Android 360° Assessment Programme

Q1 2023

MRG Effitas
EFFICACY ASSESSMENT & ASSURANCE

MRG Effitas Ltd.

MRG Effitas is a world-leading, independent IT security efficacy testing & assurance company. We are trusted by antimalware vendors across the world.

MANAGEMENT TEAM:

Chris Pickard, Chief Executive Officer
Norbert Biro, Chief Technical Officer

WEBSITE:
www.mrg-effitas.com

EMAIL:
contact@mrg-effitas.com

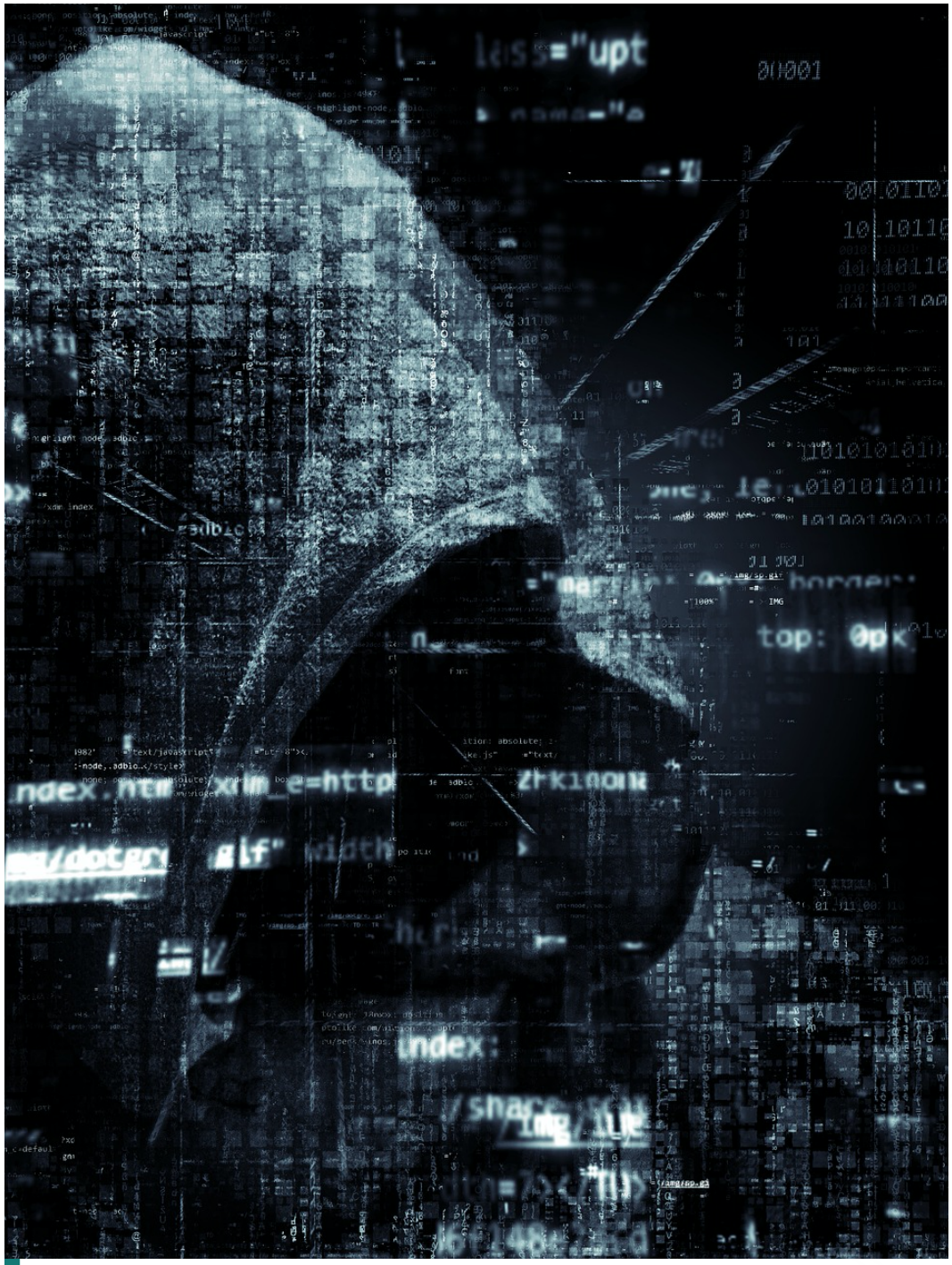
TWITTER:
@mrgeffitas

Contents

Introduction

MRG Effitas is an independent IT security research company, with a heavy focus on applied malware analysis. Besides conventional AV efficacy testing and providing samples to other players in the AV field, we regularly test APT detection appliances and enterprise grade IT security products, simulating realistic attack scenarios.

Android devices are used by over 3.1 billion people on this planet, making Android one of the most widespread IT systems. As the overall platform philosophy allows an easy-to-opt in platform with no mandated central application distribution platform, Android-based malware has been on a constant rise since the early Gingerbread days. As a result, the market for Android AVs is heaving with applications that promise loud taglines with '100% security'. A quick search on the Play Store for Antivirus products reveals literally hundreds of results – our test aims to help user decisions with a complex test regime with both In-The-Wild (ITW) and artificially crafted simulator samples and results that reflect the real-life efficacy of our test participants.



Our Mission

In providing quarterly certifications, the MRG Effitas Android 360 Programme is the de facto standard by which security vendors, financial institutions and other corporations can attain the most rigorous and accurate determination of a product's efficacy against current financial malware attacks.

We test over twelve months beginning in Quarter 2 and ending in Quarter 1, at which point (or shortly after) we publish our results. As with all our certification testing, we work with vendors, offering feedback and helping them to improve their product as we go.

Products that pass all tests during a quarter will receive the MRG Effitas certification for Android Efficacy Protection.

More information about the compliance status of this test can be found on the AMTSO website.

https://www.amtso.org/tests/mrg-effitas-Q1-2023-360-android-assessment-and-certification/

Tests Applied

MRG Effitas performed an in-depth test of several Android AV applications. The level of protection provided was measured in real-life scenarios with in-the-wild pieces of malware as well as some benign samples to map the shortcomings of the applied detection mechanisms. This report summarises the results of our efficacy tests.

Testing took place on Android 8.0.0 Genymotion emulator images and physical devices from May to July 2023, covering a significant portion of user devices on the market. To ensure maximum compatibility for samples that contain native ARM code, the ARM Translation package has also been installed on emulator images. In cases where ARM native libraries have been extensively used and the AV application could not be installed or properly run on an x86 emulator, we opted for stock Nexus 5x devices with Android 8.1.0. To ensure the cleanliness of the testing process, the Play Protect feature has been disabled.

Testing Methodology

Testing methods in earlier iterations of the 360 Android suite involved two independent test types: Early Tests, where samples were checked on the SD card, and Install Tests, which involved an actual installation of the sample.

As of Q1 2023, the standard approach has been changed to reflect a more grounded and realistic take on what is usually a series of test batch runs in a simulated environment. To reflect the actual performance of Android AV products, we follow the following flow for each sample[1]. With this revised methodology, we aim to provide a better understanding of how the individual AV products provide protection and what the limits and potential problems might be.

1. We make sure that the AV product is updated, provided with the latest signatures, and the device is connected to the Internet. Prior to testing, we make sure that the default Play Protect is disabled.
2. Using an explicit intent, the default browser on the device is navigated to a web site with neutral reputation and instructed to download the sample .apk All browser security features are turned off, URL reputation warnings are dismissed.
3. The package is downloaded, and an automated sequence is initiated to install the downloaded package. All warnings are dismissed, and all appearing system dialogs are accepted.

Should the AV display a warning or an alert...

a. ...in the download phase, the test case is counted as “Detection during download”.
b. ...after the download has finished, but the application install has not yet started, the test case is counted as “Detection before install”.
c. ...after the application install has finished, the test case is counted as “Detection after install”.

[1] A full AMTSO-approved test plan can be found on https://www.mrg-effitas.com/wp-content/uploads/2023/11/MRG-Effitas-360-Degree-Android-Certification-Methodology.pdf

Should the AV not display a warning or alert to the user, the test case is counted as a Miss.

It should be noted that on Android, installation of a piece of malware does not necessarily mean unwanted consequences for the user, as it is the first launch that kicks in any actual malicious code within. Having started the sample, however, it can have detrimental consequences from a security perspective. After the first launch, a piece of malware, after having requested SYSTEM_ALERT_WINDOW permission, can continuously display a Device Administrator or an Accessibility Admin request screen to the user. In such cases, the user is unable to get rid of the application as they have no access to the launcher, the application drawer, or the Settings application to perform an uninstall[2].

False Positive Tests

To cover all aspects of the efficacy of the participants, a limited set of samples are selected. The samples have been downloaded from a well-known 3rd party app store, exhibiting no malicious behaviour but requiring a varying range of permissions.

Samples

Malicious In-The-Wild Samples

Testing used an initial 150-sample malware set. All samples have been categorized using the following labels.

- **SMS Payment.** The application provides features to send SMS messages to premium rate numbers. Most of the selected samples were able to 'auto-send' messages, as they usually opted for the SEND_SMS permission, resulting in a direct financial loss for the victim.
- **Trojan.** Trojans are applications that display a certain set of features within their description and their overall appearance suggests some expectations regarding their functionality. However, the implemented modules require a wider range of permissions, which do not belong to the advertised functionality. A typical example is a flashlight app that can read the contact list, location information, and send them to the Internet.
- **Spyware.** We classify a sample Spyware if it leaks information that could be used to track the user (as most security-conscious users do not wish to be tracked). Ironically, most ad-propelled applications using aggressive frameworks qualify as spyware, as they leak IMEI, phone number, phone vendor and model etc. to the ad provider network.
- **Financial/banking.** This type of malware aims for direct financial abuse. A typical financial piece of malware detects if the user is logged in to a mobile banking session using either a browser or mobile banking application and, for example, might attempt to display a matching phishing site or to draw an overlay window to fool the user into thinking that the session has ended and that they need to re-

[2] Please note that to mitigate this kind of typical malware behaviour, the Android API design team reviewed the Device Administrator and the Accessibility Admin Request screens to include a checkbox that can be used to prevent the OS from displaying the screen again. This feature however, made its way only to recent revisions of the Android API.

authenticate. Typically, such samples use permissions to get the task list, combined with the SYSTEM_ALERT_WINDOW permission.

- **PUA.** [3] The term 'Potentially Unwanted Applications' denotes applications, which perform actions that are not in alignment with the security-conscious user's intentions. For instance, applications provided with aggressive advertisement modules usually make it possible for ad campaigners to track individual users, even to assign the device with the user's demographic properties through social network ad services. Effitas claims that security-conscious users are sensitive regarding their privacy, and possibly no application feature can make up for the users' private data and browsing habits to be sold over the Internet. A decent AV should let the user know if such an application is about to be installed.

Note that most samples implement several kinds of operation, therefore most samples fall into several categories. An example would be a typical piece of malware that serves malicious ads and, where possible, attempts to obtain the SEND_SMS permission to send premium rate messages.

Our ITW samples are obtained using several sources. As we strive to use fresh malware, the actual mix of samples used in test batches is highly dependent on the then-current activity of malicious threat actors. As a result, in many batches one or more families are represented with several samples, all of them fresh at the time of testing. Consequently, failing to detect that family of samples will result in multiple Misses.

Figure 1. depicts the distribution of test samples.

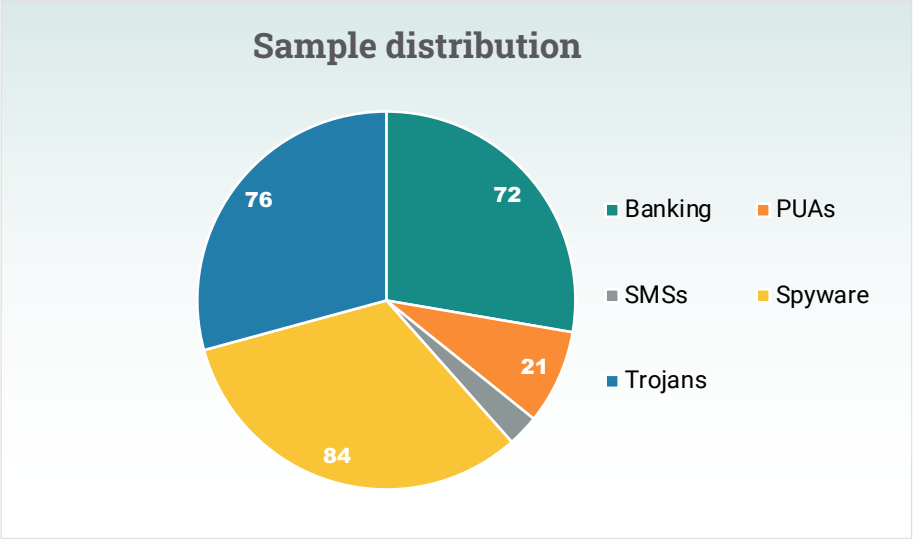


FIGURE 1. IN-THE-WILD SAMPLE DISTRIBUTION

Simulator samples

Simulators are custom samples, introduced into the testing process to put the sophistication of the detection routines to the test. Our simulators were created to simulate the attack model of a "malicious 3rd party app store providing backdoored applications" type of scenario, which means that counterfeit versions of legitimate applications are provided to the victims (many times pirated application versions can be downloaded free-of-charge). The counterfeit versions are backdoored versions of popular applications,

[3] Android applications with a social network integrated advertising module often fall into a kind of 'grey zone' from a detection perspective, as any application can be turned into a PUA, should the developers include an aggressive advertising module. Hence, we included charts, which handle PUA and non-PUA samples separately.

which, while retaining the functionality of the original application, also include malicious modules.

The samples have been created using a proof-of-concept engine using static smali byte code injection techniques, making no effort to obscure the malicious actions of the injected modules. Many of the simulator samples have been modified to implement Accessibility features, which is a common trait for several malware families.

For testing, we used 5 custom created samples. It is important to stress that these samples have not been collected or observed In-The-Wild. Our custom samples implement a well-known method exploiting the accessibility features of the Android API, which has been a popular way to read on-screen messages, SMS tokens, banking details and other sensitive information. Our samples were counterfeit versions of legitimate Android applications, but including a malicious Accessibility service, which tries to steal OTP 2FA tokens from popular tools (e.g., Google Authenticator, Azure Authenticator).

In addition, they exploit the fact that any application is allowed to create an intent to start the Google Authenticator app, which, in turn shows the OTPs right away (as opposed to having to perform a successful biometric re-authentication), making stealing the OTPs with Accessibility privileges quite trivial.

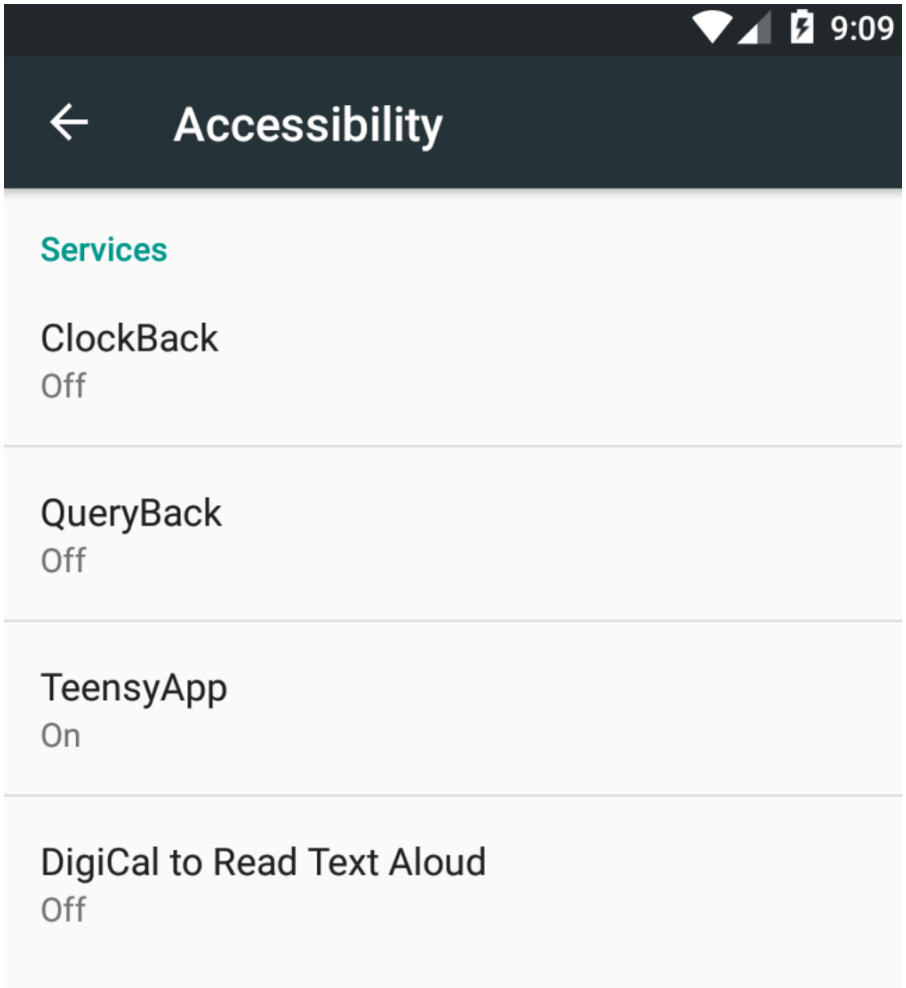


FIGURE 2. COUNTERFEIT APPLICATIONS EXPLOITING THE ACCESSIBILITY FEATURES

False positive samples

For false positive testing, a 10-sample set was used, retrieved from non-malicious 3rd party applications stores. The applications have been selected to cover a wide range of permissions and functionality.

Security Applications Tested

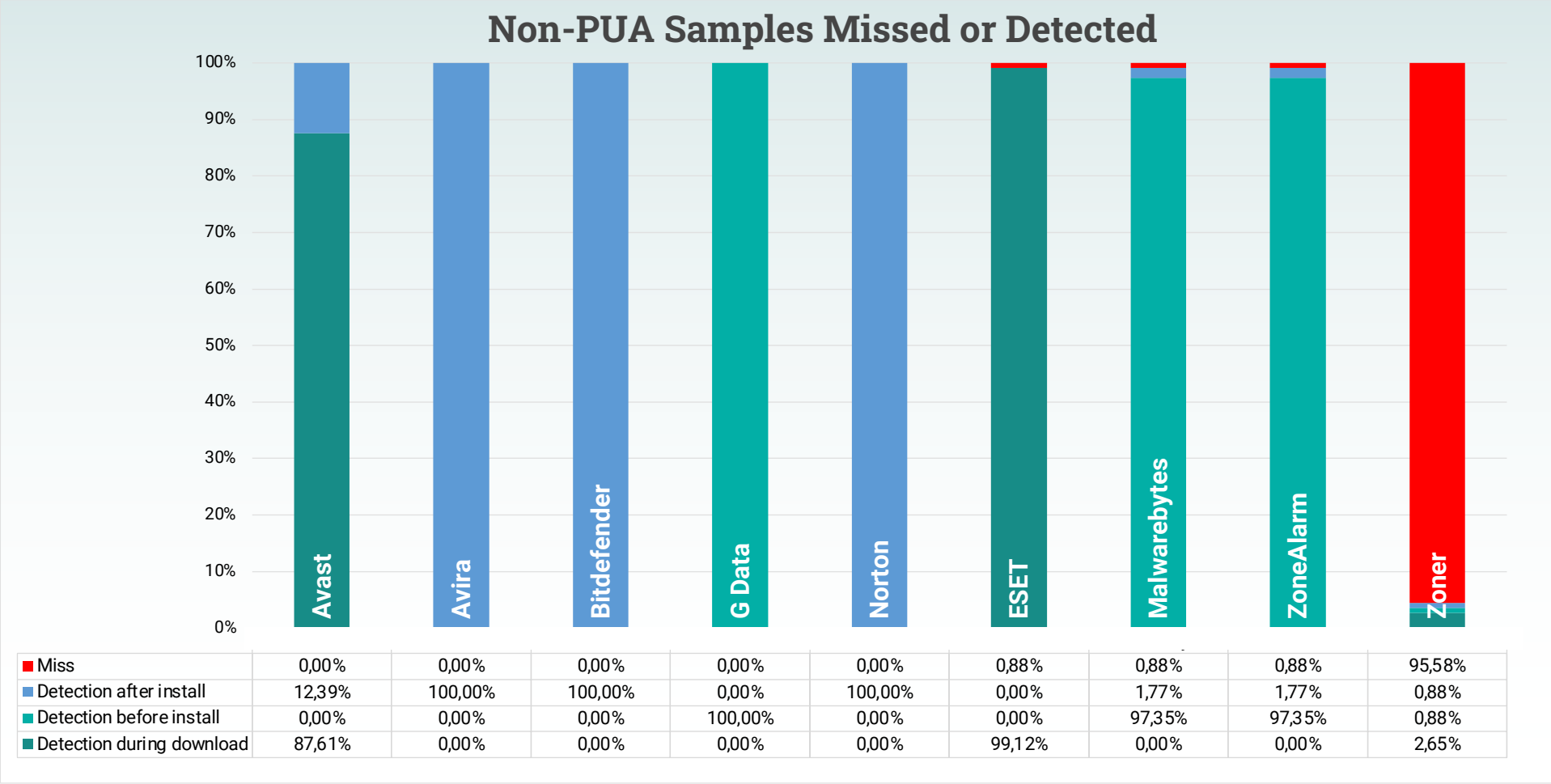
The following security suites have been selected for testing. Besides well-established vendors with considerable reputation and track history, we select smaller vendors with less market share. As the Play Store is heaving with Android security applications, we tend to select AV products with a considerable number of downloads.

Product Name	Caption	Play Store URL
Avast Software	AVG	https://play.google.com/store/apps/details?id=com.antivirus
Avira Mobile Antivirus	Avira	https://play.google.com/store/apps/details?id=com.avira.android
Bitdefender Mobile Security & Antivirus	Bitdefender	https://play.google.com/store/apps/details?id=com.bitdefender.security
ESET Mobile Security & Antivirus	ESET	https://play.google.com/store/apps/details?id=com.eset.ems2.gp
G DATA Mobile Security	G Data	https://play.google.com/store/apps/details?id=de.gdata.mobilesecurity2g
Norton Security and Antivirus	Norton	https://play.google.com/store/apps/details?id=com.symantec.mobilesecurity
Virus Cleaner, Anti-Malware	Malwarebytes	https://play.google.com/store/apps/details?id=org.malwarebytes.antimalware
ZoneAlarm Mobile Security	Zonealarm	https://play.google.com/store/apps/details?id=com.checkpoint.zonealarm.mobilesecurity
Zoner Antivirus	Zoner	https://play.google.com/store/apps/details?id=com.zoner.android.antivirus

TABLE 1. TEST PARTICIPANTS

Test Results

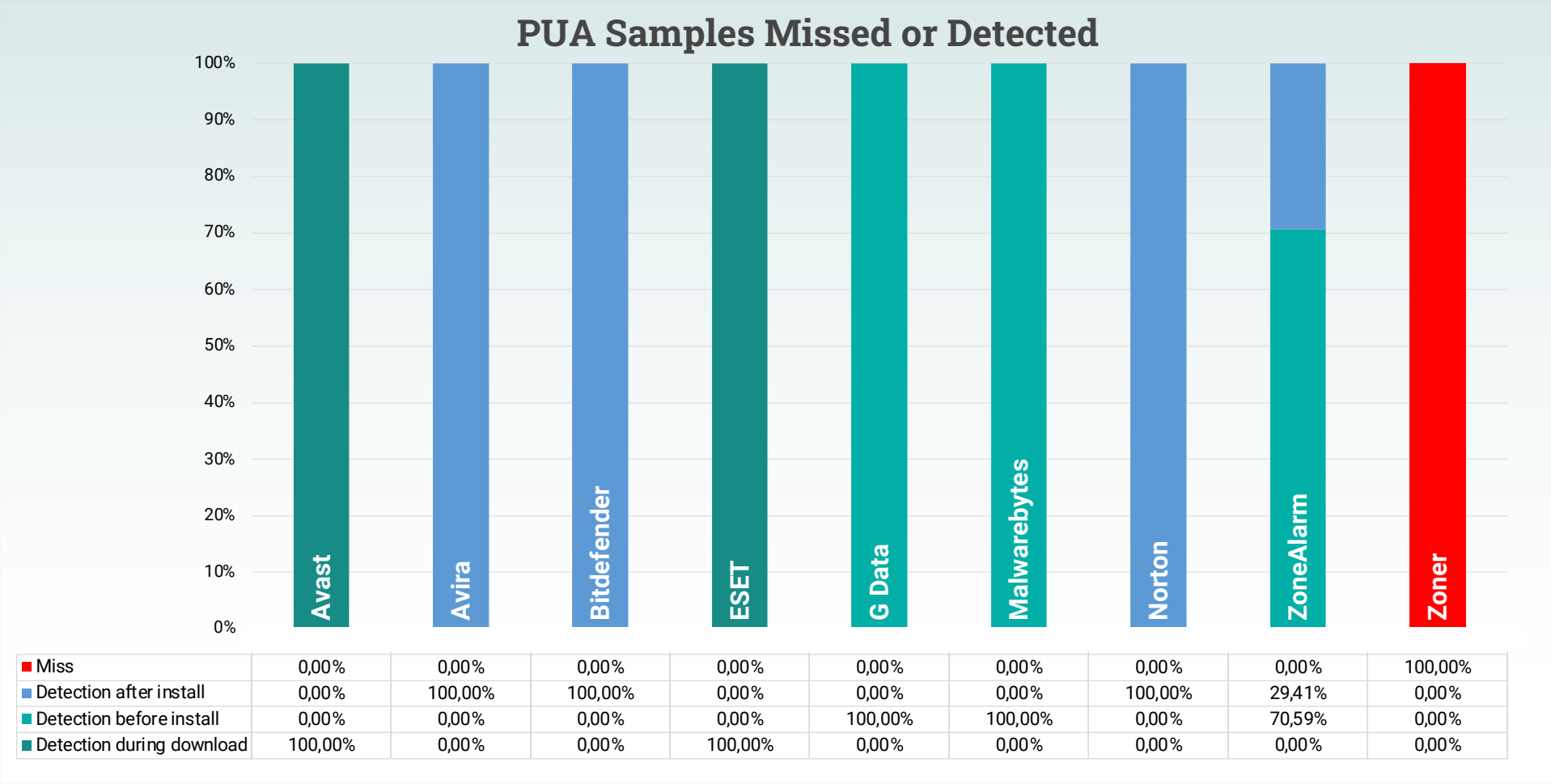
Overall non-PUA Detection



	Avast	Avira	Bitdefender	G Data	Norton	ESET	Malwarebytes	ZoneAlarm	Zoner
Miss	0,00%	0,00%	0,00%	0,00%	0,00%	0,88%	0,88%	0,88%	95,58%
Detection after install	12,39%	100,00%	100,00%	0,00%	100,00%	0,00%	1,77%	1,77%	0,88%
Detection before install	0,00%	0,00%	0,00%	100,00%	0,00%	0,00%	97,35%	97,35%	0,88%
Detection during download	87,61%	0,00%	0,00%	0,00%	0,00%	99,12%	0,00%	0,00%	2,65%

FIGURE 3. SUMMARY, NON-PUA SAMPLES

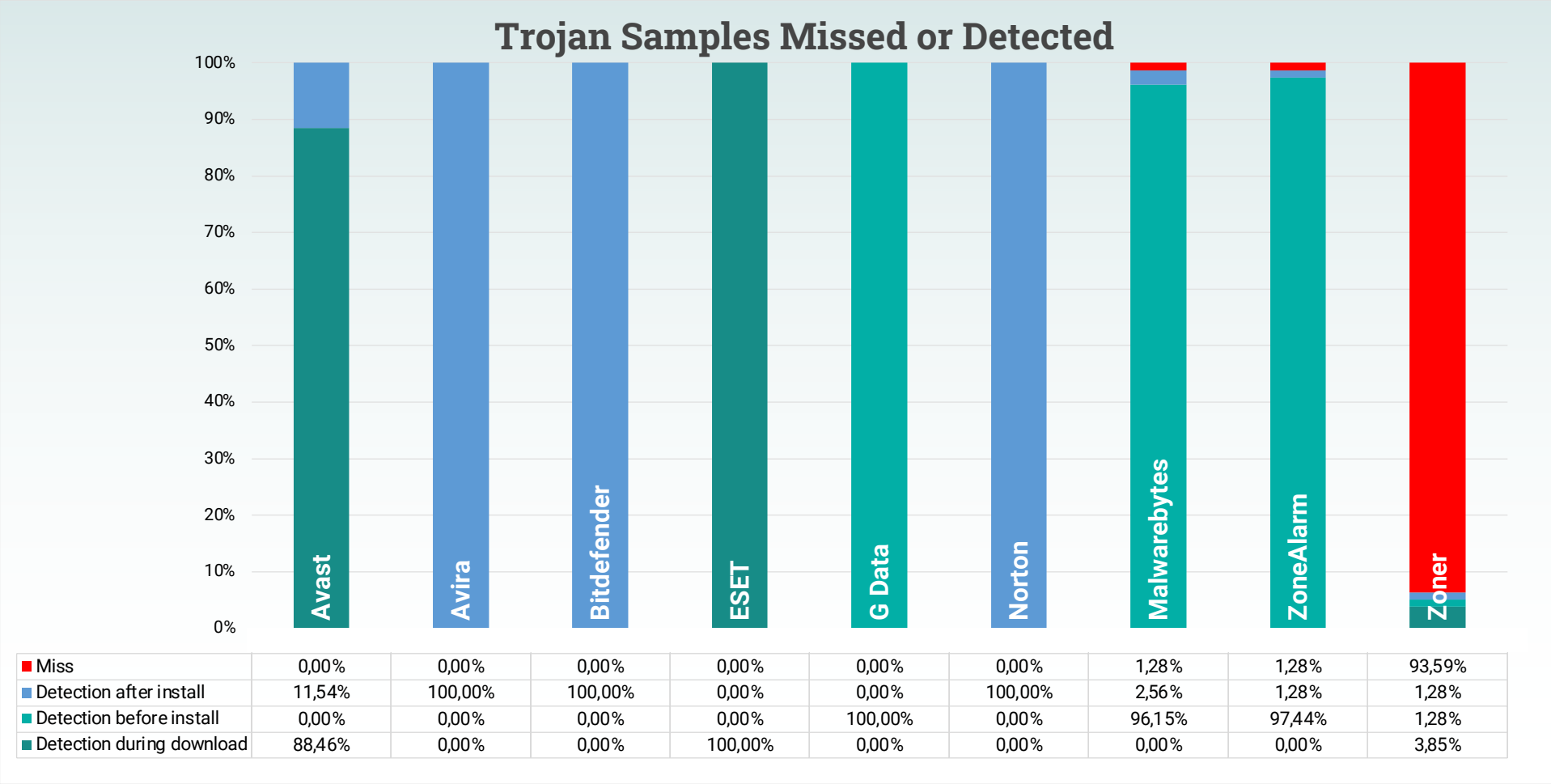
PUA Detection



	Avast	Avira	Bitdefender	ESET	G Data	Malwarebytes	Norton	ZoneAlarm	Zoner
Miss	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	100,00%
Detection after install	0,00%	100,00%	100,00%	0,00%	0,00%	0,00%	100,00%	29,41%	0,00%
Detection before install	0,00%	0,00%	0,00%	0,00%	100,00%	100,00%	0,00%	70,59%	0,00%
Detection during download	100,00%	0,00%	0,00%	100,00%	0,00%	0,00%	0,00%	0,00%	0,00%

FIGURE 4. SUMMARY, PUA SAMPLES

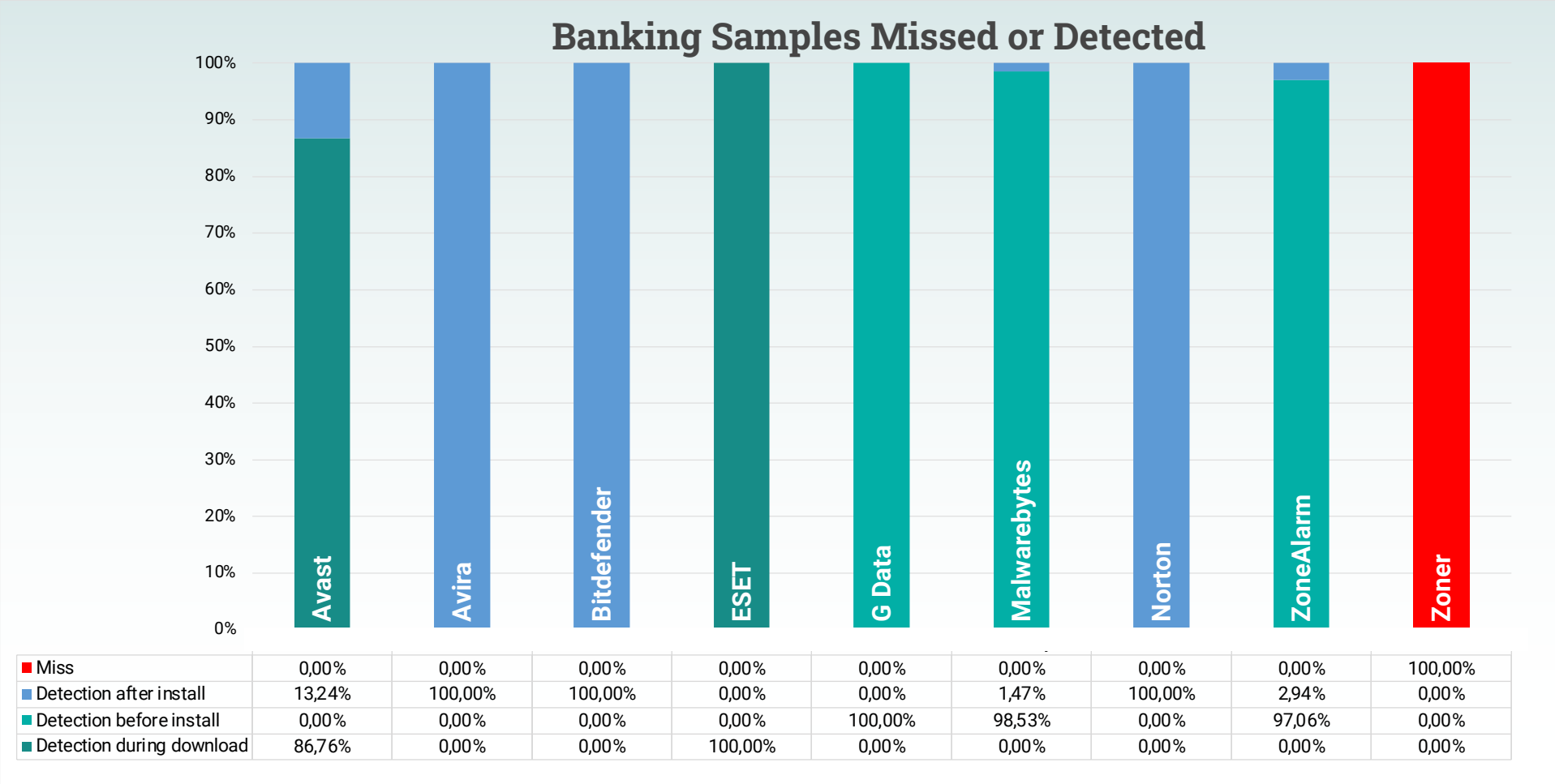
Trojan Detection



	Avast	Avira	Bitdefender	ESET	G Data	Norton	Malwarebytes	ZoneAlarm	Zoner
Miss	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	1,28%	1,28%	93,59%
Detection after install	11,54%	100,00%	100,00%	0,00%	0,00%	100,00%	2,56%	1,28%	1,28%
Detection before install	0,00%	0,00%	0,00%	0,00%	100,00%	0,00%	96,15%	97,44%	1,28%
Detection during download	88,46%	0,00%	0,00%	100,00%	0,00%	0,00%	0,00%	0,00%	3,85%

FIGURE 5. SUMMARY, TROJAN SAMPLES

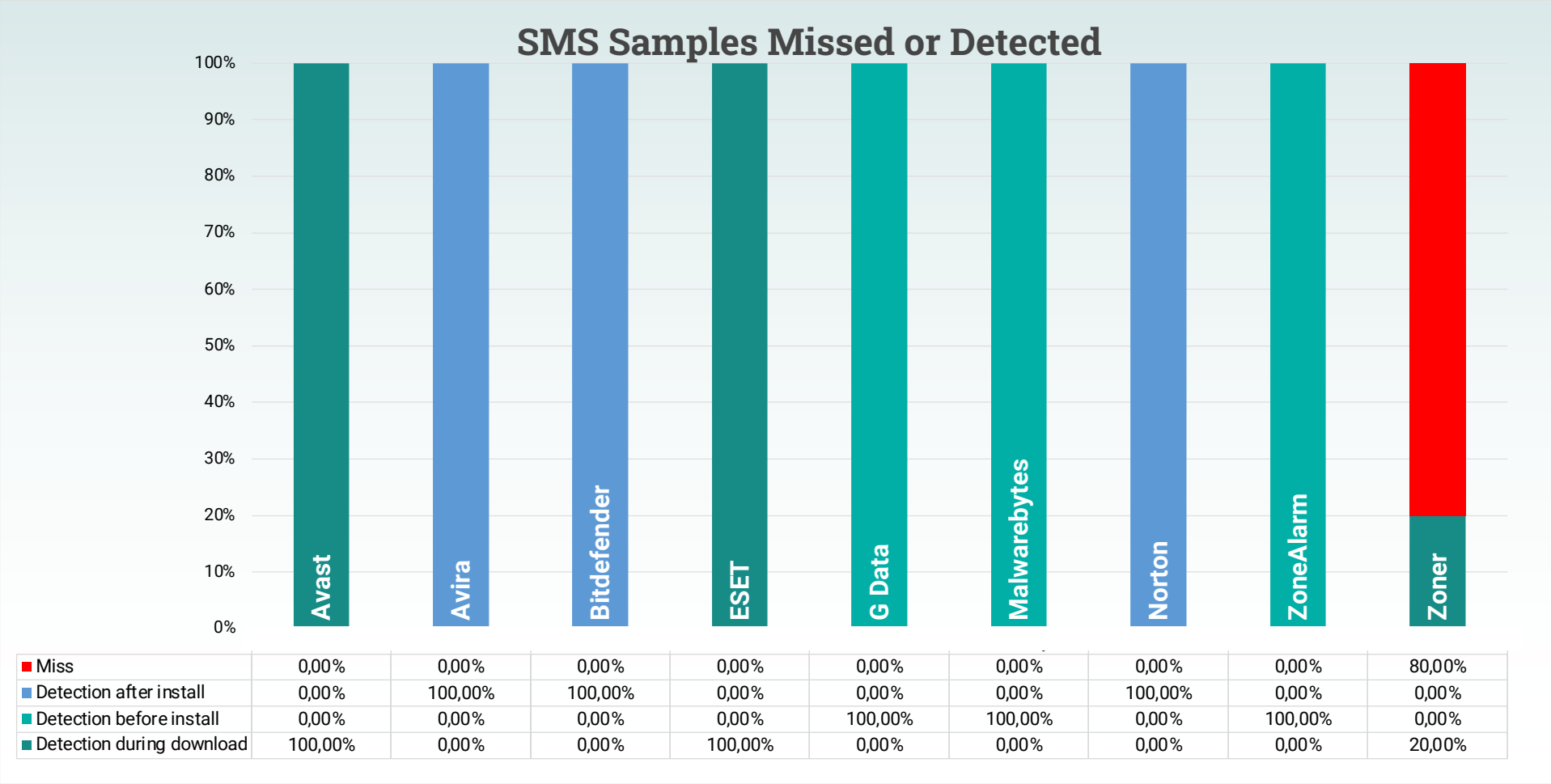
Banking Detection



	Avast	Avira	Bitdefender	ESET	G Data	Malwarebytes	Norton	ZoneAlarm	Zoner
Miss	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	100,00%
Detection after install	13,24%	100,00%	100,00%	0,00%	0,00%	1,47%	100,00%	2,94%	0,00%
Detection before install	0,00%	0,00%	0,00%	0,00%	100,00%	98,53%	0,00%	97,06%	0,00%
Detection during download	86,76%	0,00%	0,00%	100,00%	0,00%	0,00%	0,00%	0,00%	0,00%

FIGURE 6. SUMMARY, BANKING SAMPLES

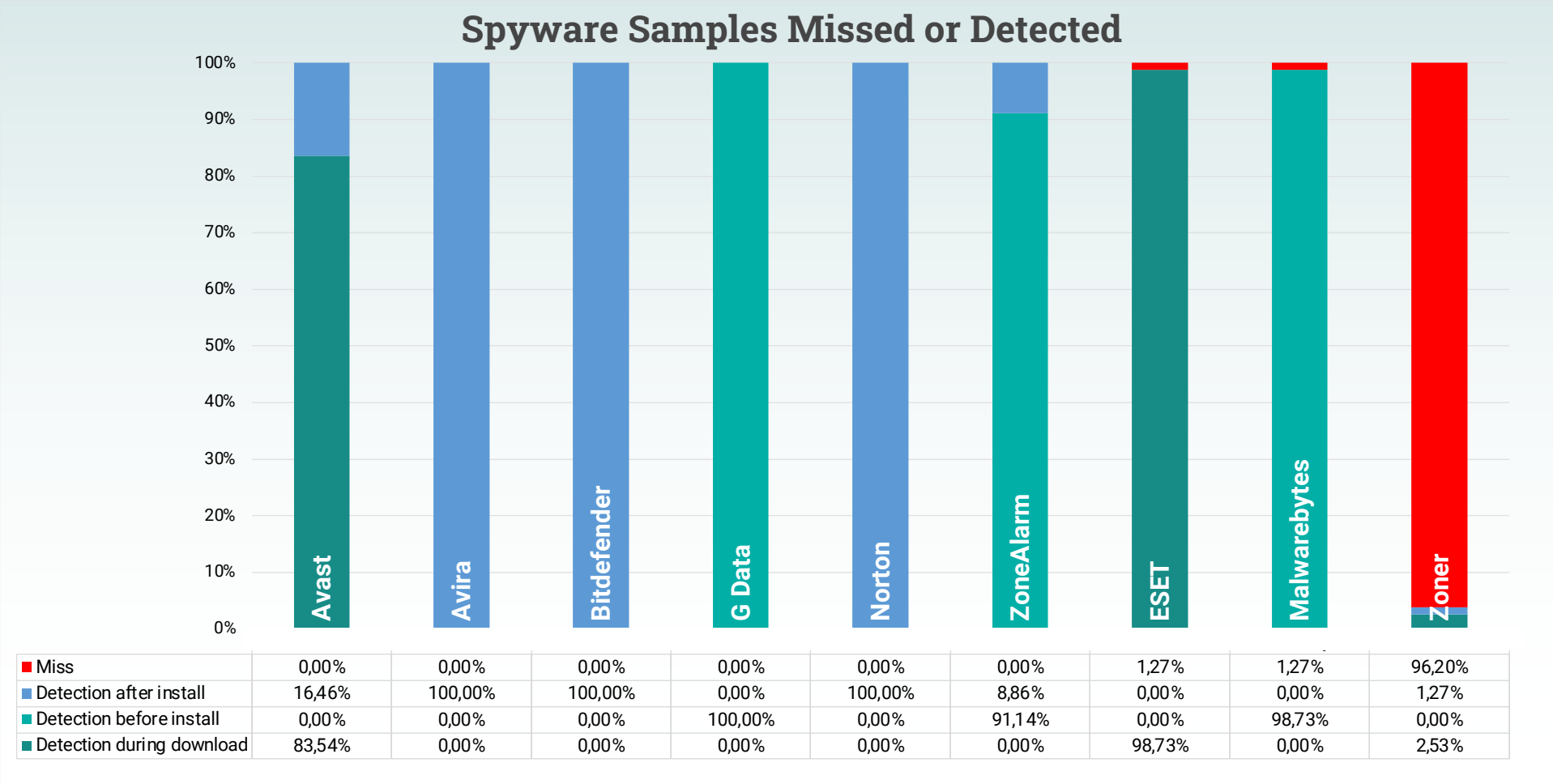
SMS Detection



	Avast	Avira	Bitdefender	ESET	G Data	Malwarebytes	Norton	ZoneAlarm	Zoner
Miss	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	80,00%
Detection after install	0,00%	100,00%	100,00%	0,00%	0,00%	0,00%	100,00%	0,00%	0,00%
Detection before install	0,00%	0,00%	0,00%	0,00%	100,00%	100,00%	0,00%	100,00%	0,00%
Detection during download	100,00%	0,00%	0,00%	100,00%	0,00%	0,00%	0,00%	0,00%	20,00%

FIGURE 7. SUMMARY, SMS SAMPLES

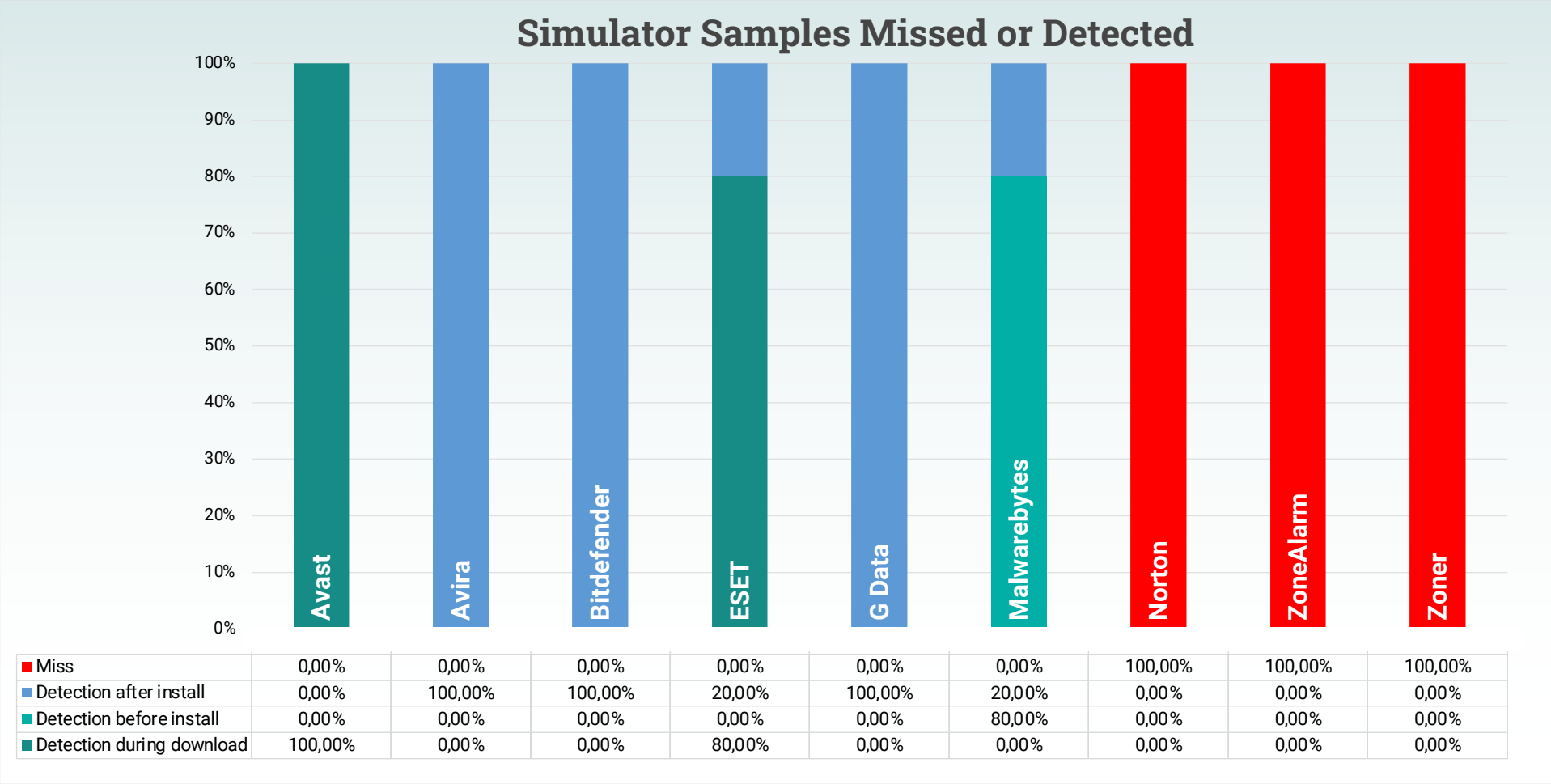
Spyware Detection



	Avast	Avira	Bitdefender	G Data	Norton	ZoneAlarm	ESET	Malwarebytes	Zoner
Miss	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	1,27%	1,27%	96,20%
Detection after install	16,46%	100,00%	100,00%	0,00%	100,00%	8,86%	0,00%	0,00%	1,27%
Detection before install	0,00%	0,00%	0,00%	100,00%	0,00%	91,14%	0,00%	98,73%	0,00%
Detection during download	83,54%	0,00%	0,00%	0,00%	0,00%	0,00%	98,73%	0,00%	2,53%

FIGURE 8. SUMMARY, SPYWARE SAMPLES

Simulator Detection



	Avast	Avira	Bitdefender	ESET	G Data	Malwarebytes	Norton	ZoneAlarm	Zoner
Miss	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	100,00%	100,00%	100,00%
Detection after install	0,00%	100,00%	100,00%	20,00%	100,00%	20,00%	0,00%	0,00%	0,00%
Detection before install	0,00%	0,00%	0,00%	0,00%	0,00%	80,00%	0,00%	0,00%	0,00%
Detection during download	100,00%	0,00%	0,00%	80,00%	0,00%	0,00%	0,00%	0,00%	0,00%

FIGURE 9. SUMMARY, SIMULATOR SAMPLES

False positive tests

In False Positive tests, all participants achieved a perfect score.

Summary

The following AV engines reached a near-perfect rounded 99% detection rate in a non-PUA sample set, therefore, they have been awarded with the MRG Effitas Certificate. Note that in the certification process, no distinction is made on where the detection has taken place, i.e. to be awarded, the MISS rate needs to be below 1%.

- Avast
- Avira
- Bitdefender
- ESET
- G Data
- Malwarebytes
- Norton
- ZoneAlarm

Conclusions

As a result of our testing efforts, a couple of conclusions can be drawn from our time with the AV engines and samples in our test lab.

Detection mechanisms

Our tests confirmed that most AVs use different methods for detection before and after installation. This is because prior to installation, a different set of metadata is available for the AV engine of a file that is stored on an SD card, than what is available after its installation.

This updated methodology clearly highlights a trend of detection mechanisms and their temporal operation. Some AV products detect threats as soon as they are downloaded, others put the emphasis on post-installation detection.

The earlier an AV detects a threat, the better it is from a user perspective. Nevertheless, given that even a successful malware installation can be saved, provided the user is warned before the first tap on the launcher icon and detection takes place shortly after installation, the test case is counted as a Pass in every scenario, where the AV displays a warning to the user.

Vendor reputation and extra services

As of Q1 2023, most of the well-established vendors reached a perfect or near-perfect score in the In-the-Wild categories. From a user perspective, this is all good news as several viable options are provided, some even without a subscription fee. In the future, we expect that the extra features (VPN, family locator, connections with desktop AV licenses etc.) will have a significant effect on user choice in the future.

As user consciousness evolves, more and more emphasis will be put on early detection, as an effective AV needs to provide functionality to scan the SD card for malicious content.

‘AV as another app’

Testing led us to the conclusion that detection for many AVs relies heavily on the metadata of installed packages (hashes, developer certificates, etc.), meaning that, unlike in a Windows based environment, an AV is unable to get an insight into the actual activity of other applications. This behaviour is in alignment with the basic Android security principles, as “AV is another app”. As a result, having already started the freshly installed samples, it is quite hard to get rid of some samples in the In-The-Wild test set, making timely and properly displayed detection an absolute must for AV engines.

Simulator detection

Most AV engines, having detected our custom simulator samples in past tests, were able to perform detection purely based on the package signature traits. This means that even though a notification has been displayed for those samples, successful detection has been a result of a mechanism and is heavily prone to false positives. As a result, in our previous Android 360 engagements many AVs had problems with detecting simulator samples.

Our test lab has been contacted on several occasions with claims that the simulators we utilise do not present a lifelike challenge for an AV engine. However, field reports show the presented scenario – when an adversary patches an existing Android application to perform hidden spying activity – is

well known technique that has been used for almost a decade[4]. The most famous campaigns using this approach is still the Dark Caracal APT[5], with an excellent analysis by Lookout[6]. For more details on the story, check out the corresponding episode of Darknet Diaries (ep. 38)[7].

Detection notification

While undertaking the 360° Android tests, we have noticed that there are significant differences in terms of user notification. When it comes to a successful detection, efficient and clear user notification is an essential part of both the efficacy of the AV application and the overall user experience.

The tested AVs choose one of the following approaches.

1. A separate activity is launched, usually with a bright red background and a couple of lines describing the nature of the threat presented by the application in question.
2. The Android notification subsystem is utilised to issue an important notification, usually displayed in the status bar, accompanied by an audible signal.

Both approaches have their merits. A separate activity like the first method is harder to dismiss or overlook, and the second method offers a more streamlined Android experience. However, Android provides a lot of options for users to customise notifications, therefore it is possible for the notification to get lost in the clutter. As a result, many of the tested apps opt for the first approach.

Furthermore, applying too many or too frequent notifications in the notification bar might result in important notifications getting lost in the noise. For instance, an AV might wish to inform the user that some background activity is taking place and to make it apparent that the device is provided with protection. However, should the user be accustomed to seeing AV notifications all the time, they might just automatically dismiss the notification as not important. As always, this is a matter of finding the right balance.

As for the wording, the overall design of the displayed notification and the consequent user choice description, there is a significant room for improvement in many apps. The Android way is to communicate as much information to the user as possible, just enough so that they can make a responsible decision. However, a responsible user must read and process the text displayed on the screen, which presents a significant mental load, especially for the less tech-savvy. As a result, a well-designed GUI can make all the difference for the everyday user, making users' choices more responsible and consequently, their devices more secure.

User data privacy as a "new" AV feature

As of 2023, with the wake of new smartphone privacy features on iOS and Android, Google is seemingly striving towards a more privacy-friendly approach, affecting the use of 3rd party user tracking for large-scale data

[4] https://threatvector.cylance.com/en_us/home/mobile-malware-and-apt-espionage-prolific-pervasive-and-cross-platform.html

[5] https://en.wikipedia.org/wiki/Dark_Caracal

[6] https://info.lookout.com/rs/051-ESQ-475/images/Lookout_Dark-Caracal_es-kf_20180118_uk_v.1.0.pdf

[7] https://darknetdiaries.com/episode/38/

analytics purposes. Although it should be born in mind that, with regards to the truth behind the tech giant's claims, a lot of research is still to be done.

However, there is a considerable amount of scepticism within the security community. Many argue that the likes of Google and Apple make a significant amount of revenue selling user data and providing and maintaining ways for data aggregators to access information, therefore, no fundamental change is to be expected. With user privacy currently under the spotlight, we expect that privacy services could offer a new way for AVs to add value for all Android users.

"Greyware" and Android API design

As of late 2022/early 2023, Android has been around for about a decade. Currently, we have 30-something different versions running on thousands of different supported hardware variations. In our experience, the majority of security related Android issues stem from design choices and platform philosophy considerations, made many years ago.

As for in-the-wild malware samples, despite the all the technical limitations of "The AV is just another app" approach, most of our public and private participants are doing a great job in detecting malicious applications even before the user has a chance to tap on Install. However, we sense that these explicitly malicious apps pose less and less risk for the everyday user, as the real risk comes from tracking/analytics SDKs, which try to gather as much data about the user as possible In order to do so, many SDKs attempt to abuse the Android API, and as they are bundled with legitimate applications with real value for the user, no AV in their right mind would flag them even as a PUA.

These apps are dubbed as "Greyware", as these apps provide actual useful features to the user, at a serious privacy cost though. A couple of ideas can be found in Nikita Kurtin's excellent Defcon 30-talk "Defaults – the Faults. Bypassing Android Permissions from All Protection Levels"[8]. The gist is that even if an app does not require any special permissions, a lot of functionality is available for them (for instance, not many know that the Toast API, which is used to create small, non-interactive "bubbles" on the screen without any permissions, can be (ab)used to create a full screen overlay, essentially carrying out a tapjacking attack).

Another (still exploitable) example is the area of tasks and task affinity: to provide a seamless UX, Android uses 'stacks' of Activities called Tasks[9] (visible when the Task Switcher is activated, kind of an 'alt-tab' on Android). Each running task has a name, and by default, every GUI based process (i.e. app) has its own Task, named with the package id. This is the default behaviour; however, an app can declare in its manifest the name of the Task it wants to be joined to. Up until the latest Android API revisions, the system made no checks on what this name is, e.g. a piece of malware is able to attach itself to the Task of a well-known application, making device based phishing a real threat.

As things stand, Google is apparently doing their best to limit the attack potential of such malicious SDKs, however, due to platform fragmentation, such malicious application components will stay with us in the foreseeable future and in our opinion, AVs should evolve to tackle these kinds of vectors.

[8] https://forum.defcon.org/node/242286
[9] https://developer.android.com/guide/topics/manifest/activity-element#aff

Stalkerware

Our sample selection process uses the vast .apk feed we also provide to our customers. Though the major parts of the selection process are automated to make our lives easier, sometimes we encounter samples, which are much more interesting than they seem at first glance and start digging into them to see what they are really doing.

During the dispute discussions with our Participants, we found that a couple instances of applications, which are debatable at best from an ethics point of view, providing 'device overwatch capabilities for concerned parents'. These apps are essentially highly intrusive spyware samples; however, the user needs to authorize their actions upon installation. According to the EULA to be accepted, the user acknowledges that this is a spyware app and understands that they are breaking the law in any other case than the 'concerned parent' scenario. Now, these agents come in all forms and formats, but the main premise is the same, and all reliable sample analyser frameworks report them as clearly malicious. During our tests, we try to veer away from using them.

Chinese Domestic Apps

Our Testing Team found a couple of interesting samples in this q's set. These apps were clearly providing actual functionality to the users, but as it turned out when performing a bit of reverse engineering on them, they also contain an excessive amount of spyware functionality.

During the discussions, it was found that these samples are widely installed by Chinese citizens living in China, and even though they are frowned upon in Europe and the US, many vendors intentionally whitelist them, with regards to them being legitimate products of legitimate Chinese companies. After a bit of correspondence, we found this weird ethical question, when one of our partners argued that if these apps should be detected, so should Facebook, TikTok and all major social media applications, as from a technical standpoint, the only notable difference is the endpoint where data is extracted to from the device.

Whether or not this is acceptable, depends on everyone's taste, however, we decided to exclude these apps from the set.

MRG Effitas

EFFICACY ASSESSMENT & ASSURANCE