Disincentivizing Malicious Users in RecDroid Using Bayesian Game Model

Bahman Rashidi* and Carol Fung
Virginia Commonwealth University, Richmond, Virginia, USA
{rashidib, cfung}@vcu.edu

Abstract

RecDroid is an Android smartphone permission control framework which provides fine-grained permission control regarding smartphone resources and recommends the permission control decisions from savvy users to inexperienced (novice) users. However, malicious users, such as dummy users created by malicious app owners, may attempt to provide untruthful responses in order to mislead the recommendation system. Although a sybil detection function can be used to detect and remove some dummy users, undetected dummy users may still be able to mislead RecDroid framework. Therefore, it is not sufficient to depend on sybil detection techniques. In this work, we investigate this problem from a game-theoretical perspective to analyze the interaction between users and RecDroid system using a static Bayesian game-theoretical formulation. In the game, both players choose the best response strategy to minimize their loss in the interactions. We analyze the game model and find both pure strategy Nash equilibrium and mixed strategy Nash equilibrium under different scenarios. Finally, we discuss the impact from several parameters of the designed game on the outcomes, and analyzed the strategy on how to disincentivize attackers through corresponding game design.

Keywords: Game Theory, Android, Recommendation System, RecDroid

1 Introduction

The mobile apps business has finally 'come of age' and this worldwide success story is now growing fast. The number of apps is exploding, driven by a huge uptick in the number of mobile devices and people are using those apps all day long. According to a new mobile phone forecast from the International Data Corporation (IDC) Worldwide Quarterly Mobile Phone Tracker, worldwide smartphone shipments will reach a total of nearly 1.4 billion units in 2014, representing an increase of 26.3% over 2013. This number is predicted to double in the next 4 years, from 1.4 billion world-wide in 2013 to 2.5 billion by 2017 [1]. The number of mobile apps has been growing exponentially in the past few years. According to the official report by Android Google Play Store, the number of apps in the store has reached 1 billion, surpassed its major competitor Apple Apps Store, in 2013 [17]. With more and more people using their smartphones and tablets to surf the web, update social networking sites, and shop and bank online, cybercriminals and malware are increasingly targeting mobile devices with new smartphone threats and mobile threats. Along with the increment of the number of smartphone apps, the privacy and security of users has turn into a serious issue, especially when the smartphones are used for sensitive tasks or for business purposes. A malicious third-party app can not only steal private information, such as contact lists, instant messages, and GPS location from the user, it can also cause financial loss of the users by making secretive premium-rate phone calls and SMS [15].

Journal of Internet Services and Information Security (JISIS), volume: 5, number: 2 (May), pp. 33-46

*Corresponding author: Department of Computer Science, Virginia Commonwealth University, Tel: +1-804-402-7575, Web: http://www.people.vcu.edu/~rashidib/

PackageInstaller component of Android OS provides users with an interface to choose whether to grant resource permission to apps at installation. Android prevents malicious applications from unauthorized access to other apps' resource by restrict the resource access of each app within their own authorized list. In the current *PackageInstaller* component setting, users have to grant all requested resource permissions upon installing an app. However, this methodology has been turned out to be ineffective to protect users' privacy and resource from malicious apps and developers. The reason is that most users do not pay attention to the resources being requested [4] when installing a new app, since have a tendency to rush through permission request screens without thinking in order to use the application immediately. This way, malicious apps can easily get users accept their erroneous requests to access resources irrelevant to their main functionality (purpose).

Although Android app developers offer a variety of ways to protect users' transactions including authentication using e-tokens, one-time passwords, confirmation of transactions through codes sent to the phone, and more, this security issue of the OS is left unsolved. To deal with this security issue, we proposed RecDroid [13, 14, 12], which is a framework that allows users to install untrusted apps under a *"probation"* mode. In the probation mode, users can make real-time permission granting decisions when apps are running. The framework facilitates a user-help-user environment, where expert users' decisions are recommended to inexperienced (novice) users. RecDroid is a crowdsourcing recommendation system that collects apps' permission requests and users' permission responses, from which an expert seeking algorithm is used to discover expert users and a voting algorithm is used to compute a secure and reliable recommendation to permission requests (*granting* or *denying*).

Providing a user-help-user environment, where expert users' decisions are recommended to inexperienced users is the main goal of RecDroid. Therefore, the decisions from expert users determine what recommendation RecDroid will provide to inexperienced users. However, this also opens the door for malicious/dishonest users to misguide RecDroid's recommendations. For instance, the developer of malicious applications can create/employ multiple dummy users and gain high level of expertise through responding to other apps. However, those dummy users send dishonest responses to targeted requests from malicious apps to mislead the recommendation from RecDroid. A Sybil detection function may be able to detect some dummy users for RecDroid system. However, a sybil detection function may not discover all dummy users and their influence still exist when attackers are savvy enough to evade the detection system. Studying what the attacker can do to make maximum profit through malicious user attack and what RecDroid can do to minimize the damage caused by attack is our focus in this proposed game model.

In order to analyze the attackers' strategies and actions in the interaction with RecDroid framework, we use a game-theoretical model analyze the behavior and strategies from both users including attackers and RecDroid. More specifically, a static Bayesian game [7] is used and Nash equilibria of the game are investigated. In this Bayesian game the defender does not know the type of his opponent(regular or malicious) so it is also an incomplete information game [5]. Through this Bayesian game formulation, we are able to help RecDroid select best strategies to play against the attacker and minimize the potential damage caused by attackers (malicious users).

The rest of the paper is organized as follows: Section 2 introduces the RecDroid framework and a background on static Bayesian games; Section 3 describes our proposed static Bayesian model and its Nash equilibria; we discuss the impact of the model parameters in Section 4; related work overview is in Section 5; and finally Section 6 concludes this paper.

2 Background

In this section, we first describe the RecDroid recommendation system and its main functionality. We then describe malicious behaviors in RecDroid and detection technology.

2.1 RecDroid Recommendation System

RecDroid is a smartphone resource permission recommendation system that assists users to make correct permission granting decisions through providing expert advises. The framework provides a fine-grain permission control model that allows users to use apps without accepting all permissions up-front at installation. The permission requests only pop up when resources are used. RecDroid users can install untrusted apps under the *probation mode*, while *trusted mode* is for trusted apps by users. In probation mode, users make real-time resource granting decisions when apps are running and request to get access to a resource. The system also facilitates a *user-help-user* environment, where expert users' responses are recommended to novice users [13, 14, 12]. The RecDroid framework provides the following functionalities for users:

- Two application installation installation modes for apps that are about to be installed: *Trusted mode* and *Probation mode*. In probation installation mode, OS receives permission requests at run-time from apps that inquire to get access to sensitive resources (e.g. GPS location, phone calls, WiFi connections) when the permission is needed. In trusted mode, the app is fully trusted and all the requested resources are all granted automatically (current version of the Android OS).
- An system to intercept and collect applications' requested resources and responses, from which recommendations are made as for what permission from which apps should and should not be granted.
- A recommendation framework to assist users with resource granting decisions, by serving users with recommendations from users with high level of expertise on the same apps and requests.
- A user-based ranking algorithm to rank security risk and threat of smartphone apps and permissions.

Figure 1 shows the RecDroid's system architecture and interactions between each pair of its components in the system.

2.2 Malicious Users in RecDroid System

In RecDroid, malicious users may want to compromise the recommendation system. For example, the owner of a malicious app (attacker) may expect the app to be installed and their resource requests accepted by inexperienced. In order to do so, they may choose to compromise RecDroid by misleading the recommendation system using malicious expert users. The strategy of attack can be the following steps:

- The attacker may create multiple malicious new users (fraudulent users). Since RecDroid uses phone ID as user identification, creating each new user requires a new phone. This step brings certain money cost to the attacker.
- Each phone needs to install a number of apps and response resource requests from them objectively in order to increase and obtain its expertise level from RecDroid. Since RecDroid only uses expert users' responses for its recommendation, malicious users have to reach expert level ranking to influence the recommendation system. This step requires certain effort cost to the attacker.

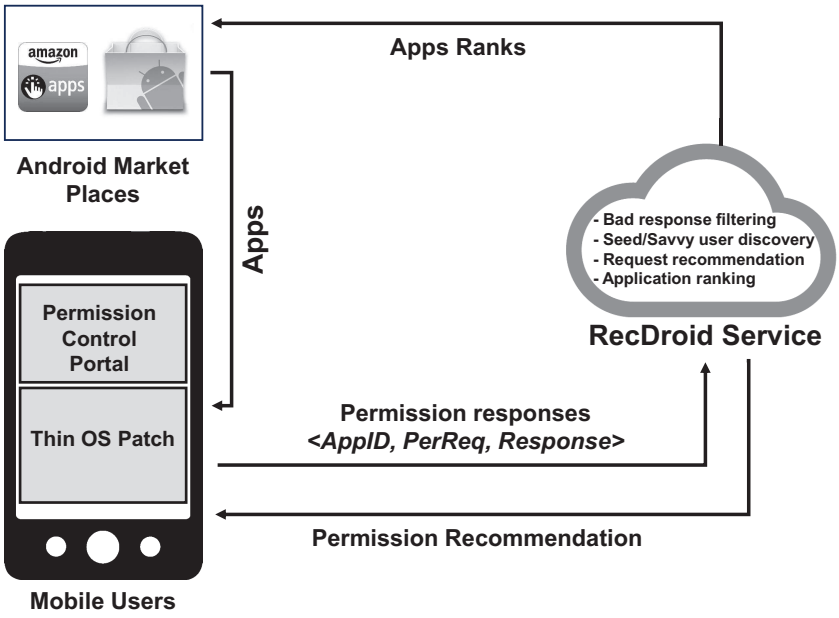


Figure 1: Architecture of the RecDroid and interactions

- After all users under control are trained to be experts, they start to install the one malicious app and response the resources requests dishonestly to mislead the recommendation system.

- In order to minimize the cost of successful attack, the step 3 should be launched as soon as the malicious app is released to public. This is because when the app has been installed by many other regular expert users, the influence from a few malicious expert users will be reduced due to the voting mechanism of that RecDroid adopts to make decisions. The best time to influence RecDroid is at the beginning.

2.3 Malicious Users Detection

In order to reduce the influence from malicious users, RecDroid has a malicious user detection function. In the detection step, RecDroid detects the type of users (malicious or normal) based on the users' behavior in responding to resource requests. There are two distinct options for RecDroid to perform malicious user detection: one is the machine-learning-based (ML) approach and the other is the human-based approach. The ML approach uses the similarity among malicious users in terms of their behaviors, and labels malicious users if they adequately similar to known malicious users in terms of behavior. For example, they were created at around the same time, installed similar set of apps, and they have similar responses to app requests. Compared to the human verification, ML-based detection costs much less and is much more efficient. However, it may not be able to detect malicious users created by sophisticated attackers. For example, malicious users can be created without sharing much commonality. On the other hand, the human-based verification approach uses a human labor to manually verify the responses to the app requests and compare them with the responses from other users. This way malicious users can be discovered and the attack fails. The human-based approach can have much lower false-positive rate and false-negative rate compared to the ML-based approach. However, the cost of human verification approach can be much higher so it shall not be used verify all applications.

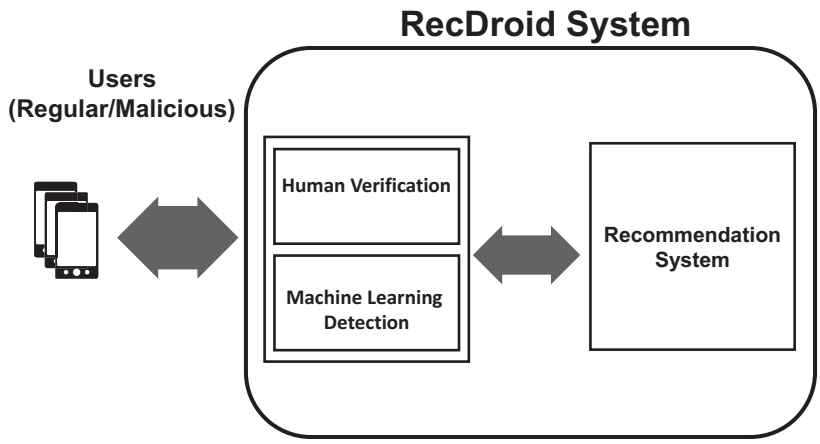


Figure 2: RecDroid system environment and detection system

Figure 2 illustrates the interaction between the users (malicious or normal) and the RecDroid framework. As show in the figure, the recommendation system takes responses from expert users (normal or malicious) and make recommendation to new users regarding whether to grant access to requests or not. An application that is chosen for verification, the recommendation is used to detect if there is a malicious users attack. If a ML-based detection method is used, then it will compare the similarity among users of the app and raise alarm if suspicious malicious users are found. After that the responses from suspicious malicious users are then removed from RecDroid system. If a human-based approach is found, the ground truth of responses are revealed and dishonest malicious users are discovered. However, if an application is not chosen to be verified, then the recommendation will be sent to new users directly. Therefore, without malicious user detection, a malicious user attack may succeed. However, malicious user detection methods may bring false-positive and false-negative. They also bring extra cost to RecDroid.

2.4 Bayesian Game Models

In game theory, a *Bayesian game* is a type of game in which information about features of the other players (i.e. payoffs) is incomplete and they are called games of *incomplete information*. Although any given player does not know the private information of an opponent (i.e. payoffs), she will have some beliefs about what the opponent knows, and we will assume that these beliefs are common knowledge [16]. For example, in card games, players have an incomplete information about the other players' cards. This is actually the case in most games. Instead, players consider a probabilistic distribution over the incomplete information.

A static Bayesian game can be modeled by introducing a node called 'Nature' as a player in a model. Nature relegates a random variable to each player which can take values of *types* for each player of game and associates probabilities or a probability density function with those users' types (in the course of the game, *nature* randomly chooses a type for each player according to the probability distribution across each player's type space) [16].

Considering the RecDroid system's features, its users (clients) and services, modeling the framework by designing a Bayesian game formulation is a feasible and effective solution. The idea behind the Bayesian game solution is that generally an attacker/defender game is an incomplete information game where the defender is uncertain about the his opponent's type (normal or malicious). A Bayesian game-

theoretical model provides defender with a framework to choose his strategies based on his belief about his opponent's type. As we described previously, the most important feature of the system is that players are aware of the type of others and that is why in this type of games we have a 'nature' node to distinguish them from each other [5]. In the next section, we describe the our proposed model and its components.

3 Game-Theoretical Model

There are two types of users in RecDroid, malicious and regular users. Detecting malicious users is a critical tasks for RecDroid. Since one attacker can create multiple malicious users, so what the RecDroid really plays with is either an attacker or a normal user. To study the interaction between the RecDroid system and users, we use a two-player static Bayesian game to model the behaviors of both parties. The static Bayesian game has two players: RecDroid users and the RecDroid framework. The RecDroid users have private information about their types and the types are unknown to the RecDroid system. However, the type of RecDroid framework is a common knowledge to all players (system and users).

Previously, we proposed a game-theoretical model [11], in which RecDroid has only to actions *Verify* and *Not Verify*. In the precious work, the verification system was only based on collected environmental information from apps and their developers and static analysis results. Since malicious users are in different levels of maliciousness, relying on these information as criteria to detect all types of malicious users (complicated and simple behavior) was not effective enough. We improved the model in a way that verification system is based on *Machine Learning* and *Human Verification* approaches. An attacker player has two strategies: *Attack* and *Not attack* when sending responses to permission requests from an app. The regular user has only one strategy: *Not attack*. When attack strategy is used, the attacker manipulates all malicious users to respond dishonestly to permission requests from an app. For example, the malicious expert users accept all malicious resource requests from an app, in order to mislead RecDroid into wrong recommendations.

Correspondingly, the RecDroid system has three strategies: *Human Verification*, *ML Verification*, and *No Verification*. When the no verification strategy is used, RecDroid makes recommendation based on all experts' responses without caring whether those responses are from malicious users or not; when ML verification is selected, RecDroid uses a machine learning approach to detect suspicious malicious users who are controlled by the same attacker, and those responses from suspicious users will not be considered by RecDroid recommendation; when human verification is chosen, a human expert will manually respond to permission requests from an app. ML verification is based on collected responses from users. In our ML verification system we consider users' responses as *response history* in order to assess the risk of considering responses in our recommendations. If assessed risk is high the response will be ignored with high probability and accept if the risk is low. This way malicious users' responses are not considered and the attack fails. Although human verification is the most accurate verification strategy in our system, it requires extra cost to the defender side due to human labor.

In order to track the payoffs of players in the game, we use ω_1 to denote the security value of the attacker, which is the gain of the attacker by performing a successful attack. For example, after a successful attack to RecDroid, a number of extra users accept malicious requests and it brings ω_1 extra profit through the attack. ω_2 is used to denote security value of RecDroid, which is the gain of a successful recommendation or the loss if compromised. For example, If RecDroid does not detect this behavior, RecDroid will make incorrect recommendations regarding the app and losses ω_2 of its security value, which can be the loss of reputation by making wrong recommendations to users. If RecDroid successfully detects attacks and removes malicious users, it gains reputation and we assume the gain is also ω_2.

Table 1: payoff matrices (RecDroid, Users)

(a) Player i is malicious

	ML Detection	Human Verification	No Detection
Attack	$(1-\alpha_m)\omega_1 - c_a$, $(\alpha_m - 1)\omega_2 - c_m$	$(1-\alpha_h)\omega_1 - c_a$, $(\alpha_h - 1)\omega_2 - c_h$	$\omega_1 - c_a, -\omega_2$
Not attack	$0, -\beta_m\omega_2 - c_m$	$0, -\beta_h\omega_2 - c_h$	$0,0$

(b) Player i is regular

	ML Detection	Human Verification	No Detection
Not attack	$0, -\beta_m\omega_2 - c_m$	$0, -\beta_h\omega_2 - c_h$	$0,0$

3.1 Normal Form

In this subsection we present the game in a static normal form. Table 1 depicts the matrices of payoffs of the Two-player game in normal form game style. In the payoff matrices, α_m and β_m indicate the detection rate (true-positive) and false alarm rate (false-positive) of RecDroid by using machine-learning detection, and $\alpha_m, \beta_m \in [0,1]$. We assume the cost of using machine detection is negligible (i.e., $c_m = 0$). We also assume human experts are reliable so that their decisions are consistently correct (i.e., $\alpha_h = 1, \beta_h = 0$). ω_1, ω_2 are the security value as we previously mentioned. The cost of attacking and human verification are denoted by c_a and c_h, where c_a, $c_h > 0$. For example, the attacker needs to spend c_a amount of money to purchase a number of smart phones and spend time to set them up into malicious mode to be able to influent the RecDroid system. The RedDroid needs to pay a seed expert c_h amount of money to verify the correct responses to the permission requests from an application, in order to detect attacks. We assume that ω_1, ω_2 are greater than c_a, c_h, since otherwise the attacker does not have incentive to attack and the RecDroid system has no incentive to use human verification, respectively.

Table 1(a) describes the payoff matrices of the RecDroid system and attackers. We can see that the expected gain for the RecDroid in (*Attack*, *ML Detection*) strategy combination is $-(1-\alpha_m)\omega_2 = (\alpha_m - 1)\omega_2$, in which $(1-\alpha_m)$ is the false-negative rate and a same for the attacker's payoff. This can be explained as: if the ML detector does not detect attack, then RecDroid generates incorrect recommendation and loses reputation. When the attacker chooses the *Not attack* strategy, the payoff for the attackers is always 0 and the RecDroid's payoff is depends on the false-positive rate and detection cost, in the *ML Detection* and *Human Verification* cases, respectively.

Table 1(b) shows the payoff matrices for the RecDroid system and regular users. The payoff value for the regular user is 0 for all RecDroid's strategies. The RecDroid's payoff depends on the strategies it plays. For example, when it plays *ML Detection* the payoff is $-\beta_m\omega_2$, which means the cost that RecDroid system lose the chance to make correct recommendation by falsely label some users to be malicious. Although we mainly focus on malicious users in the model, but we also consider the regular users and their actions in the calculations.

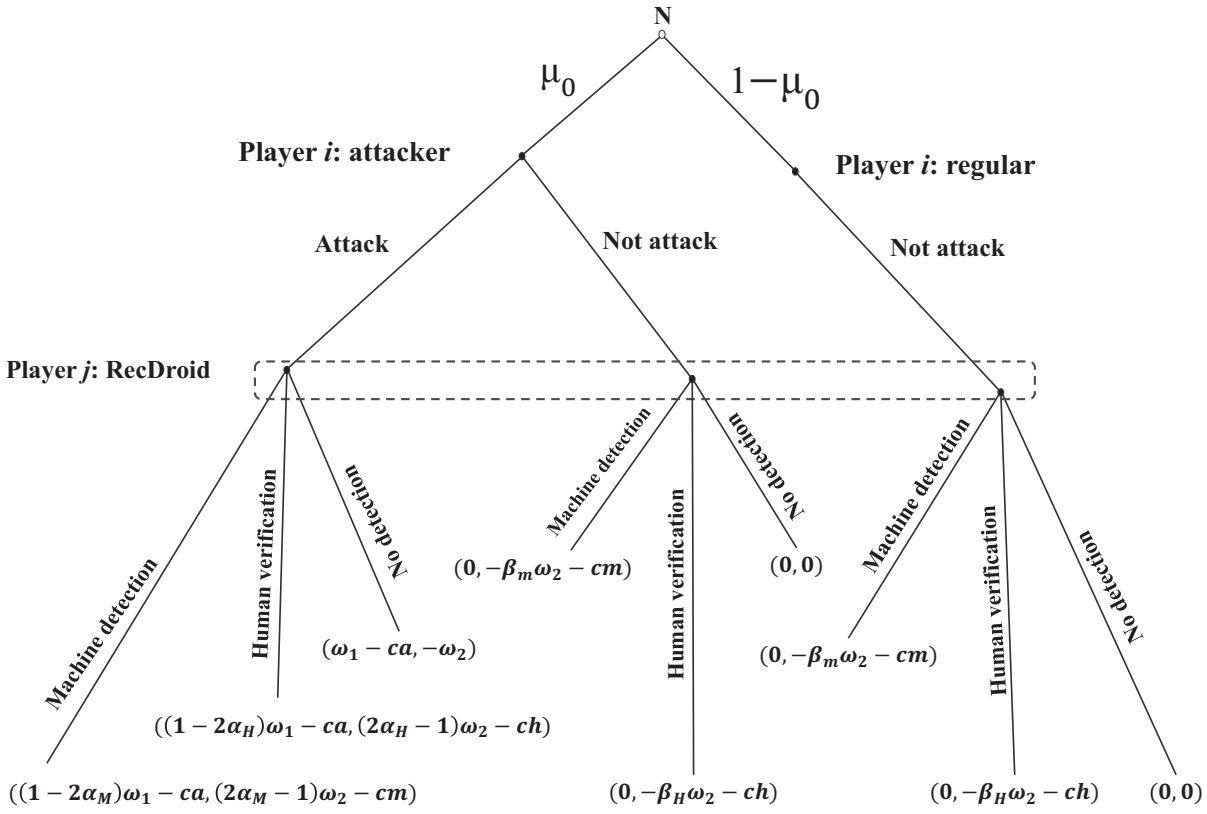


Figure 3: Extensive form of the Bayesian game

3.2 Extensive Form

In this subsection we show a different presentation form of the proposed game. Figure 3 shows the extensive form of the game. In this form we can see all the possible moves of the players and their choices on each decision state. In the figure, the root N indicates the nature node, and μ_0 represents the probability that RecDroid is playing with an attacker upon an app. The attacker may create multiple malicious smartphone users to attack the system. For regular users, their only strategy is to respond to the incoming requests honestly. RecDroid decides whether to verify the app's resource request or not depending on the μ_0 value as well as the other parameters of the game. Each terminal (leaf) node of the game tree has an 2-tuple of payoffs (RecDroid/user), which implies there is a payoff for each player at the end of every possible play.

3.3 Bayesian Nash Equilibrium (BNE)

As the main objective for all players in the game is to try to gain higher expected payoff. On one hand, attackers try to minimize the probability of being detected by RecDroid system, on the other hand, RecDroid tries to detect attackers with less detection cost. As described previously, the payoff of the strategies for both the players depends on different parameters. The parameters include the false-positive and true-positive of malicious user detection or nature (N). It is worth noting that the parameter μ_0 determined by nature has a high impact on payoffs of the players. In order to simplify the analysis,

we group the strategy *ML Detection* and *Human Verification* into the strategy *Detection* and drop their subscripts $_{h,m}$ in presentation. We will compare the two detection strategies when the *Detection* strategy is selected.

We first analyze the Bayesian Nash equilibrium of the game under different possible circumstances. The μ_0 denotes the defined probabilities, by which nature node shows the prior knowledge of the system about the users.

- The first possible case is when player *i* decides to play *Attack* if it is an attacker, *Not attack* when it is a normal user. In this case, if RecDroid decides to play *Detection* for the incoming responses from users. The RecDroid's expected payoff is as follow:

$$E_{pj}(Detection) = \mu_0((\alpha-1)\omega_2 - c) - (1-\mu_0)(\beta\omega_2 + c), \tag{1}$$

  and when the RecDroid decides to play *No Detection* the expected value would be

$$E_{pj}(NoDetection) = -\mu_0\omega_2, \tag{2}$$

  then if

$$E_{pj}(Detection) > E_{pj}(NoDetection) \tag{3}$$

$$\Rightarrow \mu_0 \geq \frac{\beta\omega_2 + c}{(\alpha+\beta)\omega_2},$$

  the best possible strategy for RecDroid is to play *Detection*. In this case if the RecDroid plays *Detection*, then the attacker only stays in strategy*Attack* if and only if $(1-\alpha)\omega_1 - c_a > 0 \Rightarrow \alpha < 1 - \frac{c_a}{\omega_1}$, which forms a Bayesian Nash Equilibrium. Therefore, under this condition, the *Attack* strategy for attackers, the *Not attack* for regular users and *Detection* for RecDroid is one pure-strategy BNE.

  In another case, if

$$\mu_0 < \frac{\beta\omega_2 + c}{(\alpha+\beta)\omega_2},$$

  the best strategy for the RecDroid is *No Detection*. Therefore, under this condition, the *Attack* strategy for attackers, the *Not attack* for regular users and *No Detection* for RecDroid is another pure-strategy BNE.

- The other case is when an attacker plays the *Not attack* and regular user plays *Not attack* strategy, regardless of μ_0, then RecDroid's dominant strategy is to play *Not verify*. If the RecDroid plays *No Detection*, then the best strategy for the attackers is to play *Attack*. Therefore, this strategy combination cannot be a BNE.

The above analysis shows that there is no pure-strategy BNE when $\mu_0 \geq \frac{\beta\omega_2 + c}{(\alpha+\beta)\omega_2}$ and $\alpha \geq 1 - \frac{c_a}{\omega_1}$. However, we can find a mixed-strategy BNE for this case. We let p denote the probability that an attacker plays *Attack*. We let q denote the probability that RecDroid plays *Detection*. We have

$$\begin{aligned} E_{pj}(Detection) = {} & p\mu_0((\alpha-1)\omega_2 - c) \\ & -(1-p)\mu_0(\beta\omega_2 + c) \\ & -(1-\mu_0)(\beta\omega_2 + c), \end{aligned} \tag{4}$$

$$E_{pj}(No-Detection) = -p\mu_0\omega_2 \tag{5}$$

then by imposing

$$\begin{aligned} E_{pj}(No-Detection) &= E_{pj}(Detection) \\ \Rightarrow p^* &= \frac{\beta\omega_2 + c}{(\alpha+\beta)\omega_2\mu_0}. \end{aligned} \tag{6}$$

In order to calculate mixed-strategy for users we impose

$$\begin{aligned} E_{pi}(Attack) &= E_{pi}(Not-attack) \\ \Rightarrow q^* &= \frac{\omega_1 - c_a}{\alpha\omega_1}. \end{aligned} \tag{7}$$

In summary, we have the mixed-strategy (($P[Attack] = p^*$ for attackers, *Not attack* for regular), $P[Detection] = q^*$ for RecDroid) under the condition that $\mu_0 \geq \frac{\beta\omega_2 + c}{(\alpha+\beta)\omega_2}$ and $\alpha \geq 1 - \frac{c_a}{\omega_1}$.

Correspondingly, we have the payoff for both players at mixed-strategy BNE are

$$\begin{aligned} E_{pi}(Attack) = E_{pi}(Not-attack) = 0, \\ E_{pj}(Detection) = E_{pj}(No-Detection) = \frac{\beta\omega_2 + c}{\alpha+\beta} \end{aligned} \tag{8}$$

3.4 Comparison Between the Two Detection Strategies

The above results shows the binary condition of the defender's strategies: *Detection* or *No Detection*. However, when the strategy *Detection* is chosen, there are actually two choices: *ML Detection* or *Human Verification*. RecDroid always chooses the strategy that brings higher payoff for the system. From Eq. [8] we have, at the mixed strategy BNE, if $c_h > \frac{\beta_m\omega_2}{\alpha_m+\beta_m}$, then the human verification strategy is chosen at the mixed-strategy BNE, otherwise the ML detection strategy is chosen at the mixed BNE.

3.5 Incentive Compatibility of RecDroid

We can summarize the implication of the BNEs that we have found above as follows:

- When the probability of attacker is small enough, aka. $\mu_0 < \frac{\beta_m\omega_2 + c}{(\alpha_m+\beta_m)\omega_2} = \frac{\beta_m\omega_2}{(\alpha_m+\beta_m)\omega_2}$ and $\mu_0 < \frac{\beta_h\omega_2 + c_h}{(\alpha_h+\beta_h)\omega_2} = \frac{c_h}{\omega_2}$, then there is no incentive for RecDroid to use any detection process for users' responses.

- When the probability of attacker is not small enough, aka. $\mu_0 \geq \min(\frac{\beta_m\omega_2}{(\alpha_m+\beta_m)\omega_2}, \frac{c_h}{\omega_2})$, then the RecDroid should either use *ML detection* or *Human verification* to check the input users responses, whichever costs less.

- However, in the second case, if RecDroid plays *Detection* with probability $q^* = \frac{\omega_1 - c_a}{\alpha\omega_1}$, there is no profit difference whether the attacker plays *attack* or *not attack*. Therefore, there is no incentive for attackers to perform attack in this case. Furthermore, if $q^* > \frac{\omega_1 - c_a}{\alpha\omega_1}$, then it is better off for attackers to play *not attack*. This way RecDroid system *disincentivize* attackers from attacking the system.

- It is worth noting that the payoff in Eq [8] is the maximum payoff RecDroid can obtain given that the system provides disincentive for attackers to attack the system.

4 Discussion

As we described previously, we have a number parameters in our proposed model and they all have impact on the outcome of the players in this Bayesian game. In this section we discuss two important and main parameters α and μ_0, which are essential with a high impact in our formulations. When the RecDroid's belief of μ_0 is high (high probability), which means the probability that the RecDroid system playing with an attacker is high, then RecDroid should play *verify* strategy with a high probability in order to get optimal payoff. If the parameter α is high and *beta* is low, which means the human verifier is more reliable, then the probability of RecDroid playing *verify* is low.

From Equation (7) we can see that the border-line *verify* probability q^* is influenced by parameters ω_1, c_a, and α. Higher ω_1, lower c_a, and lower α imposes higher higher probability of verity strategy. It is because (1) the cost of being attacked by system is higher, the RecDroid system should be more caucus and play more verify strategy. (2) the lower that cost of attack, the attackers will be more likely to attack and therefore, RecDroid should increase probability of verification. (3) the lower α is, the less reliable the human experts, then RecDroid should increase the probability of verification.

In addition, since verification system is based on human and machine learning approaches, it causes some challenges. First, using human verification as a tool to verify the incoming responses requires extra cost to the defender side due to human labor. Second, since applying machine learning approach needs enough collected information from users such as their responses history and environmental information, at the beginning of running the system we can not provide the ML system with enough information for bootstrapping step of the system.

5 Related Work

Liu et al. [9] proposed a game-theoretical approach to model the interactions between a DDoS attacker and a network administrator (system). They model the network and infer the attackers' intents, objectives, and strategies to observe the importance of modeling and its effects on risk assessment and harm prediction. Jormokka et al. [8] presented a few examples of static game models with complete information (players' information) where each example represents an information warfare scenario. Lye et al. [10] and Alpcan et al. [2] proposed two game-theoretical solutions to analyze the interactions between malicious attackers of system, IDS, and security of a computer network. In both works, they focus on the existing network parameters and the interactions between attacker and defender (system). Zhu, et al. [21, 20] proposed game-theoretical models to incentivize collaboration in intrusion detection networks (IDN). In this work, a game-theoretical model is proposed to analyze the behavior of IDSs in network and incentivize their participation by strategical game policy design. Due to the complexity of attackers' activities, many efforts have been proposed towards the risk assessment, modeling the attackers' activities (behavior), and cybersecurity strategies [3, 18, 19, 6].

In spite of the existing similarities between the RecDroid system and its architecture and related proposed works such as having an attacker in one side and a defender on the other side, there are differences, which seem to be significant and make these proposed models inapplicable to the RecDroid framework. For example, interaction between the RecDroid framework and users is different and more complicated than these models. Therefore, due to this inconsistency between the proposed models and RecDroid framework, we need to design a game model, which is more consistent to the RecDroid features. The proposed model should be able to model the interactions (request/response) between the game's players.

6 Conclusion

In order to analyze the interaction (permission request/user's response) between RecDroid recommendation framework and users (attacker/malicious, regular/normal), we proposed a static Bayesian game-theoretical model. Since RecDroid does not have enough knowledge about the users' type (maliciousness feature of users), we try to maximize and enhance the security of the framework's recommendations to users through training the system. As any two-player game needs to be provided with strategies spaces, we defined the possible strategy space for both the players (RecDroid system, users) based on a static game scenario. We also discussed the parameters of the proposed game that influence the final outcome of the players and challenges which are caused by Machine Learning approach and Human verification system as *verify* strategy of the RecDroid. In the proposed static Bayesian game, the RecDroid always assume fixed prior probabilities about the types of his opponent throughout the entire game period. Using Machine Learning and Human verification approaches can improve the accuracy of the prepared recommendations by RecDroid. We proved that the proposed static Bayesian game leads to a mixed-strategy BNE when the RecDroid's belief of player i (users) being malicious is high (high probability) and to a pure-strategy BNE when the RecDroid's belief of player i being malicious is low.

Using game theory to optimize the RecDroid's strategies to minimize the damage from the attacks by malicious users (fraudulent users) is our main focus in this work. As we described before, using ML and Human verification approaches causes some serious challenges, but RecDroid can use the proposed model to detect the malicious responses from normal (non-malicious) responses of the users and disregard them from recommendation system. Although it can increase the accuracy of the system, there is still a way to improve the model. The proposed model was a static Bayesian game and modeling the system in a dynamic (updating the current players' beliefs during the time) way can improve the accuracy of recommendations more. Using a dynamic model can help the RecDroid to use the *activity history* of the users and the game to improve its prior knowledge, α, and consequently the malicious user detection rate. Although using a dynamic model bring more overhead such as more calculation, updating the prior knowledge can improve the accuracy significantly.

References

[1] Smartphone users worldwide will total 1.75 billion in 2014. http://www.emarketer.com/Article/Smartphone-Users-Worldwide-Will-Total-175-Billion-2014/1010536.

[2] T. Alpcan and T. Basar. A game theoretic analysis of intrusion detection in access control systems. In *Proc. of the 43rd IEEE Conference on Decision and Control (CDC'04), Nassau, Bahamas*, volume 2, pages 1568–1573. IEEE, December 2004.

[3] L. Carin, G. Cybenko, and J. Hughes. Cybersecurity Strategies: The QuERIES Methodology. *Computer*, 41(8):20–26, 2008.

[4] A. P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, and D. Wagner. Android permissions: User attention, comprehension, and behavior. In *Proc. of the 8th Symposium on Usable Privacy and Security (SOUPS'12), Washington, D.C., USA*, pages 3:1–3:14. ACM, July 2012.

[5] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, Cambridge, Massachusetts, August 1991.

[6] C. J. Fung and R. Boutaba. Design and management of collaborative intrusion detection networks. In *Proc. of the 2013 IFIP/IEEE International Symposium on Integrated Network Management (IM'13), Ghent, Belgium*, pages 955–961. IEEE, May 2013.

[7] J. C. Harsanyi. Games with incomplete information played by bayesian players, i-iii part i. the basic model. *Management science*, 14(3):159–182, 1967.

[8] J. Jormakka and V. Molsa. Modeling information warfare as a game. *Journal of Information Warfare, 2005*, 4(2):12–25.

[9] P. Liu, W. Zang, and M. Yu. Incentive-based modeling and inference of attacker intent, objectives, and strategies. *ACM Transactions on Information and System Security (TISSEC)*, 8(1):78–118, February 2005.

[10] K. Lye and J. Wing. Game strategies in network security. *International Journal of Information Security*, 4(1/2):71–86, February 2005.

[11] B. Rashidi and C. Fung. A Game-Theoretic Model for Defending Against Malicious Users in RecDroid. In *Proc. of the 14th IEEE/IFIP IM2015 Workshop on Security for Emerging Distributed Network Technologies (DISSECT'15), Ottawa, Canada*, May 2015.

[12] B. Rashidi, C. Fung, G. Bond, S. Jackson, M. Pare, and T. Vu. Demo: RecDroid– An Android Resource Access Permission Recommendation System. In *Proc. of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'15), Hangzhou, China*. ACM, June 2015.

[13] B. Rashidi, C. Fung, and T. Vu. RecDroid: A Resource Access Permission Control Portal and Recommendation Service for Smartphone Users. In *Proc. of the 2014 ACM MobiCom workshop on Security and privacy in mobile environments (SPME'14), Maui, Hawaei, USA*, pages 13–18. ACM, September 2014.

[14] B. Rashidi, C. Fung, and T. Vu. Dude, Ask The Experts!: Resource Access Permission Recommendation with RecDroid. In *Proc. of the 14th IFIP/IEEE International Symposium on Integrated Network Management (IM'15), Ottawa, Canada*. IEEE, May 2015.

[15] W. Rothman. Smart phone malware: The six worst offenders. `http://www.nbcnews.com/tech/mobile/smart-phone-malware-six-worst-offenders-f125248`.

[16] S. Tadelis. *Game Theory: An Introduction*. Princeton University Press, January 2013.

[17] H. Victor. Android's google play beats app store with over 1 million apps, now officially largest. http://www.phonearena.com/news/Androids-Google-Play-beats-App-Store-with-over-1-million-apps-now-officially-largest_`id45680`.

[18] C. Xiaolin, T. Xiaobin, Z. Yong, and X. Hongsheng. A Markov Game Theory-Based Risk Assessment Model for Network Information System. In *Proc. of the 2008 International Conference on Computer Science and Software Engineering, Wuhan, Hubei, China*, volume 3, pages 1057–1061. IEEE, December 2008.

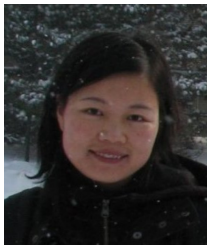
[19] X. You and Z. Shiyong. A kind of network security behavior model based on game theory. In *Proc. of the 4th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'03), Chengdu, China*, pages 950–954. IEEE, August 2003.

[20] Q. Zhu, C. Fung, R. Boutaba, and T. Basar. A game-theoretical approach to incentive design in collaborative intrusion detection networks. In *Proc. of the 2009 International Conference on the Game Theory for Networks (GameNets'09), Istanbul, Turkey*, pages 384–392. IEEE, May 2009.

[21] Q. Zhu, C. Fung, R. Boutaba, and T. Basar. GUIDEX: A game-theoretic incentive-based mechanism for intrusion detection networks. *IEEE Journal on Selected Areas in Communications*, 30(11):2220–2230, 2012.

Author Biography

Bahman Rashidi is a PhD student in Virginia Commonwealth University, USA. He received his BSc and MSc in computer engineering from University of Isfahan and Iran University of Science and Technology, Tehran, Iran, in 2011 and 2014 respectively. He is mainly interested in the Distributed Systems, Cloud Computing, Mobile Computing, Mobile Devices, and Privacy. Currently, he is doing research on a Malware detection framework for smartphones. He is the recipient of Distinguished Masters Student of the year in research award for two consecutive years in 2012 and 2013 from Iran University of Science and Technology and Outstanding Early-career Student Researcher, Virginia Commonwealth University, Apr 2015.



Carol Fung received her Bachelor degree and Master degree in computer science from the university of Manitoba (Canada), and her PhD degree in computer science from the university of Waterloo (Canada). Her research interests include collaborative intrusion detection networks, social networks, security issues in mobile networks and medical systems, location-based services for mobile phones, and machine learning in intrusion detection. She is the recipient of the young professional award in IEEE/IFIP IM 2015, Alumni Gold Medal of university of Waterloo in 2013, best dissertation awards in IM2013, the best student paper award in CNSM2011 and the best paper award in IM2009. She received numerous prestige awards and scholarships including Google Anita Borg scholarship, NSERC Postdoc fellowship, David Cheriton Scholarship, NSERC Postgraduate Scholarship, and President's graduate scholarship. She has been a visiting scholar at POSTECH (South Korea), a software engineer intern at Google, and a research intern at BlackBerry.