>0$, for Disclosure to be the optimal choice of workers $j\in\Gamma_u^*$, the following must hold.

$$\xi^*c_u<\left(\delta-(1-\rho^*)\widehat{\delta}\right)\sum_{j\in\Gamma_u^{k*}}\theta_j \quad (1.23)$$

For workers $i\in\Gamma_{nu}^{k*}$, Disclosure decreases the probability of being hacked by a Zero-Day, but it also increases their probability of being hacked since the hacker can exploit the N-Day vulnerability that these workers are not willing to defend against. However,

[51] These are the high-type workers such that they are not the pivotal worker. The pivotal worker is the worker with $\theta_k=\frac{c_u}{\rho^*(v+D)}$ that is indifferent between updating and not updating.

workers $j\in\Gamma_u^{k*}$ are more likely to want a Disclosure regime since they both obtain the benefit of hackers having less vulnerabilities to search over as well as protection from the N-Day exploits since they will sometimes update.

Now to examine the welfare over all the workers, where the optimal policy depends on the welfare equations

$$\begin{aligned}\sum_{i\in I}U_d(A_d^{\alpha*},A_d^{(1-\alpha)*},A_i^*,\theta_i)=\rho^*&\left(v\sum_{j\in\Gamma_u^{k*}}\theta_j+p_k^*(v\theta_k-c_u)-(1-p_k^*)D\theta_k-D\sum_{i\in\Gamma_{nu}^{k*}}\theta_i\right)\\&+(1-\rho^*)\left(\widehat{\delta}\left(-D\sum_{i\in I}\theta_i\right)+(1-\widehat{\delta})\left(v\sum_{i\in I}\theta_i\right)\right)-\xi^*c_u\end{aligned}\tag{1.24}$$

$$\sum_{i\in I}U_{nd}(A_{nd}^*,\theta_i)=(1-\delta)\left(v\sum_{i\in I}\theta_i\right)-\delta\left(D\sum_{i\in I}\theta_i\right)\tag{1.25}$$

Then by comparing the two welfare equations, the following condition describes the optimal policy.

$$\sum_{i\in\Gamma_{nu}^{k*}}\theta_i+\left(\frac{D}{v+D}-p_k^*-\widehat{\delta}\right)\theta_k+\frac{(\xi^*+p_k^*)c_u}{\rho^*(v+D)}\lessgtr\left(\frac{\delta-(1-\rho^*)\widehat{\delta}}{\rho^*}\right)\sum_{i\in I}\theta_i\tag{1.26}$$

As with the medium cost case, the left-hand side represents the cost under a Disclosure policy and the right-hand side represents the Non-Disclosure regime. The first term on the left-hand side is the value of the set of low-type workers lost due to the exploitation of the disclosed vulnerability. The following term are the expected costs faced by the pivotal worker k. The final term on the left-hand side is the cost associated with updating for the high-type workers.

The right-hand side is the expected damages done by search under Non-Disclosure less the damages done by search under Disclosure[52]. Since $\frac{c_u}{\rho^*(v+D)}=\theta_k$, then Equation

[52] Recall that the hacker is now willing to mix between Search and Exploit.

1.26 can be written as

$$\sum_{i\in\Gamma_{nu}^{k*}}\theta_i+\left(\frac{D}{v+D}-\widehat{\delta}+\xi^*\right)\theta_k\lesseqgtr\left(\frac{\delta-(1-\rho^*)\widehat{\delta}}{\rho^*}\right)\sum_{i\in I}\theta_i \tag{1.27}$$

Hence, the optimal policy under low search costs is as follows.

Theorem 1.4. *Let* $c_s<\widehat{\delta}D\sum_{i\in I}\theta_i$. *Then Inequality 1.27 yields three distinct cases.*

1. *If* $\sum_{i\in\Gamma_{nu}^{k*}}\theta_i+\left(\frac{D}{v+D}-\widehat{\delta}+\xi^*\right)\theta_k<\left(\frac{\delta-(1-\rho^*)\widehat{\delta}}{\rho^*}\right)\sum_{i\in I}\theta_i$, *then Disclosure is the optimal policy.*
2. *If* $\sum_{i\in\Gamma_{nu}^{k*}}\theta_i+\left(\frac{D}{v+D}-\widehat{\delta}+\xi^*\right)\theta_k>\left(\frac{\delta-(1-\rho^*)\widehat{\delta}}{\rho^*}\right)\sum_{i\in I}\theta_i$, *then Non-Disclosure is the optimal policy.*
3. *If* $\sum_{i\in\Gamma_{nu}^{k*}}\theta_i+\left(\frac{D}{v+D}-\widehat{\delta}+\xi^*\right)\theta_k=\left(\frac{\delta-(1-\rho^*)\widehat{\delta}}{\rho^*}\right)\sum_{i\in I}\theta_i$, *then both Disclosure and Non-Disclosure are optimal.*

Proof. Notice that the welfare of the workers is given by Equations 1.24 and 1.25 the following is obtained,

$$\sum_{i\in\Gamma_{nu}^{k*}}\theta_i+\left(\frac{D}{v+D}-p_k^*-\widehat{\delta}\right)\theta_k+\frac{(\xi^*+p_k^*)c_u}{\rho^*(v+D)}<\left(\frac{\delta-(1-\rho^*)\widehat{\delta}}{\rho^*}\right)\sum_{i\in I}\theta_i$$

Notice that $\frac{c_u}{\rho^*(v+D)}=\theta_k$, and thus the equation can be rewritten as

$$\sum_{i\in\Gamma_{nu}^{k*}}\theta_i+\left(\frac{D}{v+D}-\widehat{\delta}+\xi^*\right)\theta_k<\left(\frac{\delta-(1-\rho^*)\widehat{\delta}}{\rho^*}\right)\sum_{i\in I}\theta_i$$

Hence, $\Psi^*=\{Disclosure\}$.

The other cases trivially follow. □

Disclosure is the optimal policy so long as the losses of being exploited by an N-Day and paying the cost of updating is less than the expected losses of a Zero-Day attack.

1.5 Discussion

According to Sym (2016) the cost of finding Zero-Days has significantly increased over the last couple of years. This shift has altered the environment from one akin to the Medium Search Cost case to one more closely approximated by the High Search Cost case. Thus, under the games listed above, the optimal policy will shift from Disclosure being sometimes optimal toward Non-Disclosure always being optimal.

This is also due to the findings in Sym (2018) that clearly shows that the ease of hacking machines that have not updated has increased. For example, only 2.3% of people are on the latest version of Android. As vendors continue to release updates, and workers are refusing to update their machines, hackers can take full advantage of easy hacks. Therefore, the complementary effects of the cost of finding Zero-Days increasing while the vendors are attempting to release more updates that are not being updated by workers should cause a change in policy.

However, the policy change may not have to take the form of forcing Non-Disclosure. Instead, what if the vendor can choose to change the game? In Section 1.5.1, I analyze how the forthcoming change to Microsoft 7 and 10 updating procedures could change the game. I will set up the following game under which Microsoft has just introduced a new monthly charge to receive updates. Microsoft intends to implement this policy starting on January 14^{th}, 2020, which, coincidentally, is the same day that Windows 7 will no longer be supported. But with a large number of Windows users still using Windows 7, Microsoft needed to come up with a policy to protect these users and maintain their market share. Even though the policy, as outlined below, not only affects Windows 7, but will also have significant impacts on Windows 10 users, I will focus on Windows 7 users decisions on and after January 14^{th}, 2020.

1.5.1 Extension: Microsoft's New Disclosure Policy

On September 6^{th}, 2018, Microsoft posted a blog article entitled[53] "Helping Customers Shift to a Modern Desktop" in which they laid out their new updating, i.e. disclosure, policy. Starting on January 14^{th}, 2020, Windows 7 users will no longer receive their usual second Tuesday updates, but will be able to pay for "Extended Support" from Microsoft under which Microsoft will release updates *for your machine* for a given fee[54].

To model this, the cost of updating must increase due to this forthcoming fee. Let $\phi_u > 0$ be the new service charge paid by the worker to keep their machine up to date. However, this is not the only available choice to the worker anymore. The worker can also choose to shift toward using a different version, i.e. Windows 10, for which the worker must pay a cost $c_v > 0$. If the worker shifts toward using the new version of the software, then the hacker is not able to attack the worker, not even via Zero-Days.

Assumption 1.2. *The cost of changing to the new version of the software, c_v, is assumed to be such that there potentially exists at least one worker that is now willing to change to the new version, i.e.* $\frac{c_v}{\delta(v+D)} \in (\theta_1, \theta_m)$.

The first thing to notice is that, due to the availability of other version of software to the worker, the hacker is immediately effected by the new policy. The hacker's payoffs are now decreasing in the number of workers that are willing to install the new software version.

I make the following assumption on the availability of the released vulnerability, N-Day, to the hacker[55]

[53] See https://www.microsoft.com/en-us/microsoft-365/blog/2018/09/06/helping-customers-shift-to-a-modern-desktop/

[54] The size of the fee has yet to be revealed, but this fee will increase over time.

[55] This is a strong assumption, but reasonable in a static model. In a dynamic model this assumption could be relaxed to account for the timing between the disclosure of a vulnerability and the release of an update. A dynamic model could also allow for external groups reporting and disclosing vulnerabilities.

Assumption 1.3. *In order to observe the released vulnerability, the hacker must pay ϕ_u, but does not have to pay c_u.*

If the hacker wants to gain access to the disclosure of the vulnerability, the hacker must pay the subscription fee for the "Extended Support", ϕ_u. However, the hacker does not have to pay c_u since the hacker could do something like enroll an old computer in the updating scheme in order to be notified of vulnerabilities. Consequently, the cost of exploiting N-Days has increased since $\phi_u{>}0$. To be clear, Microsoft's new policy is fascinating since it has the potential to increase the cost of exploiting N-Days while also decreasing the effectiveness of Zero-Days against Windows 7.

New Policy Game Given this new policy, I now explicitly define the new game by the following game tree for this new type of Disclosure policy. To make the game tree more readable, I have split it into two sub-games determined by nature, the Non-Disclosure branch in Figure 1.3 and the Disclosure branch in Figure 1.4. I still assume that the probability of the vendor, Microsoft, finding a vulnerability[56] is α. Since the worker can now change the version of software she is using, she is able to make strategic decisions in both games.

Now to set up the Non-Disclosure branch of the game tree. The vendor was unable to find a vulnerability, and thus the hacker is only able to search for a Zero-Day, i.e. the hacker can only choose an action, $A_M^{(1-\alpha)}$, from the set $\{Search,\ Exit\}$. Searching for a Zero-Day is not as effective as in the above games due to the fact that workers are now able to change their software version to avoid being attacked. The worker choice is to either continue using the old version Old, or to start using the new version of the software, New, with an action $A_{M,i}^{(1-\alpha)}$.

In Figure 1.3, the payoffs are laid out for the workers as well as the payoffs for the hacker attacking that worker. The utility of worker i, $U^i_{M;nd}{:}\{Search,Exit\}\times\{New,Old\}\times\theta_i$, is dependent on the actions of the hacker, worker i's action, and the

[56] Should we assume that this is constant? Will α decrease due to a decreasing investment in finding these vulnerabilities? These are good questions, but they are beyond the scope of this paper.

worker i's type. I am going to include all players that use the old software in Γ_{nu}, and all workers that switch versions in Γ_v. Recall, there are no directed attacks in this game, hence the payoff functions for the hacker still need to be presented.

I will start here with the trivial case of the hacker choosing to leave the game: *Exit*. If this is the case, then the hacker's payoff

$$\Pi_X^{M(1-\alpha)}(\theta,\{\Gamma_{nu},\Gamma_v\})=0$$

When the hacker decides to search for a Zero-Day in the old version of the software, he receives a payoff of

$$\Pi_S^{M(1-\alpha)}(\theta,\{\Gamma_{nu},\Gamma_v\})=\delta(n)\left(D\sum_{i\in\Gamma_{nu}}\theta_i\right)-c_s$$

The next step is to formalize the Disclosure branch of the tree under Microsoft's new policy. As the vendor has found and released an update with the probability α, both the hacker and workers have an extra action they could take. The hacker has the same set of actions in this case as in the Disclosure case above to pick from, i.e. he picks an action, A_M^α, from the set $\{Exploit,Search,Exit\}$. This new policy also allows the worker the ability to not update, update, or switch software versions, or choose $A_{M,i}^\alpha\in\{New\ Version,Update,Not\ Update\}$.

The utility of worker i is now $U_{M;d}^i$:$\{Exploit,Search,Exit\}\times\{New\ Version,Update,Not\ Update\}\times\theta_i$. Leaving the final step in establishing the game created by Microsoft's new policy as describing the payoff functions of the hacker.

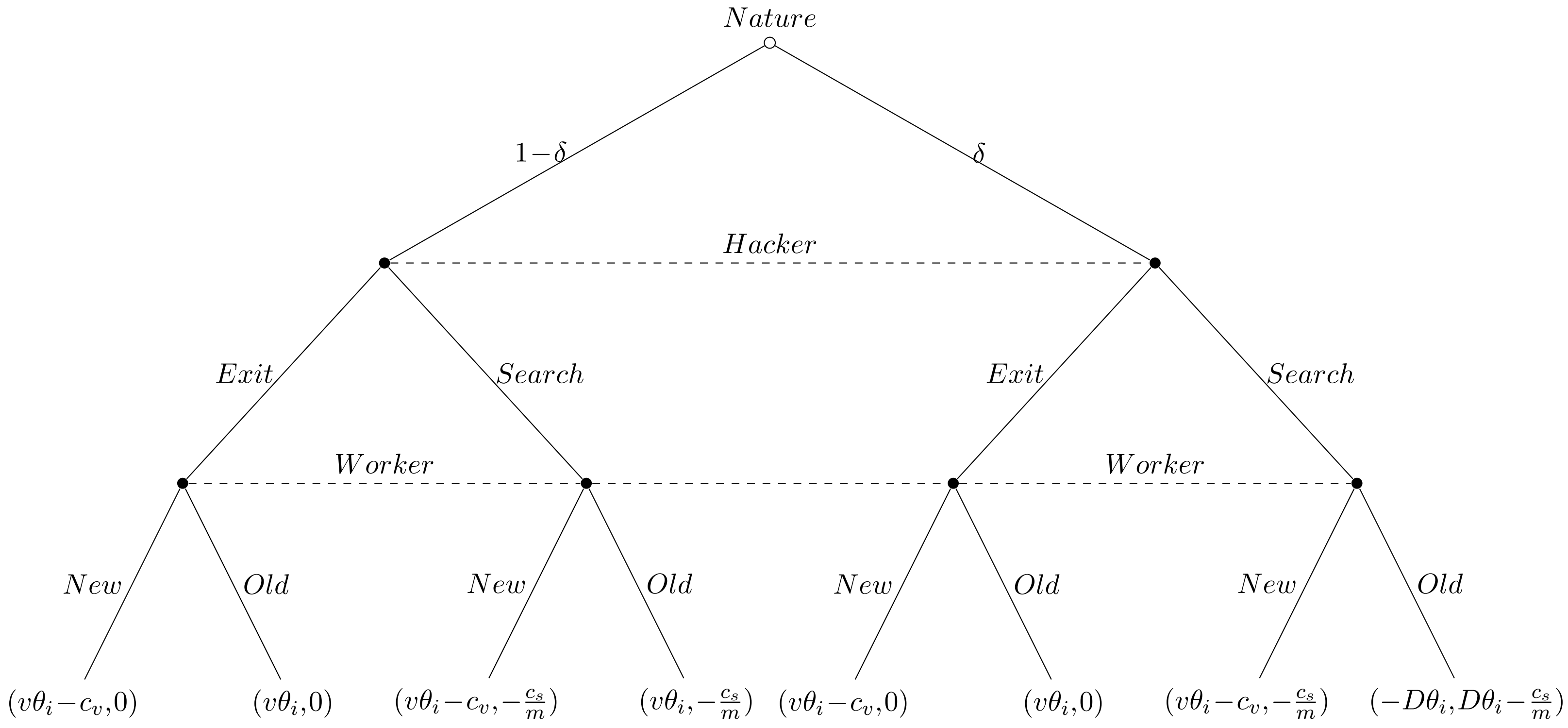


Figure 1.3: Non-Disclosure Branch of Game for Worker i

When the hacker chooses (*Exploit*,*Search*), he receives the expected payoff of

$$\Pi^{M}_{(E,S)}(\theta,\{\Gamma_{nu},\Gamma_{u},\Gamma_{v}\})=\alpha\left[D\sum_{i\in\Gamma_{nu}}\theta_i-\phi_u\right]+(1-\alpha)\left[\delta D\sum_{i\in\Gamma_{nu}\cup\Gamma_u}\theta_i-c_s\right] \tag{1.28}$$

In expectation, the hacker receives the sum of damages done to all workers in Γ_{nu} when a vulnerability is disclosed, and, when the vulnerability is not found, he receives the expected value of a Zero-Day less the cost of searching. The expected value of a Zero-Day has decreased since the hacker is now unable to attack any worker that has decided to shift toward the use of the new version.

However, if the hacker decides to *Exit* instead of *Search* when no vulnerability is disclosed, then the expected payoff of the hacker is

$$\Pi^{M}_{(E,X)}(\theta,\{\Gamma_{nu},\Gamma_{u},\Gamma_{v}\})=\alpha\left[D\sum_{i\in\Gamma_{nu}}\theta_i-\phi_u\right] \tag{1.29}$$

If the hacker decides to search for Zero-Days when an update is released, then the expected payoff to the hacker from searching or exiting when no vulnerability is found by the vendor are as follows.

$$\Pi^{M}_{(S,S)}(\theta,\{\Gamma_{nu},\Gamma_{u},\Gamma_{v}\})=\alpha\left[\widehat{\delta}D\sum_{i\in\Gamma_{nu}\cup\Gamma_u}\theta_i\right]+(1-\alpha)\left[\delta D\sum_{i\in\Gamma_{nu}\cup\Gamma_u}\theta_i\right]-c_s \tag{1.30}$$

$$\Pi^{M}_{(S,X)}(\theta,\{\Gamma_{nu},\Gamma_{u},\Gamma_{v}\})=\alpha\left[\widehat{\delta}D\sum_{i\in\Gamma_{nu}\cup\Gamma_u}\theta_i-c_s\right] \tag{1.31}$$

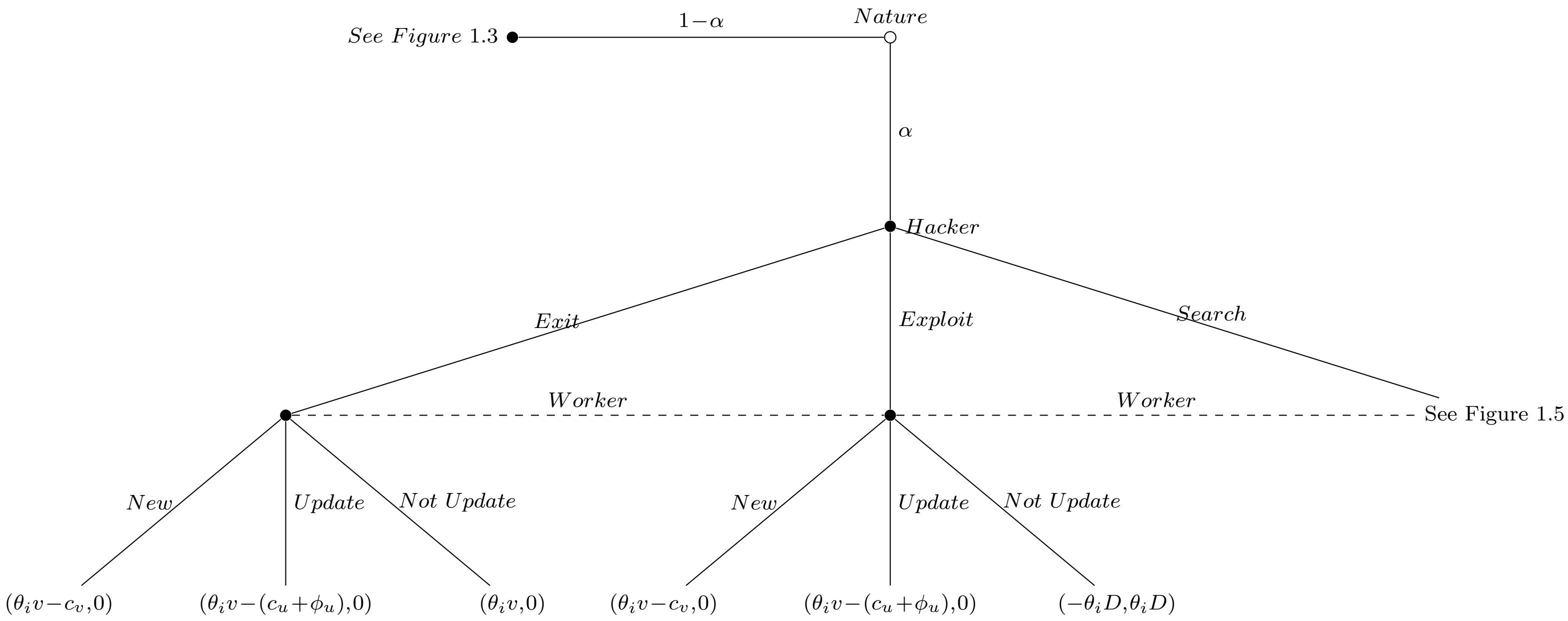


Figure 1.4: Disclosure Policy Game for Worker i

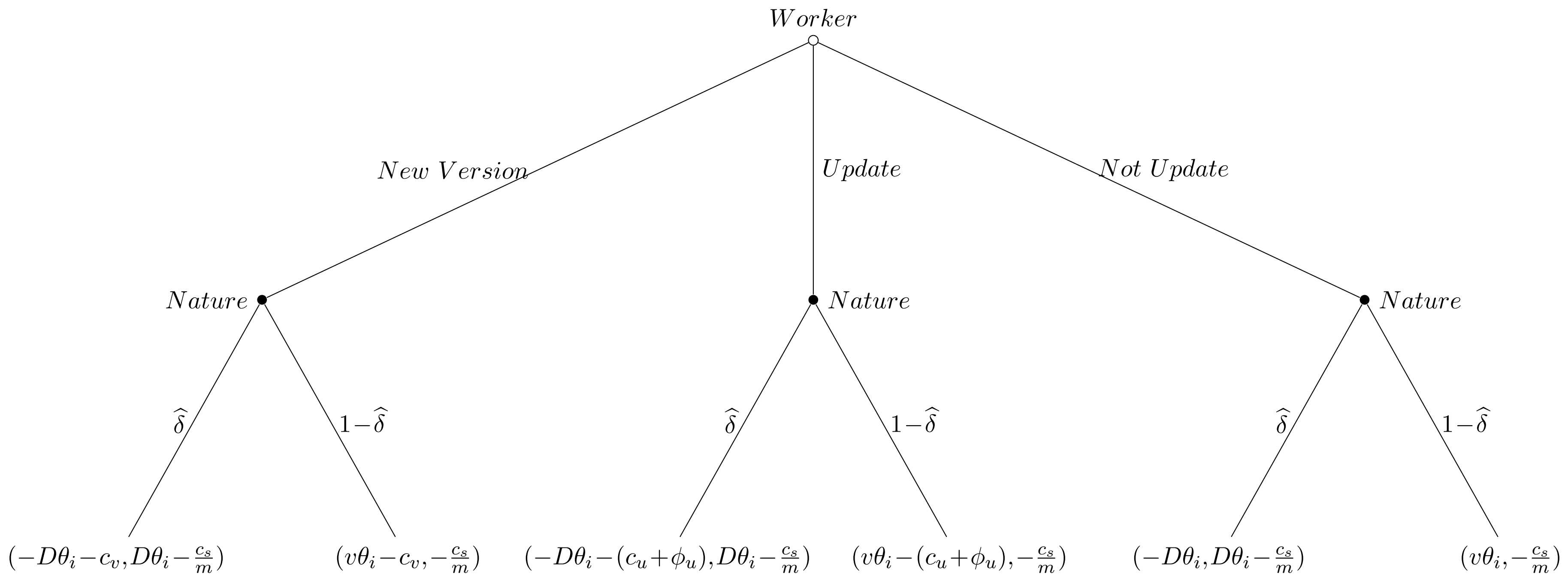

$Worker$
$New\ Version$
$Update$
$Not\ Update$
$Nature$
$Nature$
$Nature$
$\widehat{\delta}$
$1-\widehat{\delta}$
$\widehat{\delta}$
$1-\widehat{\delta}$
$\widehat{\delta}$
$1-\widehat{\delta}$
$(-D\theta_i-c_v,D\theta_i-\frac{c_s}{m})$
$(v\theta_i-c_v,-\frac{c_s}{m})$
$(-D\theta_i-(c_u+\phi_u),D\theta_i-\frac{c_s}{m})$
$(v\theta_i-(c_u+\phi_u),-\frac{c_s}{m})$
$(-D\theta_i,D\theta_i-\frac{c_s}{m})$
$(v\theta_i,-\frac{c_s}{m})$

Figure 1.5: Search Sub-Branch of Disclosure Game for Worker i

The final set of payoffs the hacker could receive are given by the hacker deciding to exit the market when a vulnerability is disclosed.

$$\Pi^{M}_{(X,S)}(\theta,\{\Gamma_{nu},\Gamma_{u},\Gamma_{v}\})=(1-\alpha)\left[\delta D\sum_{i\in\Gamma_{nu}\cup\Gamma_{u}}\theta_i-c_s\right] \tag{1.32}$$

$$\Pi^{M}_{(X,X)}(\theta,\{\Gamma_{nu},\Gamma_{u},\Gamma_{v}\})=0 \tag{1.33}$$

Equilibria Now that the game is formalized, the hacker and the workers can solve for the Nash equilibrium of the game given Microsoft's new policy. There are four main drivers of the Nash equilibria in this model,

(a) Do there exist any workers using the old version of the software? If so, do there exist any workers that choose not to update when an update is released?

(b) Under the "Non-Disclosure" branch of the game, does the cost of finding a Zero-Day exceed the expected profits of searching? I.e.

$$c_s\lesseqgtr\delta D\sum_{i\in I}\theta_i. \tag{1.34}$$

(c) Under the "Disclosure" branch of the game, does the cost of finding a Zero-Day exceed the expected profits of searching? I.e.

$$c_s\lesseqgtr\widehat{\delta}D\sum_{i\in I}\theta_i. \tag{1.35}$$

(d) Does the cost of updating exceed the cost of switching to the new version of the software package? I.e.

$$c_v\lesseqgtr c_u+\phi_u \tag{1.36}$$

Notice that the first three impact the hacker's decision, while the last point is going to impact the high-type workers' best response functions. The derivation of the best response functions for this policy are in Appendix A.3. While in this section I will describe the Nash equilibria under the different cost scenarios[57].

For every case listed below, there also exist three cases on the Disclosure branch of the game as the answer to: Does the cost of searching for an N-Day exceed the payoff?

$$\phi_u \lesseqgtr D \sum_{i \in I} \theta_i \tag{1.37}$$

In order to simplify the notation, I will first solve for the equilibria in the Non-Disclosure game followed by the equilibria in the Disclosure game.

1.5.1.1 Non-Disclosure

When search costs exceed the expected payoff of search under Non-Disclosure[58], the hacker will always play (X). Given the hacker strategy of exiting the game, all workers will not update. Therefore, the equilibrium of the Non-Disclosure branch is $((X),(nu)_{i \in I})$.

Next, if $c_s < \delta D \sum_{i \in I} \theta_i$, then search costs are less than the expected payoff of search under Non-Disclosure. Via the best responses of both workers and the hacker in Section A.3, the Nash equilibria under medium search costs are as follows in Theorem 1.5. Define $\Omega_M \equiv \left\{ k \in I | \theta_k \geq \frac{c_v}{\delta(v+D)} \right\}$.

Theorem 1.5. *Let $k_{min} \in \Omega_M$ be the minimal worker in Ω_M. Then Under Non-Disclosure and low search costs, if*

$$c_s < \delta D \sum_{i \in I \setminus \Omega_M} \theta_i \tag{1.38}$$

[57] The Knife-Edge Case and the Low Search Cost case are in Appendix A.1 and A.4, respectively.

[58] I.e. $c_s > \delta D \sum_{i \in I} \theta_i$

Then the Nash equilibrium is

$$\left(A_M^{(1-\alpha)*},\left(A_{M,i}^{(1-\alpha)*}\right)_{i\in I}\right)=\left((S),((nu)_{i\in\Gamma_{nu}^{k_{min},nd*}},(v)_{j\in\Gamma_{v}^{k_{min},nd*}})\right) \tag{1.39}$$

Where $\Gamma_{nu}^{k_{min},nd*}=\{i\in I|\theta_i<\theta_{k_{min}}\}$, *and* $\Gamma_{v}^{k_{min},nd*}=\{j\in I|\theta_j\geq\theta_{k_{min}}\}$.

Otherwise, there exists a pivotal worker $k^*\in\Omega_M$ *and a mixed strategy for worker* k^* *strategy,* $p_{k^*}^{v*}\in[0,1]$, *such that*

$$c_s=\delta\left(D\sum_{i\in\Gamma_{nu}^{k^*,nd*}}\theta_i+(1-p_k^{v*})D\theta_{k^*}\right) \tag{1.40}$$

Then the Nash equilibrium is

$$\left(A_M^{(1-\alpha)*},\left(A_{M,i}^{(1-\alpha)*}\right)_{i\in I}\right)=\left((\rho^*(S),(1-\rho^*)(X)),((nu)_{i\in\Gamma_{nu}^{k^*,nd*}},(p_{k^*}^{v*}(v),(1-p_{k^*}^{v*})(nu)),(v)_{j\in\Gamma_{v}^{k^*,nd*}})\right) \tag{1.41}$$

Where $\rho^*=\frac{c_v}{\theta_{k^*}\delta(v+D)}$, $\Gamma_{nu}^{k^*,nd*}=\{i\in I|\theta_i<\theta_{k^*}\}$, *and* $\Gamma_{v}^{k^*,nd*}=\{j\in I|\theta_j>\theta_{k^*}\}$.

Proof. Notice that, if Inequality 1.38 holds, then the best response of the hacker is to search for a Zero-Day. Given the hacker strategy of always searching, then the best response of high-type workers, workers $j\in\Omega_M$, is to install the new version of the software. The best response for all other workers is to do nothing, i.e. $i\in\Gamma_{nu}^*$. Thus 1.39 is the Nash Equilibrium.

Given the hacker strategy of $(\rho^*(S),(1-\rho^*)(X))$ where $\rho^*=\frac{c_v}{\theta_{k^*}\delta(v+D)}$, then all workers $i\in I$ such that $\theta_i<\frac{c_v}{\rho^*(v+D)}$ have the best response of $i\in\Gamma_{nu}^{k^*,nd*}$. Then if $\theta_j>\frac{c_v}{\rho^*(v+D)}$, then worker j will install the new version of the code, i.e. $j\in\Gamma_{v}^{k^*,nd*}$. Additionally, for worker k^* such that $\theta_{k^*}=\frac{c_v}{\rho^*(v+D)}$ is indifferent between installing the new version and not updating the old version with any probability $p_k^v\in[0,1]$. Since $p_{k^*}^{v*}\in[0,1]$, then $(p_{k^*}^{v*}(v),(1-p_{k^*}^{v*})(nu))$ is in worker k^*'s best response.

Given the worker strategy $((nu)_{i\in\Gamma_{nu}^{k^*,nd*}},(p_{k^*}^{v*}(v),(1-p_{k^*}^{v*})(nu)),(v)_{j\in\Gamma_{v}^{k^*,nd*}})$ such that there exists $k^*\in\Omega_M$ and $p_{k^*}^{v*}\in[0,1]$ to satisfy Equation 1.40, then the hacker is indifferent between (S) and (X), and is willing to play any mixed strategy $\rho\in[0,1]$. Since $\rho^*=\frac{c_v}{\theta_{k^*}(v+D)}\in[0,1]$, then ρ^* is the best response of the hacker. Therefore, Equation 1.41 is the Nash equilibrium. □

1.5.1.2 Disclosure

Now to solve for the Nash equilibria under the Disclosure branch of the game in Figure 1.4. On the Disclosure branch, both the hacker and the workers have three actions they could each take. In Section 1.3.3, the equilibria cases followed from the relation between the cost of searching and the expected payoffs from searching. However, due to the new action available to the workers, (v), and the enrollment fee, ϕ_u, there now exist extra cases dependent on Equations 1.36 and 1.37.

Beginning with high exploitation costs and either medium, knife-edge, or high search costs, then the Nash Equilibrium is as follows.

Theorem 1.6. *If there are both high or medium search costs and high exploitation costs, i.e.* $c_s>\widehat{\delta}D\sum_{i\in I}\theta_i$ *and* $\phi_u>D\sum_{i\in I}\theta_i$*, then the Nash equilibrium of the game is*

$$(A_M^{\alpha*},(A_{M,i}^{\alpha*})_{i\in I})=((X),(nu)_{i\in I}) \tag{1.42}$$

Proof. Notice that both searching for Zero- and N-Days are too costly, therefore, the hacker will always exit the game. Given this strategy, the workers will all not update. Hence, this is the Nash equilibrium. □

The other case to examine is when the exploitation costs of the N-Day are low. In other words, the last case is to examine when the updating fee charged by Microsoft is smaller than the profits gained by the hacker when no worker updates the old version of the software or installs the new version of the software.

Theorem 1.7. *If $c_s > \widehat{\delta} D \sum_{i\in I}\theta_i$ and $\phi_u \leq D\sum_{i\in I}\theta_i$, while the workers face $c_v < c_u + \phi_u$, and*

$$\phi_u < D \sum_{i\in I\setminus\Omega_M} \theta_i \tag{1.43}$$

Then the Nash equilibrium is

$$(A_M^{\alpha *}, (A_{M,i}^{\alpha *})_{i\in I}) = \Big((E), \big((nu)_{i\in\Gamma_{nu}^{d*}}, (v)_{j\in\Gamma_v^{d*}} \big) \Big) \tag{1.44}$$

Where $\Gamma_{nu}^{d*} = \{i \in I\setminus\Omega_M\}$ and $\Gamma_v^{d*} = \{j\in\Omega_M\}$.

Otherwise if $c_s > \widehat{\delta} D \sum_{i\in I}\theta_i$ and $\phi_u \leq D\sum_{i\in I}\theta_i$, while the workers face $c_v < c_u + \phi_u$, and there exists $k^* \in \Omega_M$ and a mixed strategy for worker k^*, $p_{k^*}^{v*} \in [0,1]$, such that

$$\phi_u = D \sum_{i\in\Gamma_{nu}^*} \theta_i + (1 - p_{k^*}^{v*}) D \theta_{k^*} \tag{1.45}$$

Then the Nash equilibrium of the game is

$$(A_M^{\alpha *}, (A_{M,i}^{\alpha *})_{i\in I}) = \Big((\rho^*(E), (1-\rho^*)(X)), \big((nu)_{i\in\Gamma_{nu}^{d*}}, (p_{k^*}^{v*}(v), (1-p_{k^*}^{v*})(nu)), (v)_{j\in\Gamma_v^{d*}} \big) \Big) \tag{1.46}$$

Where $\Gamma_{nu}^{d*} = \{i\in I | \theta_i < \theta_{k^*}\}$, $\Gamma_v^{d*} = \{j \in I | \theta_j > \theta_{k^*}\}$, and $\rho^* = \frac{c_v}{\theta_{k^*}(v+D)}$.

Proof. If Inequality 1.43 holds, then the best response of the hacker is to exploit the N-Day. Given that the hacker is playing (E) and the cost of installing the new version is cheaper than installing the updates on the old version, workers $j\in\Omega_M$ will play (v), and workers $i \in I\setminus\Omega_M$ will play (nu). Therefore, this is the Nash equilibrium.

Given $(\rho^*(E), (1-\rho^*)(X))$ where $\rho^* = \frac{c_v}{\theta_{k^*}(v+D)}$, then for any worker i such that $\theta_i < \theta_{k^*}$, then $i \in \Gamma_{nu}^{d*}$. For worker j such that $\theta_j > \theta_{k^*}$, then $i\in\Gamma_v^{d*}$. Lastly, worker k^* is indifferent between (v) and (nu), thus, since $p_{k^*}^{v*} \in [0,1]$, $(p_{k^*}^{v*}(v), (1-p_{k^*}^{v*})(nu))$ is worker k^*'s best response.

Next, if there exists $k^*\in\Omega_M$ and $p^{v*}_{k^*}\in[0,1]$ such that

$$\phi_u = D\sum_{i\in\Gamma^*_{nu}}\theta_i + (1-p^{v*}_{k^*})D\theta_{k^*} \tag{1.47}$$

Then the hacker is indifferent between any mixed strategy of the form $(\rho(E),(1-\rho)(X))$ for $\rho\in[0,1]$. Since $\rho^*=\frac{c_v}{\theta_{k^*}(v+D)}\in[0,1]$, then this is in the hacker's best response. Thus, it is a Nash equilibrium. □

Corollary 1.1. *If $c_s>\widehat{\delta}D\sum_{i\in I}\theta_i$ and $\phi_u\leq D\sum_{i\in I}\theta_i$, while the workers face $c_v>c_u+\phi_u$, and Inequality 1.43 holds, then the Nash equilibrium is*

$$(A^{\alpha*}_M,(A^{\alpha*}_{M,i})_{i\in I}) = \Big((E),\big((nu)_{i\in\Gamma^{d*}_{nu}},(u)_{j\in\Gamma^{d*}_u}\big)\Big) \tag{1.48}$$

Where $\Gamma^{d*}_{nu}=\{i\in I\backslash\Omega_M\}$ and $\Gamma^{d*}_u=\{j\in\Omega_M\}$.

Otherwise if $c_s>\widehat{\delta}D\sum_{i\in I}\theta_i$ and $\phi_u\leq D\sum_{i\in I}\theta_i$, while the workers face $c_v>c_u+\phi_u$, and there exists $k^*\in\Omega_M$ and a mixed strategy for worker k, $p^{v*}_{k^*}\in[0,1]$, such that

$$\phi_u = D\sum_{i\in\Gamma^*_{nu}}\theta_i + (1-p^{u*}_{k^*})D\theta_{k^*} \tag{1.49}$$

Then the Nash equilibrium of the game is

$$(A^{\alpha*}_M,(A^{\alpha*}_{M,i})_{i\in I}) = \Big((\rho^*(E),(1-\rho^*)(X)),\big((nu)_{i\in\Gamma^{d*}_{nu}},(p^{u*}_{k^*}(u),(1-p^{u*}_{k^*})(nu)),(v)_{j\in\Gamma^{d*}_u}\big)\Big) \tag{1.50}$$

Where $\Gamma^{d*}_{nu}=\{i\in I|\theta_i<\theta_{k^*}\}$, $\Gamma^{d*}_u=\{j\in I|\theta_j>\theta_{k^*}\}$, and $\rho^*=\frac{c_u+\phi_u}{\theta_{k^*}(v+D)}$.

Again, since updating protects the worker the same amount as installing the new version, but since updating is cheaper, then the high-type workers will update, and thus follows similarly to Theorem 1.7.

Welfare Analysis

Now to investigate whether this new "Extended Coverage" will be a welfare improving policy. This section flows as follows: First, define the optimal policy; Then, under each of the different cost scenarios[59], the welfare improving policy will be solved for.

Definition 1.2. *The optimal policy* $\Psi^*\in\{Microsoft,\ Disclosure, Non-Disclosure\}$ *is chosen such that:*

$$\Psi^*=argmax_{\psi\in\{M,d,nd\}}\left\{\sum_{i\in I}U_M(A_M^{\alpha*},A_M^{(1-\alpha)*},A_{M,i}^{\alpha*},A_{M,i}^{(1-\alpha)*},\theta_i),\sum_{i\in I}U_d(A_d^{\alpha*},A_d^{(1-\alpha)*},A_i^*,\theta_i),\sum_{i\in I}U_{nd}(A_{nd}^*,\theta_i)\right\} \tag{1.51}$$

Where $((A_d^{\alpha*},A_d^{(1-\alpha)*}),(A_i^*)_{i\in I})$*,* (A_{nd}^*)*, and* $((A_M^{\alpha*},A_M^{(1-\alpha)*}),(A_{M,i}^{\alpha*},A_{M,i}^{(1-\alpha)*})_{i\in I})$ *are the Nash equilibria of the Disclosure, Non-Disclosure, and Microsoft policies, respectively.*

As in Section 1.4, the optimal policy is determined by the utility maximizing policy. The optimal policy is dependent on the types of search costs that are faced by the hackers, and thus the following sections outline the optimal policies under both high and medium search costs[60].

1.5.1.3 High Search Cost

Recall that the equilibria of the Microsoft policy game are split into two cases: when ϕ_u is high and when ϕ_u is low. Notice that when $\phi_u>D\sum_{i\in I}\theta_i$, then the Nash equilibrium of the Microsoft game is for the hacker to always exit and the workers to never update or install the new version of the software. These two cases can be identified by

$$\phi_u\lesseqgtr D\sum_{i\in I}\theta_i \tag{1.52}$$

Theorem 1.8. *Let* $c_s>\delta D\sum_{i\in I}$*. Then the two cases satisfying Inequality 1.52 are*

1. *If* $\phi_u>D\sum_{i\in I}\theta_i$*, then both Microsoft and Non-Disclosure are optimal policies.*

[59] I will focus on the High and Medium search cost cases in this section, the Low cost case is in Appendix A.5.

[60] Low search cost welfare is discussed in Appendix A.5

2. If $\phi_u \leq D\sum_{i\in I}\theta_i$, then Non-Disclosure is the optimal policy.

Proof. Notice that

$$\sum_{i\in I} U_M(A_M^{\alpha *}, A_M^{(1-\alpha)*}, A_{M,i}^{\alpha *}, A_{M,i}^{(1-\alpha)*}, \theta_i) = v\sum_{i\in I}\theta_i \tag{1.53}$$

Then by Theorem 1.2

$$\sum_{i\in I} U_M(A_M^{\alpha *}, A_M^{(1-\alpha)*}, A_{M,i}^{\alpha *}, A_{M,i}^{(1-\alpha)*}, \theta_i) = \sum_{i\in I} U_{nd}(A_{nd}^{*}, \theta_i) > \sum_{i\in I} U_d(A_d^{\alpha *}, A_d^{(1-\alpha)*}, A_i^{*}, \theta_i) \tag{1.54}$$

Therefore, $\Psi^* = \{Microsoft,\ Non-Disclosure\}$.

The other case trivially follows. □

Therefore, for the new policy to be effective under high search costs, the extended service fee must be large. Also notice that if $\phi_u \leq D\sum_{i\in I}\theta_i$, i.e. the exploitation fee is low, then the Nash equilibrium of the hacker exit when a vulnerability is not found and to mix between exploitation of the N-Day and exiting the game. Then, Microsoft is preferred to Disclosure when

$$\rho_M^* \left[\sum_{i\in \Gamma_{nu}^{M*}} \theta_i + (1-p_{k^*}^{M*})\theta_{k^*} \right] + \xi^{M*} c_v < (v+D) \sum_{i\in \Gamma_{nu}^{d*}} \theta_i \tag{1.55}$$

1.5.1.4 Medium Search Cost

Under medium search costs, the optimal policy decision must be split into cases that are firstly dependent on the exploitation fee to solve for the Nash equilibria of the Microsoft game, as in the high search cost case. Differing from high search costs, now the optimal policy is also relies on the cost of installing updates relative to the cost of the new version of the software[61]. If there are high exploitation costs, $\phi_u > D\sum_{i\in I}\theta_i$,

[61] Recall that under the Non-Disclosure policy, the hacker will search for a Zero-Day. When an update is released in the Disclosure policy, the hacker will play (E).

then via Theorem 1.6, the hacker will always leave the game, i.e. play (X). On the other hand, if exploitation costs are low, i.e. Theorem 1.7, then the hacker will either exploit the N-Day or mix between (E) and (X).

Under high exploitation costs all workers will not update nor switch versions. However, under low exploitation costs, the low-type workers choose to not update their machines nor switch to the new version of the software, while high-type workers will either install the new software version or update their machines dependent on the relative cost of updating to installation of the new version.

Notice that, under high exploitation costs, the welfare equation for all workers is

$$\sum_{i\in I} U_M(A_M^{\alpha *}, A_M^{(1-\alpha)*}, A_{M,i}^{\alpha *}, A_{M,i}^{(1-\alpha)*}, \theta_i) = v\sum_{i\in I}\theta_i \tag{1.56}$$

Therefore, compared to Disclosure, the workers do not need to either update or be hacked via the released patch, and compared to Non-Disclosure, the hacker is not going to be searching for a Zero-Day, and thus the workers will not bear the burden of the expected damages. Hence, as discussed in Theorem 1.9, the new policy proposed by Microsoft is optimal.

The next case to discuss is when the exploitation cost is low, $\phi_u \leq D\sum_{i\in I}\theta_i$, and the cost of installing the new version is less than the cost of updating, $c_v \leq c_u + \phi_u$. Therefore, the workers' welfare equation is

$$\begin{aligned}\sum_{i\in I}U_M(A_M^{\alpha *},A_M^{(1-\alpha)*},A_{M,i}^{\alpha *},A_{M,i}^{(1-\alpha)*},\theta_i)=\alpha&\left(v\sum_{j\in\Gamma_v^*}\theta_j+p_{k^*}^{v*}(v\theta_{k^*}-c_v)-(1-p_{k*}^{v*})D\theta_k-D\sum_{i\in\Gamma_{nu}^*}\theta_i-\xi_v^*c_v\right)\\&+(1-\alpha)\left[v\sum_{j\in\Gamma_v^*}\theta_j-\xi_v^*c_v+p_{k*}^{v*}(v\theta_{k*}-c_v)\right.\\&+((1-\delta)(v-c_v)-\delta D)(1-p_{k*}^{v*})\theta_{k*}\\&\left.-\delta D\sum_{i\in\Gamma_{nu}^*}\theta_i+(1-\delta)v\sum_{i\in\Gamma_{nu}^*}\theta_i\right]\end{aligned}\tag{1.57}$$

Comparing the new Microsoft policy to Disclosure and Non-Disclosure, the following inequality describes when the new Microsoft policy is optimal.

$$\alpha\rho_M^*(1-\delta)\left[\sum_{i\in\Gamma_{nu}^{M*}}\theta_i+(1-p_{k^*}^{v*})\theta_{k^*}\right]+\xi_v^*\frac{c_v}{v+D}\leq\min\left\{\delta\sum_{i\in I}\theta_i,\ \alpha\sum_{i\in\Gamma_{nu}^{d*}}\theta_i+(1-\alpha)\delta\sum_{i\in I}\theta_i+\xi_v^*\frac{c_u}{v+D}\right\}\tag{1.58}$$

The left-hand side is the cost paid by the workers under the “Extended Support” policy for Windows 7, where as the right-hand side describes the costs associated with Non-Disclosure and Disclosure, respectively. Now to break down the left-hand side. The first term are the costs paid by low-type workers and the pivotal worker when she does not install the new version of the code when the hacker is able to exploit the N-Day. The final term on the left hand side is the cost paid by the high-type workers when they install the new version of the software.

Finally, if exploitation costs are low but the cost of installing the new version is higher than that of updating the old version, i.e. $c_v > c_u + \phi_u$, then, under the Disclosure branch of the Microsoft game, the high-type workers will update. Whereas, in the Non-Disclosure branch of the Microsoft game, the high-type workers will install the new version of the software to protect their computers[62]. This yields the following condition for when "Extended Support" of Windows 7 is the optimal policy.

$$\begin{aligned}\alpha\rho_M^{d*}\left[\sum_{i\in\Gamma^{M*}_{nu,nd}}\theta_i+(1-p_{k^*}^{u*})\theta_{k^*}\right]+(1-\alpha)\rho_M^{nd*}\left[\sum_{i\in\Gamma^{M*}_{nu,nd}}\theta_i+(1-p_{k^*}^{v*})\theta_{k^*}\right]+\xi^*\frac{\alpha(c_u+\phi_u)+(1-\alpha)c_v}{v+D}\\ \le\min\left\{\delta\sum_{i\in I}\theta_i,\ \alpha\sum_{i\in\Gamma^{d*}_{nu}}\theta_i+(1-\alpha)\delta\sum_{i\in I}\theta_i+\xi^*\frac{c_u}{v+D}\right\}\end{aligned} \tag{1.59}$$

Again, the left-hand side is the cost paid by workers under the Microsoft policy, while the right-hand side is the minimum of the costs paid by the workers under Non-Disclosure and Disclosure, respectively. The first term on the left-hand side is the damage done to the low-type workers since they do not update and the pivotal worker k when she does not update on the Disclosure branch of the Microsoft game due to the hacker exploiting the N-Day. The second term is the damage done on the Non-Disclosure branch of the Microsoft game to the low-type workers and the pivotal worker k when she does not install the new version of the code given the hacker is searching for a Zero-Day. The final term is the cost of either updating when a vulnerability is found or installing the new version when the vendor does not find a vulnerability that the high-type workers pay.

Given these conditions, Microsoft's new policy is an element of the optimal policy set, Ψ^*, under the conditions given in the following theorem.

Theorem 1.9. *Let $\widehat{\delta}D\sum_{i\in I}\theta_i\le c_s<\delta D\sum_{i\in I}\theta_i$. Then the cases satisfying Inequality 1.52 are*

1. *If there are high exploitation costs, i.e. $\phi_u>D\sum_{i\in I}\theta_i$, then Microsoft is the optimal policy.*

[62] Therefore, $\rho_M^{d*}\neq\rho_M^{nd*}$

2. *If there are low exploitation costs,* $\phi_u \leq D\sum_{i\in I}\theta_i$*, low version costs,* $c_v \leq c_u + \phi_u$*, and Inequality 1.58 is satisfied, then Microsoft is an optimal policy.*

3. *If there are low exploitation costs, low version costs, and Inequality 1.58 is not satisfied, then Microsoft is not an optimal policy.*

4. *If there are low exploitation costs, high version costs, i.e.* $c_v > c_u + \phi_u$*, and Inequality 1.59 is satisfied, then Microsoft is an optimal policy.*

5. *If there are low exploitation costs, high version costs, and Inequality 1.59 is not satisfied, then Microsoft is not an optimal policy*

Proof. For Case 1, Notice that by Equation 1.56 and Theorem 1.3, $\Psi^* = \{Microsoft\}$.

For Case 2, Notice that by Equation 1.57 and given Equation 1.58, $Microsoft \in \Psi^*$. Notice that $Microsoft$ is the only element if Equation 1.58 is strict.

The other cases trivially follow. □

Notice that ϕ_u can be used as a weapon to harm hackers. In order for Microsoft's new policy to be effective under medium search costs, the optimal extended service fee and cost of installing the new version are interdependent. The first way for Microsoft to maximize worker welfare is to pick a very large support fee, i.e. high exploitation costs. This prices the hacker out of the market, while also allowing for the workers to not have to pay to install updates or update their software version since the hacker is priced out of the exploitation market. However, under low exploitation costs, for the Microsoft policy to maximize worker welfare they must choose c_v such that either Inequality 1.58 or Inequality 1.59 hold.

1.6 Conclusion

The optimal policy debate should be centered around how these policies influence both the hacker's and workers' behavior[63]. The ease with which the hacker is able to

[63] As Sun Tzu said: "Know thy self, know thy enemy. A thousand battles, a thousand victories."

infiltrate the network can be decreased via appropriate disclosure policies. Since the cost of searching for Zero-Days has drastically increased over the last couple of years, the hacker desires more disclosure to decrease his costs. The policies of Non-Disclosure and Microsoft's new policy both decrease hacker interference in the network as well as increase overall worker welfare.

2 The User's Guide to Solving Games via the Modular Gröbner Basis Approach

2.1 Introduction

Many economic problems are highly non-linear which, in many cases, pose significant computational challenges. There have been many attempts to solve these problems, but these methods require stringent simplifications of the models and do not allow economists to solve for all equilibria of the complex models. Building off of Arnold (2003), we have developed a tool, the Modular Groebner Basis Approach (MGBA), which can solve for all equilibria in highly non-linear economic models via Groebner bases.

In rough terms, a Gröbner basis is a generalization of Gaussian elimination to polynomial systems. The idea is to "triangularize" a system of polynomial equations symbolically, not numerically, in order to find all solutions to the initial system. However, Gröbner basis computation can have some difficulties, that are discussed in detail in Section 2.3.3. This paper describes the Modular Gröbner Basis Approach (MGBA) that is able to overcome these computational obstacles to solve for the Gröbner basis of the original system.

The following is a "triangularized" polynomial system.

$$0{=}x{+}z^{17}{+}z^{4}{+}z^{2}{+}5$$

$$0{=}y{-}4z^{23}{+}8z^{2}{-}12$$

$$0{=}z^{7}{-}8z^{4}{+}z^{2}{-}z{+}1$$

The final equation is entirely in z and thus, in this case, we can solve for all seven roots of z. The other important fact to notice about polynomial "triangularization" is that all equations, other than the final one, are linear in the remaining variables, x and y; therefore, plugging in the roots of z, the solutions for x and y can be obtained. Given an initial system of polynomial equations, Buchberger's algorithm[64] can be applied to solve for the Gröbner basis of the initial system, i.e. the "triangularized" system that has the same roots as the initial system. The beauty of a Gröbner basis is that the "triangularized" system has the same roots as the initial polynomial system.

A perceived issue is that many economics problems may not be polynomial, but these problems can be "polynomialized", i.e. converted into a set of polynomial equations. For example, suppose that the following equation containing rational powers is included in the system of otherwise polynomial equations,

$$K^{\frac{1}{3}}L^{\frac{2}{3}}-Y=0$$

In order to "polynomialize" this equation, we must first apply the following change of variables: $\mathbb{K}^3=K$ and $\mathbb{L}^3=L^2$. Then, by removing the initial equation and adjoining the equations in 2.1, we have a system of polynomial equations.

$$0=\mathbb{K}\mathbb{L}-Y \tag{2.1}$$

$$0=\mathbb{L}^3-L^2 \tag{2.2}$$

$$0=\mathbb{K}^3-K \tag{2.3}$$

Therefore, a Gröbner basis can be found for any non-linear economic problem that can be solved via a system of "polynomializable" equations, e.g. rational functions, functions with rational powers, etc.

The paper is layed-out as follows: The literature review can be found in Section 2.2, followed by a formal introduction to the definitions from algebraic geometry used within this paper in Section 2.3. Then, the core section of the paper, a description

[64] For more information on Buchberger's algorithm, see Cox et al. (2007) and Section 2.3.

of the algorithm of the MGBA is in Section 2.4. The next two sections, Sections 2.5 and 2.6, build up examples of how to apply the MGBA. We then conclude in Section 2.7.

2.2 Literature Review

The literature describing modular Gröbner basis methods has been around since Ebert (1983), and has focused on describing lucky primes[65]. Our paper builds off of the work found in Arnold (2003), by applying high-power computing to resolve the problem of finding lucky primes.

A growing literature has recently emerged applying techniques from algebraic geometry to economic problems. Including how to solve for all pure-strategy equilibria, Judd et al. (2012) and Kubler et al. (2014), applying polynomial programming to solve for generalized Nash equilibria, Couzoudis and Renner (2013), solving dynamic quantity precommitment games, Renner (2015), and apply polynomial optimization techniques to principal-agent problems, Renner and Schmedders (2015).

2.3 Preliminaries

In order to define a Gröbner Basis, we will begin by defining a set of notation followed by an explanation of the solution method for finding a Gröbner basis, Buchberger's Algorithm. The last step is to describe intermediate coefficient swell, which is the main computational problem facing Buchberger's algorithm.

[65] For a description of lucky primes, see Appendix C.

2.3.1 Definitions/Notation

Let R be a ring[66] and K be a field[67]. Then, a polynomial ring is defined as

Definition 2.1. *A polynomial ring, $K[\boldsymbol{x}]$, in $\boldsymbol{x}$ over a field K is defined as the set of polynomials in $\boldsymbol{x}$ of the form:*

$$p_0+p_1\boldsymbol{x}+\cdots+p_m\boldsymbol{x}^m$$

where $p_0,\ldots,p_m\in K$.

Note that $\mathbf{x}$ can be a vector of variables, i.e. $\mathbf{x}\equiv(x_0,\ldots,x_n)$, where $\mathbf{x}^\alpha=\prod_{i=1}^n x_1^{\alpha_1}\cdots x_n^{\alpha_n}$ for any multi-index $\alpha=(\alpha_1,\ldots,\alpha_n)$. Next we define a polynomial ideal.

Definition 2.2. *An ideal, I, is a subset of elements of ring R that forms an additive group*[68] *s.t. $\forall x\in R$ and $y\in I$, then $xy\in I$ and $yx\in I$.*

For simplicity[69], let $\mathbb{C}[\mathbf{x}]$ denote the multivariate polynomial ring over the field of complex numbers.

Definition 2.3. *A subset $I\subset\mathbb{C}[\boldsymbol{x}]$ is an ideal if it satisfies:*

1. $0\in I$

2. *If $f,g\in I$, then $f+g\in I$*

3. *If $f\in I$ and $g\in\mathbb{C}[\boldsymbol{x}]$, then $f\cdot g\in I$*

[66] A ring, R, is a set S together with two binary operators $+$ and $*$ satisfying the following conditions: 1) Additive associativity, 2) Additive commutability, 3) Additive identity, 4) Additive inverse, 5) Multiplicative distributivity, 6) Multiplicative associativity.

[67] A field, K, is a ring, R, that also satisfies the following conditions: 1) Multiplicative commutativity, 2) Multiplicative identity, 3) Multiplicative inverse.

[68] A group is a non-empty set Θ on which there is defined a bianary operation $\cdot$ satisfying the following properties: 1) *Closure:* If a and b belong to Θ, then $a\cdot b$ is also in Θ, 2) *Associativity:* $a\cdot(b\cdot c)=(a\cdot b)\cdot c$ for all $a,b,c\in\Theta$, 3) *Identity:* There is an element $1_\Theta\in\Theta$ such that $a\cdot 1_\Theta=1_\Theta\cdot a=a$ for all $a\in\Theta$, 4) *Inverse:* If $a\in\Theta$, then there exists an element $a^{-1}\in\Theta$ such that $a\cdot a^{-1}=a^{-1}\cdot a=1$.

[69] Economic problems tend to live within the set of complex numbers.

The type of ideal we are concerned with are those generated by a finite set of polynomials $\{f_1,\ldots,f_s\}$ which is defined as:

$$\langle f_1,\ldots,f_s\rangle\equiv\{f|f=g_1f_1+\cdots+g_sf_s,\ g_i\in\mathbb{C}[x_1,\ldots,x_n]\}$$

Hence, we will denote the polynomial ideal I as $I=\langle f_1,\ldots,f_s\rangle$, where $f_1,\ldots,f_s$, such that f_i, for each i, is drawn from the polynomial ring $\mathbb{C}[x_1,\ldots,x_n]$.

The next step is to define the ordering of monomials followed by the formal definition of a Gröbner basis. Monomial ordering answers the question: Which monomial should come first x^3y^2z or x^3yz^2?

Definition 2.4. *Let $\mathbb{C}$ be the field of complex numbers. A monomial ordering on $\mathbb{C}[x_1,\ldots,x_n]$ is any relation $\succeq$ on $\mathbb{Z}^n_+$, i.e. all weakly positive integers, s.t.:*

1. *$\succeq$ is a total ordering on $\mathbb{Z}^n_+$*

2. *$\succeq$ is a well-ordering on $\mathbb{Z}^n_+$*

3. *If $\alpha\succeq\beta$ and $\gamma\in\mathbb{Z}^n_+$, then $\alpha+\gamma\succeq\beta+\gamma$*

Then we say $x^\alpha\succeq x^\beta$ if and only if $\alpha\succeq\beta$.

One example of monomial ordering is lexicographical ordering. Let $\alpha=(\alpha_1,\ldots,\alpha_n)$ and $\beta=(\beta_1,\ldots,\beta_n)$, such that $\alpha,\beta\in\mathbb{Z}^n_+$. An ordering is lexicographical, $\alpha\succeq_{lex}\beta$ if and only if, in the difference $\alpha-\beta\in\mathbb{Z}^n$, the left-most non-zero entry is positive. So we say that $x^\alpha\succeq_{lex}x^\beta$ if $\alpha\succeq_{lex}\beta$. For example, given $x_1\succeq_{lex}x_2\succeq_{lex}x_3$, then $x_1^2\succeq_{lex}x_1x_2\succeq_{lex}x_1x_3\succeq_{lex}x_2^2\succeq_{lex}x_2x_3\succeq_{lex}x_3^2$.

Another example of a monomial ordering, and the main ordering used in Section 2.4, is the Degree Reverse Lexicographical ordering, or degrevlex. For degrevlex, $x^\alpha\succeq_{degrevlex}x^\beta$ if and only if either: (1) $\alpha_1+\cdots+\alpha_n>\beta_1+\cdots+\beta_n$ or (2) $\alpha_1+\cdots+\alpha_n=\beta_1+\cdots+\beta_n$ and if there exists an $i\in\{1,\ldots,n\}$ such that $\alpha_j=\beta_j$ for all $j>i$, and

$\alpha_i > \beta_i$. For example, given $x_1 \succeq_{degrevlex} x_2 \succeq_{degrevlex} x_3$, then $x_1^2 \succeq_{degrevlex} x_1 x_2 \succeq_{degrevlex} x_2^2 \succeq_{degrevlex} x_1 x_3 \succeq_{degrevlex} x_2 x_3 \succeq_{degrevlex} x_3^2$.

2.3.2 Gröbner Basis Introduction

Let $\mathbb{C}[x_1,\ldots,x_n]$ be the polynomial ring over the complex numbers, $\succeq$ be a monomial ordering, and $I = \langle f_1,\ldots,f_s \rangle$ be a polynomial ideal for $f_i \in \mathbb{C}[x_1,\ldots,x_n]$ for all $i \in \{1,\ldots,s\}$. First, we provide a few key polynomial definitions.

Definition 2.5. *Let $f \equiv p_0 + p_1 \boldsymbol{x} + \cdots + p_m \boldsymbol{x}^m$ be any polynomial, then we have the following terms:*

1. *$LC(f)$ is the leading coefficient of f, i.e. p_m.*
2. *$LM(f)$ is the leading monomial[70] of f, i.e. $\boldsymbol{x}^m$.*
3. *$LT(f)$ denote the leading term of f, i.e. $p_m \boldsymbol{x}^m$.*

Notice that each term is determined by the monomial ordering on f.

Definition 2.6. *Let a monomial ordering on $\mathbb{C}[x_1,\ldots,x_n]$ be fixed. If $I = \langle f_1,\ldots,f_s \rangle$ is a polynomial ideal over $\mathbb{C}[x_1,\ldots,x_n]$, then a finite subset $G = \{g_1,\ldots,g_s\} \subset K[x_1,\ldots,x_n]$ is a Gröbner basis[71] if*

$$\langle LT(g_1),\ldots,LT(g_s) \rangle = \langle LT(f_1),\ldots,LT(f_s) \rangle .$$

The next step is to setup all of the terms required to define Buchberger's algorithm, i.e. an algorithm used to solve for the Gröbner basis of a given polynomial ideal I. In order to solve Buchberger's algorithm, we define an S-polynomial of any two polynomials f and g, denoted as $S(f,g)$. Let $f,g \in \mathbb{C}[x_1,\ldots,x_n]$ be non-zero polynomials, and let α be the powers of the $LT(f)$ and β be the powers of the $LT(g)$.

[70] This is also known as a leading power product.

[71] Notice that the Gröbner basis is a monomial ideal.

Then define $\gamma=(\gamma_1,\ldots,\gamma_n)$, where $\gamma_i\equiv\max(\alpha_i,\beta_i)$ for all $i\in\{1,\ldots,n\}$. We call x^γ the least common multiple of $LM(f)$ and $LM(g)$, written as $x^\gamma=LCM(LM(f),LM(g))$.

Definition 2.7. *The S-polynomial of f and g is the combination:*

$$S(f,g)=\frac{x^\gamma}{LT(f)}f-\frac{x^\gamma}{LT(g)}g$$

Recall that, for every pair of polynomials (f_0,f_1) s.t. $f_1\neq 0$, polynomial division provides a quotient Q and a remainder R s.t. $f_0=f_1Q+R$ and either $R=0$ or the degree of R is less than the degree of f_1, i.e. $deg(R)<deg(f_1)$. Moreover, (Q,R) is the unique pair of polynomials having this property, and the process of obtaining the uniquely defined polynomials Q and R from f_0 and f_1 is called *Euclidean division.*

In order to determine if a Gröbner basis G has been obtained, we must first define the term "the remainder on division of a polynomial by a list of polynomials". Let $\widehat{f}^G$ be the remainder on division of f by the list of polynomials $G=\{g_1,\ldots,g_s\}$, meaning that f is divided, in some order, by each element of G.

Definition 2.8. ***Buchberger's Criterion:*** *Let I be a polynomial ideal. Then a basis $G=\{g_1,\ldots,g_s\}$ is a Gröbner basis for I iff for all pairs $i\neq j$, the remainder on division of $S(g_i,g_j)$ by G is zero.*

Let $I=\langle f_1,\ldots,f_s\rangle\neq\langle 0\rangle$ be a polynomial ideal. Then I has a Gröbner basis and it can be constructed in a finite number of steps[72].

Buchberger's Algorithm[73]: Let $F=\{f_1,\ldots,f_s\}$ be a set of polynomials defining[74] $I\neq\{0\}$. For each pair of polynomials $f_i,f_j\in F$, calculate $S(f_i,f_j)$ and divide it by the polynomials in F obtaining $\widehat{S}^F$. If $\widehat{S}^F\neq 0$, add $\widehat{S}^F$ to F, and start again with $F^{'}=F\cup\{\widehat{S}^F\}$. Repeat until all S-polynomials of the polynomials in $F^{'}$ have remainder 0 after division by $F^{'}$.

[72] Computational Problem: If it does not reduce to a zero dimensional variety, then we must worry about witness sets (this is for later research using a computer algebra language called Bertini.).

[73] For more information on Buchberger's algorithm, see Cox et al. (2007).

[74] I.e. $I=\langle f_1,\ldots,f_s\rangle$.

2.3.3 Intermediate Coefficient Swell

The major obstacle that the algorithm in Section 2.4 overcomes in Gröbner basis computations is called intermediate coefficient swell. During the computation of Buchberger's algorithm, many intermediate polynomial are computed. In some of these intermediate polynomials, the coefficients that are computed explode in length. This is called intermediate coefficient swell.

For example, consider the following quadratic system of polynomials[75] over the variables x_0, x_1,..., x_8.

$$
\begin{aligned}
&f_0 \equiv x_0+2x_1+2x_2+2x_3+2x_4+2x_5+2x_6+2x_7+2x_8-1\\
&f_1 \equiv 2x_1x_8+2x_0x_7+2x_1x_6+2x_2x_5+2x_{34}-x_7\\
&f_2 \equiv 2x_2x_8+2x_1x_7+2x_0x_6+2x_1x_5+2x_2x_4+x_3^2-x_6\\
&f_3 \equiv 2x_3x_8+2x_2x_7+2x_1x_6+2x_0x_5+2x_1x_4+2x_2x_3-x_5\\
&f_4 \equiv 2x_4x_8+2x_3x_7+2x_2x_6+2x_1x_5+2x_0x_4+2x_1x_3+x_2^2-x_4\\
&f_5 \equiv 2x_5x_8+2x_4x_7+2x_3x_6+2x_2x_5+2x_1x_4+2x_0x_3+2x_1x_2-x_3\\
&f_6 \equiv 2x_6x_8+2x_5x_7+2x_4x_6+2x_3x_5+2x_2x_4+2x_1x_3+2x_0x_2+x_1^2-x_2\\
&f_7 \equiv 2x_7x_8+2x_6x_7+2x_5x_6+2x_4x_5+2x_3x_4+2x_2x_3+2x_1x_2+2x_0x_1-x_1\\
&f_8 \equiv x_0^2+2x_1^2+2x_2^2+2x_3^2+2x_4^2+2x_5^2+2x_6^2+2x_7^2+2x_8^2-x_0
\end{aligned}
$$

In order for this system of quadratic polynomials to be a polynomial ideal, it must satisfy the conditions in Definition 2.3. Let $I \equiv \langle f_0,f_1,f_2,f_3,f_4,f_5,f_6,f_7,f_8\rangle$.

1. Notice that if $x_0=x_1=\cdots=x_8=0$, then $f_8=0$, and thus $0\in I$.

2. Since $0,1\in\mathbb{C}[x_0,\ldots,x_8]$, then notice that for any $f,g\in I$, $1\cdot f+1\cdot g+0\cdot\sum_{h\in I\{f,g\}}h$ is an element in I.

[75] This is known as Katsura 8 from physics to describe the random Ising model, See Katsura (1986).

3. Again, by the definition of the set generated by I, notice that if $f \in I$ and $g \in \mathbb{C}[x_0,\ldots,x_8]$. Then, since $0 \in \mathbb{C}$, then $f \cdot g + 0 \cdot \sum_{h \in I \setminus \{f\}} \in I$.

Therefore, $I \equiv \langle f_0,f_1,f_2,f_3,f_4,f_5,f_6,f_7,f_8 \rangle$ is a polynomial ideal.

Notice that no polynomial in I has a coefficient greater than two, and there is no term that is of a power larger than quadratic. However, while computing one of the S-polynomials in Buchberger's algorithm, the following polynomial is produced:

$$(667943\ldots)x_3x_4^2x_5^2x_7^2x_8+\ldots$$

Where the coefficient contains roughly 55,000 digits, causing the computation to slowdown and eventually to fail. Notice that there are no large coefficients or powers within the original ideal I that would indicate that the coefficients within the intermediate steps of triangularization of the polynomials would explode in size.

Due to intermediate coefficient swell, directly computing the Gröbner basis, G, of the system F is not possible, so we present the MGBA, and give a few economic examples of how to use the algorithm.

2.4 The Theory of MGBA

This section presents the Modular Gröbner Basis Approach (or MGBA), which is a computationally efficient method of eliminating intermediate coefficient swell as a problem faced when trying to compute a Gröbner basis via techniques from number theory[76]. We build up the algorithm by first discussing the three steps of the algorithm, and then present how the MGBA overcomes the intermediate coefficient swell problem[77].

The MGBA can be broken into three distinct steps, as seen in Figure 2.1. Given

[76] Mostly from modular arithmetic.

[77] In Appendix C we define "Lucky Primes" and discuss how to feasibly overcome the computational issues surrounding them.

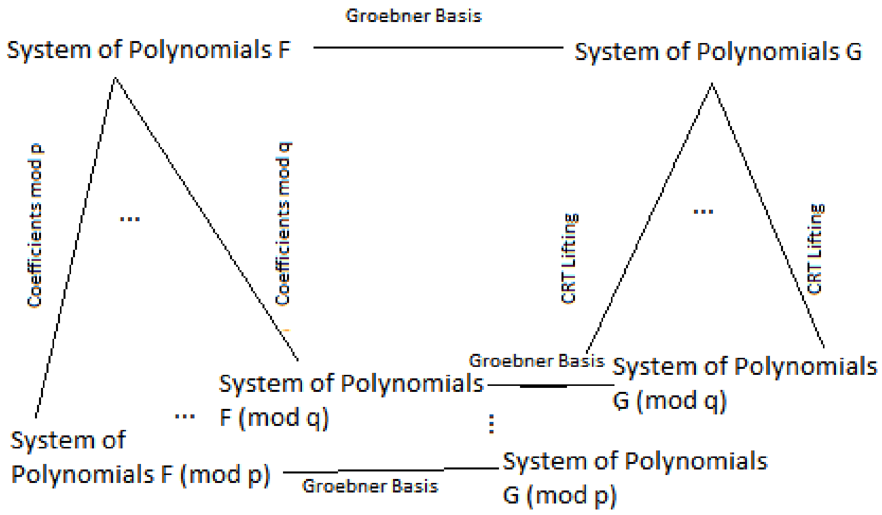


Figure 2.1: MGBA

an initial system of polynomial equations F, thus initial ideal $I{=}\langle f_1,\ldots,f_s\rangle$, and a set of primes $P{=}\{p_1,\ldots,p_m\}$, the first step[78] is to compute the coefficients modulo a prime p. The second step[79] is to then compute the modular Gröbner basis, denoted as $G(mod\ p)$, of the system $F(mod\ p)$. The final step[80] is to use the Chinese Remainder Theorem[81] (CRT) to lift a set of modular Gröbner bases, given a set of prime numbers, and check whether we have obtained the correct solution G.

In order to apply modular arithmetic, let $K[x_1,\ldots,x_n]{=}\mathbb{Z}[x_1,\ldots,x_n]$ be the polynomial ring over the integers[82] and given an initial polynomial system $F{\subset}K[x_1,\ldots,x_n]$, the monomial ordering used is degrevlex, and a set of primes P, then we can apply the MGBA to solve for the Gröbner basis G.

[78] The vertical lines on the left-hand side of Figure 2.1.

[79] The lower horizontal lines of Figure 2.1.

[80] The vertical lines on the right-hand side of Figure 2.1.

[81] Arnold (2003) also describes via p-Adic methods, see her paper for more details.

[82] Many economic problems can be converted into such a form, for example see Equations 2.20-2.22.

2.4.1 Polynomial System Mod p

The first goal is to transform the initial set of polynomials, F, into $F(mod\, p_i)$, denoted as F_{p_i}, i.e. the set such that the coefficients are elements of the field[83] $K_{p_i}{=}\mathbb{Z}_{p_i}$ instead of K, for all $p_i{\in}P$.

Let's examine the following example system $F{\equiv}\{f_1,\ f_2,\ f_3\}$.

$$f_1{\equiv}3245xy{+}476z^5{-}z$$

$$f_2{\equiv}436x{-}56z^2{-}z{+}89$$

$$f_3{\equiv}56z^9{+}z^5{-}78$$

Let $p{=}29$, then to compute F_{29}, we must compute every coefficient in f_i modulo 29 for all $i{\in}\{1{,}2{,}3\}$. For example, the first coefficient in f_2 is 436 which is congruent to 1 modulo 29. Therefore, the first coefficient in $f_{2,29}$ is 1. Then computing the other coefficients in F_{29}, we obtain

$$f_{29,1}{\equiv}26xy{+}12z^5{+}28z$$

$$f_{29,2}{\equiv}x{+}2z^2{+}28z{+}2$$

$$f_{29,3}{\equiv}27z^9{+}z^5{+}9$$

Let $\Phi{\equiv}\{F_{p_1},{\ldots},F_{p_m}\}$ be the set of modular initial polynomial systems for each $p_i{\in}P$. Notice that computing Φ is easy to parallelize, i.e. we can compute each F_{p_i} separately.

[83] Let $p{\neq}0$ be an integer. We say that two integers a and b are *congruent modulo p* if there is an integer k s.t. $a{-}b{=}kp$, and in this case we write

$$a{\equiv}b(mod\ p).$$

2.4.2 Modular Gröbner Basis

The next stage of the algorithm is to compute the modular Gröbner basis G_{p_i} for each F_{p_i} in Φ, creating a set of modular Gröbner bases $\Gamma=\{G_{p_1},\ldots,G_{p_m}\}$. The benefit of this modular approach is that there is no possibility of intermediate coefficient swell. Now we can simply apply Buchberger's algorithm[84] to compute G_{p_i} for each $p_i \in P$.

To continue with the example from above, we obtain G_{29}:

$$g_{29,1} \equiv x+2z^2-z+2$$

$$g_{29,2} \equiv -y-4z^8-14z^7-3z^6-2z^5+4z^4+8z^3-8z-4$$

$$g_{29,3} \equiv z^9+14z^5+10$$

Again, notice that finding the modular Gröbner basis for each $F_{p_i} \in \Phi$ is also trivial to parallelize.

2.4.3 Lifting/Checking to the Solution

In this section, we will define how to lift the elements of Γ to find a potential solution, then how to check to see if the lifted potential solution is a Gröbner basis of the initial set F. We will first define how coefficients are lifted via the Chinese Remainder Theorem (CRT), followed by an example.

Chinese Remainder Theorem[85] (CRT): Let p and q be two relatively prime odd integers. For every system of simultaneous congruences:

$$x \equiv a (mod\ p)$$

$$x \equiv b (mod\ q)$$

[84] Using degrevlex as the monomial ordering.

[85] Notice that this definition can easily be extended to the simultaneous congruences of more than two modular equations. For a formal definition of the CRT algorithm see Appendix B.

There exists a unique solution $\bar{x}$ modulo pq, i.e. $\bar{x}\equiv c(mod\ pq)$, where $\frac{-pq}{2}\leq\bar{x}\leq\frac{pq}{2}$.

Given the set of prime numbers P, the goal of the CRT is to lift the coefficients of a given subset of modular Gröbner bases. Let Ω be a subset of the set of modular Gröbner bases for each $p_i\in P$, i.e. $\Omega\subset\Gamma\equiv\{G(mod\ p_1),G(mod\ p_2),\ldots,G(mod\ p_m)\}$. Then, lifting all of the coefficients of $g_{p_i,j}$ over all primes contained in Ω, we create a potential solution $\widehat{G}_\Omega$.

As an example, suppose that we have $\Omega\equiv\{G_5,G_7,G_{11},G_{13}\}\subset\Gamma$ such that

$$\begin{aligned}
g_{5,1}&\equiv 4x^5+3x+1\\
g_{7,1}&\equiv 3x^5+x+6\\
g_{11,1}&\equiv 9x^5+7x+4\\
g_{13,1}&\equiv 8x^5+6x+11
\end{aligned}$$

By applying the CRT, we notice that the coefficient for x^5 modulo 5,005 is -2,124. Then, computing the other coefficients, we have the following polynomial $g_{\Omega,1}\in\widehat{G}_\Omega$:

$$g_{\Omega,1}\equiv -2124x^5-202x+1896$$

Now, returning to the example in Section 2.4.1, if we lift the modular Gröbner bases until the coefficients are in the integers[86], then we obtain a $\widehat{G}_\Omega$, for some Ω, defined in Equations 2.4-2.6.

[86] I.e. the coefficients are no longer changing as the lifting continues.

$$\widehat{g}_{\Omega,1} \equiv 436x - 56z^2 - z + 89 \tag{2.4}$$

$$\begin{aligned}\widehat{g}_{\Omega,2} \equiv\ & 26445714624043051560485490998640640y - 19859525057522153482953271384473 60z^8 \\ & - 12418033817859479908323464035958784z^7 - 29344953427515658230350116301045 76z^6 \\ & - 19683402043691824036031308706611200z^5 - 43477256422669598136139295269847 04z^4 \\ & - 12245780645929839319405963069358 08z^3 - 68879107874494007049946273499054 08z^2 \\ & - 18232060171665387363878293565603 84z - 10914300965461252214430964371161088\end{aligned} \tag{2.5}$$

$$\widehat{g}_{\Omega,3} \equiv 56z^9 + z^5 - 78 \tag{2.6}$$

After every lifting, a $\widehat{G}_{\Omega}$, for the given Ω, is obtained. Notice that the creation of $\widehat{G}_{\Omega}$ is independent of the creation of a $\widehat{G}_{\Omega'}$, for some $\Omega' \neq \Omega$. Therefore, the CRT step of the algorithm can also be computed in parallel.

The last step is to check if $\widehat{G}_{\Omega}$ is the Gröbner basis, G, of the initial system of polynomial equations F.

Theorem 2.1. *Let $\widehat{G}_{\Omega}$ be a set of polynomials such that $LM(\widehat{G}_{\Omega}) = LM(G_{p_i})$ for all p_i that are contained in Ω, $\widehat{G}_{\Omega}$ is a Gröbner basis for the ideal that it generates, $\left\langle \widehat{G}_{\Omega} \right\rangle$, and $I \subseteq \left\langle \widehat{G}_{\Omega} \right\rangle$. Then $I = \left\langle \widehat{G}_{\Omega} \right\rangle$.*

Proof. See Theorem 7.1 in Arnold (2003). □

Notice that, since only $I \subseteq \left\langle \widehat{G}_{\Omega} \right\rangle$ is required, and we do not have to test the inverse set containment, the computation is simplified. Showing the inverse containment, $I \supseteq \left\langle \widehat{G}_{\Omega} \right\rangle$, "is as difficult a problem as computing a Gröbner basis for I," Arnold (2003). Another noteworthy aspect of the MGBA is that P is not mentioned in Theorem 2.1. The algorithm is able to solve for the Gröbner basis so long as there exists a set of lucky primes. Since, given a large enough set of primes P, and enough computing ability, we do not need to solve for lucky primes, hence more discussion of lucky primes can be found in Appendix C.

Since we have created a large set of potential Gröbner bases, each can be checked separately, and thus this last step is also ideal for a parallel environment. Hence the entire algorithm can be paralellized.

2.5 Example 1: Duopoly Model

The first example we discuss is a Bertrand pricing game with two firms and three types of consumers. This is a pedagogical example given to illustrate the various steps that are needed to transform an economic problem into a system of polynomial equations such that the MGBA can be applied to find all pure-strategy Nash equilibria in a single game with continuous strategies. We will begin by setting up the game, followed by defining demand and production, then we transform the equations into a system of polynomial equations F, and concluding by applying the MGBA to solve the game.

Game Setup: Following the typical setup of a Bertrand duopoly game, the two firms, X and Y, simultaneously choose their prices in order to maximize their profits and face the same marginal cost[87] m. However, instead of the typical setup of a representative agent, let there be three types of consumers, N={1,2,3}, and let the two goods be imperfect substitutes, denoted x and y, that are produced by firms X and Y, respectively. Then denote the price of x (y) as p_x (p_y).

Demand: Consumers of type 1 and 3 both have linear demand and only want to consume good x and y, respectively. Then Consumer 1's demand for goods x and y

[87] This is not a requirement to use MGBA, but is used as a simplifying assumption.

are

$$D_x^1(p_x,p_y)=A-p_x \tag{2.7}$$

$$D_y^1(p_x,p_y)=0 \tag{2.8}$$

While Consumer 3's demand functions are

$$D_x^3(p_x,p_y)=0 \tag{2.9}$$

$$D_y^3(p_x,p_y)=A-p_y \tag{2.10}$$

For type 2 consumers, the two goods are imperfect substitutes with a constant elasticity of substitution between the two goods, (σ), and a constant elasticity of demand for the composite good, (γ). Also, let n be a constant. Thus, the demand functions for this consumer type are given by

$$D_x^2(p_x,p_y)=np_x^{-\sigma}\left(p_x^{1-\sigma}+p_y^{1-\sigma}\right)^{\frac{\gamma-\sigma}{\sigma-1}} \tag{2.11}$$

$$D_y^2(p_x,p_y)=np_y^{-\sigma}\left(p_x^{1-\sigma}+p_y^{1-\sigma}\right)^{\frac{\gamma-\sigma}{\sigma-1}} \tag{2.12}$$

Therefore, the total demand for each good are defined as

$$D_x(p_x,p_y)=D_x^1(p_x,p_y)+D_x^2(p_x,p_y)+D_x^3(p_x,p_y) \tag{2.13}$$

$$D_y(p_x,p_y)=D_y^1(p_x,p_y)+D_y^2(p_x,p_y)+D_y^3(p_x,p_y) \tag{2.14}$$

Production: Now we define the firms' problems. First we assume that m is the unit cost of production for both firms. Therefore, the profit to firm X is

$$\Pi_x(p_x,p_y)=(p_x-m)D_x(p_x,p_y) \tag{2.15}$$

Similarly, the profit for firm Y is

$$\Pi_y(p_x,p_y)=(p_y-m)D_y(p_x,p_y) \tag{2.16}$$

The marginal profits for each firm are then $\frac{\delta\Pi_x(p_x,p_y)}{\delta p_x}$ and $\frac{\delta\Pi_y(p_x,p_y)}{\delta p_y}$. The equilibrium prices must satisfy the necessary conditions of

$$\frac{\delta\Pi_x(p_x,p_y)}{\delta p_x}=\frac{\delta\Pi_y(p_x,p_y)}{\delta p_y}=0 \tag{2.17}$$

Polynomial Form: As a simple example, we will choose the following parameterization of the model[88]: $m=1$, $A=50$, $n=2700$, $\gamma=2$, and $\sigma=3$. Therefore, the marginal profits are

$$\Pi_x^{'}=0=50-p_x+(p_x-1)\left(\frac{2700}{p_x^6\left(p_x^{-2}+p_y^{-2}\right)^{\frac{3}{2}}}-1-\frac{8100}{p_x^4\left(p_x^{-2}+p_y^{-2}\right)^{\frac{1}{2}}}\right)+\frac{2700}{p_x^3\left(p_x^{-2}+p_y^{-2}\right)^{\frac{1}{2}}} \tag{2.18}$$

$$\Pi_y^{'}=0=50-p_y+(p_y-1)\left(\frac{2700}{p_y^6\left(p_x^{-2}+p_y^{-2}\right)^{\frac{3}{2}}}-1-\frac{8100}{p_y^4\left(p_x^{-2}+p_y^{-2}\right)^{\frac{1}{2}}}\right)+\frac{2700}{p_y^3\left(p_x^{-2}+p_y^{-2}\right)^{\frac{1}{2}}} \tag{2.19}$$

The solutions of the game are defined as the solutions of Equations 2.18 and 2.19; however, to apply the MGBA all of the equations must be in polynomial form. Therefore, the next step is to "polynomialize" these equations.

Define $Z=\left(p_x^{-2}+p_y^{-2}\right)^{\frac{1}{2}}$. Then, by squaring both sides we obtain

$$0=Z^2-\left(p_x^{-2}+p_y^{-2}\right)$$

Note that this is still not polynomial, but if we multiply through by $p_x^2p_y^2$, then it is

[88] For this to be a tractable example, the parameters of γ and σ must be rational numbers. Otherwise, the marginal profits can not be converted into polynomials. Notice that m, A, and n must be elements of a field.

in polynomial form.

$$f_1 \equiv -p_x^2 - p_y^2 + Z^2 p_x^2 p_y^2 \tag{2.20}$$

With this definition of Z, we obtain the following marginal profits from Equations 2.18 and 2.19.

$$f_2 \equiv -2700 + 2700 p_x + 8100 Z^2 p_x^2 - 5400 Z^2 p_x^3 + 51 Z^3 p_x^6 - 2 Z^3 p_x^7 \tag{2.21}$$

$$f_3 \equiv -2700 + 2700 p_y + 8100 Z^2 p_y^2 - 5400 Z^2 p_y^3 + 51 Z^3 p_y^6 - 2 Z^3 p_y^7 \tag{2.22}$$

Now we can define our initial set of polynomial equations F as Equations 2.20, 2.21, and 2.22. Then, following the method shown in the Katsura example above, we have our initial ideal $I = \langle f_1, f_2, f_3 \rangle$. Then, given I we can compute the ideal J generated by the Gröbner basis G. The system G has the same roots as the initial system F. Now by applying the MGBA to the initial system F to solve for the Gröbner basis G, we are able to obtain the solutions to the Bertrand game.

Solutions: Applying the MGBA as described in Section 2.4, we obtain all possible solutions to the initial system of equations F. The goal is to eliminate all solutions that are not economically relevant.

In this case there are 62 solutions, of which 44 are complex and 18 are real. Nine of the real solutions contain negative values, which are not feasible prices. Thus, there are nine candidates for equilibria. Checking the second-order conditions of the firms' optimization problems eliminates another five solutions. Lastly, checking for global optimality, we observe there are two Bertrand equilibria.

$$(p_x, p_y) = (2.168,\ 25.157) \tag{2.23}$$

$$(p_x, p_y) = (25.157,\ 2.168) \tag{2.24}$$

Comments: In general, when can Bertrand duopoly games be solved as a system of polynomial equations, i.e. via the MGBA? Notice that the answer is derived by the form of aggregate demand, D_i, for each firm and the first order condition of each aggregate demand function, $\frac{\delta D_i}{\delta p_i}$.

To be more precise, in order to solve for the equilibrium set of prices, the necessary condition of

$$\frac{\delta \Pi_i}{\delta p_i}=(1-m)D_i+(p_i-m)\frac{\delta D_i}{\delta p_i}=0 \tag{2.25}$$

must be satisfied for each firm. Notice that, for each firm $i\in\{X,Y\}$, the necessary condition takes the form of a sum of two terms with a polynomial times the aggregate demand function, D_i, and the first order condition of D_i in terms of p_i.

The first case in which MGBA can be applied to Bertrand duopoly games[89] is when D_i and $\frac{\delta \Pi_i}{\delta p_i}$, for all i, are "polynomializable". A function is "polynomializable" if there exists a change of variables such that the original function can be replaced by a system of polynomials, see Equations 2.20-2.22 as an example.

The other case in which the MGBA can be applied is when D_i and $\frac{\delta \Pi_i}{\delta p_i}$, for all i, can be approximated via polynomials[90]. This can be reasonably applied to many Bertrand systems.

2.6 Example 2: Manifold Dynamic Programming

Now to present a novel all-solution technique for dynamic games and dynamic general equilibrium models, a manifold approach to dynamic programming. I will first discuss the conventional setup to dynamic programming, followed by the manifold dynamic programming approach via a simple growth example, and concluding with a discussion of complexity reduction for the algorithm.

Conventional DP: Let k_0 be the current state and k be the next period's state. The payoff function of the state variables is $U(k_0,k)$, and the value function for the

[89] Notice that there is no requirement that there need be only two firms, this solution technique could be applied to models with a larger number of firms.

[90] E.g. Chebyshev polynomials.

next period is $V_{old}(k)$ while $V_{new}(k)$ i the value function for today.

Given this notation, the Bellman equation is

$$V_{new}(k_0)=\max_{k} U(k_0,k)+\beta V_{old}(k) \tag{2.26}$$

Where the objective function is $U(k_0,k)+\beta V_{old}(k)$, and the choice variable is k.

Manifold DP Approach: In order to solve a dynamic programming problem we will first create an implicit expression for the value function, followed by describing the Bellman equation and creating the system of equations that can be solved via the MGBA.

The first step is to create an implicit expression for the value function tomorrow

$$val_{old}=ImpV_{old}(v,k) \tag{2.27}$$

Where $v=V(k)$ if and only if $val_{old}=0$. The Bellman equation for the implicitly defined V_{old} can now be defined as

$$\begin{aligned} &\max_{v,k} U(k_0,k)+\beta v \\ &\quad s.t.\ ImpV_{old}(v,k)=0 \end{aligned} \tag{2.28}$$

Notice that the objective function is now $U(k_0,k)+\beta v$, with choice variable of v and k, given the constraint $ImpV_{old}(v,k)=0$. Therefore, the Lagrangian is

$$\mathcal{L}=U(k_0,k)+\beta v+\lambda ImpV_{old}(v,k) \tag{2.29}$$

With the following first-order conditions

$$\mathcal{L}_k = U_k(k_0, k) + \lambda ImpV_{old,k}(v, k) \tag{2.30}$$

$$\mathcal{L}_v = \beta + \lambda ImpV_{old,v}(v, k) \tag{2.31}$$

The new value function, val_{new}, can now be defined as

$$val_{new} = v_{new} - v\beta - U(k_0, k) \tag{2.32}$$

Where $v_{new} = V_{new}(k_0)$ if and only if $val_{new} = 0$.

Collecting all of the equations that define the solution to the Bellman equation at a typical k_0, we obtain

$$val_{old} = ImpV_{old}(v, k) \tag{2.33}$$

$$\mathcal{L}_k = U_k(k_0, k) + \lambda ImpV_{old,k}(v, k) \tag{2.34}$$

$$\mathcal{L}_v = \beta + \lambda ImpV_{old,v}(v, k) \tag{2.35}$$

$$val_{new} = v_{new} - v\beta - U(k_0, k) \tag{2.36}$$

Simple Growth Example: In order to show how to apply a manifold approach to dynamic games and dynamic general equilibrium problems, we will describe how to convert a simple growth example into a system of equations that can be solved via the MGBA. First we specify the production function, utility function, and discount factors, then we solve for the corresponding equations to Equations 2.33-2.36.

To set up the model, we will need to describe both a production function and a utility function, as well as giving values to the parameters of the model, i.e. the depreciation rate, δ, and the discount rate, β. First, we describe the firm problem, then the household problem. Let the production function be $f(k) = (1-\delta)k + k^{\frac{1}{3}}$, where $\delta = \frac{1}{5}$. Then, let the utility function be $u(c) = \frac{-1}{c}$ and $\beta = \frac{9}{10}$.

Now we can specify the value and payoff functions. Let the initial guess be

$$V_{old}(k)=-k^{-2} \tag{2.37}$$

$$ImpV_{old}(v,k)=v^3k+1 \tag{2.38}$$

$$U(k_0,k)=u(f(k_0)-k)=\frac{-1}{k^{\frac{1}{3}}-\frac{k}{5}} \tag{2.39}$$

$$val_{new}(k)=v^3k+1 \tag{2.40}$$

By computing Equations 2.33-2.36 for this problem, we obtain

$$val_{old}=1+kv^3 \tag{2.41}$$

$$\mathcal{L}_k=\frac{-1}{\left(k_0^{\frac{1}{3}}-k+\frac{4}{5}k_0\right)^2}+v^3\lambda \tag{2.42}$$

$$\mathcal{L}_k=\frac{9}{10}+3kv^2\lambda \tag{2.43}$$

$$val_{new}=v_{new}-\frac{9}{10}v+\frac{-1}{k_0^{\frac{1}{3}}-k+\frac{4}{5}k_0} \tag{2.44}$$

In order to apply the MGBA, we must first "polynomialize" the system, yielding the following system of equations.

$$0=k_0-K_0^3 \tag{2.45}$$

$$val_{old}=1+kv^3 \tag{2.46}$$

$$\mathcal{L}_k=-50-45kv+45K_0v+36K_0^3v+50kv_{new}-50K_0v_{new}-40K_0^3v_{new} \tag{2.47}$$

$$\mathcal{L}_v=3\left(3+10kv^2\lambda\right) \tag{2.48}$$

$$val_{new}=-25+25k^2v^3\lambda-50kK_0v^3\lambda+25K_0^2v^3\lambda-40kK_0^3v^3\lambda+40k_0^4v^3\lambda+16K_0^6v^3\lambda \tag{2.49}$$

Notice that this is a system with four unknowns, (λ, k, v, v_{new}), and one parameter, k_0.

A Gröbner basis of this system yields a triangularization of the system defined in Equations 2.45-2.49, and, most importantly, the first polynomial will involve v_{new} and k_0. This then gives an implicit function solution for v_{new}, today's value, in terms of k_0, today's capital. Therefore, we feed in an implicit function for tomorrow's value function, and get out an implicit definition of today's value function.

To solve via value function iteration, we back up in time, so the "tomorrow" value is the solution to the last iteration

$$\begin{aligned}0=&2196-7200k+25200k^2-50400k^3+63000k^4-50400k^5+25200k^6-7200k^7\\&+900k^8-\Big(2800-19200k+76800k^2-227200k^3+496800k^4-792000k^5\\&+924000k^6-792000k^7+495000k^8-220000k^9+66000k^{10}-12000k^{11}\\&+1000k^{12}\Big)v+\Big(228-2400k+8400k^2-16800k^3+21000k^4-16800k^5+8400k^6\\&-2400k^7+300k^8\Big)v^2+\Big(30-120k+180k^2-120k^3+30k^4\Big)v^3+v^4\end{aligned} \tag{2.50}$$

$$0=20\left(-1-k+2k_0-k_0^2\right)+2kv\lambda \tag{2.51}$$

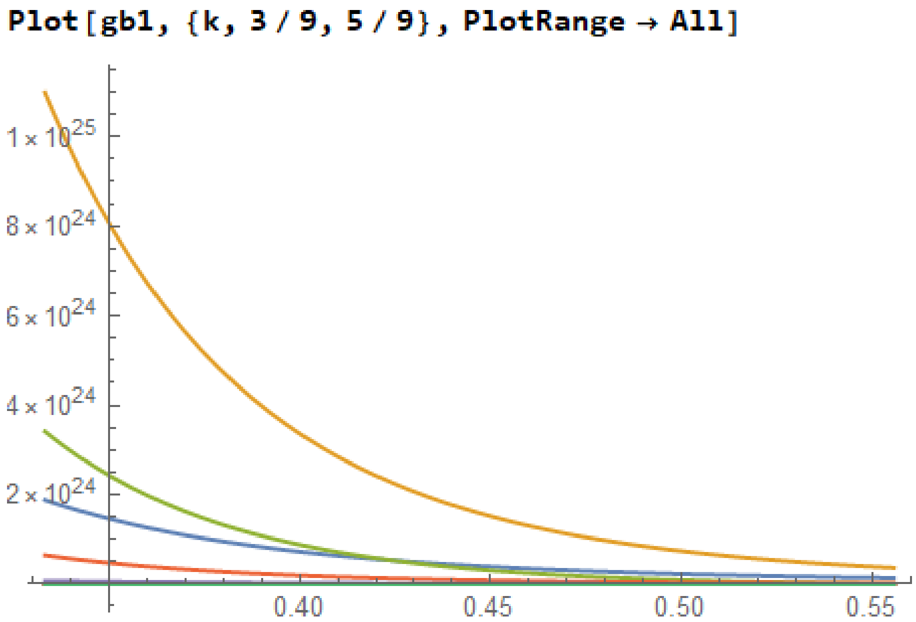
$$0=\frac{9}{10}+k^2\lambda \tag{2.52}$$

$$0=10\left(-1-k+2k_0-k_0^2\right)^2-\frac{9}{10}v+v_{new} \tag{2.53}$$

Now we compute the Gröbner basis again to obtain the new value function implicitly expressed in terms of k_0. The solution explodes very quickly. The complexity of the value function is increasing rapidly. Could it be that we have to give up on value function iteration? No. We will discuss how to reduce the computational complexity in the next section.

Complexity Reduction: The value function we have computed holds for all k, even for negative, very large, and complex k's. We care only about k in a small range of economically reasonable values. Let's study the polynomial val_{new}: It is degree 14 in v, and the coefficients are polynomials in k.

Let's plot the coefficient functions over the relevant range of capital.



```
Plot[gb1, {k, 3 / 9, 5 / 9}, PlotRange → All]
```

Figure 2.2: Coefficient Functions

Notice that in Figure 2.2 the functions are rather well behaved, thus implying that we can approximate them with low degree polynomials over this relevant range. By applying Chebyshev polynomial approximation of the high-degree coefficient functions in k, we can now decrease the highest power of v_{new} in val_{new} from a 52 degree polynomial to an 11 degree polynomial. If we do this for each power of v_{new} in val_{new}, then we get an approximation of val_{new}, which can now be the old value function manifold for the next iteration of value function iteration.

2.7 Conclusions

Many problems in economics focus on solving for an unknown function. Polynomial methods vastly increase the flexibility of the available models economists can use. Advances in algorithms and hardware make possible problems that were intractable

20 years ago. With the MGBA implemented both in Mathematica and in high-power computing environments, we can solve economics problems vastly more realistic than currently possible. Some potential applications and extensions of the MGBA are: economics of imperfect competition, competition analysis, dynamic games, equilibrium problems with equilibrium conditions, and financial markets. This work will also be of interest to the general community that uses Gröbner basis methods to solve polynomial systems[91].

[91] E.g. See the physics example in Section 2.3.3.

3 The Compression Value

3.1 Introduction

The field of cooperative game theory is centered on analyzing cooperation between different coalitions of players, and how to distribute payments across these players. There are two types of cooperative games: Transferable Utility (TU) games and Non-Transferable Utility (NTU) games. A game in which the players within each coalition are able to divide a single number, such as money, that can be interpreted as the pay-off or utility, among themselves in a mutually agreeable manner. NTU games, on the other hand, are described by a set of pay-off, or utility, vectors that are indexed by the members of the coalition for each of the available coalitions.

The theory of TU and NTU games have a long and storied history in economics. I will be focusing on one of the most prevalent solution techniques found in both the theoretical and applied cooperative game theory literature, the Shapley value. The Shapley value was introduced in Shapley (1953) for TU games to provide a solution technique that was characterized by a reasonable set of axioms[92]. The two main arguments that the Shapley value makes are that each player should be paid according to their marginal contribution and that each player's pay-off is not determined by their name. The NTU Shapley value, described in Shapley (1969), attempts to apply an axiomatic approach similar to that of TU games to NTU games[93].

As computational abilities have drastically increased over the last twenty years, however the field of cooperative games has fallen behind. That being said, there has been considerable work done to develop efficient computational methods to solve TU games. Chalkiadakis et al. (2012) describes the two major issues of identifying

[92] The axioms that characterize the Shapley value are: 1) Symmetry 2) Null Player Condition 3) Efficiency 4) Additivity. See Shapley (1953) for more information.

[93] The set of axioms for the NTU Shapley value are: 1) Non-Emptiness 2) Efficiency 3) Conditional Additivity 4) Unanimity 5) Scale Covariance 6) IIA 7) Closure Invariance. See Aumann (1985b).

compact representations for games and efficiently computing solution concepts for these games. They lay out the major solution concepts for TU games, and build up a framework for many games, but the difficulty of solving for many NTU solution concepts means that computational techniques are scarce.

Within the large applied theory literature that uses the Shapley value, many simplifications on the models must be made to allow for computational techniques to be feasibly applied. The first common simplifying assumption is to use a TU game. For example, van Campen et al. (2017) discusses both the computational problems facing the TU Shapley value and the need for “better approximations for the Shapley value . . . in the ranking procedure of individuals in networks of a terrorist, insurgent or criminal nature.”

CoinJoin games have also used the Shapley value to describe how coalitions are formed within the exchange of cryptocurrencies. For example, Arce and Böhme (2018) build an NTU game to describe the price that cryptocurrency users place on anonymity. In order to solve for the Shapley value of the given NTU game, they enforce that the optimal Pareto weights of the NTU Shapley value are egalitarian[94]. In this chapter of the dissertation, I discuss a new solution technique, the compression value, that is able to approximate the NTU Shapley value to allow for more complex models to be built to appropriately reflect CoinJoin games as well as coalition formation within criminal circuits such as hacking rings and terrorist groups.

In order to create an algorithm to solve for the NTU Shapley value, Andersen and Lind (1999) introduce a simplex approach to solve for the NTU Shapley value for games that are defined by a multiple objective linear program, MOLP. While this can be effective as for MOLP games or games that can be closely approximated by such a game, there still does not exist a general solution algorithm.

To advance the field of computational cooperative game theory and move toward

[94] I.e. $\lambda^*=(1,\ldots,1)$.

a general solution algorithm for the NTU Shapley value, I begin by defining a new solution technique, the compression value, and present an efficient algorithm to solve for the compression value. Lastly, I describe how the compression value can be used to approximate the NTU Shapley value.

The compression value of an NTU game is the linearly scaled TU Shapley value of the TU representation of the original game. This solution technique satisfies a reasonable set of properties that are discussed in Section 3.3.1. The compression value allows for an efficient algorithm to solve for this solution technique. The algorithm to solve for the compression value that is presented in this paper also presents an algorithm that is able to approximate the NTU Shapley value. The compression value has the primary benefit of giving a good approximation of the NTU Shapley value if the vector of Pareto weights associated with the NTU Shapley value is near the initial guess.

The paper begins with a brief introduction to TU and NTU games and the notation used in this paper within Section 3.2, followed by the main result, found in Definition 3.3, of this paper, the definition of the compression value and the algorithm used to solve for it, in Section 3.3. I then present a descriptive example of the compression value in Section 3.4, then I conclude in Section 3.5.

3.2 Preliminaries

A TU coalitional game is the tuple $G=(N,v)$, where $N=\{1,\ldots,n\}$ is the set of players and $v(\cdot)$ is a characteristic function that assigns a real number, $v(S)$, to each coalition $S\subset N$. In any characteristic function, the empty set of players, $\emptyset$ is assumed to give a zero payoff, i.e. $v(\emptyset)=0$.

An NTU coalitional game is defined as the tuple $\Gamma=(N,V)$, where $N=\{1,\ldots,n\}$ is still the set of players and $V(\cdot)$ is a function that assigns a subset $V(S)$ of $\mathbb{R}^S$ to each coalition $S\subset N$, such that:

Assumption 3.1. *For each coalition S, the set $V(S)$ is closed, convex, and comprehensive. Moreover, $0{\in}V(S)$ and $V(S){\cap}\mathbb{R}^S_+$ is bounded.*

Assumption 3.2. *$V(N)$ is smooth*[95].

Assumption 3.3. *If $x{,}y{\in}\delta V(N)$ and $x{\geq}y$, then $x{=}y$.*

Assumptions 3.1 and 3.2 are standard properties. Assumption 3.3 says that the frontier of the grand coalition payoff-set contains only strict Pareto-optima.

The TU Shapley value[96], $\phi(v)$, of a TU game v on N attempts to provide a "fair" distribution of payments to each player in the sense that each player is rewarded according to their contribution. This payoff distribution is defined as the vector, $\phi(v){\in}\mathbb{R}^N$, such that the i^{th} coordinate of the vector is given by

$$\phi_i(v){\equiv}\sum_{S\subseteq N\backslash\{i\}}\frac{|S|!(|N|{-}|S|{-}1)!}{|N|!}(v(S{\cup}\{i\}){-}v(S)) \tag{3.1}$$

The TU Shapley value provides a unique solution for each game G that satisfies a desirable set of properties. These properties have been studied in great detail in the literature, e.g. see Shapley (1953), Andersen and Lind (1999), etc. For example, if v is superadditive[97], then the TU Shapley value must be individually rational in the sense that $\phi_i(v){\geq}v(\{i\})$ for each player i in N.

Due to the nice properties exhibited by the TU Shapley value, there was a desire to find an appropriate NTU counterpart. Thus, the NTU Shapley value was introduced in Shapley (1969). The NTU Shapley value is defined as follows.

[95] A convex set $V(N)$ in $\mathbb{R}^N$ is said to be smooth if it has a unique supporting hyperplane at each point of its frontier, δV.

[96] See Shapley (1953).

[97] A characteristic function v is superadditive if and only if, for every pair of coalitions $S{,}T{\subset}N$, if $S{\cap}T{=}\emptyset$, then $v(S{\cap}T){\geq}v(S){+}v(T)$.

Definition 3.1. *Let $\lambda \in \mathbb{R}^N_{++}$ and let $S \subset N$, then define*

$$v^\lambda(S) \equiv sup\left\{\sum_{i\in S}\lambda_i x_i | x \in V(S)\right\} \tag{3.2}$$

Then, a vector $x \in V(N)$ is an NTU Shapley value of V, $\Phi(V,\lambda)$, if there exists a vector $\lambda \in \mathbb{R}^N_{++}$ such that $\lambda_i x_i = \phi_i(v^\lambda)$ for all $i \in N$.

Since $\Phi(V,\lambda)$ is equivalent to a TU Shapley value for the characteristic function v^λ over N, then it is reasonable to assume that the NTU Shapley value may be a "fair" distribution of payoffs for the game Γ. For more on this see Chapter 9.9 of Myerson (1997). Given Assumption 3.1, then the NTU Shapley value exists, but is not necessarily unique. Similar to many numerical root finding methods, the compression value algorithm described in the following section, Section 3.3, provides an approximation of one of the NTU Shapley values[98].

3.3 Compression Value

The compression value is a solution technique that satisfies a reasonable set of properties, that will be discussed later in this section, as well as providing an approximation of the NTU Shapley value that is easy to compute. This section begins with a definition of the compression value and a description of the algorithm. I then prove the existence of the compression value under a certain set of assumptions, and discuss the number of solutions that exist. Following this, I present a discussion of some of the properties satisfied by the compression value in Section 3.3.1.

In order to define the compression value and establish the algorithm to solve for this solution technique, I will first define the Egalitarian TU Representation. The Egalitarian TU Representation of an NTU game is the separating hyperplane that

[98] Future work is needed to examine which NTU Shapley value the compression value tends toward, if there are multiple solutions.

describes the TU version of $\Gamma{=}(N,V)$ with equal Pareto weight for each player[99].

Definition 3.2. *The Egalitarian TU Representation of (N,V) is v^e such that for all $S{\subset}N$*

$$v^e(S){\equiv}sup\left\{\sum_{i\in S}e_ix_i|x{\in}V(S)\right\} \tag{3.3}$$

Where $e{=}(1,\ldots,1){\in}\mathbb{R}^N_+$, i.e. there are equal Pareto weights.

Now that all of the hardware in place, I will define the compression value and describe the algorithm that can be applied to the NTU game Γ to obtain the compression value that is denoted as $\Phi(v^e,t^*)$.

Definition 3.3. *Let $\Gamma{=}(N,V)$ be an NTU game and v^e is the Egalitarian TU Representation of Γ. Then there exists $t^*{\in}\mathbb{R}_+$ such that $t^*{\cdot}\phi(v^e){\in}\delta V$ where $\phi(v^e)$ is the TU Shapley value of the game (N,v^e). The compression value is defined as $\Phi(v^e,t^*){=}t^*{\cdot}\phi(v^e)$.*

The compression value is so named since it takes the TU representation Shapley value, and "compresses", or linearly scales, $\phi(v^e)$ until it is feasible within the NTU game. The next question to be dealt with is: Does the compression value exist, and is it unique? In the following theorem, Theorem 3.1, I show that under Assumptions 3.1 and 3.2, the compression value exists. I also discuss when the compression value is unique.

Theorem 3.1. *Let $\Gamma{=}(N,V)$ be an NTU game such that V satisfies Assumptions 3.1-3.2, then the compression value exists. Moreover, if V satisfies Assumption 3.3, then there either exist one, two, or infinite compression values.*

Proof. Let $\phi(v^e)$ be the Shapley value of the Egalitarian TU Representation of Γ. Then there are two cases.

[99] This is an initial guess of the appropriate Pareto weights. If there is a reason to assume that the Pareto weights will not be equivalent across players, then an alternate $\widehat{\lambda}{\in}\mathbb{R}^N_+$ should be used to increase the accuracy of the compression value approximation.

1. $\phi(v^e) \in \delta V$, then existence is proved.

2. $\phi(v^e) \notin \delta V$, then construct $\Phi(v^e, t)$ as a linear function through $\mathbf{0}$ and $\phi(v^e)$. Notice that, via Assumptions 3.1-3.2, δV is a smooth, bounded, and convex function. Since $\mathbf{0} \in V(S)$, and $\Phi(v^e, t)$ has at least one point, $\mathbf{0}$, that is feasible and one point, $\phi(v^e)$, that is not feasible, then there exists a t^* such that $\Phi(v^e, t^*)$ is on the Pareto frontier.

Given a strict Pareto frontier, i.e. Assumption 3.3 holds, then there exist two cases.

1. Given $\mathbf{0} \in V(S)$ for all S, but if $\mathbf{0} \notin \delta V$, then $\Phi(v^e, t)$ intersects δV at a single point.

2. If $\mathbf{0} \in \delta V$, then $\mathbf{0}$ is a solution. If $\mathbf{0}$ is the only solution, then the proof is complete. Otherwise, there exist two cases:

   (a) If the Egalitarian TU Representation is equivalent to δV, then every element x of the Pareto frontier is a solution[100].

   (b) The last case is when δV is strictly convex, $\mathbf{0}$ is not an element of the Egalitarian TU Representation, and $\mathbf{0} \in \delta V$. Then either $\mathbf{0}$ is the only solution, or $\Phi(v^e, t)$ passes through the interior of $V(N)$, thus there exists $y \in \delta V$ such that $y = \Phi(v^e, t^*)$ for some $t^* \in \mathbb{R}_+$.

□

Now that the compression value has been defined and existence has been proved, the properties of this solution technique are discussed.

[100] In this case, the NTU Shapley value is equivalent to the $\phi(v^e)$, so computing the compression value is not necessary.

3.3.1 Properties

The goal of this section is to introduce a set of properties that the compression value satisfies[101].

Non-Emptiness Notice that via Theorem 3.1, so long as V satisfies Assumptions 3.1 and 3.2, then $\Phi(v^e,t^*)$ is non-empty.

Efficiency The compression value is also efficient, i.e. $\Phi(v^e,t^*)\subset\delta V$. Notice that this is obtained by definition.

Closure Invariance As with efficiency, closure invariance holds by definition.

Unanimity If U_T is the unanimity game on a coalition T, then $\Phi(u^e_T,t^*)=\left\{\frac{1_T}{|T|}\right\}$, by definition of the Shapley value on the TU game.

Scale Covariance If $\eta\in\mathbb{R}^N_+$, then the solution to $\Gamma=(N,\eta V)$ is $\eta\Phi(v^e,t^*)$, by the affine nature of the solution.

Inessential Games If $\mathbf{0}\in\delta V$, then $\mathbf{0}\in\Phi(v^e,t^*)$, by the construction of $\Phi(v^e,t)$.

3.3.2 Algorithm

Now I will describe the steps of the algorithm to solve for the compression value, $\Phi(v^e,t^*)$, of Γ.

Step 1: *Construct the Egalitarian TU Representation*

The first step is to construct the Egalitarian TU Representation of Γ. Following Definition 3.2, v^e is obtained.

Step 2: *Compute the Shapley value of* v^e

Since v^e is a TU game, then the game $G=(N,v^e)$ has a unique Shapley value $\phi(v^e)$ that can be solved via Equation 3.1.

Step 3: *Construct a linear function* $\Phi(v^e,t)$

Given the points $\phi(v^e)$ and $\mathbf{0}=(0,\ldots,0)$, both in $\mathbb{R}^{|N|}$, then define $\Phi(v^e,t)\equiv t\cdot\phi(v^e)$ for $t\in\mathbb{R}_+$, i.e. the line in $|N|$ dimensions through the points $\phi(v^e)$ and $\mathbf{0}$.

[101] Future research will be spent finding a full set of axioms that identify the compression value.

Step 4: *Solve for the optimal* t^* *to obtain the compression value* $\Phi(v^e, t^*)$

The last step is to set the linear function $\Phi(v^e, t)$ equal to the set of strictly Pareto optimal allocations, δV, i.e. $t^* \cdot \phi(v^e) \in \delta V$ for some $t^* \in \mathbb{R}_+$. Due to Assumptions 3.1-3.3, there exists a t^*, and thus a solution set denoted $\Phi(v^e, t^*)$, that solves $\delta V = t^* \cdot \phi(v^e)$. This is discussed in the following theorem.

3.4 Example

Within this section, I will show how to solve for the compression value, and discuss the conditions under which the compression value is able to approximate the NTU Shapley value of a classic NTU games found in the cooperative game theory literature, the market for gloves[102].

Suppose there is a pure exchange economy with three players, $N = \{1,2,3\}$, and two goods $X = \{x,y\}$. Let x and y be be perfect complements, e.g. x be left gloves and y be right gloves.

Consider the following initial endowments for a given $\varepsilon \in [0,1]$:

$$z_1 = (1-\varepsilon, 0) \tag{3.4}$$

$$z_2 = (0, 1-\varepsilon) \tag{3.5}$$

$$z_3 = (\varepsilon, \varepsilon) \tag{3.6}$$

Now to describe the utility function for each player in N.

$$u_1(x,y) = \min\{x,y\} \tag{3.7}$$

$$u_2(x,y) = \min\{x,y\} \tag{3.8}$$

$$u_3(x,y) = \frac{x+y}{2} \tag{3.9}$$

[102] See Shafer (1980), Roth (1980), Roth (1986), Aumann (1985a), and Vidal-Puga (2008)

Players 1 and 2 want "matching pairs of gloves", whereas player 3 just wants to use the material from the gloves.

This economy can be described as an NTU game via the following characteristic function V.

$$V(\{1\})=\{(x_1)|x_1\leq 0\} \tag{3.10}$$

$$V(\{2\})=\{(x_2)|x_2\leq 0\} \tag{3.11}$$

$$V(\{3\})=\{(x_3)|x_3\leq \varepsilon\} \tag{3.12}$$

$$V(\{1,2\})=\{(x_1,x_2)|x_1+x_2\leq 1-\varepsilon,x_1\leq 1-\varepsilon,x_2\leq 1-\varepsilon\} \tag{3.13}$$

$$V(\{1,3\})=\left\{(x_1,x_3)|x_1+x_3\leq \frac{1+\varepsilon}{2},x_1\leq \varepsilon,x_3\leq \frac{1+\varepsilon}{2}\right\} \tag{3.14}$$

$$V(\{2,3\})=\left\{(x_2,x_3)|x_2+x_3\leq \frac{1+\varepsilon}{2},x_2\leq \varepsilon,x_3\leq \frac{1+\varepsilon}{2}\right\} \tag{3.15}$$

$$V(\{1,2,3\})=\{(x_1,x_2,x_3)|x_1+x_2+x_3\leq 1,x_1\leq 1,x_2\leq 1,x_3\leq 1\} \tag{3.16}$$

The NTU Shapley value[103] gives a payoff of $\left(\frac{5(1-\varepsilon)}{12},\frac{5(1-\varepsilon)}{12},\frac{5\varepsilon+1}{6}\right)$.

By applying the algorithm in Section 3.3, I will show that the compression value of the game (N,V) is $\left(\frac{5-3\varepsilon}{4(3+\varepsilon)},\frac{5-3\varepsilon}{4(3+\varepsilon)},\frac{5\varepsilon+1}{2(3+\varepsilon)}\right)$.

Step 1: *Construct the Egalitarian TU Representation*

[103] See Shafer (1980).

The Egalitarian TU representation of the game is:

$$v^e(\{1\}){=}v(\{2\}){=}0 \tag{3.17}$$

$$v^e(\{3\}){=}\varepsilon \tag{3.18}$$

$$v^e(\{1{,}2\}){=}1{-}\varepsilon \tag{3.19}$$

$$v^e(\{1{,}3\}){=}v(\{2{,}3\}){=}\frac{1{+}\varepsilon}{2} \tag{3.20}$$

$$v^e(\{1{,}2{,}3\}){=}1 \tag{3.21}$$

$$ \tag{3.22}$$

Step 2: *Compute the Shapley value of* v^e

Now given $(N{,}v)$, the Shapley value of the TU game is $\phi(v^e){=}\left(\frac{5-3\varepsilon}{12}{,}\frac{5-3\varepsilon}{12}{,}\frac{5\varepsilon+1}{6}\right)$.

Step 3: *Construct a linear function* $\Phi(v^e{,}t)$

Solving for the line between the origin and $\phi(v^e)$, the following is obtained.

$$x_1(t){=}\frac{5{-}3\varepsilon}{12}t \tag{3.23}$$

$$x_2(t){=}\frac{5{-}3\varepsilon}{12}t \tag{3.24}$$

$$x_3(t){=}\frac{5\varepsilon{+}1}{6}t \tag{3.25}$$

Step 4: *Solve for the optimal* t^* *to obtain the compression value* $\Phi(v^e{,}t^*)$

When $\varepsilon{=}0$, then $\phi(v^e){=}\left(\frac{5}{12}{,}\frac{5}{12}{,}\frac{1}{6}\right)$, which is feasible since $\sum_{i\in N}\phi_i(v^e){=}1$ and $\phi_i(v^e){\leq}1$ for all $i{\in}N$. Therefore, for $\varepsilon{=}0$, the compression value is $\left(\frac{5}{12}{,}\frac{5}{12}{,}\frac{1}{6}\right)$.

However, if $\varepsilon{>}0$, then $\phi(v^e)$ is not feasible since $\sum_{i\in N}\phi_i(v^e){=}1{+}\frac{1}{3}\varepsilon{>}1$. Solving for the line between the origin and $\phi(v^e)$, the following is obtained.

$$x_1(t)=\frac{5-3\varepsilon}{12}t \tag{3.26}$$
$$x_2(t)=\frac{5-3\varepsilon}{12}t \tag{3.27}$$
$$x_3(t)=\frac{5\varepsilon+1}{6}t \tag{3.28}$$

The next goal is to find the feasible allocation, which means $\sum_{i\in N}x_i(t^*)=1$. Notice that $t^*=\frac{3}{3+\varepsilon}$, implying Equations 3.26-3.28 are equivalent to

$$x_1(t^*)=\frac{5-3\varepsilon}{4(3+\varepsilon)} \tag{3.29}$$
$$x_2(t^*)=\frac{5-3\varepsilon}{4(3+\varepsilon)} \tag{3.30}$$
$$x_3(t^*)=\frac{5\varepsilon+1}{2(3+\varepsilon)} \tag{3.31}$$

Therefore, the compression value is $\Phi(v^e,t^*)=\left(\frac{5-3\varepsilon}{4(3+\varepsilon)},\frac{5-3\varepsilon}{4(3+\varepsilon)},\frac{5\varepsilon+1}{2(3+\varepsilon)}\right)$.

Approximation: The key point to be discussed is how good of an approximation the compression value is for the NTU Shapley value. A distance measure between the compression value and the NTU Shapley value can be obtained for this example.

$$||\Phi(V,\lambda^*)-\Phi(v^e,t^*)||_2=\sqrt{2\left(\frac{5(1-\varepsilon)}{12}-\frac{5-3\varepsilon}{4(3+\varepsilon)}\right)^2+\left(\frac{5\varepsilon+1}{6}-\frac{5\varepsilon+1}{2(3+\varepsilon)}\right)^2} \tag{3.32}$$

Notice that when $\varepsilon=0$, the compression value and the Shapely value both equal $\left(\frac{5}{12},\frac{5}{12},\frac{1}{6}\right)$ since the Pareto weight vector of the NTU Shapley value is the egalitarian solution. Hence, when the initial guess of the Pareto weights is near the NTU Shapley value Pareto weights, λ^*, the approximation is very good.

However, as ε goes to one, the two values diverge. When $\varepsilon=1$, then under the Shapley value, player 3 has full control of the game and keeps all of the "gloves" for himself, i.e. the Shapley value is (0,0,1). However, under the compression value, the solution is $\left(\frac{1}{8},\frac{1}{8},\frac{3}{4}\right)$. Therefore, as λ^* moves away from e, the compression value loses its power as an approximation of the NTU Shapley value.

3.5 Conclusion and Future Research

The compression value is a novel solution technique that has a computationally efficient algorithm. The compression value can be used as an approximation of the NTU Shapley value so long as λ^* is near the egalitarian Pareto weights.

The goal of future research on this solution technique will both include an axiomitization of the compression value, as well as, applications of the compression value within price anonymity games[104] and hacker crime rings in order to better understand coalition formation. I will also continue on this work to improve the algorithm to give a better approximation of the NTU Shapley value when the Egalitarian TU Representation is far from λ^*.

[104] See Arce and Böhme (2018).

Bibliography

(2016). 2016 internet security threat report (istr). Technical report, Symantec.

(2016). Flipping the Economics of Attacks. Technical report, Ponemon Institute.

(2018). 2018 internet security threat report (istr). Technical report, Symantec.

Andersen, K. A. and Lind, M. (1999). Computing the ntu-shapley value of ntu-games defined by multiple objective linear programs. *International Journal of Game Theory*, 28(4):585–597.

Arce, D. and Böhme, R. (2018). Pricing anonymity. *Financial Cryptography and Data Security.*

Arnold, E. (2003). Modular algorithms for computing gröbner bases. *Journal of Symbolic Computation*, 35(4):403–419.

Arora, A., Nandkumar, A., and Telang, R. (2006). Does information security attack frequency increase with vulnerability disclosure? an empiricial analysis. *Information Systems Frontiers*, 8(5):350–362.

Arora, A., Telang, R., and Xu, H. (2008). Optimal policy for software vulnerability disclosure. *Management Science*, 54(4):642–656.

Aumann, R. (1985a). On the non-transferable utility value: A comment on the roth-shafer examples. *Econometrica*, 53:667–677.

Aumann, R. J. (1985b). An axiomization of the non-transferable utility value. *Econometrica*, 53:599–612.

Becker, G. S. (1968). Crime and punishment: An economic approach. *Springer.*

Blume, L. and Zame, W. (1992). The algebraic geometry of competetive equilibrium. *Economic Theory and International Trade*, pages 53–66.

Blume, L. and Zame, W. (1994). The algebraic geometry of perfect and sequential equilibrium. *Econometrica*, 62(4):783–794.

Borkovsky, R. (2017). The timing of version releases: A dynamic duopoly model. *Quantitative Marketing and Economics*, 15(3):187–239.

Borkovsky, R., Doraszelski, U., and Kryukov, Y. (2010). A user's guide to solving dynamic stochastic games using the homotopy method. *Operations Research*, 58(4):1116–1132.

Borkovsky, R., Doraszelski, U., and Kryukov, Y. (2012). A dynamic quality ladder model with entry and exit: Exploring the equilibrium correspondence using the homotopy method. *Quantitative Marketing and Economics*, 10(2):197–229.

Brandes, U. and Fleischer, D. (2005). Centrality measures based on current flow. *Proc. 22nd Symp. Theoretical Aspects of Computer Science (STACS '05)*, pages 533–544.

Cerdeiro, D., Dziubinski, M., and Goyal, S. (2017). Individual security, contagion, and network design. *Journal of Economic Theory*, 170:182–226.

Chalkiadakis, G., Elkind, E., and Wooldridge, M. (2012). *Computational Aspects of Cooperative Game Theory.* Morgan & Claypool Publishers, Williston, VT.

Choi, J. P., Fershtman, C., and Gandal, N. (2010). Network security: Vulnerabilities and disclosure policy. *The Journal of Industrial Economics*, 58(4):868–894.

Couzoudis, E. and Renner, P. (2013). Computing generalized nash equilibria by polynomial programming. *Mathematical Methods of Operations Research*, 77(3):459–472.

Cox, D., Little, J., and O'Shea, D. (2007). *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra.* Springer-Verlag New York, 3 edition.

Coyne, C. and Leeson, P. (2005). Who's to protect cyberspace? *Journal of Law, Economics & Policy*, 2:473–496.

Dziubinski, M. and Goyal, S. (2017). How do you defend a network? *Theoretical Economics*, 12(1):331–376.

Ebert, G. (1983). Some comments on the modular approach to gröbner bases. *ACM SIGSAM Bulletin*, 17:28–32.

Frei, S., Schatzmann, D., Plattner, B., and Trammell, B. (2010). Modeling the security ecosystem – the dyanmics of (in)security. *Economics of Information Security and Privacy Chapter 6.*

Goyal, P., Batra, S., and Singh, A. (2010). A literature review of security attack in mobile ad-hoc networks. *International Journal of Computer Applications*, 9(12).

Goyal, S. and Vigier, A. (2014). Attack, defense and contagion in networks. *Review of Economic Studies*, 81(4):1518–1542.

Grossklags, J., Johnson, B., and Christin, N. (2010). The price of uncertainty in security games. *Economics of Information Security and Privacy Chapter 2.*

Hong, Y. and Neilson, W. (2018). Cybercrime and punishment: A rational victim model. *Working Paper.*

Ion, I., Reeder, R., and Consolvo, S. (2015). "...no one can hack my mind": Comparing expert and non-expert security practices. *Symposium on Usable Privacy and Security (SOUPS).*

Judd, K. (1998). *Numerical Methods in Economics*. MIT Press.

Judd, K., Renner, P., and Schmedders, K. (2012). Finding all pure–strategy equilibria in games with continuous strategies. *Quantitative Economics*, 3(2).

Kalambe, K. and Apte, S. (2017). An exhaustive survey on security solutions in manets. *International Journal of Computer Science and Engineering*, 5.

Katsura, S. (1986). Theory of spin glass by the method of the distribution function of an effective field. *Progress of Theoretical Physics Supplement*, 87:139–154.

K.C., A. (2012). Software vulnerabilities: Key factors impacting on response time of software vendors in releasing patches for software vulnerabilities. *LAP LAMBERT Academic Publishing*.

Kubler, F., Renner, P., and Schmedders, K. (2014). Chapter 11 – computing all solutions to polynomial equations in economics. *Handbook of Computational Economics*, 3:599–652.

Kubler, F. and Schmedders, K. (2010a). Competitive equilibria in semi-algebraic economies. *Economic Theory and International Trade*, 145(1):301–330.

Kubler, F. and Schmedders, K. (2010b). Tackling multiplicity of equilibria with gröbner bases. *Operations Research*, 58(4):1037–1050.

Kuehn, A. and Mueller, M. (2016). Analyzing bug bounty programs: An institutional perspective on the economics of software vulnerabilities. *TPRC Conference Paper*.

Lalar, S. (2014). Security in manet: Vulnerabilities, attacks & solutions. *International Journal of Multidisciplinary and Current Research*.

Laszka, A., Zhao, M., and Grossklags, J. (2016). Banishing misaligned incentives for validating reports in bug-bounty platforms. *European Symposium on Research in Computer Security*, pages 161–178.

Maghsudi, S. and Hossain, E. (2016). Mulit-armed bandits with application to 5g small cells. *IEEE Wireless Communications*, 23:64–73.

Myerson, R. B. (1997). *Game theory - Analysis of Conflict.* Harvard University Press.

Owen, G. (1972). A value for non-transferable utility games. *International Journal of Game Theory*, 1:95–109.

Ozment, A. (2004). Bug auctions: Vulnerability markets reconsidered. *Workshop on the Economics of Information Security.*

Png, I. P., Tang, C. Q., and Wang, Q.-H. (2006). Hackers, users, information security. *WEIS Conference Precedings.*

Renner, P. (2015). Quantity precommitment and bertrand competition: A dynamic games approach. *SSRN.*

Renner, P. and Schmedders, K. (2015). A polynomial optimization approach to principal-agent problems. *Econometrica*, 83(2):729–769.

Rescorla, E. (2005). Is finding security holes a good idea? *Security Privacy, IEEE*, 3(1):14–19.

Roth, A. (1980). Values of games without side payments: some difficulties with current concepts. *Econometrica*, 48:457–465.

Roth, A. (1986). The non-transferable value: a reply to aumann. *Econometrica*, 54:981–984.

Schanuel, S., Simon, L., and Zame, W. (1991). The algebraic geometry of games and the tracing procedure. *Game Equilibrium Models II*, pages 9–43.

Shafer, W. (1980). On the existence and interpretation of value allocations. *Econometrica*, 48:467–477.

Shapley, L. (1953). A value for n-person games. In Kuhn, H. and Tucker, A., editors, *Contributions to the Theory of Games II*, pages 307–317. Princeton University Press, Princeton NJ.

Shapley, L. (1969). Utility comparison and the theory of games. In Kuhn, H. and Tucker, A., editors, *La Decision, Agregation et Dynamique des Ordres de Preference*, pages 251–263. Editions du Centre de la Recherche Scientifique, Paris.

Stephenson, K. and Zelen, M. (1989). Rethinking centrality: Methods and examples. *Social Networks*, 11:1–37.

van Campen, T., Hamers, H., Husslage, B., and Lindelauf, R. (2017). A new approximation method for the shapley value applied to the wtc 9/11 terrorist attack. *Social Network Analysis and Mining*, 8(1).

Vidal-Puga, J. (2008). Forming coalitions and the shapley ntu value. *European Journal of Operational Research*, 190:659–671.

A Mathematical Appendix

A.1 The Knife-Edge Case

The following cases satisfy $c_s{=}\delta D\sum_{i\in I}\theta_i$.

A.1.1 Non-Disclosure: Knife-Edge Case

In the the knife-edge case, $c_s{=}\delta D\sum_{i\in I}\theta_i$, which causes the expected profits of searching for Zero-Days to be equal to zero, which is equivalent to the hacker's the outside option. Notice that in the Non-Disclosure game, all payoffs available to the hacker are equal zero, thus all strategies are in the best response. Since the hacker is the only decision maker, the Nash equilibria of the Non-Disclosure game are $A^*_{nd}{=}\{(\rho_{nd}S{+}(1{-}\rho_{nd})X)\}_{\forall\rho_{nd}\in[0,1]}$.

A.1.2 Disclosure: Knife-Edge Case

In the case where the expected costs of a Zero-Day attack in the "Non-Disclosure" branch equals the search cost, then the hacker will be indifferent between $A^{(1-\alpha)*}_d{=}(S)$ and $A^{(1-\alpha)*}_d{=}(X)$. Notice that the expected value of searching for a Zero-Day under the "Disclosure" branch is $\widehat{\delta}D\sum_i\theta_i$ which is strictly less than $\delta D\sum_i\theta_i$. Therefore, searching for Zero-Days gives negative payoffs, while exiting the game yields a zero payoff. Given that at least one worker will never update, then exploiting the N-Day will give strictly positive payoffs, hence $A^{\alpha*}_d{=}(E)$

Thus, the set $\{(\rho_d(E,S){+}(1{-}\rho_d)(E,X))\}_{\rho_d\in[0,1]}$ is the best response for the hacker[105]. By the same reasoning as in Section 3.2.1.1, the best response of worker i is to not update, i.e. $i{\in}\Gamma_{nu}$, if $\theta_i{<}\frac{c_u}{v+D}$. Otherwise, for worker j such that $\theta_j{>}\frac{c_u}{v+D}$, updating is optimal[106]. Then the Nash equilibria are

$$((A^{\alpha*}_d,A^{(1-\alpha)*}_d),(A^*_i)_{i\in I}){=}((\rho_d(E,S){+}(1{-}\rho_d)(E,X)),(nu)_{i\in\Gamma_{nu}},(u)_{j\in\Gamma_u})\ \forall\rho_d{\in}[0,1] \tag{A.1}$$

A.1.3 Welfare: Knife-Edge Case

For all workers i such that[107] $i{\in}\Gamma^*_{nu}$, Non-Disclosure is the optimal strategy since the hacker will always exploit any released update. Thus, the only workers that may prefer Disclosure are the workers that have a positive probability of updating. Disclosure is only an optimal policy so long as the following is satisfied.

[105] Given that there exists at least one worker that will not update, i.e. there exists a worker i such that $\theta_i{<}\frac{c_u}{v+D}$. If this were not assumed then all strategies could yield the same payoff, and any mixed strategy over all strategies would give the same value.

[106] If $\theta_i{=}\frac{c_u}{v+D}$, then any mixture $p_j{\in}[0,1]$ of *Update* and *Not Update* are all equivalent to the worker.

[107] Here $\Gamma^*_{nu}{\equiv}\{i{\in}I|\theta_i{<}\frac{c_u}{v+D}\}$.

$$\alpha\big[(1-\rho_d^*\delta)\sum_{i\in\Gamma_{nu}^*}\theta_i-\rho_d^*\delta\sum_{j\in\Gamma_u^*}\theta_j+\xi^*\frac{c_u}{v+D}\big]<(\rho_{nd}^*-\rho_d^*)\delta\sum_{i\in I}\theta_i \tag{A.2}$$

To analyze when Disclosure is optimal, the equation must be broken down into cases. The first case is when both Equations A.3 and A.4 are positive or both cases are negative.

$$(1-\rho_d^*\delta)\sum_{i\in\Gamma_{nu}^*}\theta_i-\rho_d^*\delta\sum_{j\in\Gamma_u^*}\theta_j+\xi^*\frac{c_u}{v+D} \tag{A.3}$$

And

$$(\rho_{nd}^*-\rho_d^*) \tag{A.4}$$

Notice that in this case, increasing α decreases the desirability of Disclosure. This happens because Disclosure is useful for the workers as a way of decreasing the probability of a successful Zero-Day attack, since $\delta>\widehat{\delta}$.

Then when Equation A.3 is negative while Equation A.4 is positive, then the damage done when a hacker searches exceeds the damages done when the hacker exploits only. The hacker also has a higher probability of searching under Non-Disclosure than under Disclosure. Therefore, Disclosure is always the optimal policy.

Lastly, if Equation A.3 is positive while Equation A.4 is negative, then the hacker has a higher probability of searching under Disclosure than under Non-Disclosure. While all workers pay damages from the exploitation of the Zero-Days, these damages are exceeded by losses of the non-updating workers' hack as well as the cost assumed by the updating workers of c_u. Hence, Non-Disclosure is optimal.

As a note, Frei et al. (2010) find that hacker behavior has not changed despite the massive efforts by both security engineers and software vendors to find vulnerabilities, i.e. increases in α in this model. Even though α is included in both the hacker's profit function and the worker's problem, it does not show up in the optimal policy decision except in the knife-edge case.

A.1.4 Microsoft Non-Disclosure Best Response: Knife-Edge Case

Given the worker strategy $\Gamma_{nu}{=}I$, then the hacker is indifferent between *Search* and *Exit*. However, for any worker strategy such that $\Gamma_{nu}{\subset}I$, then $A_M^{(1-\alpha)*}{=}(X)$ is the best response.

A.1.5 Microsoft Disclosure Best Response: Knife-Edge Case

Given that the cost of searching is equivalent to the expected revenue if the vendor does not release an update, then

$$c_s{>}\widehat{\delta}D\sum_{i\in I}\theta_i$$

Therefore, (S) is not in the best response for any worker action since $\delta{>}\widehat{\delta}$.

Given a worker strategy $(\Gamma_{nu},\Gamma_u,\Gamma_v)$, the hacker's best response is $A_M^{\alpha*}{=}(X)$ if Equation A.37 holds. However, if Equation A.38 holds, then the hacker's best response is $A_M^{\alpha*}{=}(E)$.

A.2 Continuum of Workers Disclosure Game Equilibrium

To present a clean, closed form solution for the Nash equilibria within the low search cost case, I will present the case of a continuum of workers distributed over [0,1]. The value of worker i is now defined by the function $\theta{:}I{\to}[0{,}1]$, where $I{=}[0{,}1]$. Also, define $\Omega{\equiv}\left\{j{\in}I|\theta(j){\geq}\frac{c_u}{v+D}\right\}$

As above, the first step is to derive the best response of the workers. Notice that, again, if the hacker chooses the action $(E{,}S)$, then worker i is willing to update so long as

$$\theta(i){>}\frac{c_u}{v+D} \tag{A.5}$$

On the other hand, if $\theta(i){<}\frac{c_u}{v+D}$, then the cost of updating is too large for worker

i to be willing to update. Lastly, for the pivotal worker i such that $\theta(i)=\frac{c_u}{v+D}$, she is willing to mix between updating, $p(i)$, and not, $1-p(i)$, for any probability $p(i)$.

Next is to examine the best response of the workers if the hacker plays (S,S). Since updating does not protect the worker from being hacked and the cost of updating, c_u, is strictly greater than zero, then updating is not worthwhile for any worker. Thus, the best response for every worker is to not update.

Lastly, analysis of the workers' best response wouldn't be complete without their response to any mixed strategy, $\rho\in(0,1)$ of (E,S) and $1-\rho$ of choosing (S,S), of the hacker. Given that the hacker chooses the mixed strategy with the probability ρ^* of choosing (E,S), worker i's best response is to update so long as

$$\theta(i)>\frac{c_u}{\rho^*(v+D)} \tag{A.6}$$

Otherwise, worker i will not update if $\theta(i)<\frac{c_u}{\rho^*(v+D)}$. The last case is that worker i will be the pivotal worker if $\theta(i)=\frac{c_u}{\rho^*(v+D)}$.

Now we need to derive the best response of the hacker to any strategy of the workers. Recall that, under Non-Disclosure $A_d^{(1-\alpha)*}=(S)$. Then, under Disclosure, if all workers do not update, then $A_d^{\alpha*}=(E)$ is the hacker's best response since the cost of searching is strictly positive and the probability of finding a Zero-Day is strictly less than one.

To analyze all other cases, Equation 1.5 must first be rewritten as

$$\rho\left[D\int_{i\in\Gamma^*_{nu}}\theta(i)\right]+(1-\rho)\left[\widehat{\delta}D\int_0^1\theta(i)-c_s\right] \tag{A.7}$$

Hence, for any worker strategy (Γ_{nu},Γ_u), there exists a threshold value $\theta^*\in[0,1]$ such that all workers i with $\theta_i<\theta^*$ do not update, $i\in\Gamma^*_{nu}$, and workers j such that $\theta_j>\theta^*$ update, $j\in\Gamma^*_u$. Given a specific threshold value, the hacker's best response is to set $\rho^*=1$ if the cost of searching in addition to the loss in exploitation benefits exceed the expected value of searching for a Zero-Day, i.e. Equation A.8 holds.

$$c_s > \widehat{\delta} D \int_0^1 \theta(i) - D \int_0^{\theta^*} \theta(i) \tag{A.8}$$

However, if Equation A.9 holds, then the hacker's best response is to search for Zero-Days. In other words, the hacker will send $\rho \to 0$ giving a best response of $A_d^{\alpha *} = (S, S)$.

$$c_s < \widehat{\delta} D \int_0^1 \theta(i) - D \int_0^{\theta^*} \theta(i) \tag{A.9}$$

The final case is given by the following equation.

$$c_s = \widehat{\delta} D \int_0^1 \theta(i) - D \int_0^{\theta^*} \theta(i) \tag{A.10}$$

Then the best response of the hacker is to mix with any $\rho^* \in [0,1]$ since he is indifferent between *Exploit* and *Search*.

Theorem A.1. *If, for $\theta^*(k^*) = \frac{c_u}{v+D}$ i.e. the minimal worker in Ω, Inequality A.8 holds, then the Nash Equilibrium is*

$$((A_d^{\alpha *}, A_d^{(1-\alpha)*}), (A_i^*)_{i \in I}) = ((E,S), (nu)_{i \in \Gamma_{nu}^*}, (p(k)^*(u), (1-p(k)^*)(nu)), (u)_{j \in \Gamma_u^*})$$

Where $\theta^*(k) = \frac{c_u}{v+D}$, $\Gamma_{nu}^* = \{i \in I | \theta(i) < \theta(k)\}$, $\Gamma_u^* = \{i \in I | \theta(i) > \theta(k)\}$, and p_k^* is any mixture of updating and not updating.

Otherwise, there exists a Nash equilibrium such that there exists a pivotal worker $k^* \in \Omega$ such that Equation A.10 holds, and the Nash equilibrium is

$$((A_d^{\alpha *}, A_d^{(1-\alpha)*}), (A_i^*)_{i \in I}) = ((\rho^*(E,S), (1-\rho^*)(S,S)), (nu)_{i \in \Gamma_{nu}^*}, (p(k^*)^*(u), (1-p(k^*)^*)(nu)), (u)_{j \in \Gamma_u^*})$$

For any mixed strategy for worker[108] *k^*, $p(k^*)^* \in [0,1]$, and where $\rho^* = \frac{c_u}{\theta(k^*(v+D)}$, $\Gamma_{nu}^* =$*

[108] Since worker k^* is measure zero, and thus does not impact Equation A.10.

$\{i{\in}I|\theta(i){<}\theta^*(k^*)\}$, *and* $\Gamma_u^*{=}\{i{\in}I|\theta(i){>}\theta^*(k^*)\}$.

Proof. If Inequality A.8 holds, then the hacker's best response is (E,S). Then, given the hacker strategy of (E,S), the best response of low-type workers is $\Gamma_{nu}^*{=}\left\{i{\in}I|\theta(i){<}\frac{c_u}{v+D}\right\}$, while high-type workers' best response is $\Gamma_u^*{=}\left\{j{\in}I|\theta(j){>}\frac{c_u}{v+D}\right\}$, and worker k^* is indifferent between updating and not updating. Since worker k^* is of measure zero, then for every $p(k^*){\in}[0,1]$ are Nash equilibria.

If Equation A.10 holds, then the hacker is indifferent between any mixture of exploiting and searching. Then the strategy $\rho^*{=}\frac{c_u}{\theta(k^*(v+D)}$ makes the worker k^* indifferent between updating and not updating since $\theta^*(k^*){=}\frac{c_u}{\rho^*(v+D)}$. Therefore, all low-type workers' best response is $\Gamma_{nu}^*{=}\{i{\in}I|\theta(i){<}\theta^*(k^*)\}$, and high-type workers will update, i.e. $\Gamma_u^*{=}\{i{\in}I|\theta(i){>}\theta^*(k^*)\}$. Since this solution is in the best response of all workers and the hacker, this is the Nash equilibrium. □

A.3 Microsoft's New Policy Best Response Derivation

I will describing the best response functions for the workers and hacker under the "Non-Disclosure" branch, followed by the best response of the workers and hacker under "Disclosure" branch of the game. Once the best responses are solved for,

A.3.1 Non-Disclosure Worker Best Response

Beginning with the best response of the workers on the "Non-Disclosure" branch of the game, i.e. Figure 1.3, the worker can choose between switching to the new version or continue using the old version of the software. If the hacker's action is *Exit*, then worker i will continue to use the old version, $i{\in}\Gamma_{nu}$, since they are not at risk of being attacked and don't have to pay the fee of switching software versions, c_v.

The other pure-strategy available to the hacker is *Search*. The worker then must decide whether the payoff of changing versions exceeds the expected loss from using the old version.

$$v\theta_i - c_v \lesseqgtr -\delta D\theta_i + (1-\delta)v\theta_i \tag{A.11}$$

Which can be written in a similar manner as was observed above

$$\theta_i \lesseqgtr \frac{c_v}{\delta(v+D)} \tag{A.12}$$

By Assumption 1.2, there exist workers that will be willing to switch to the new version, while others will want to continue using the old software package. For high-type worker j, meaning that $\theta_j > \frac{c_v}{\delta(v+D)}$, then the best response to $Search$ is to switch to the new version of the software, i.e. $j \in \Gamma_v$. However[109], if $\theta_i < \frac{c_v}{\delta(v+D)}$, then worker i would prefer to continue using the old software, $i \in \Gamma_{nu}$.

The last strategy the workers need to have a response to is the mixed strategy $(\rho(S),(1-\rho)(X))$. The set of workers whose best response is to not update, $i \in \Gamma^*_{nu}$, so long as worker i's expected payoffs satisfy $\theta_i < \frac{c_v}{\rho\delta(v+D)}$. However, if $\theta_j > \frac{c_v}{\rho\delta(v+D)}$, then worker j's best response is to install the new version of the software, i.e. $j \in \Gamma^*_v$. Finally, if there exists a worker k such that $\theta_k = \frac{c_v}{\rho\delta(v+D)}$, then worker k is indifferent between using the old and the new software versions, and will be willing to mix with any probability $p^{nd}_k \in [0,1]$ of installing the new version.

A.3.2 Non-Disclosure Hacker Best Response

The hacker's best response is dependent on the relationship between the cost of searching for and the expected payoff of exploiting Zero-Days.

$$c_s \lesseqgtr \delta D \sum_{i \in \Gamma_{nu}} \theta_i \tag{A.13}$$

[109] As in the previous cases, if $\theta_i = \frac{c_v}{\delta(v+D)}$, then worker i is indifferent between software packages.

A.3.2.1 High Search Cost

Given any worker strategy, if $c_s > \delta D \sum_{i\in I}\theta_i$, then, in expectation, *Search* is too costly. Therefore, the best response of the hacker is (X).

A.3.2.2 Low Search Cost

Now to examine the low cost case, i.e. $c_s < \delta D \sum_{i\in I}\theta_i$. Given a workers' strategy $(\Gamma_{nu},(p_k(v),(1-p_k)(nu)),\Gamma_v)$, then we define $\Omega_M^{nd} \equiv \left\{ i\in I | \theta_i \geq \frac{c_v}{\delta(v+D)} \right\}$. Therefore, the expected payoff of the hacker for a mixed strategy $\rho(S),(1-\rho)(X)$ is

$$\rho\left[D\sum_{i\in\Gamma_{nu}}\theta_i+(1-p_k)D\theta_k-c_s\right] \tag{A.14}$$

Then the hacker's best response is *Exit*, $\rho\to 0$, if the expected value of searching is less than the opportunity cost of searching for a Zero-Day, i.e.

$$c_s>\delta\left[D\sum_{i\in\Gamma_{nu}}\theta_i+(1-p_k)D\theta_k\right] \tag{A.15}$$

On the other hand, if the inequality is inverted, then the best response of the hacker is $\rho\to 1$.

$$c_s<\delta\left[D\sum_{i\in\Gamma_{nu}}\theta_i+(1-p_k)D\theta_k\right] \tag{A.16}$$

The last possible case is when, for some $k\in\Omega_M^{nd}$ and there exists a $p_k\in[0,1]$,

$$c_s=\delta\left[D\sum_{i\in\Gamma_{nu}}\theta_i+(1-p_k)D\theta_k\right] \tag{A.17}$$

Then the hacker is indifferent between *Search* and *Exit*.

A.3.3 Disclosure Worker Best Response

On to the best responses of the "Disclosure" branch of the game. There are three separate cases to analyze to solve for the best response of the workers. Under each case, I will begin with the best response of the workers to hacker pure strategies followed by mixed hacker strategies.

A.3.3.1 Low New Version Costs

The first case is when the cost of switching to the new software version is less than the cost to keep the old version up to date against all disclosed vulnerabilities, $c_v < c_u + \phi_u$. Given the hacker plays (X), then to avoid paying any cost of updating or switching versions, the best response of all workers is to not update, i.e. $\Gamma^*_{nu} = I$.

When the hacker decides to search for a Zero-Day, (S) is played, then worker i will either switch to the new version or not update. If i were to update, then she would have to pay $c_u + \phi_u$, which is greater than the cost of switching versions, but would not be protected from search. However, if i installs the new version of the software, then she only has to pay ϕ_u and is protected if the hacker is successful in his attack. For high-type workers, $j \in I$ such that $\theta_j > \frac{c_v}{\delta(v+D)}$, $j \in \Gamma^*_v$ is the best response. For low-type workers, $i \in I$ such that $\theta_i < \frac{c_v}{\delta(v+D)}$, $i \in \Gamma^*_{nu}$ is the best response. If $\exists\ k \in I$ such that $\theta_j = \frac{c_v}{\delta(v+D)}$, then worker k is indifferent between any mixture, $p_k \in [0,1]$, of switching to the new version and not updating. Thus, $\Gamma^*_u = \emptyset$ if the hacker searches for a Zero-Day and $c_v < c_u + \phi_u$.

The last pure strategy that can be played by the hacker is (E). Given the Then the worker must balance the cost of installing the new software version against the cost of being hacked if she chooses not to update or purchase the new version of the software. Since updating and installing the new version of the software both protect the worker, then, given the fact that $c_v < c_u + \phi_u$, every worker would choose installing the new version over updating. Therefore, high-type workers, $j \in I$ such that $\theta_j > \frac{c_v}{v+D}$, will install the new version of the code, $j \in \Gamma_v^*$, while low-type workers, workers $i \in I$ such that $\theta_i < \frac{c_v}{v+D}$, will not update, $i \in \Gamma_{nu}^*$. Then, as in each of these cases, if there is a worker k such that $\theta_k = \frac{c_v}{v+D}$, worker k is indifferent between not updating the old version or switching to the new version. Again, $\Gamma_u^* = \emptyset$.

The final cases, under the assumption that $c_v < c_u + \phi_u$, are to examine are when the hacker plays a mixed strategy. Since installing the new version of the software is cheaper than updating, the worker will never update since the new version protects against both N-Day and Zero-Day exploits and is cheaper.

Then given a mixed strategy $(\rho_M^2(S), (1-\rho_M^2)(X))$ of the hacker, then each worker $i \in I$ much choose between not updating or installing the new version[110]. If the worker is a high-type worker, i.e.

$$\theta_i > \frac{c_v}{\rho_M^2 \widehat{\delta}(v+D)} \tag{A.18}$$

she will choose to install the new software version, i.e. $i \in \Gamma_v^*$. However, if Equation A.18 is inverted, then worker i will not update, $i \in \Gamma_{nu}^*$. Lastly, if Equation A.18 holds with equality, then the worker is indifferent between joining Γ_{nu}^* and Γ_v^*. Thus the worker's best response is to mix with any probability $p_i \in [0,1]$, such that p_i is the probability of installing the new version.

Next, given the hacker strategy of $(\rho_M^1(E), \rho_M^2(S))$, such that both $\rho_M^1, \rho_M^2 \in [0,1]$ and $\rho_M^1 + \rho_M^2 = 1$, then worker i will install the new version of the software if

$$\theta_i > \frac{c_v}{\rho_M^1 D + \rho_M^2 \widehat{\delta}(v+D)} \tag{A.19}$$

For similar reasons as above, if Equation A.19 is inverted, then not updating is the best response, and when it holds with equality, then any mixture of not updating and installing the new version is in the best response.

Moreover, if the hacker decides to play $(\rho_M^1(E), (1-\rho_M^1)(X))$, then worker i's best

[110] This is due to the fact that installing the update is costly and does not protect against Zero-Days.

response is $i \in \Gamma_v^*$ when

$$\theta_i > \frac{c_v}{\rho_M^1 (v+D)} \tag{A.20}$$

Conversely, when $\theta_i < \frac{c_v}{\rho_M^1 (v+D)}$, her best response is $i \in \Gamma_{nu}^*$. Finally, if $\theta_i = \frac{c_v}{\rho_M^1 (v+D)}$, then any mixture of not updating and installing the new version of the software is in the best response for worker i.

The final hacker strategy that needs to be analyzed under the assumption that $c_v < c_u + \phi_u$ is $(\rho_M^1(E), \rho_M^2(S), (1-\rho_M^1-\rho_M^2)(X))$. Worker i's best response to this strategy is to switch to the new software version, $i \in \Gamma_v^*$, so long as

$$\theta_i > \frac{c_v}{(\rho_M^1 + \rho_M^2 \widehat{\delta})(v+D)} \tag{A.21}$$

Again, if this equation is inverted, then $i \in \Gamma_{nu}^*$ is her best response, while she is willing to mix, with any probability, between not updating and using the new software version if it holds with equality.

A.3.3.2 High New Version Costs

The next case is when the cost of switching to the new version of the code is more expensive than simply updating the old version, $c_v > c_u + \phi_u$. Again, I will begin by solving for the workers' best response to the pure strategies of the hacker followed by the workers' best response to mixed hacker strategies.

Now to start the analysis of the workers' best response under the condition $c_v > c_u + \phi_u$ and given the hacker strategy of X. Since $c_v > c_u + \phi_u > 0$, then the best response for every worker is to not update, i.e. $\Gamma_{nu}^* = I$ and $\Gamma_u^* = \Gamma_v^* = \emptyset$.

The following case is when the hacker decides to search for a Zero-Day, (S). Notice that the protection to a worker of updating the old software version is equivalent to not updating, but if the worker updates she must pay $c_u + \phi_u$. Hence the worker would prefer not updating the old software over updating the old software if the hacker is going to play (S). Installing the new version, however, does protect the worker from a potential Zero-Day attack. Again, for high-type workers[111], $j \in I$ such that $\theta_j > \frac{c_v}{\delta(v+D)}$, will install the new software version, while low-type workers, $i \in I$ such that $\theta_i < \frac{c_v}{\delta(v+D)}$, will not update and risk the possibility of being hacked. If there exists $k \in I$ such that $\theta_k = \frac{c_v}{\delta(v+D)}$, then worker k is indifferent between installing the new version of the software and not updating the old version.

Furthermore, the hacker could play (E). Given the exploitation of the N-Day by the hacker, workers $i \in I$ such that $\theta_i < \frac{c_u+\phi_u}{v+D}$ will not want to update or download the new version, and thus $i \in \Gamma^*_{nu}$ is their best response. Since both updating and the new software version both defend against Zero-Day attacks, and due to the fact that updating is less costly than the new version, high-type workers[112] will want to update, i.e. $j \in \Gamma^*_u$. Additionally, as in every case, if there exists a worker k such that $\theta_k = \frac{c_u+\phi_u}{v+D}$, then worker k is indifferent between updating and not updating. Therefore, $\Gamma^*_v = \emptyset$.

The next set of hacker actions to analyze are when the hacker mixes, beginning with $(\rho^2_M(S),(1-\rho^2_M)(X))$. Recall that updating is costly, but will do nothing to protect the worker from being attacked, thus $i \in \Gamma_u$ is not a best response to this hacker strategy. Therefore, the worker must decide between the new version or using the old version with no updates. If Equation A.18 holds, then $i \in \Gamma^*_v$ is worker i's best response to $(\rho^2_M(S),(1-\rho^2_M)(X))$. As in the above case, if the inequality is inverted, the worker will not update, and if Equation A.18 holds with equality, then the worker is indifferent between the new version of the software and not updating the old software version.

Now, if the hacker plays $(\rho^1_M(E),(1-\rho^1_M)(X))$ for some $\rho^1_M \in (0,1)$, then the worker is unwilling to install the new version since updating will protect the worker just as well as the new version will and the new version costs more than updating. Thus,

[111] Recall that these workers do exist via Assumption 1.2.

[112] $j \in I$ such that $\theta_j > \frac{c_u+\phi_u}{v+D}$

the worker will decide between updating and not updating via the cutoff equation

$$\theta_i \lesseqgtr \frac{c_u + \phi_u}{\rho_M^1 (v+D)} \tag{A.22}$$

The best response is to update if greater, not update if less, and indifferent if equal.

When the hacker plays $(\rho_M^1(E), \rho_M^2(S))$, where $\rho_M^1, \rho_M^2 \in [0,1]$ and $\rho_M^1 + \rho_M^2 = 1$, then no strategy can be immediately eliminated. Worker i will not update, $i \in \Gamma_{nu}^*$ is the best response, so long as

$$\theta_i < \min\left\{ \frac{c_v}{\rho_M^1 D + \rho_M^2 \widehat{\delta}(v+D)} , \frac{c_u + \phi_u}{\rho_M^1 (v+D)} \right\} \tag{A.23}$$

The other evaluation left is whether high-type workers will update the old software or install the new version of the software. When the three following questions hold

$$\frac{c_v}{\rho_M^1 D + \rho_M^2 \widehat{\delta}(v+D)} \geq \frac{c_u + \phi_u}{\rho_M^1 (v+D)} \tag{A.24}$$

$$\rho_M^2 \widehat{\delta}(v+D) - \rho_M^1 v = 0 \tag{A.25}$$

$$\theta_i > \frac{c_u + \phi_u}{\rho_M^1 (v+D)} \tag{A.26}$$

Then, since $c_u + \phi_u < c_v$, all high-type workers will update, i.e. Γ_u^* is the best response[113]. On the other hand, if

$$\frac{c_v}{\rho_M^1 D + \rho_M^2 \widehat{\delta}(v+D)} < \frac{c_u + \phi_u}{\rho_M^1 (v+D)} \tag{A.27}$$

[113] However, if Inequality A.24 holds while $\theta_i = \frac{c_u + \phi_u}{\rho_M^1 (v+D)}$, then worker i is indifferent between updating and not updating the old version of the software.

Then when Inequalities A.25 and A.26 hold, then[114] $i \in \Gamma_u^*$. But when

$$\frac{c_v}{\rho_M^1 D + \rho_M^2 \widehat{\delta}(v+D)} < \theta_i < \frac{c_u + \phi_u}{\rho_M^1 (v+D)} \tag{A.28}$$

$i \in \Gamma_v^*$ is her best response.

The last case to be examined under this hacker strategy is when $\rho_M^2 \widehat{\delta}(v+D) - \rho_M^1 v \neq 0$. As above, any worker i such that Inequality A.23 holds, then $i \in \Gamma_{nu}^*$ is her best response. Then, for all workers j such that

$$\theta_j > \min\left\{ \frac{c_v}{\rho_M^1 D + \rho_M^2 \widehat{\delta}(v+D)} \, , \, \frac{c_u + \phi_u}{\rho_M^1 (v+D)} \right\} \tag{A.29}$$

then she will either install the new version or update the old version. Worker j will install the new version of the software, $j \in \Gamma_v^*$, when

$$\theta_j > \frac{c_v - (c_u + \phi_u)}{\rho_M^2 \widehat{\delta}(v+D) - \rho_M^1 v} \tag{A.30}$$

However, when

$$\theta_j < \frac{c_v - (c_u + \phi_u)}{\rho_M^2 \widehat{\delta}(v+D) - \rho_M^1 v} \tag{A.31}$$

Worker j's best response is $j \in \Gamma_u^*$. Worker j will be indifferent between updating the old version and installing the new version of the software when

$$\theta_j = \frac{c_v - (c_u + \phi_u)}{\rho_M^2 \widehat{\delta}(v+D) - \rho_M^1 v} \tag{A.32}$$

The final hacker strategy is $(\rho_M^1(E), \rho_M^2(S), (1 - \rho_M^1 - \rho_M^2)(X))$. The best response of the workers is the same as the $(\rho_M^1(E), \rho_M^2(S))$ case if Inequalities A.23 and A.31

[114] Of course, if Inequality A.27 holds while $\theta_i = \frac{c_u + \phi_u}{\rho_M^1 (v+D)}$, then worker i is indifferent between updating the old version and purchasing the new version of the software. Also, if $\theta_i = \frac{c_v}{\rho_M^1 D + \rho_M^2 \widehat{\delta}(v+D)}$, then worker i is indifferent between not updating the old version and installing the new version.

are replaced by

$$\theta_i < \min\left\{ \frac{c_v}{(\rho_M^1 + \rho_M^2 \widehat{\delta})(v+D)} \ , \ \frac{c_u + \phi_u}{\rho_M^1 (v+D)} \right\} \tag{A.33}$$

$$\theta_i < \frac{c_v}{\rho_m^2 \widehat{\delta}(v+D)} \tag{A.34}$$

respectively.

A.3.3.3 Knife-Edge New Version Costs

The final case in examining the best response function of the workers is when the cost of changing the old software to the new version es equal to the cost of updating the old software, $c_v = c_u + \phi_u$. Building up the best response of the workers as in the above cases, I first describe the workers' best response to the hackers exiting the game. Given that updating and installing the new version are costly, all workers will not update, $\Gamma_{nu}^* = I$, as in the above cases.

Given that the hacker chooses the action (S), then, as discussed above, the only action available to the worker that ensures protection is (v). Therefore, if the worker is of a high-type, which in this case means $j \in I$ such that $\theta_j > \frac{c_v}{\delta(v+D)}$, will install the new version, $j \in \Gamma_v^*$, while workers such that $\theta_i < \frac{c_v}{\delta(v+D)}$ will not update, $i \in \Gamma_{nu}^*$. If there is a worker k such that $\theta_k = \frac{c_v}{\delta(v+D)}$, then worker k is indifferent between purchasing the new version of the software and not updating the old version of the software package.

The last pure strategy that could be played by the hacker is (E). Given this strategy, the best response of worker i is dependent on[115]

$$\theta_i \lesseqgtr \frac{c_u + \phi_u}{v+D} = \frac{c_v}{v+D} \tag{A.35}$$

Workers $i \in I$ with low θ_i values best response is $i \in \Gamma_{nu}^*$, while workers $j \in I$ with high θ_j values are indifferent between installing the update or downloading the new version

[115] Notice that both updating and installing the new version will protect the worker.

of the software.

The next set of hacker actions to analyze are when the hacker mixes, beginning with $(\rho_M^2(S),(1-\rho_M^2)(X))$. This case is identical to the case in Section A.3.3.2.

Given the hacker strategy $(\rho_M^1(E),(1-\rho_M^1)(X))$, any worker $i\in I$ such that

$$\theta_i<\frac{c_u+\phi_u}{\rho_M^1(v+D)}=\frac{c_v}{\rho_M^1(v+D)} \tag{A.36}$$

will not update, i.e. her best response is $i\in\Gamma_{nu}^*$. All the high-type workers are indifferent between updating the old version and installing the new version, thus, all high-type workers will mix with any probability $p_j\in[0,1]$, where p_j is the probability that $j\in\Gamma_v^*$. Notice that if $\theta_k=\frac{c_u+\phi_u}{\rho_M^1(v+D)}=\frac{c_v}{\rho_M^1(v+D)}$, then worker k is indifferent between every strategy available.

If the hacker plays $(\rho_M^1(E),\rho_M^2(S))$ for some $\rho_M^1,\rho_M^2\in[0,1]$ and $\rho_M^1+\rho_M^2=1$, then the best response of the worker is identical to the $(\rho_M^1(E),\rho_M^2(S))$ case in Section A.3.3.2. Similarly, when the hacker plays $(\rho_M^1(E),\rho_M^2(S),(1-\rho_M^1-\rho_M^2)(X))$, all of the workers' best responses satisfy the same equations as in Section A.3.3.2 when the hacker plays $(\rho_M^1(E),\rho_M^2(S),(1-\rho_M^1-\rho_M^2)(X))$.

A.3.4 Disclosure Hacker Best Response: High Search Cost

Now to describe the best response of the hacker under a worker strategy of $(\Gamma_{nu},\Gamma_u,\Gamma_v)$, then the hacker's best response is dependent on the cost of being allowed to observe the update, ϕ_u.

When the cost of learning of the vulnerability exceeds the profits of hacking all of the workers, i.e.

$$\phi_u>D\sum_{i\in\Gamma_{nu}}\theta_i \tag{A.37}$$

Then the best response of the hacker is $A_M^{\alpha *}=(X)$ since both the expected payoff of the N-Day and the Zero-Day are strictly negative. Nevertheless, if the cost of the

N-Day is strictly less than the exploitation revenues,

$$\phi_u < D \sum_{i \in \Gamma_{nu}} \theta_i \tag{A.38}$$

The result will be that the hacker's best response is $A_M^{\alpha *} = (E)$.

A.3.5 Disclosure Hacker Best Response: Medium Search Cost

Notice that, as in the above cases, search under Disclosure yields a negative expected payoff, and, since exiting the game gives a zero payoff, (S) is not in the hacker's best response. Again, as in the above case, given a worker strategy $(\Gamma_{nu}, \Gamma_u, \Gamma_v)$, the hacker's best response is to exit the game, $A_M^{\alpha *} = (X)$, if Equation A.37 holds. However, if Equation A.38 holds, then the hacker's best response is $A_M^{\alpha *} = (E)$.

A.3.6 Disclosure Hacker Best Response: Low Search Cost

The final best response to solve for is when searching gives positive expected payoffs to the hacker. When the workers' strategy is $\Gamma_v = I$, then the hacker's best response is $A_M^{\alpha *} = (X)$. However, if $\Gamma_u = I$, then[116] $A_M^{\alpha *} = (S)$. Then if $\Gamma_{nu} = I$, the hacker must evaluate whether (E) or (S) yields higher expected payoffs. The hacker's best response is $A_M^{\alpha *} = (E)$ when

$$(1 - \widehat{\delta}) D \sum_{i \in I} \theta_i > \phi_u - c_s \tag{A.39}$$

However, search becomes optimal, $A_M^{\alpha *} = (S)$, if

$$(1 - \widehat{\delta}) D \sum_{i \in I} \theta_i < \phi_u - c_s \tag{A.40}$$

[116] When $c_s = \widehat{\delta} D \sum_{i \in I} \theta_i$, then the hacker is indifferent between exiting the game and searching for a Zero-Day.

The last case is where the hacker is indifferent between searching and exploiting, i.e. willing to choose any probability $\rho \in [0,1]$ of exploiting.

$$(1-\widehat{\delta})D\sum_{i\in I}\theta_i=\phi_u-c_s \tag{A.41}$$

Define $\Gamma_m=\{k\in I|k\notin\Gamma_v\cup\Gamma_u\cup\Gamma_{nu}\}$ as the set of all workers that will play some mixed strategy. All other a worker strategies can be written as[117] $(\Gamma_{nu},(p_k^u(u),p_k^v(v),(1-p_k^u-p_k^v)(nu))_{k\in\Gamma_m},$
$\Gamma_u,\Gamma_v)$ for some p_k^u, $p_k^v\in[0,1]$. To solve for the hacker's best response, this must be broken down into cases since exploiting the N-Day is no longer free.

A.3.6.1 High Updating Fees

The first case to examine is when $\phi_u\geq D\sum_{i\in I}\theta_i$. Under this condition, the hacker will never choose (E) since gaining access to the N-Day is too expensive, relative to searching for a Zero-Day. Therefore, the best response of the hacker is $A_M^{\alpha*}=(X)$ so long as

$$c_s>\widehat{\delta}D\left[\sum_{i\in\Gamma_{nu}\cup\Gamma_u}\theta_i+\sum_{k\in\Gamma_m}(1-p_k^v)\theta_k\right] \tag{A.42}$$

Whereas, if

$$c_s<\widehat{\delta}D\left[\sum_{i\in\Gamma_{nu}\cup\Gamma_u}\theta_i+\sum_{k\in\Gamma_m}(1-p_k^v)\theta_k\right] \tag{A.43}$$

Then $A_M^{\alpha*}=(S)$. The last case is when the hacker is indifferent between choosing any probability $\rho\in[0,1]$ of (S) and $1-\rho$ of (X).

$$c_s=\widehat{\delta}D\left[\sum_{i\in\Gamma_{nu}\cup\Gamma_u}\theta_i+\sum_{k\in\Gamma_m}(1-p_k^v)\theta_k\right] \tag{A.44}$$

[117] None of the Γ sets are equal to I since these have been covered above.

A.3.6.2 Low Updating Fees

When $\phi_u < D\sum_{i\in I}\theta_i$, the hacker may be willing to exploit, and thus his expected payoff is

$$\rho^1\left[D\left(\sum_{i\in\Gamma_{nu}}\theta_i+\sum_{k\in\Gamma_m}(1-p_k^u-p_k^v)\theta_k\right)\right]+\rho^2\left[\widehat{\delta}D\left(\sum_{i\in\Gamma_{nu}\cup\Gamma_u}\theta_i+\sum_{k\in\Gamma_m}(1-p_k^v)\theta_k\right)-c_s\right] \tag{A.45}$$

Where $\rho^1+\rho^2\leq 0$, since the hacker could always choose (X).

When both

$$D\left(\sum_{i\in\Gamma_{nu}}\theta_i+\sum_{k\in\Gamma_m}(1-p_k^u-p_k^v)\theta_k\right)<0 \tag{A.46}$$

$$\widehat{\delta}D\left(\sum_{i\in\Gamma_{nu}\cup\Gamma_u}\theta_i+\sum_{k\in\Gamma_m}(1-p_k^v)\theta_k\right)-c_s<0 \tag{A.47}$$

The hacker will send both ρ^1 and ρ^2 to zero, i.e. $A_M^\alpha=(X)$. If Inequality A.46 instead holds with equality, then the hacker would be indifferent between (E) and (X). Similarly, the hacker would be indifferent between (S) and (X) if Inequality A.47 held with equality. If both Inequalities A.46 and A.47 hold with equality, then the hacker is indifferent between all three actions.

Now, assuming at least one of the following hold, the hacker will either choose to exploit the N-Day or search for a Zero-Day.

$$D\left(\sum_{i\in\Gamma_{nu}}\theta_i+\sum_{k\in\Gamma_m}(1-p_k^u-p_k^v)\theta_k\right)>0$$

$$\widehat{\delta}D\left(\sum_{i\in\Gamma_{nu}\cup\Gamma_u}\theta_i+\sum_{k\in\Gamma_m}(1-p_k^v)\theta_k\right)-c_s>0$$

Thus, the hacker's best response is $A_M^\alpha=(E)$, i.e. $\rho\to 1$, when

$$c_s>D\left(\widehat{\delta}\sum_{j\in\Gamma_u}\theta_j+(1-\widehat{\delta})\sum_{i\in\Gamma_{nu}}\theta_i+\sum_{k\in\Gamma_m}\left[(1-p_k^u-p_k^v)-\widehat{\delta}(1-p_k^v)\right]\theta_k\right) \tag{A.48}$$

Otherwise, if

$$c_s < D\left(\widehat{\delta}\sum_{j\in\Gamma_u}\theta_j + (1-\widehat{\delta})\sum_{i\in\Gamma_{nu}}\theta_i + \sum_{k\in\Gamma_m}\left[(1-p^u_k-p^v_k)-\widehat{\delta}(1-p^v_k)\right]\theta_k\right) \tag{A.49}$$

Then the hacker will send ρ^2 to one, i.e. $A^{\alpha}_{M}=(S)$.

However, when

$$c_s = D\left(\widehat{\delta}\sum_{j\in\Gamma_u}\theta_j + (1-\widehat{\delta})\sum_{i\in\Gamma_{nu}}\theta_i + \sum_{k\in\Gamma_m}\left[(1-p^u_k-p^v_k)-\widehat{\delta}(1-p^v_k)\right]\theta_k\right) \tag{A.50}$$

Then the hacker will choose any strategy such that $\rho^1+\rho^2=1$.

A.4 Microsoft Nash Equilibrium: Other Cases

A.4.1 Medium Search Cost: Knife Edge Worker Costs

Theorem A.2. *If $c_s > \widehat{\delta}D\sum_{i\in I}\theta_i$ and $\phi_u < D\sum_{i\in I}\theta_i$, while the workers face $c_v=c_u+\phi_u$, and Equation 1.43 holds, then the Nash equilibria are any convex combinations of*

$$(A^{\alpha*}_{M},(A^{\alpha*}_{M,i})_{i\in I}) = \left((E),\left((nu)_{i\in\Gamma^{d*}_{nu}},(v)_{j\in\Gamma^{d*}_{v}}\right)\right) \tag{A.51}$$

And

$$(A^{\alpha*}_{M},(A^{\alpha*}_{M,i})_{i\in I}) = \left((E),\left((nu)_{i\in\Gamma^{d*}_{nu}},(v)_{j\in\Gamma^{d*}_{u}}\right)\right) \tag{A.52}$$

Where $\Gamma^{d*}_{nu}=\left\{i\in I|\theta_i<\frac{c_v}{(v+D)}\right\}$ and $\Gamma^{d*}_{v}=\left\{j\in I|\theta_j>\frac{c_v}{(v+D)}\right\}$.

Otherwise, if there exists $k^*\in\Omega_M$ and a mixed strategy for worker k, $p^{v*}_{k^*}\in[0,1]$, such that

$$\phi_u = D\sum_{i\in\Gamma^*_{nu}}\theta_i + (1-p^{v*}_{k^*}-p^{u*}_{k^*})D\theta_k \tag{A.53}$$

Then the Nash equilibrium of the game is

$$(A_M^{\alpha*},(A_{M,i}^{\alpha*})_{i\in I})=\Big((\rho^*(E),(1-\rho^*)(X)),\big((nu)_{i\in\Gamma_{nu}^{d*}},(p_{k^*}^{v*}(v),p_{k^*}^{u*}(u),(1-p_{k^*}^{v*}-p_{k^*}^{u*})(nu)),(v)_{j\in\Gamma_v^{d*}}\big)\Big) \tag{A.54}$$

Where $\Gamma_{nu}^{d*}=\{i\in I|\theta_i<\theta_{k^*}\}$, $\Gamma_v^{d*}=\{j\in I|\theta_j>\theta_{k^*}\}$, *and* $\rho^*=\frac{c_v}{\theta_{k^*}(v+D)}=\frac{c_u+\phi_u}{\theta_{k^*}(v+D)}$.

Notice that this is equivalent to Theorem 1.7, only that the worker is indifferent between updating the old software and installing the new version. Since the hacker does not search, whether the worker installs the new version or updates both yield the same payoff to the hacker.

A.4.2 Low Search Cost

Notice that the Non-Disclosure equilibrium is the same as in the medium search cost case, i.e. Theorem 1.5 holds. Then the Disclosure equilibrium is as follows.

Theorem A.3. *Let* $k_{min}\in\Omega_M$ *be the minimal worker in* Ω_M. *Under Disclosure, if* $\phi_u\geq D\sum_{i\in I}\theta_i$ *and*

$$c_s<\widehat{\delta}D\sum_{i\in I}\theta_i \tag{A.55}$$

Then the Nash equilibrium is

$$\Big(A_M^{\alpha*},\big(A_{M,i}^{\alpha*}\big)_{i\in I}\Big)=\big((S),((nu)_{i\in\Gamma_{nu}^{d*}},(v)_{j\in\Gamma_v^{d*}})\big) \tag{A.56}$$

Where $\Gamma_{nu}^{d*}=\{i\in I|\theta_i<\theta_{k_{min}}\}$, *and* $\Gamma_v^{d*}=\{j\in I|\theta_j\geq\theta_{k_{min}}\}$.

Otherwise, there exists a pivotal worker $k^*\in\Omega_M$ *and a mixed strategy for worker* k^* *strategy,* $p_{k^*}^{v*}\in[0,1]$, *such that*

$$c_s=\delta\left(D\sum_{i\in\Gamma_{nu}^{d*}}\theta_i+(1-p_k^{v*})D\theta_{k^*}\right) \tag{A.57}$$

Then the Nash equilibrium is

$$\left(A_M^{\alpha*},\left(A_{M,i}^{\alpha*}\right)_{i\in I}\right)=\left((\rho^*(S),(1-\rho^*)(X)),((nu)_{i\in\Gamma_{nu}^{d*}},(p_{k^*}^{v*}(v),(1-p_{k^*}^{v*})(nu)),(v)_{j\in\Gamma_v^{d*}})\right) \tag{A.58}$$

Where $\rho^*=\frac{c_v}{\theta_{k^*}\delta(v+D)}$, $\Gamma_{nu}^{k^*,d*}=\{i\in I|\theta_i<\theta_{k^*}\}$, *and* $\Gamma_v^{k^*,d*}=\{j\in I|\theta_j>\theta_{k^*}\}$.

Proof. If Inequality A.55 holds, then the hacker's best response is to play (S). Given the hacker strategy of (S), then $\Gamma_{nu}^{d*}=\{i\in I|\theta_i<\theta_{k_{min}}\}$ and $\Gamma_v^{d*}=\{j\in I|\theta_j\geq\theta_{k_{min}}\}$ are the workers' best responses. Therefore, A.56 is the Nash equilibrium.

If Equation A.62 holds, then the hacker is indifferent between searching for a Zero-Day and exiting the game. Then notice that $\rho^*=\frac{c_v}{\theta_{k^*}\delta(v+D)}$ causes worker k^* to be indifferent between moving to the new version and not updating the old version. Accordingly, A.63 is the Nash equilibrium. □

Theorem A.4. *Let* $k_{min}\in\Omega_M$ *be the minimal worker in* Ω_M*. Under Disclosure, if* $\phi_u<D\sum_{i\in I}\theta_i$, $c_v\leq c_u+\phi_u$, *and* $c_s<\widehat{\delta}D\sum_{i\in I}\theta_i$, *then there are three cases for the Nash Equilibrium*

1. *If* $D\sum_{i\in I}\theta_i-\phi_u<\widehat{\delta}D\sum_{i\in I}\theta_i-c_s$, *then the Nash equilibrium is*

$$\left(A_M^{\alpha*},\left(A_{M,i}^{\alpha*}\right)_{i\in I}\right)=\left((S),((nu)_{i\in\Gamma_{nu}^{d*}},(v)_{j\in\Gamma_v^{d*}})\right) \tag{A.59}$$

Where $\Gamma_{nu}^{d*}=\{i\in I|\theta_i<\theta_{k_{min}}\}$, *and* $\Gamma_v^{d*}=\{j\in I|\theta_j\geq\theta_{k_{min}}\}$.

2. *If* $D\sum_{i\in I}\theta_i-\phi_u>\widehat{\delta}D\sum_{i\in I}\theta_i-c_s$, *then the Nash equilibrium is*

$$\left(A_M^{\alpha*},\left(A_{M,i}^{\alpha*}\right)_{i\in I}\right)=\left((E),((nu)_{i\in\Gamma_{nu}^{d*}},(v)_{j\in\Gamma_v^{d*}})\right) \tag{A.60}$$

Where $\Gamma_{nu}^{d*}=\{i\in I|\theta_i<\theta_{k_{min}}\}$, *and* $\Gamma_v^{d*}=\{j\in I|\theta_j\geq\theta_{k_{min}}\}$.

3. If $D\sum_{i\in I}\theta_i-\phi_u=\widehat{\delta}D\sum_{i\in I}\theta_i-c_s$*, then the Nash equilibrium is*

$$\left(A_M^{\alpha*},\left(A_{M,i}^{\alpha*}\right)_{i\in I}\right)=\left((\rho_d(E),(1-\rho_d)(S)),((nu)_{i\in\Gamma_{nu}^{d*}},(v)_{j\in\Gamma_v^{d*}})\right)\ \forall\rho_d\in[0,1] \tag{A.61}$$

Where $\Gamma_{nu}^{d*}=\{i\in I|\theta_i<\theta_{k_{min}}\}$*, and* $\Gamma_v^{d*}=\{j\in I|\theta_j\geq\theta_{k_{min}}\}$*.*

Otherwise, there exists a pivotal worker $k^*\in\Omega_M$ *and a mixed strategy for worker* k^* *strategy,* $p_{k^*}^{v*}\in[0,1]$*, such that*

$$c_s=\delta\left(D\sum_{i\in\Gamma_{nu}^{d*}}\theta_i+(1-p_k^{v*})D\theta_{k^*}\right) \tag{A.62}$$

Then the Nash equilibrium is

$$\left(A_M^{\alpha*},\left(A_{M,i}^{\alpha*}\right)_{i\in I}\right)=\left((\rho^*(S),(1-\rho^*)(X)),((nu)_{i\in\Gamma_{nu}^{d*}},(p_{k^*}^{v*}(v),(1-p_{k^*}^{v*})(nu)),(v)_{j\in\Gamma_v^{d*}})\right) \tag{A.63}$$

Where $\rho^*=\frac{c_v}{\theta_{k^*}\delta(v+D)}$*,* $\Gamma_{nu}^{k^*,d*}=\{i\in I|\theta_i<\theta_{k^*}\}$*, and* $\Gamma_v^{k^*,d*}=\{j\in I|\theta_j>\theta_{k^*}\}$*.*

A.5 Microsoft Welfare: Low Search Cost

B CRT Algorithm

```
**input** : $x \equiv a_i (mod\ p_i)$ for $i = 1, ..., m$ and $\forall i, j \in \{1, ..., m\}$, $gcd(p_i, p_j) = 1$
**output:** $x \equiv \bar{x} (mod\ P)$

*Define:* $P \leftarrow p_1 \cdots p_m$;
**for** $i \leftarrow 1$ **to** $m$ **do**
    *Define:* $z_i \leftarrow \frac{P}{p_i}$;
    *Solve:* $y_i \leftarrow z_i^{-1} (mod\ p_i)$;
**end**
*Define:* $\bar{x} \leftarrow a_1 y_1 z_1 + \cdots + a_m y_m z_m$
```

Algorithm 1: Chinese Remainder Theorem

C Lucky Primes

A formal discussion of Lucky Primes can be found in Arnold (2003). We present a far more brief description for two reasons, the formal description can be found elsewhere, and by using high-power computing environments we do not have to worry about solving for lucky primes upfront.

As Arnold (2003) states: "Roughly, a prime p is lucky for the computation if we do not lose too much algebraic information when viewing the object to be computed modulo p." Arnold's algorithm is as follows:

1. Find a lucky prime with high probability

2. Use a Hensel algorithm or the CRT to lift

3. Check the result

The main difference in our approach, found in Figure 2.1, is that we do not attempt to solve for lucky primes, but instead, using a very large set of primes, we use the highly parallel nature of the algorithm to allow the computer to worry about lucky primes. We compute a large number of modular Gröbner bases, one for each prime, then we lift over subsets of the primes, checking results as we go, until we find a solution. Notice that this type of computation, while it would benefit from low-latency, is able to be computed easily in high-latency systems.