Group 2- Android Malware

Introduction

In this report, we study the behavioral economics of the various actors involved in the security issue. The security issue is the malware presence on the platform caused because of the developers leaving vulnerabilities in their apps. We consider three actors in this report: users, platform owners and app developers. We first discuss a countermeasure for each of the actors involved, analyze the costs and benefits of each of the strategies, analyze if there are any incentives for the actors to actually implement those strategies. Then we take a look at the externalities caused by these strategies. Finally, we revisit the two metrics we designed in assignment block 2 and try to explain the variance in them using one external factor -- app size.

Dataset

The given dataset contains information regarding games and software that has been uploaded on Baidu and 360, two of the most popular android application stores in China. For every app, it contains information regarding its category, software version, size, software package, number of downloads, download URL, last update date, and download date.

Actors Involved

- Users (Problem owners),
- Platform owners,
- App developers

Countermeasures

Users - risk avoidance

Users can mitigate the risk of infecting their phones with malicious apps by migrating to another platform. We assume here that the users are security aware so they will migrate to another platform if they get an infection from the one they are currently using. However, the platform they migrate to might or might not be safer than the current one – there will only be less perceived risk.

Platform owners - deploying a scanner

Platform owners can deploy a scanner on their platform that checks every app that the developers upload for any vulnerabilities by comparing from publically known issues (e.g. via CVE database[1]). It can also check the customers' feedback that might indicate early signs of malware infections.

Developers - secure development and patching their apps

Developers can have a proactive approach of following a secure development strategy and then keep their apps up to date to reduce the risk of having vulnerabilities in their apps which can be exploited and cause various problems for the users. This strategy might not totally eliminate the chance of apps getting exploited but it would significantly reduce the security issue.

Cost/Benefit analysis

<table>
<tr><th colspan="2">Users</th></tr>
<tr><td>(Cons) Costs:
Getting technically locked-in if not enough platforms to migrate to.
Getting used to the new platform and its regulations, causing loss of productivity.
Some commonly used apps are not available on the other platform.
Specific benefits (discounts) that the old platform had, will not be available anymore.</td><td>(Pros) Benefits:
There is less perceived risk on the new platform.
If the user is not security sensitive or is not able to find a safer platform (in terms of risk) then there is no benefit to implement this countermeasure.</td></tr>
<tr><td colspan="2">Altogether there will be some transferring issues (switching costs) at the beginning. But at the end the user should be safer and feel more comfortable for switching a platform, assuming that the user is security sensitive.</td></tr>
</table>

<table>
<tr><th colspan="2">Platform owners</th></tr>
<tr><td>(Cons) Costs:
It costs money to deploy the scanner, train employees and then maintaining it.
Keeping the scanner up to date, so that it recognizes new malware samples.</td><td>(Pros) Benefits:
None, if the users are not security sensitive.</td></tr>
</table>

[1] http://www.cvedetails.com/

Altogether the platform owner will create a safer platform, which will result in a higher overall operational cost. This cost will only be compensated for if the users are actually security sensitive. Otherwise, it ends up creating an externality.

Developers	
(Cons) Costs: - Regular patching is expensive, especially after deployment of a software. Bugs found in the field cost 50-200 times more than the ones fixed earlier on [3,7]. - Cost of secure development is higher as it involves more planning and risk assessment even before the software is built [8]. - Constantly observing their app in case of a security breach; this monitoring gives the developer extra costs [9].	(Pros) Benefits: - Proactive patching results in less future exploits. - Having a secure and protected app gives a good reputation to the developer in terms of positive reviews which could result in more downloads, especially if the users are security aware.
For the developers, there is a lot of extra work if they want to keep their apps up to date and have to constantly patch every security issue. This extra cost will only be compensated for if the users actually care enough to switch to an alternative in case an infection occurs.	

Incentives

Users

The users have the biggest incentive to take the countermeasure because they are the ones directly getting affected by the malware. As mentioned previously, switching does not ensure less malware infections but it does reduce the perceived risk of infections. Moreover, only the users who are security sensitive will consider migrating. For other users, it might be too much hassle for no perceived gains.

Platform Owners

The incentive behind the platform owners is not easy to analyze. At first glance, one could say that platform owners do not have an incentive to clean up the malware on their platform because it does not

affect them directly, unless the law enforces to act upon such incidents. On the other hand, one can assume that the platform owners will be indirectly affected by malicious apps because of the loss of their reputation. However, in practice, even if malicious apps are discovered on a famous app store, it is very unlikely that the users will migrate on a different platform and as a result the platform will not be harmed [6]. In short, whether there is an incentive for the platform owners depends on the security sensitivity of the users. If the users prefer platforms that provide security, they will switch to that platform, creating the need for platform owners to also put security measures in place to create a safer platform.

Developers

As discussed above, users get directly affected by the malware and the app developers are responsible, assuming for the cases where the app they have created is vulnerable which gets exploited. In the digital world, attribution is often the most challenging task [5] when an incident happens. If the users do not understand the difference between platform owners and app developers, it is possible that the users blame the platform instead of the app developer for an infection. Therefore, the incentive for whether the developer patches his/her apps is derived from the security awareness of its target audience and his/her own desire to maximize revenue (In the case of a paid app, revenue can be money, but in case of a free app, revenue can be user data on which the developer can run further market segmentation algorithms for highly targeted ads (e.g. Facebook ads)).

Another important point is that if the users do not care about their privacy, then the developers also have no incentive to take any action; especially if that means app downtime. Case in point, and a personal experience: Sarahah app[2] was an app to provide anonymous feedback. It was introduced a few months back and became an instant hit with more than 18 Million android and iPhone users worldwide. However, a few weeks later, Zachary Julian, a senior security analyst at Bishop Fox found that the app steals entire contact lists and sends to a C&C as soon as it is installed on the phone[3]. Following that discovery, the authors of this report informed their contacts that their privacy was at risk, but majority of them responded along the lines of ‘we have nothing to hide’ and ‘Facebook/Google steals more information; contact lists don’t matter anyway’. Sarahah’s developers said that they were collecting this information for a friend-finding-feature that did not yet exist, and still does not [4]. Perhaps if there was more pressure from the users, the developers might consider suspending the contact-list-stealing feature, at least until the friend-finding-feature was released.

[2] https://sarahah.com/

[3] https://thehackernews.com/2017/08/sarahah-privacy.html

[4] https://nakedsecurity.sophos.com/2017/08/31/people-rating-app-sarahah-slurps-up-contacts-for-feature-that-doesnt-exist/

Externalities

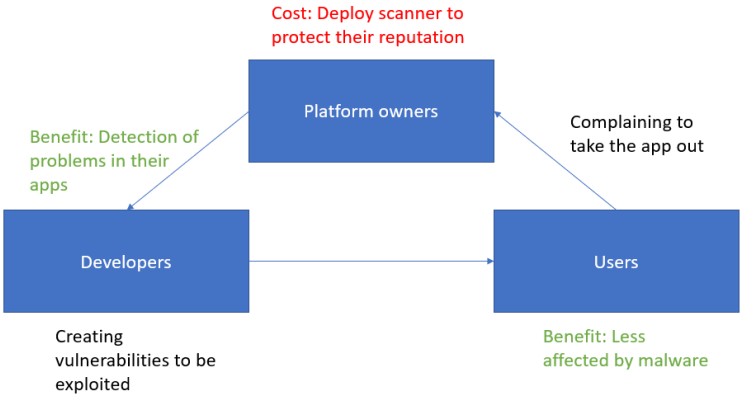

Figure 1: Externality because of platform owners' strategy

Platforms are two-sided markets that attract both buyers and sellers. They face two-sided network effects. There is a delicate balance to be struck in order to simultaneously grow both sides of the market [4]. As shown in Figure 1, a market externality is created by misaligned costs and benefits if the platform owners implement their countermeasure of deploying an app scanner. The users are directly affected by the malware, while the developers are responsible for the security issue, assuming that they leave vulnerabilities in the apps that are later exploited. However, the platform owners have no incentive on cleaning the mess up as they are only bearing the cost of implementing the scanner with little to no benefit.

Unless the party that actually is responsible for the malware is held liable, the market will stay inefficient. To fix this issue, for example, platform owners can come up with rules that make developers obligated to compensate the loss that their apps caused the users. This would drive up the cost of the apps, in some cases, not even having free apps. Then, unless the users understand the importance of secure apps, they will not buy the apps and migrate to another platform that provides cheaper insecure solutions, which is the case of information asymmetry which is another cause of market failure.

Security Performance

The first metric measures a property (maliciousness) of a platform, while the second one (impact) measures a property of the users. However, the maliciousness of a platform can help users decide which platform to select, and impact can help both developers and platform owners get a feedback of their security performance.

Metrics

We will first explain the metrics, then we will use the additional factor to explain its role in the variance of the two metrics.

To evaluate the metrics, we assume those apps to be malicious which were last updated more than 50 days ago. We make this assumption because there is no security related information in our dataset. We chose 50 days as a suitable number based on the distribution of the last-updated-days values. Here, an important distinction should be made. We use this assumption of 50 days only to evaluate the metric. Later, when we discuss the additional factor, app size, we associate it with the last-updated-day's value directly and assume that the older an app was last updated, the higher the odds of it getting malicious. We do this to make the analysis between two ordinal series, and to incorporate the uncertainty factor.

Metric 1: Maliciousness of a platform

$$maliciousness = \frac{\#apps_m}{\#apps}$$

Where:

$\#apps_m$ = number of malicious apps

$\#apps$ = total number of apps on a platform

The first metric measures the degree of maliciousness of a platform based on the fraction of malicious apps that are present on that platform. This metric is directly relevant to the platform owners, but can help users of a platform in choosing a suitable platform as well. If security of a platform is a factor based on which different platforms are ranked, then this metric will help form that ranking.

This metric will be 0 if no app is malicious and 1 if all apps are malicious. In this way, it is a normalized metric that can be used to compare multiple platforms.

Table 1 shows the maliciousness of Baidu and 360. It is clear that 360 is much more malicious than Baidu, and the reason is solely the number of malicious apps. This metric highlights the fact that if two platforms have the same number of apps, where one platform regularly patches its apps (or rather the developers of that platform do that) while the other does not, the latter one will be more malicious than the former. It also highlights that if a much smaller platform (in terms of number of apps hosted) does not regularly patch the apps, compared to a much bigger platform that imposes such restrictions of regular app updates, the smaller platform will be more malicious than the bigger one, even though the sizes of the platform suggest otherwise.

Table 1: Evaluation of metrics

	# apps	# downloads	# malicious apps	# malicious downloads	maliciousness	impact
baidu	2258	6753740305	1403	2621453730	0.6213463242	0.2411746033
360	18846	5443398109	17004	3793291588	0.9022604266	0.6287500598

Metric 2: Impact on users

$$impact = \frac{\#apps_m * \#downloads_m}{\#apps * \#downloads}$$

Where:

$\#apps_m$ = number of malicious apps

$\#downloads_m$ = number of times malicious apps were downloaded

$\#apps$ = total number of apps on a platform

$\#downloads$ = total number of times all apps were downloaded

The second metric measures the collective impact on the users of a platform because of malicious apps on that platform. This metric is directly relevant to the users of a platform, but the developers may take it into account if they want to check user satisfaction of their apps, for example. This metric's value also lies between 0 and 1, where 1 means total impact on all users, while 0 means no impact on any user.

This metric highlights the following difference in security performance: The impact on users is directly affected by the number of malicious apps that the users are using (or in other words, a platform is hosting) and the number of times those apps have been downloaded. For example, suppose that a platform 'A' only hosts one app which is downloaded by 10 million people, and another platform 'B' hosts 1 million apps but the total user base of that platform is 5 million people. Platform 'A' will have a much higher impact if that one app that it hosts was to go malicious, as compared to platform 'B', where if half of the apps were to go malicious, it will only have half of that impact.

Table 1 shows the values of impact for the users of Baidu and 360. Clearly, the impact for users of 360 is much higher than the users of Baidu, even though Baidu is a much more popular platform (and has many more downloads than 360). This shows that if the metric's value is 1, it means that all the apps on a platform are malicious so there is a total impact. However, if the metric's value is 0, it can either mean that no app on that platform is malicious, or that nobody downloaded any malicious app from that platform. The latter case remains true even if all the apps on that platform were malicious - If the users do not use the app, there is no impact.

Factors causing variance

According to the authors in [2], the **app size** is directly correlated with the popularity of the app. The larger size also makes apps more complex and that makes them prone to vulnerabilities which might make them malicious, if hacked.

Our hypotheses are shown in Figure 2. The larger app size will make it more popular [2], which is measured by the number of downloads (of course, controlled by the platform size). The popularity is directly correlated to the impact metric. The larger size of the app also makes it more complex (because of more features) and we assume that it is less likely to be updated. Therefore, the complexity will make it more prone to be vulnerable, increasing the odds of it getting infected. Therefore, it will contribute to the maliciousness metric. **An important assumption** here is that the more days it has been since the app was last updated, the higher the probability that it will get malicious.

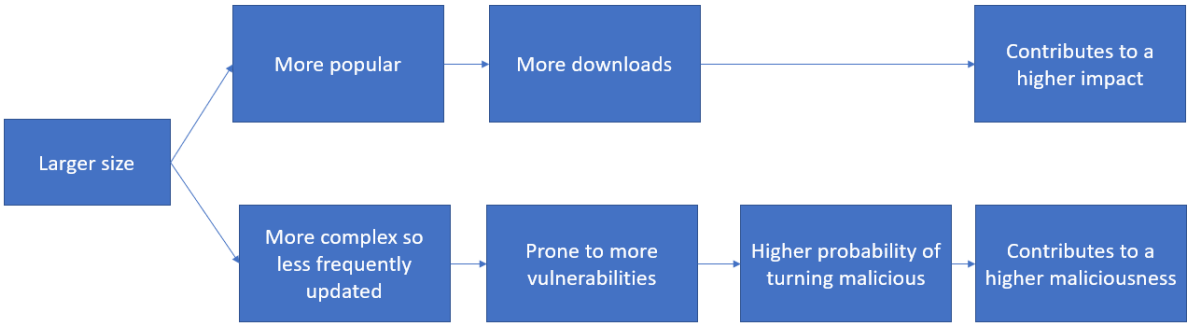

Figure 2: Hypotheses for external factors

Data collection

The app size for each of the app was part of the existing dataset. The sizes were stated in GB, MB, and kB for different apps, so we converted them all to MBs to consolidate the data.

Statistical Analysis

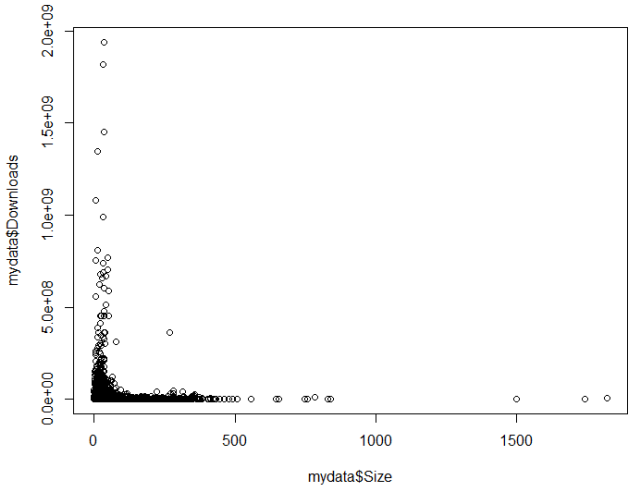

Figure 3: Correlation -- App size vs. Downloads

P-value = 2.486e-06 < 0.01 [significant]

Figure 3 shows the correlation between the app size and the number of downloads of each app. The correlation coefficient is 0.0199 which shows a slightly positive correlation saying that the larger the size of the app, the more downloads it will likely get with 95% confidence interval. This confirms our first hypothesis and follows from [2] that app size correlated with popularity of apps.

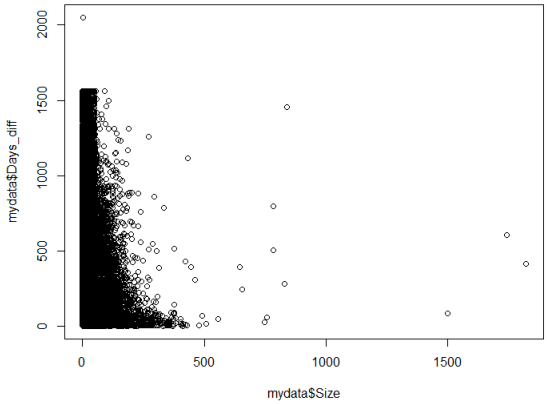

Figure 4: Correlation -- App size vs. Update days

P-value = 2.2e-16 < 0.01 [significant]

In Figure 4, we plot the app size against the number of days it has been since the last update. The correlation coefficient is -0.3611 which shows a negative correlation saying that the larger the size of the app, the more recently it was updated. This refutes our second hypothesis. This could mean that the smaller apps are the ones getting delayed updates, so they are more likely to get infected, and hence, contribute to maliciousness metric more than larger apps.

Assumptions

- We assume that popularity is directly measured by the number of downloads
- We assume that safe and protected apps are positive marketing for platform and developer.
- The impact of a malicious app on a certain user is uniform. We use these assumptions because we do not have the data to actually calculate these values, and these assumptions help make sense of the metrics.

Limitations

- The metrics are at the platform level granularity, while the data which will explain the variance is collected on app level. The difference in granularity might have an impact on the final conclusion not being that accurate.
- The assumption that safe apps are positive marketing only makes sense if the users consider security to be a showstopper in terms of app usage.

Conclusion

In this report, we analyzed the security strategies implemented by users, developers, and platform owners -- their costs/benefits, incentives and externalities caused by the misaligned costs/benefits. Lastly, we evaluate the effect of app size on the two metrics -- maliciousness of a platform and the collective impact on the user. We conclude that the app size is directly correlated with the popularity of the app, and hence contribute directly to the impact on the user. However, the smaller app sizes are more likely to get infected, so there is a negative correlation between maliciousness and the app size.

References

[1] http://www.aabri.com/manuscripts/131583.pdf

[2] Tian, Y., Nagappan, M., Lo, D., & Hassan, A. E. (2015, September). What are the characteristics of high-rated apps? a case study on free android applications. In *Software Maintenance and Evolution (ICSME), 2015 IEEE International Conference on* (pp. 301-310). IEEE.

[3] https://www.app-press.com/blog/whats-the-cost-to-maintain-an-app

[4] https://edge.edx.org/courses/course-v1:DelftX+WM0824+Fall_2015/courseware/4324fb5934c34d838a1c4f8243bde49b/3bea1331475d495cae85ec7d6f78353a/?child=last

[5] Brenner, S. W. (2006). At light speed: Attribution and response to cybercrime/terrorism/warfare. *J. Crim. L. & Criminology*, *97*, 379.

[6] Abawajy, Jemal. (2012). User preference of cyber security awareness delivery methods. Behaviour & Information Technology - Behaviour & IT. 33. 1-12. 10.1080/0144929X.2012.708787.

[7] *Boehm, Barry W. and Philip N. Papaccio. 'Understanding and Controlling Software Costs,' IEEE Transactions on Software Engineering, v. 14, no. 10, October 1988, pp. 1462-1477*

[8] https://www.us-cert.gov/bsi/articles/knowledge/business-case-models/estimating-benefits-from-investing-in-secure-software-development

[9] https://thinkmobiles.com/blog/how-much-cost-make-app/