applied sciences

MDPI

Article

mPrivacy: A Privacy Policy Engine and Safeguard Mechanism in Mobile Devices

Zhong Zhang and Minho Shin *

Department of Computer Engineering, Myongji University, Yongin 17058, Korea; zhangzhong219017@hotmail.com
* Correspondence: mhshin@mju.ac.kr

Abstract: Within the scope of mobile privacy, there are many attack methods that can leak users' private information. The communication between applications can be used to violate permissions and access private information without asking for the user's authorization. Hence, many researchers made protection mechanisms against privilege escalation. However, attackers can further utilize inference algorithms to derive new information out of available data or improve the information quality without violating privilege limits. In this work. we describe the notion of Information Escalation Attack and propose a detection and protection mechanism using Inference Graph and Policy Engine for the user to control their policy on the App's privilege in information escalation. Our implementation results show that the proposed privacy protection service is feasible and provides good useability.

Keywords: mobile privacy; privacy protection mechanism; information escalation; inference algorithm

Citation: Zhang, Z.; Shin, M. mPrivacy: A Privacy Policy Engine and Safeguard Mechanism in Mobile Devices. *Appl. Sci.* **2021**, *11*, 11629. https://doi.org/10.3390/app112411629

Academic Editors: Dohoon Kim and Younkyu Lee

Received: 30 October 2021
Accepted: 2 December 2021
Published: 8 December 2021

1. Introduction

In the past few years, the number of mobile phone has grown rapidly, and the number of mobile device users will continue to grow in the future [1,2]. Along with the number of mobile phone users, the number of malicious applications in mobile devices has also increased [3,4]. Since the system in a personal computer is different from a mobile device, the existing protection mechanism in a personal computer is not compatible with the mobile device. Many kinds of research have been conducted to propose the user's privacy protection mechanisms in mobile devices. Nevertheless, there are still many different ways of attacking the user's private information. This information can be accessed directly through the platform, or indirectly inferred from available resources. Currently, the systematic approach to such attacks is lacking. This limits the development of data protection and secure mobile platforms.

There is a lot of private information on users' mobile devices. One typical important piece of private information is location information. The application can obtain the user's location through the global positioning system (GPS). Although the application's access to GPS information is managed from the mobile system, the application still has other ways to track the user's location. One way is called pedestrian dead reckoning. This is a navigation process that uses only motion sensors. The user's moving speed and distance can be measured using the motion sensors, and thus the current position can be estimated based on the previous position, the user's moving speed, and moving distance [5]. Several researchers have developed accurate pedestrian dead reckoning algorithms to obtain the user's indoor position [6] or outdoor position [7].

The information from motion sensors can improve the location information [8]; moreover, it can even be used to infer the user's password [9,10] or activities [11]. Besides the information from the physical sensors, the information from logical sensors, such as short message service (SMS), call logs, and calendars, could be used to infer various sensitive information, including user emotions [12–14] and social relationships [15].

To protect the user's privacy, the device platform controls the permissions of the application to the resources in the device, and only authorized applications can access the allowed resources. Whether the access control is dynamic or not, attackers can still circumvent the permission mechanism by communicating with other applications with different permission sets. The attackers can use *colluding* [16,17] or *confused deputy* attacks [18,19]. By *colluding*, the attackers can collude with other malicious applications. By *confused deputy*, the attackers can leverage legitimate interactions with a benign application. In the literature, this is called a privilege escalation attack. Furthermore, attackers can increase the accuracy of private information available to themselves by increasing the number of samples from sensors. This *inference attack* [10] leaks more information than is allowed to the attacker, and does not necessarily violate the access permission policy. In order to model all such permissions-based attacks along with inference-based attacks, we define *information escalation attack* as an attack in which the attacker can gain more or higher quality information than they are allowed to possess.

Researchers have made efforts on privacy mechanisms, allowing or prohibiting access when an application violates the privacy policy. However, we believe that privacy is not an issue of access control, but more an information control issue. To compromise privacy and utility, privacy policies that intentionally limit the amount or quality of information disclosed to third parties could provide benefits to both the users and service providers. This kind of privacy policy could be better than inflexible access control policies. In order to develop this privacy mechanism, we systematically designed privacy policies and execution mechanisms for information escalation attacks.

In this paper, our contribution can be summarized as follows:

- Defined a generic privacy attack model called information escalation attack.
- Proposed a novel privacy-policy model based on the inference graph.
- We designed our privacy-policy engine based on Android systems.
- We designed and implemented one location escalation attack.
- We tested our system with a designed location escalation attack.

The rest of this work is structured as follows: Section 2 is about related works. Section 3 describes the privacy problem in the mobile device and introduces the concept of information escalation attack. Section 4 explains the concept of our approach, the system design, and the privacy policy. In Section 5, the implementation and experiment results are shown. Section 6 is about discussion and future work. Finally, in Section 7 are the conclusions.

2. Related Works

There are various inference algorithms, which could infer sensitive information. These algorithms could be used to violate the user's privacy. Various privacy-preserving mechanisms have been made to protect the user's private information. In this chapter a few of the inference algorithms and the privacy-preserving mechanisms are mentioned.

2.1. Inference Algorithms

One type of sensitive information is location information. Many inference algorithms can infer the user's location or location traces. The attacker can further infer other kinds of sensitive information, using location information. We collected various location inference algorithms.

In [8], the authors used the accelerometer, gyroscope, GPS, and map information to obtain more accurate location information. The data from the accelerometer are used to infer the step distance, count of step, and the angularity of turn. The data from the gyroscope are used to detect whether the user has made a turn or not from time to time. Combining the data from the accelerometer and gyroscope, the algorithm can gain the turn information with accurate turn degree and relative direction. In the end, the algorithm performs map matching, using all previously inferred information, the location from the

GPS as a start point, and external map. The accuracy of the GPS in their experiments is 15 m. Their algorithm is able to obtain the location in 5 m.

A train route tracking inference algorithm is introduced in [20]. To infer the user's train routes information, the authors used data from the accelerometer, linear accelerometer, gyroscope, magnet, supervised machine learning classifier, and external train routes map as input to the inference algorithm. Linear accelerometer data can be generated by excluding the gravity information from the accelerometer data. With a pre-generated classifier and data from the previously mentioned sensors, the algorithm detects all the time whether the user is walking, using the train, or in other motion modes. Then, the departure and arrival time information is detected, using the motion and time information. Comparing the departure and arrival time information with the train route map, which includes the timetable of all train routes, the algorithm can predict the user's train routes.

Using the motion sensors, especially the accelerometer, location-related information can be inferred. A few other algorithms were made to gain user location information using motion sensors in [5–7,21,22].

Previous location-related information is revealed via physical sensors and external information. Some other privacy-related information can also be inferred using physical sensors or logical sensors.

In [23] the authors developed a keystrokes-of-keyboards snooping algorithm. If the mobile phone is located near the keyboard, the inference algorithm can detect the keystrokes. To do this inference, data from two microphones in the phone are used at first to inference the relative position of the phone. Because the arrival time of the keystrokes signal to two different microphones is varied, the keystrokes can be distributed into different key groups. After analyzing a few keystrokes, the algorithm can guess theoretical key groups based on the position of the phone for the next step. Using the theoretical key groups and keystrokes signals, the keystrokes are distributed into groups. For example, "e, d, x", and "r, f, c" are groups for which the arrival time of keystrokes signal can be applied to the same mathematical curve. Acoustic features of keystroke sounds in the same key groups are different. The algorithm uses the statistical Mel-frequency cepstral coefficients (MFCC) features of keystrokes to predict which key is the keystroke in the key group. In the end, the algorithm can inference every keystroke by combining the key groups' information and the key position information in the group.

In [12], one algorithm for happiness recognition using a few logical sensors and Bluetooth is introduced. The author made use of the relationship of the user's emotion to the call log, SMS, Bluetooth, weather condition, and personality traits. A pre-generated classifier based on machine learning is used to predict the user's emotion. The classifier distributes the emotion into different classes with various regularity of the phone call, usage of SMS, and Bluetooth. The weather condition and the user's personality traits help to filter the classifier so that the algorithm can provide better accuracy.

2.2. Privacy-Preserving Mechanisms

Chakraborty, et al. made "ipShield" [24], a framework that provides control of sensors in phone at runtime to make a risk prediction for the user's privacy. The user can obtain a list of inferences using different sensors and choose the allowed inferences. The system returns recommendations of the sensors management and invites the user to make the decision. They collect and store the accuracy of the inference algorithms in the database. However, they only analyzed the raw data from sensors and did not consider that the information coming from the inference process can also be used to infer the information.

Sikder, et al. made a context-aware sensor-based attack detection system called "6thsense" [25]. This system observes the change sensors' data and compares the sensors' behavior with the trained data upon to user's different activities. With this method, they can distinguish the application's normal behavior from malicious behavior. They only mentioned the physical sensors and did not provide the control of the data quality.

In the same year, Liu, et al. published their work "DEEProtect" [26]. They collected an amount of inferences algorithms of private information that use sensors' data. Based on these inference models, the system makes an analysis between useful and sensitive information inference ability of the application. They obfuscated the data from sensors to reduce the risk for sensitive information, and in exchange, degraded the utility of applications. This work did not consider the different capabilities of different inference processes for the same information.

"Taintdroid" [27] provides a way to control the data flow. The suspicious data are marked with "taint". By monitoring the flow of marked data, the system can detect the colluding between the applications. If the application attempts to transfer the "taint" marked data outside the mobile device, this application might be malicious. This work was made to prevent privilege escalation attacks.

In "T2Droid" [28], Yalew, et al. monitored application programming interface (API) calls and system calls made from applications to detect malicious applications. They monitored sensitive calls so that they could monitor the behavior of the applications. To trace API calls, they used "Xposed", and to trace the system calls, they used "strace".

In our previous work [29], we monitored the API calls from applications and made policies for each application. The user can set the policy and privacy level corresponding to different information's quality. In this work, we extended our previous work [30]. We implemented our system with privacy policies and information escalation graph to prevent the information escalation attack.

3. Problem Formulation

There are various mobile devices, such as smartphones, laptops, and tablets. Such mobile devices share the same system model and attack model to a large extent. In this section, we describe the system model and attack model in detail. Our approach can be applied to any mobile device under the same model.

3.1. System Model

Figure 1 shows the system model of the mobile device. The system consists of the application layer, platform layer, and physical sensor layer. Users can install and interact with third-party applications at the application layer. The third-party application may come from an attacker. The platform layer is produced by the platform provider. The platform provides an operating system (OS) and services that enable third-party applications to access physical sensors (such as accelerometers, microphones, and gyroscopes) and logical sensors (such as address books, calendars, and text messages). Physical sensors and logical sensors may directly expose users' private information. For example, a third-party application can use GPS sensors to learn about the user's location or learn about the user's activities from schedules in the calendar. In particular, the information of the logical sensor is usually personal information. The attacker could use the network interface as a route of information from external sources. Access to the network interface can expose contextual information of the user. For example, the IP address or Wi-Fi environment can reveal the location of the user. The density of access points around the user can reveal what kind of areas the user is located in. For these reasons, we also consider the network interface as a physical sensor.

The information that the application can use may come from the mobile device itself, or it may come from external resources, such as cloud servers, other mobile devices, ambient sensors, and internet of things (IoT) devices. Although information from external sources may not directly expose personal information, it can still expose users' contextual information, such as location, physical environment, activities, and interests. In this work, attacks related to external sources are out of scope.

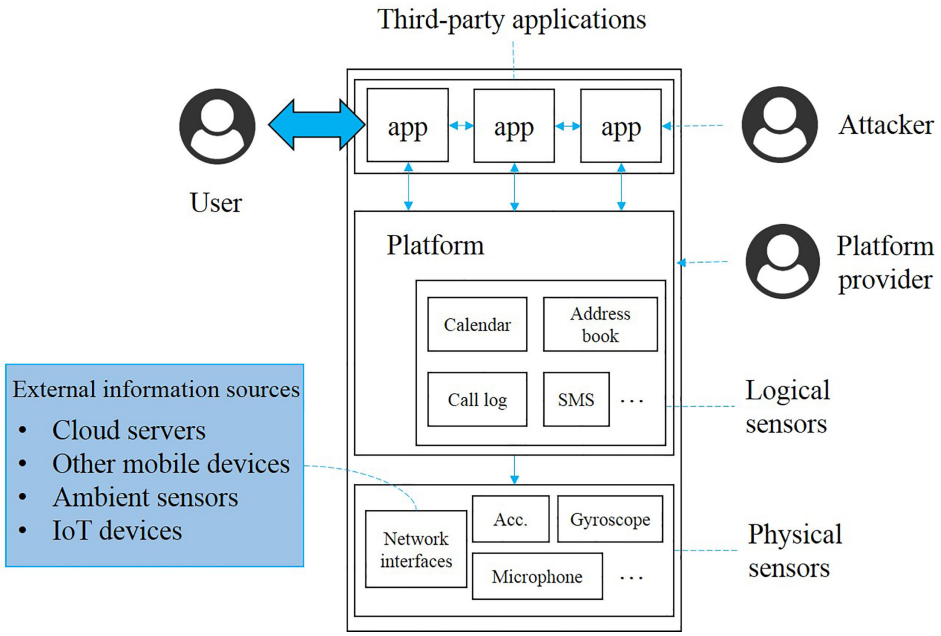


Figure 1. System model.

Recent mobile platforms have provided static (for example, Android, version $\leqslant 5$) or dynamic (for example, Android, version $\geqslant 6$) [31] permission control mechanisms. In Android 11, the latest Android version at the time of writing, there are some updates for privacy such as "one-time permissions" and "permissions auto-reset" [32]. Whether static or dynamic, the fundamental Android approach to privacy problems remains the same; it only tries to control the application's access to resources with a binary decision (allow or disallow). The decision comes only from the user. Therefore, from the information escalation attack's point of view, both mechanisms employed in Android are not effective. On the contrary, our approach overcomes the limitations of such privilege-based protections to defend against the more sophisticated attackers who focus on information rather than privilege. Therefore, we believe that the limitation of our experiments with old versions of Android does not affect the validity of our approach.

3.2. Adversary Model

The attacker may be a malicious developer or an infected application provider. They can all provide malicious applications to users.

Attackers may try to access the user's private information, such as location, contact list, and password, without permission. The information obtained may be sold to the data requester or published. In order to access the user's private information, an attacker can develop a malicious application and upload it to the application store. Users may install malicious applications without notice. The attacker may also send an email to the victim, which contains a download link of the malicious application. Attackers can also develop malicious websites and force users to download malicious applications with or without notice.

In our work, we trust the platform provider, who implemented the mobile platform and the system services and applications, such as contacts application or location services. This assumption allows us to focus on third-party application developers as potential adversaries and limit the attack surface to the application layer.

3.3. Attack Model

Although the mobile device's system restricts the application's access to information sources, the user's private information may still be leaked without the user's authorization.

In this case, many studies have discussed privilege escalation attacks. However, we found that escalation is not limited to permissions, as information can also be escalated.

Information escalation attacks are when an attacker accesses more information without permission or obtains information with higher accuracy than allowed. Privilege escalation is part of information escalation. If privilege escalation occurs, the user's private information is leaked. Therefore, this concept is always within the scope of information escalation.

- **Privilege escalation:** This type of attack is also known as a permission escalation attack. Malicious applications are designed to access protected resources in mobile devices. Through this attack, malicious applications can gain more permissions than expected and leak sensitive information without user authorization [19]. Privilege escalation can be achieved by the confused deputy or colluding [33]. The original sensitive information can be leaked without going through the inference process, or it can be used to infer other information.
- **Non-privilege information escalation:** When information escalation occurs, the attacker may not need to violate any permissions. The information that the attacker wants to obtain may be information that is not included in the access control, or it may be information of higher quality. Non-privilege information escalation can also be accomplished by the confused deputy and colluding. Sensitive information can be leaked directly without inference, or it can be leaked indirectly through inference. In the following text, we explain some examples.

The confused deputy is an attack in which an attacker can control malicious and infected applications to violate permissions or access resources. For example, an attacker can let the system browser download files or use the system scripting environment to send text messages without user authorization [33]. In the case of a non-privileged information attack, application A has the user's health status information, while application B does not. If application B uses the vulnerability of A to obtain health status information, this attack is a non-privilege information attack by confused deputy. The system has no permission control over the user's health status information, so this situation is not a privilege escalation attack.

A colluding attack is when a group of malicious applications cooperate to access more resources than one of them can access. If these applications are developed by the same developer and have the same certificate, they can share their permissions and resources. In other words, if one application in a group of applications can access permission or resource, all other applications in the same group can share its permission or resource [17]. In the example of the confused deputy in the previous paragraph, if A and B are developed by the same attacker, A can send information to B. In this case, A and B collude to conduct a non-privileged information escalation attack.

Information escalation attacks not only mean that the malicious application directly accesses disallowed information from another application, but also that the malicious application infers some other information that is different from the information used as input for the reasoning process, or that enhances the information accuracy. The inference attack is an attack method used by attackers to obtain sensitive information through data analysis. The attacker can access the information without direct permission [34,35]. Combining inference with colluding or confused deputy attacks, attackers can carry out more complex attacks. In the aforementioned example, application B can use the user's health status information and behavior information to infer the user's disease information. This attack is non-privileged information escalation with inference.

To give an example of combining privilege escalation and inference, application A has GPS permissions, and application B is allowed to access the accelerometer and gyroscope with high frequency. The pedestrian dead reckoning algorithm in [8] can infer a more accurate position through the accelerometer, gyroscope, and GPS. The attacker uses A and B colluding or confused deputy so that B can use the information in the GPS. A more accurate position can be inferred using the algorithm mentioned earlier. The location information does not allow B, and its quality is higher than the GPS.

When an attacker tries to obtain sensitive information without inference, we can set up control over all information available to the application to protect the users' privacy. If an attacker uses inference to perform an information upgrade attack, it is difficult to detect. Many pieces of research on inference algorithms are conducted on various information. Therefore, information escalation endangers sensitive pieces of information. We are attempting to propose an approach that can protect the users' privacy in consideration of this attack.

In addition to colluding, confused deputy, and inference, there are other possible privilege escalation attacks. Attackers can use vulnerabilities in the system kernel to carry out privilege escalation attacks [36]. In this work, we do not consider privilege escalation attacks through the vulnerabilities in the platform itself.

3.4. Privacy Policy Decision

Since the attack methods are varied and some are complex for the user to understand, the decision of privacy policy might be difficult for the user. If the attacker attempts to make the privilege escalation, the user can set the permissions of applications to protect this kind of attack. Only some of the privilege escalation can be protected, because the users might not know which permission would endanger their privacy. When the attacker applies non-privileged information escalation, the users might not know which information could be revealed while using the application. The system provides the yes or no control; to use the applications smoothly, the users might ignore the permission control and just allow all permissions. The privacy policy should be designed to help the user to make privacy policy decisions more easily.

4. Approach

Our method is based on the *inference escalation graph*, which illustrates the inference relationship between different types of information. We dynamically monitor the data access of each application and check it according to the information escalation graph to identify the information that may be inferred and its corresponding quality. We compare the accessible information and its quality with the privacy policy set by the user to determine access allowance. When the access violates the policy, we either disallow the access or manipulate the return values to satisfy the privacy policy.

4.1. Definitions of Information Escalation Graph

The information escalation graph (IEG) is a directed graph composed of three types of nodes and edges. A node represents a type of information or information processing type. On the other hand, directed edges represent the inference direction from raw data or less abstract information to more abstract or processed information. The three types of nodes and edges are described in detail as follows.

- **Source node:** A source node represents a physical or logical sensor represented by a rectangle. A source node provides raw data to other types of nodes.
- **Process node:** A process node represents an implementation of inference algorithms that take multiple inputs and generate higher-level information. This kind of inference includes simple arithmetic calculations on more complex machine learning algorithms. A process node is represented by a diamond.
- **Information node:** An information node represents all types of information, except raw data. This includes simple combinations of raw data or significant abstractions of raw data. An information node is represented by an oval.
- **Edges:** An edge represents the direction of inference from lower-level to higher-level information. An edge is represented by an arrow.

4.2. Graph Based Approach

Figure 2 is an example of an information escalation graph of three position trajectory inference algorithms using different combinations of accelerometer, magnet, and gyroscope.

The inference algorithm uses various processes to infer a variety of lower-level information, which is expected to infer the position trajectory. There are five processes to infer various information. The information to be inferred in this graph is the moving distance, turning angle, and location trace. All inference processes use time information. Process 1 is used to infer the moving distance using data from the accelerometer [37]. Processes 2, 3, and 4 represent three different pedestrian dead reckoning algorithms to infer the turning angle of the user. Process 2 uses the accelerometer and magnet sensors [21], Process 3 uses magnet and gyroscope sensors [22], and Process 4 uses all three sensors [6]. These three inference processes return the turning angle information in different qualities. Finally, Process 5 uses a location in the GPS as the starting point, the moving distance, and the turning angle indicating the user's moving direction to infer the user's location trajectory.

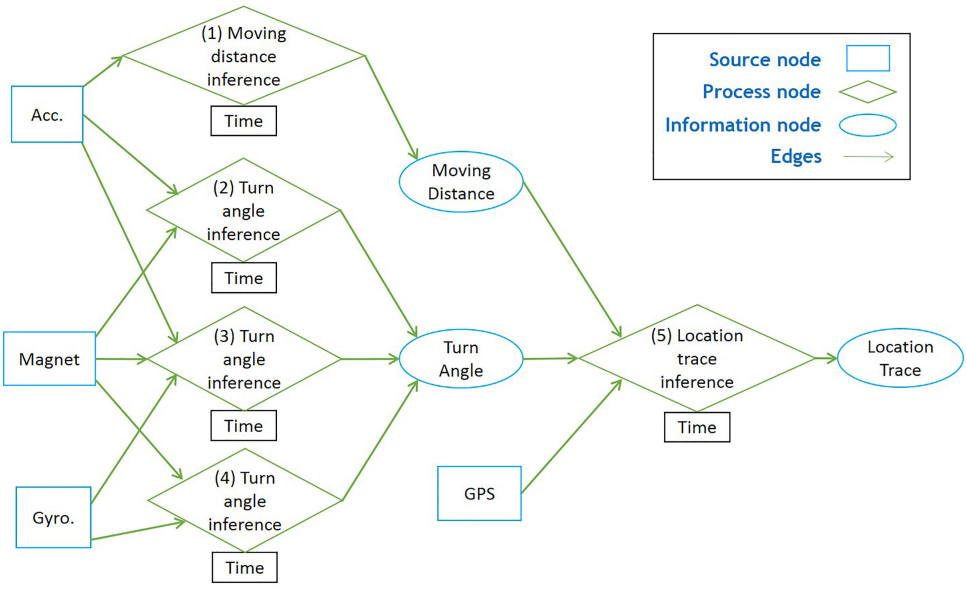


Figure 2. An example of an information escalation graph.

The three pedestrian dead reckoning algorithms in the graph use an accelerometer to infer the moving distance. The difference between the three algorithms is that they use different processes to infer the turning angle. Algorithm 1 uses an accelerometer and magnet, Algorithm 3 uses a magnet and gyroscope, and Algorithm 2 uses all three sensors. The three inference processes will return turning angle information of different qualities. The quality of the turning angle will affect the quality of the location trace in inference Process 5. Assume that the accuracies provided by the inference Processes 2, 3, and 4 are 80%, 90%, and 50%, and the corresponding inferred location trace accuracies are medium, high, and low. When the user allows the application to obtain the mid-level location information, the application should be prohibited from using the gyroscope. If the user chooses to allow a high level, the application is allowed to use all three sensors. However, if the user wants to select a low level, in this case, there seems to be no solution. Algorithm 1 and Algorithm 3 both require all three sensors to infer the location trace. Algorithm 1 can infer a more accurate location trace than Algorithm 3. If we disable the accelerometer or magnet to destroy the inference of Algorithm 1, then Algorithm 3 will also be destroyed. We cannot Allow algorithm 3 while disabling Algorithm 1. In this case, we can break Algorithm 1 by adjusting the frequency of the sensor while maintaining Algorithm 3. Another similar situation is that the application requires all three sensors to work properly. Our approach can maintain the utility of the application while protecting users' privacy.

From this example, we can see how to use graphs to implement privacy policies. The implementation of the policy enforcement is based on some predefined logical rules.

4.3. System Architecture

In our system, the policy manager realizes the user's privacy decision into the privacy policy. There is also a privacy service and policy database for further functions.

Figure 3 is an overview of our system architecture. From the entities' point of view, the policy manager interacts with the user at the application layer. The privacy service monitors and controls the API usages of the application. The data in the policy database are used to help the privacy service implement policy enforcement. From the perspective of the workflow, first, the privacy service monitors and collects the API usages of each application. The privacy service can obtain the information that the application wants to access from the usages of the API. With the help of the IEG in the policy database, the privacy service learns the set of accessible or inferable information from the set of resources that the application is allowed to use. When an application attempts to access certain resources that result in access to prohibited information, the privacy service will block or allow API calls. In some cases, it can also manipulate the results of API calls to ensure compliance with privacy policies.

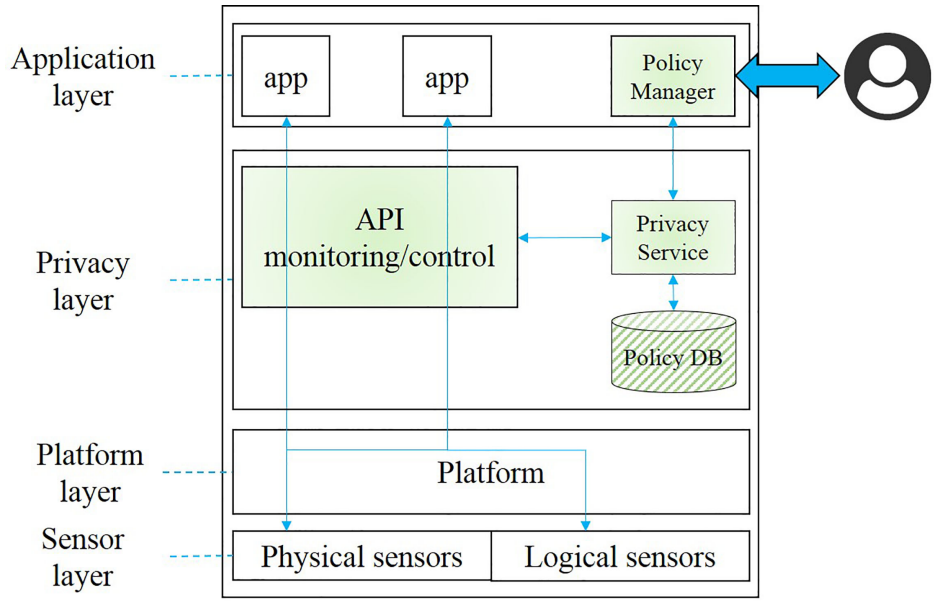


Figure 3. System architecture.

For example, there is an application that allows access to the accelerometer, magnet, gyroscope, and GPS. As mentioned in Section 3.3, the app can use pedestrian dead reckoning to infer positions with higher accuracy. Our approach detects resource usage from the application, predicts that the higher accuracy location can be revealed, and executes the corresponding policy enforcement.

Colluding attacks are also easily detected by our mechanism as we monitor which information becomes available to each application, as far as the information exchange between malicious applications is over standard mechanisms, such as intents or binder. However, if those applications use the covert channel that cannot be detected by monitoring the API calls, they cannot be detected. The covert channel-based attacks are difficult to prevent and there are proposals in the literature.

Since our system design is applied to Android systems, for the API monitoring and the data flow detection that we need for this work, we refer to API monitoring using "Cydia Substrate" [38], "Xposed" [39], "TaintDroid" [27], and our previous work [29].

4.4. Privacy Policy

In our application, the user can add one of the policies for different information (e.g., location, and photo) themselves by interacting with the policy manager. The policy

manager informs the privacy service to add or update the privacy policy in the policy database. One privacy policy consists of the time of the monitoring, target application, information to protect, and protection method. The user can set the target application, information to protect, and protection method; the time of monitoring is all the time. Before the user adds the privacy policy, the privacy service takes the information that the applications can gain as input into the IEG, the IEG performs the path searches, and gives the information that might be inferred as output. The privacy service informs the policy manager of the endangered information with the corresponding application. When the user adds the privacy policy, the warning message of the endangered information with the corresponding application is shown to the user.

To make it easier for the user to make privacy policy decisions, we propose four protection methods.

- **Allow:** The application is allowed to use the original information.
- **Deny:** The application is forbidden to use any information.
- **User confirmation:** Whether the application is allowed to use the information or not depends on the user's decision.
- **Information manipulation:** Application is allowed to use the manipulated version from the information.

Choosing to allow, the target application is allowed to use the information. Choosing to deny, the target application receives no information. When the user chooses the user confirmation, our application monitors the operation of the target application based on the privacy policy stored in the policy database. If the target application attempts to obtain the information, which is under monitoring, the privacy service sends an alarm message to the user.

5. Implementation and Evaluation

We made our own information escalation attack to test our approach. In this section, the attack is explained in detail, and the evaluation metric and the environmental setup are introduced.

5.1. Attack Example

To show information escalation attack and test our protection mechanism, we made an information escalation attack on location. Many researchers made k-anonymity algorithms to protect the user's location privacy [40]. The user's location information is provided as a cloaking region. Our attack aims to escalate the cloaking region using access point (AP) information.

Figure 4 shows the IEG of this attack. The wireless network has a certain reachable region and unique identification information, which can reveal the geographic information of the network. We define this external information in the graph as AP coverage information. In practice, the AP coverage information can be manually collected as [41]. It is used together with the wireless network connection information from the user's mobile device to detect the AP region, where the user is staying. Then the AP region and the cloaking regions can be drawn on the same map to detect the overlapping region, and hence, we can gain the smaller region. Both AP region detection and overlapping detection processes are kept running from time to time.

To apply this attack, we have one mobile device, which provides a location blurring service for privacy and one malicious application that can access wireless network information. We pre-define the AP coverage information for the malicious application to utilize. The attack has two cases depending on the contents of the AP coverage information.

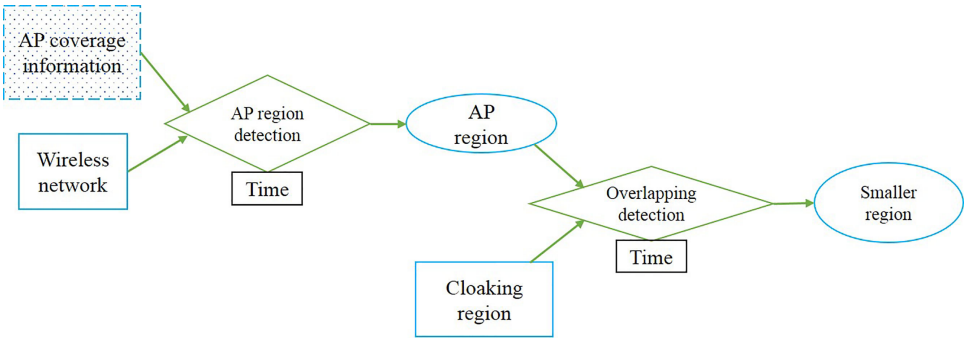


Figure 4. Location Attack IEG.

The Wi-Fi signal has various signal strengths in different regions. As in Figure 5, the Wi-Fi signal is full in a circle with a radius of 10 m. The Wi-Fi reachable region is a circle with a radius of 30 m. The overlapping region in the left part of the figure is $\leqslant 21.71$ square meters. In the right figure is the overlapping region $\leqslant 740.92$ square meters. The rates of escalated regions compared to the cloaking regions are $21.71/2827.43 \times 1000 \approx 7.68$ and $740.92/2827.43 \times 1000 \approx 262.05$. We can escalate the cloaking region into a relatively small region with the full signal strength region, but with the weak signal region, a rough region.

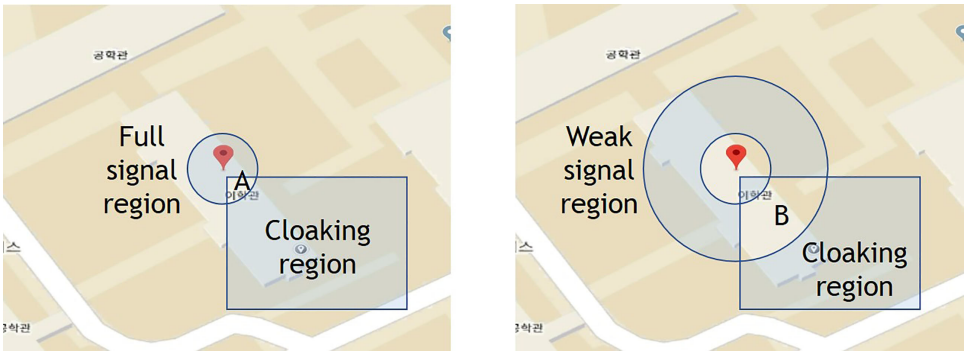


Figure 5. Location attack using AP region with various signal strength.

5.2. Evaluation Metric

To evaluate our protection mechanism against the information escalation attack with our attack program, we collected the size of the revealed region resulting from the overlapping of the AP coverage region and the cloaking region. We tested various sizes of the cloaking region. To show whether the modification of the Wi-Fi signal is beneficial to the user, we compared the size of overlapping regions before and after the modification of the Wi-Fi signal.

The API monitoring/control and the IEG analyzing phase may cause the system delay. We evaluated the runtime with various numbers of hooks, input data for IEG, and various sizes of IEG to analyze the system latency.

5.3. Experimental Setup

The protection mechanism is written in Java and can run on mobile devices with the Android system. A virtual device with Android version 4.2.2 was used for the experiment. In practice, we found that the application can gain the location with GPS or network provider; however, the GPS provider is not always available and the location from the network provider has low accuracy. In our experiment, we set one accurate location for GPS with a virtual machine manually. We assumed that the system uses the cloaking region to protect the user's privacy, and there is one service that generates the cloaking region corresponding to the user's accurate location.

As shown in Figure 6, our program detects that the "maliciouswifiscanner" could violate the user's location and thus shows the notice to the user. The user can select the

target application, information to protect, and the protection method to add one privacy policy. We currently implemented the protection mechanism for the photo and location information. In the following experiments, "maliciouswifiscanner" in the target application is chosen, location information in the information to protect is chosen, and information manipulation in the protection method is chosen. The protection methods are explained in Section 4.4. Once the policy is added, our program monitors the target application all the time and applies the policy.

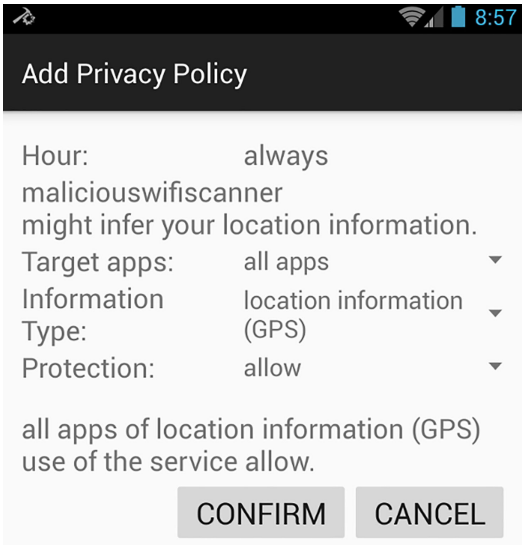


Figure 6. Add privacy policy.

As shown in Figure 7, when the application "maliciouswifiscanner" attempts to obtain the location information, our application sends the user the alarm message. The user can choose to allow or deny the usage of the information. If the user wants to allow the target application to use the information and at the same time concerned about their privacy, by information manipulation, our application accordingly modifies the information and allows the target application to use the modified information.

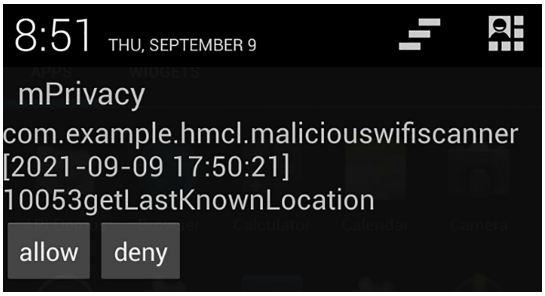
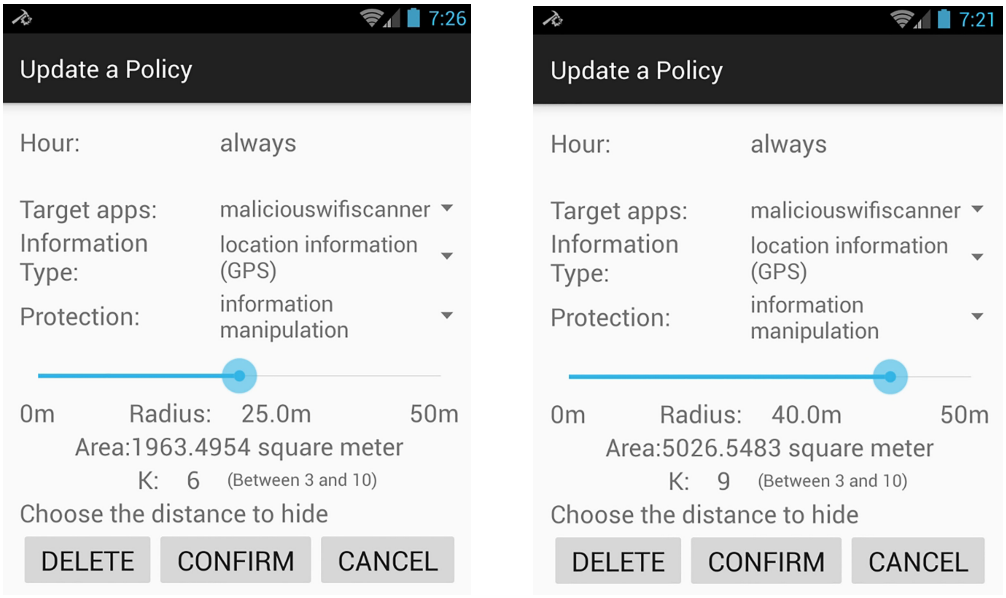


Figure 7. Alarm message for user confirmation.

The user can choose to apply different levels of privacy. As shown in Figure 8, the user can set the privacy level of the location with the size of radius for the location. The value of radius is from 0 m to 50 m. Here we have one example of privacy setting using 25 m as the radius in Figure 8a. In Figure 8b, the setting example has a relatively higher privacy level with a 40 m radius. The target application is only allowed to use the location information with accuracy below the privacy policy. Our program checks whether the location that the application can access violates the policy or not, and makes corresponding policy enforcement. In the case of our attack, the size of cloaking region and the privacy policy setting are compared, and the IEG of the Wi-Fi location attack is used to protect against the information escalation attack. To simplify the calculation of the overlapping

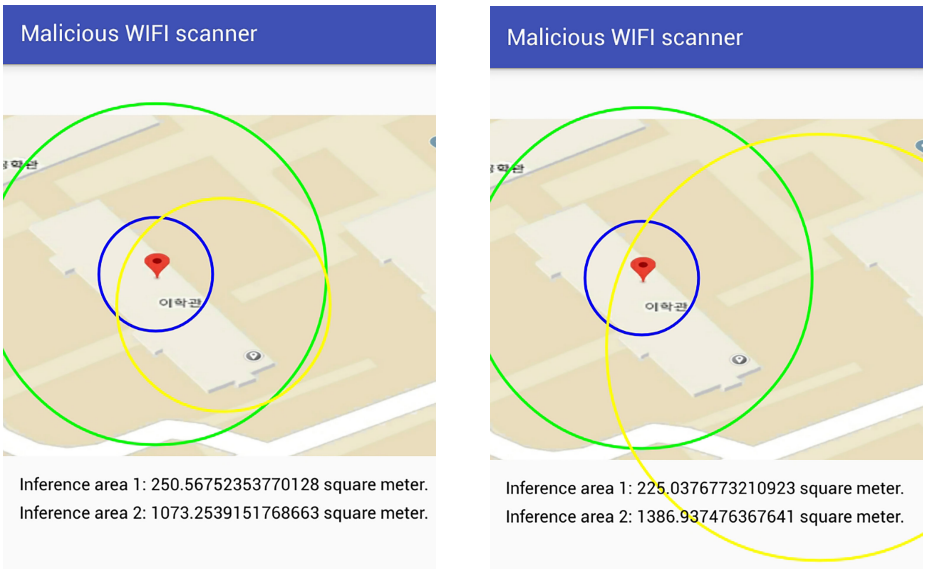
region, we assume that the shape of the cloaking region is a circle. We pre-define the AP coverage region as a circle. The overlapping region of two circles is easy to calculate.



(**a**) (**b**)

Figure 8. Setting of privacy for location: (**a**) setting of low privacy, (**b**) setting of high privacy.

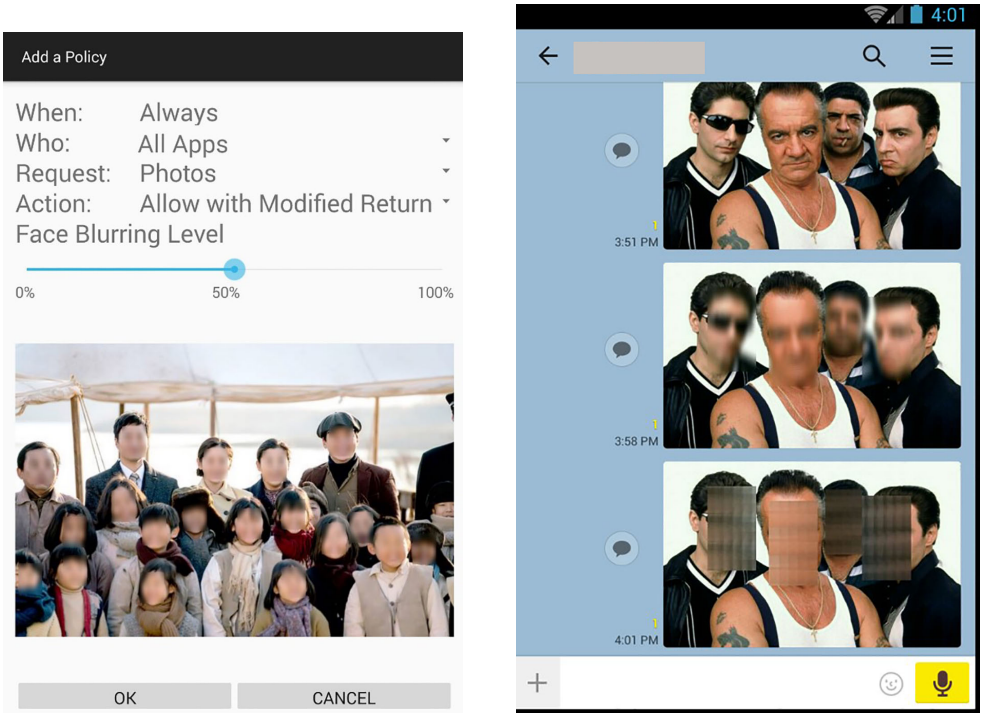
The results are shown in Figure 9. The malicious application has two AP coverage information as shown in different sizes of circles.The blue circle is AP coverage with the high signal strength. The green circle shows the range of AP coverage with the weak signal strength. The third circle in yellow is the cloaking region. The "inference area 1" and "inference area 2" are the two overlapping areas of the two AP coverage circles and the cloaking region. In Figure 9a, the setting of the privacy level is relatively low, and the inferred areas are smaller than the inferred areas in Figure 9b. The bigger the inferred area, the better the user's privacy is protected.



(**a**) (**b**)

Figure 9. Location inference results: (**a**) location inference result with low privacy, (**b**) location inference result with high privacy.

We also implemented photo manipulation. In Figure 10, the left side shows the privacy setting. The users can choose to blur the faces in the photo when they want to share the photos. There are two intensities for the users to choose from. The results are on the right side of the figure. In Figure 10b, the first photo on the top is the original photo, the photo below is the picture blurred with 50% intensity, and the last picture is blurred with 100% intensity. The faces in the second picture are slightly blurred. We can still see the face contours. The faces are fully blurred in the last picture.



(**a**) (**b**)

Figure 10. Photo manipulation: (**a**) setting of privacy for photo, (**b**) face blur results.

5.4. Evaluation Results

We collected the value of overlapping regions with the size of cloaking regions.

Figure 11 shows the relation between the revealed area and the different sizes of the cloaking region. The x-axis is the cloaking region size in meters squared. The y-axis is the revealed area in meters squared. The line of "no attack" shows how many areas from the original size of cloaking region is revealed. The line marked with triangles shows how many areas from the original size of cloaking region is revealed with protection from our program. The line of "privacy policy area" shows the user's privacy policy setting. The user's privacy is violated if the revealed area is below the line of "privacy policy area". If the information escalation happens using our attack, the line marked with diamonds shows that the revealed area is always under the line of the privacy policy area. Using our protection mechanism, which detects the information escalation attack with IEG and modifies the Wi-Fi signal, the revealed area can be enhanced. We can see the values after modification are higher than the requirements of the user's privacy policy settings.

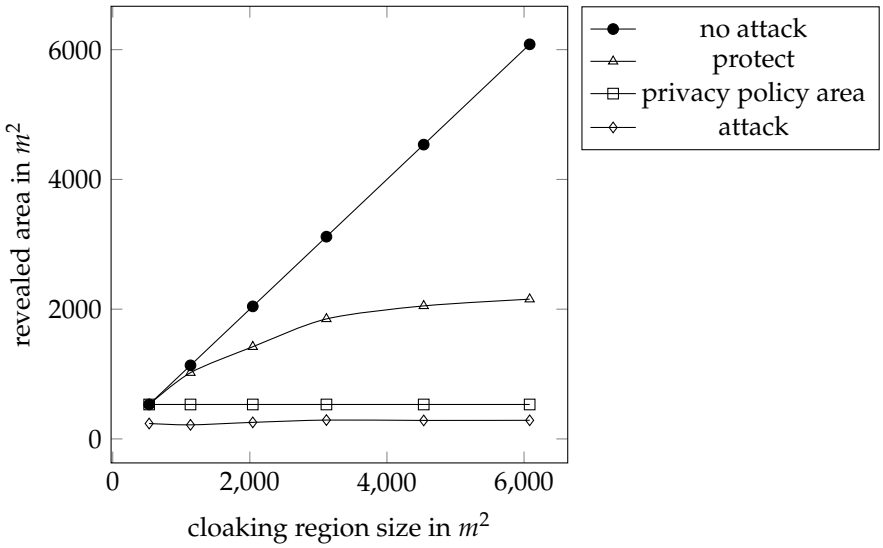


Figure 11. Experiments with different cloaking region size.

Figure 12 shows the relation between the revealed area and the different privacy policy settings. The x-axis is the privacy policy area in meters squared. The y-axis is the revealed area in meters squared. If there is no information escalation attack, our protection mechanism allows the application to use the cloaking region as shown in the line of "no attack", which is larger than the user's privacy policy setting marked with squares. With the information escalation attack, the user's privacy is violated as shown with the line of "Attack". Using our protection mechanism, the revealed area can be enlarged until they are above the line of "Privacy Policy Area".

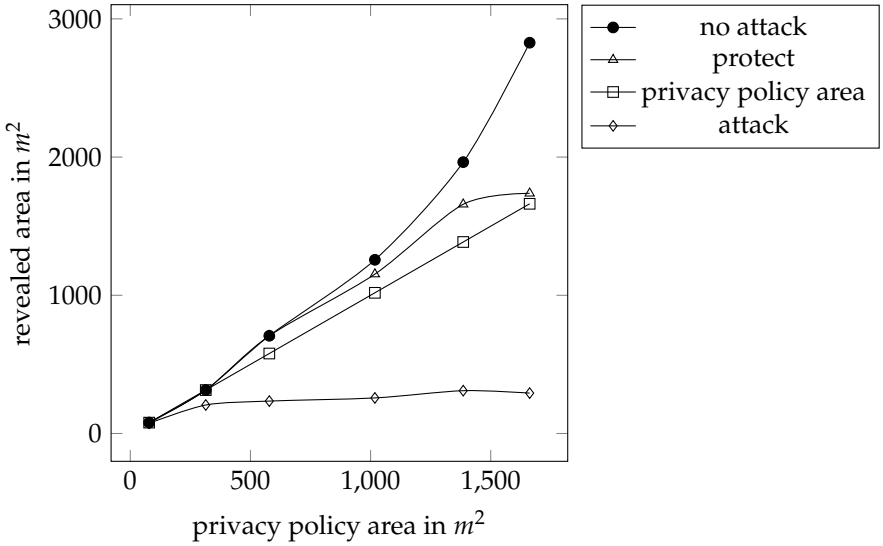


Figure 12. Experiments with different privacy policy area.

We did not experiment with the performance and scalability of the proposed privacy protection mechanism. We expect that the performance overhead due to the Xposed hooking mechanism is not negligible because it was done in the Java implementation part of the system. However, it is expected that the performance of the mechanism can be optimized when the mechanism is actually implemented natively into the Android system by the platform/device manufacturer (e.g., Google or Samsung).

We tested the runtime of graph search with two different sizes with Android UnitTest. The first one has 6 nodes and the time for graph search is 1 ms. The second one has 16 nodes and the corresponding time for graph search is 2 ms. The average searching time

for each node is about 0.14 ms. To measure the efficiency and scalability of our IEG, we will further test the runtime using a large size of the graph in the future.

6. Discussion and Future Work

We have two implementations for different Android versions. We used "Cydia Substrate" and "Xposed" for the APIs hooking. Because "Cydia Substrate" is only supported on Andriod versions 2.3 through 4.3, we made further implementation using "Xposed" for the later Android versions. The implementation with "Xposed" is not done, so we only tested our implementation with "Cydia Substrate". In the future, we will further develop our system with "Xposed" and make more experiments in operation. We will integrate the idea from "TaintDorid" into IEG. The information under monitoring and control could be marked using tags and given to the IEG. For now, we create the IEG manually with collected data. In the future, we can try to use a machine learning algorithm to build our IEG.

7. Conclusions

In this work, we explained information escalation with examples. We proposed a protection mechanism based on IEG; our system enables the user to control private information while considering the quality of information. The privacy policy can be applied to physical sensors, logical sensors, and other information that we can monitor and control. The system design is explained with pictures in detail. Our privacy protection mechanism was tested with our location escalation attack along with different values of cloaking regions and privacy policy settings. The experiment results show that our system can protect the user's privacy while keeping the information quality according to the privacy policy setting.

Author Contributions: Conceptualization, M.S. and Z.Z.; methodology, M.S.; software, Z.Z.; validation, Z.Z.; investigation, Z.Z.; resources, M.S.; writing—original draft preparation, Z.Z.; writing—review and editing, M.S.; visualization, Z.Z.; supervision, M.S.; project administration, M.S.; funding acquisition, M.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1F1A1055324). ※ MSIT: Ministry of Science and ICT.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Smartphone Users 2026 | Statista. Available online: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ (accessed on 23 October 2021).
2. Number of Mobile Devices Worldwide 2020–2025 | Statista. Available online: https://www.statista.com/statistics/245501/multiple-mobile-device-ownership-worldwide/ (accessed on 23 October 2021).
3. Global Android Malware Volume 2020 | Statista. Available online: https://www.statista.com/statistics/680705/global-android-malware-volume/ (accessed on 23 October 2021).
4. McAfee Mobile Threat Report. Available online: https://www.mcafee.com/content/dam/consumer/en-us/docs/2020-Mobile-Threat-Report.pdf (accessed on 23 October 2021).
5. Beauregard, S.; Harald, H. Pedestrian dead reckoning: A basis for personal positioning. In Proceedings of the 3rd Workshop on Positioning, Navigation and Communication, Hannover, Germany, 11–12 March 2006; pp. 27–35.
6. Kang, W.; Youngnam, H. SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization. *IEEE Sens. J.* **2015**, *15*, 2906–2916. [CrossRef]
7. Mosenia, A.; Dai, X.; Prateek, M.; Niraj, K. PinMe: Tracking a smartphone user around the world. *IEEE Trans. Multi. Scale Comput. Syst.* **2018**, *4*, 420–435. [CrossRef]
8. Zhu, X.; Li, Q.; Chen, G. APT: Accurate outdoor pedestrian tracking with smartphones. In *2013 Proceedings IEEE INFOCOM*; IEEE: Piscataway, NJ, USA, 2013; pp. 2508–2516.
9. Xu, Z.; Kun, B.; Zhu, S. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In Proceedings of the Fifth ACM Conference on Security and Privacy in Wireless and Mobile Networks, Tucson, ZA, USA, 16–18 April 2012; pp. 113–124.

10. Owusu, E.; Han, J.; Das, S.; Perrig, A.; Zhang, J. Accessory: Password inference using accelerometers on smartphones. In Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications, San Diego, CA, USA, 28–29 February 2012; pp. 1–6.
11. Hassan, M.M.; Uddin, M.Z.; Mohamed, A.; Almogren, A. A robust human activity recognition system using smartphone sensors and deep learning. *Future Gener. Comput. Syst.* **2018**, *81*, 307–313. [CrossRef]
12. Bogomolov, A.; Lepri, B.; Pianesi, F. Happiness recognition from mobile phone data. In Proceedings of the 2013 International Conference on Social Computing, Washington, DC, USA, 8–14 September 2013; pp. 790–795.
13. Bogomolov, A.; Lepri, B.; Ferron, M.; Pianesi, F.; Pentland, A. Daily stress recognition from mobile phone data, weather conditions and individual traits. In Proceedings of the 22nd ACM International Conference on Multimedia, New York, NY, USA, 20–24 October 2014; pp. 477–486.
14. LiKamWa, R.; Liu, Y.; Lane, N.D.; Zhong, L. Can your smartphone infer your mood. In *PhoneSense Workshop*; 2011; pp. 1–5. Available online: http://www.yecl.org/publications/likamwa11phonesense.pdf (accessed on 23 October 2021).
15. Devlic, A.; Reichle, R.; Wagner, M.; Pinheiro, M.K.; Vanrompay, Y.; Berbers, Y.; Valla, M.; Context inference of users' social relationships and distributed policy management. In Proceedings of the 2009 IEEE International Conference on Pervasive Computing and Communications, Galveston, TX, USA, 9–13 May 2009; pp. 1–8.
16. Schlegel, R.; Zhang, K.; Zhou, X.Y.; Intwala, M.; Kapadia, A.; Wang, X. Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones. *NDSS* **2011**, *11*, 17–33.
17. Marforio, C.; Aurélien, F.; Srdjan, C. *Application Collusion Attack on the Permission-Based Security Model and Its Implications for Modern Smartphone Systems*; ETH Zurich: Zürich, Switzerland, 2011.
18. Felt, A.P.; Wang, H.J.; Moshchuk, A.; Hanna, S.; Chin, E. Permission Re-Delegation: Attacks and Defenses. *USENIX Secur. Symp.* **2011**, *30*, 88.
19. Davi, L.; Dmitrienko, A.; Sadeghi, A.R.; Winandy, M. Privilege escalation attacks on android. In Proceedings of the International Conference on Information Security, Miyazaki, Japan, 23–25 June 2010; pp. 346–360.
20. Watanabe, T.; Akiyama, M.; Mori, T. Tracking the human mobility using mobile device sensors. *IEICE Trans. Inf. Syst.* **2017**, *100*, 1680–1690. [CrossRef]
21. Nagpal, P.S.; Rashidzadeh, R. Indoor positioning using magnetic compass and accelerometer of smartphones. In Proceedings of the 2013 International Conference on Selected Topics in Mobile and Wireless Networking (MoWNeT), Montreal, QC, Canada, 19–21 August 2013; pp. 140–145.
22. Von Der Hardt, H.-J.; Didier, W.; René, H. The dead reckoning localization system of the wheeled mobile robot ROMANE. In Proceedings of the 1996 IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems (Cat. No. 96TH8242), Washington, DC, USA, 8–11 December 1996; pp. 603–610.
23. Liu, J.; Wang, Y.; Kar, G.; Chen, Y.; Yang, J.; Gruteser, M. Snooping keystrokes with mm-level audio ranging on a single phone. In Proceedings of the 21st Annual International Conference on Mobile Computing and Networking, Paris, France, 7–11 September 2015; pp. 142–154.
24. Chakraborty, S.; Shen, C.; Raghavan, K.R.; Shoukry, Y.; Millar, M.; Srivastava, M. ipshield: A framework for enforcing context-aware privacy. In Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14), Seattle, WA, USA, 2–4 April 2014; pp. 143–156.
25. Sikder, A.K.; Aksu, H.; Uluagac, A.S. 6thsense: A context-aware sensor-based attack detector for smart devices. In Proceedings of the 26th USENIX Security Symposium (USENIX Security 17), Vancouver, BC, Canada, 16–18 August 2017; pp. 397–414.
26. Liu, C.; Chakraborty, S.; Mittal, P. Deeprotect: Enabling inference-based access control on mobile sensing applications. *arXiv* **2017**, arXiv:1702.06159.
27. Enck, W.; Gilbert, P.; Han, S.; Tendulkar, V.; Chun, B.G.; Cox, L.P.; Jung, J.; McDaniel, P.; Sheth, A.N. Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones. *ACM Trans. Comput. Syst.* **2014**, *32*, 1–29. [CrossRef]
28. Yalew, S.D.; Maguire, G.Q.; Haridi, S.; Correia, M. T2Droid: A TrustZone-based dynamic analyser for Android applications. In *2017 IEEE Trustcom/BigDataSE/ICESS*; IEEE: Piscataway, NJ, USA, 2017; pp. 240–247.
29. Shin, M.; Kim, J. Privacy Preserving Watchdog System in Android Systems. In Proceedings of the 2017 International Conference on Platform Technology and Service (PlatCon), Busan, Korea, 13–15 February 2017; pp. 1–5.
30. Zhang, Z.; Yoon, S.; Shin, M. The design of graph-based privacy protection mechanisms for mobile systems. In Proceedings of the 2019 International Conference on Platform Technology and Service (PlatCon), Jeju, Korea, 29–31 January 2019; pp. 1–6.
31. Permissions on Android | Android Developers. Available online: https://developer.android.com/guide/topics/permissions/overview (accessed on 23 October 2021).
32. Privacy in Android 11: Android Developers. Android Developers. Available online: https://developer.android.com/about/versions/11/privacy (accessed on 14 November 2021).
33. Bugiel, S.; Davi, L.; Dmitrienko, A.; Fischer, T.; Sadeghi, A.R. *Xmandroid: A New Android Evolution to Mitigate Privilege Escalation Attacks*; Technical Report TR-2011-04; Technische Universität Darmstadt: Darmstadt, Germany, 2011.
34. Krumm, J. Inference attacks on location tracks. In Proceedings of the International Conference on Pervasive Computing, Toronto, ON, Canada, 13–16 May 2007; pp. 127–143.
35. Shafer, G. Detecting Inference Attacks Using Association Rules. 2001. Available online: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.628.4525 (accessed on 23 October 2021).

36. Shabtai, A.; Fledel, Y.; Kanonov, U.; Elovici, Y.; Dolev, S.; Glezer, C. Google android: A comprehensive security assessment. *IEEE Secur. Priv.* **2010**, *8*, 35–44. [CrossRef]
37. Jahn, J.; Batzer, U.; Seitz, J.; Patino-Studencka, L.; Boronat, J.G. Comparison and evaluation of acceleration based step length estimators for handheld devices. In Proceedings of the 2010 International Conference on Indoor Positioning and Indoor Navigation, Zurich, Switzerland, 15–17 September 2010; pp. 1–6.
38. Cydia Substrate. Available online: http://www.cydiasubstrate.com/ (accessed on 23 October 2021).
39. Xposed Module Repository. Available online: https://repo.xposed.info/ (accessed on 23 October 2021).
40. Talukder, N.; Ahamed, S.I. Preventing multi-query attack in location-based services. In Proceedings of the Third ACM Conference on Wireless Network Security, Hoboken, NJ, USA, 22–24 March 2010; pp. 25–36.
41. Public Wi-Fi Database. Geo-Location API. "Public API of Position by ...". Available online: https://www.mylnikov.org/archives/1170 (accessed on 23 October 2021).