The AppChk Crowd-Sourcing Platform: Which Third Parties are iOS Apps Talking To?

Oleg Geier, Dominik Herrmann

The AppChk Crowd-Sourcing Platform: Which third parties are iOS apps talking to?

Oleg Geier[1] and Dominik Herrmann[1]

University of Bamberg, Germany
{oleg.geier,dominik.herrmann}@uni-bamberg.de

Abstract. In this paper we present a platform which is usable by novice users without domain knowledge of experts. The platform consisting of an iOS app to monitor network traffic and a website to evaluate the results. Monitoring takes place on-device; no external server is required. Users can record and share network activity, compare evaluation results, and create rankings on apps and app-groups. The results are used to detect new trackers, point out misconduct in privacy practices, or automate comparisons on app-attributes like price, region, and category. To demonstrate potential use cases, we compare 75 apps before and after the iOS 14 release and show that we can detect trends in app-specific behavior change over time, for example, by privacy changes in the OS. Our results indicate a slight decrease in tracking but also an increase in contacted domains. We identify seven new trackers which are not present in current tracking lists such as *EasyList*. The games category is particularly prone to tracking (53 % of the traffic) and contacts on average 36.2 domains with 59.3 requests per minute.

Keywords: privacy · transparency · citizen science

1 Introduction

Modern smartphone apps communicate with several services at runtime, for instance, for debugging and tracking as well as displaying advertisements [7, 13]. So far, there are no easily accessible means that allow users to analyze the communication behavior of apps. This lack of transparency makes informational self-determination hard to achieve. A user study on information asymmetries between app providers and users comes to the conclusion that *strengthening user's control* (36 %) and *increasing transparency* (16 %) are two of the top three requested measures [3].

While privacy research on desktop devices is well established [12, 14, 15, 18] and has resulted in a number of transparency enhancing tools, there is a lack of tools for privacy research on smartphone apps, in particular for the iOS ecosystem. Existing tools are hard to set up or require a working Jailbreak to run; only experts can use these solutions. Further, the results of one-off studies are outdated a few months after publication and often apply to the considered set of apps only. Many of these publications are not easily reproducible because

they require a special experimental setup. Common setups require tethering the device to a computer, *jailbreaking* or *rooting* the device, routing the network traffic through a proxy, or patching the kernel to intercept system calls. Setting up the environment requires expert knowledge and time, which limits the target audience and the number of tested apps. Additionally, chances are lower that the study will be replicated whenever a new OS or app update is available.

We propose the *AppChk* platform to ease the evaluation for both, privacy experts and the general public. Our aim is to offer an easily accessible platform one can use without prior knowledge, which is future-proof, and keeps security and privacy measures intact. With *AppChk* we want to establish an on-going citizen science project to raise awareness for privacy practices in iOS apps; and create incentives for app providers to reduce third-party tracking. *AppChk* consists of two components, an iOS app[1] and a website (https://appchk.de). The app records application-specific network traffic; the website displays the results visually. *AppChk* allows its users to uncover known trackers as well as other high-frequented domains that are not considered a tracker yet. *AppChk* demands minimal user trust by following a privacy-by-design approach: The app uses an *on-device* VPN tunnel, i. e., traffic is not routed over our servers, and the app considers the headers of DNS queries only. No logging activity leaves the device unless the user opts in and chooses to upload a traffic recording to the *AppChk* website.

2 Related Work

Previous studies found that many third-party Android libraries collect Personally Identifiable Information (PII) and share it with advertising companies [7, 13]. These libraries often require more permissions than the application would need, to gain access to PII data like IMEI, IMSI, location, and sensor-data – in some cases even users' email addresses, email subjects, and IP addresses [13].

Claesson et al. [2] find that apps become increasingly consumed by the advertising business. Grindr, a gay dating platform which performs particularly bad, shares data with 53 domains, 36 of which are related to advertising. At least seven contacted domains receive the user's gender, age, IP address, GPS location, and a unique device identifier; in four cases even a unique user id.

The detection of these threats is researched extensively on Android [2, 4, 8, 13, 16]. However, there are only a few studies on iOS. This is in part due to the more restricted environment and the closed source nature of the OS [9–11]. Kurtz et al. have developed the testing platform SNOOP-IT for a dynamic analysis of iOS applications [10]. The analysis requires a jailbroken device and traces all system calls via `objc_msgSend` messages. Whenever a dynamic library is loaded, an API hook injects a tracing module to record all privacy-related API calls at runtime. The authors extended their approach later with an automated testing capability (DiOS [11]). DiOS allowed them to scale up the study and test 1136 apps. Their

[1] https://github.com/ubapsi/appchk-app

implemenation is based on Apple's UI Automation framework to simulate finger taps and perform screen navigation. Depending on the desired level of detail, their setup allows to test up to 500 apps daily. In their experiment almost half of all tested apps use tracking and advertising libraries.

Our setup is closely related to the one proposed by Amrein [1]. The SpySpy app monitors network traffic directly on the phone. Although the author states that SSL interception is an unwanted security risk, he does not abandon it completely. He argues that SSL interception is necessary to fully evaluate privacy risks. The proposed solution operates in two phases, an app screening and a network monitoring phase. The first phase uses a MitM server to intercept HTTPS connections to detect privacy violations. The second phase uses an on-device proxy to monitor network traffic. The proxy uses the results of the first phase to warn users about specific apps if necessary. Users can see the app-analysis results directly in the app.

Maass et al. propose *PrivacyScore* [15], a platform for website analysis and comparison. Their work goes beyond pure tracking analysis and evaluates websites based on security measures and recommended privacy practices. One of the core features are comparison lists. Websites from the same peer group are ranked according to the scoring on privacy and security features. The ranking creates an incentive for website operators to improve by reducing tracking [14]. Further, the authors provide a tool for data protection authorities and activists to verify the claims made by providers. With *AppChk* we aim to provide a similar service for iOS apps.

Apart from research, there are also tools used in practice; εxodus [4] and TrackerControl [8]. Both projects consider Android apps only. TrackerControl exposes tracking and allows its users to block tracking selectively. TrackerControl uses an on-device network proxy for monitoring. εxodus displays tracker usage, it is intended as an app index or app catalog. Their database is based on static analysis and contains 84855 applications and 340 trackers (as of February 2021).

3 Our Approach

We use an on-device *NEPacketTunnelProvider* proxy to capture all network traffic of the device. To the user this is presented as a VPN service. The advantage of an on-device proxy is that potentially sensitive data like browsing history and user-specific domains are not sent off to another party during analysis. Using on-device avoids pitfalls such as misconfigured VPN servers, which may leak IPv6 traffic or allows DNS hijacking [17]. Finally, on-device proxies do not have any impact on the speed of data transmission (no additional latency, no throughput limit). The device connects directly to the requested target.

The *AppChk* app only considers the domain names of outgoing connections. We do not look inside the traffic and hence do not depend on breaking TLS encryption. This ensures future reproducibility as it does not require a working Jailbreak or special setup. The *AppChk* app displays all network requests in realtime (cf. Fig. 1 left).

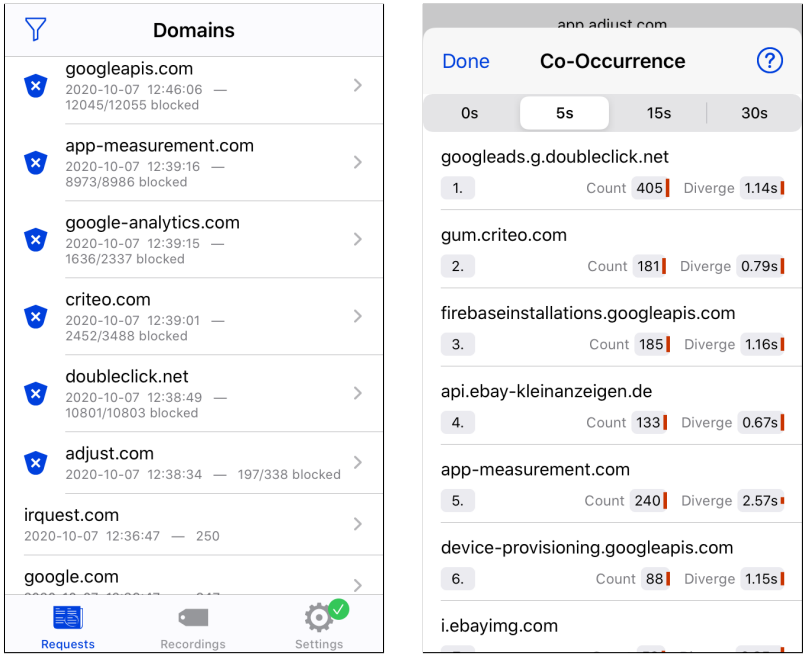

Fig. 1. Realtime Domain Requests (left), Co-Occurrence Analysis (right)

3.1 Design Goals

Our primary goal is to provide a platform that can easily be set up and be used by everyone, including novice users. *AppChk* should keep existing **privacy and security** measures intact. Therefore, we can not rely on a Jailbreak (new OS versions will close previous vulnerabilities), TLS interception (privacy invasion and data integrity), or an external proxy server (disclosing browsing history, requiring trust in the service provider). To be **future-proof**, our app works with TLS enabled and uses only documented APIs, which allows us to release the app on Apple's AppStore. Making *AppChk* **available** in the AppStore lowers the bar to participate in research. App evaluation becomes a continuous process and latest releases of frequently used applications get evaluated more quickly. We provide the *AppChk* app including its source code for public interest to keep the service active and up-to-date. *AppChk* app and website are designed with **ease of use** in mind. The app's design is unobtrusive yet helpful and the website aids novice users to judge whether a particular app uses tracking. With enough user-contributed reports, the website will foster **comparability** between apps, which might incentivize app vendors to enter into a competition for more privacy-friendly apps in their respective category (cf. [14]).

Privacy by Design A recent Washington Post article by Geoffrey Fowler analyzed the newly introduced privacy labels for apps [5]. Fowler used *Privacy Pro*, a

MitM VPN app by the company *Disconnect*. Contrary to *AppChk*, *Privacy Pro* routes the traffic over an external VPN server. This does not only require trust in the provider but may also put privacy at risk. The design of *AppChk* is not subject to this limitation. Users can download and use *AppChk* without prior explanation by an expert and will immediately see what network connections the device establishes. Users can further record the network activity within a short time interval of a few minutes and, if explicitly chosen, share their recordings on the *AppChk* website. As a user's recordings may include domain names deemed sensitive, the *AppChk* app displays what information has been collected. Users can review and delete individual domain names to sanitize the upload.

Data Minimization *AppChk* does not collect more information than strictly necessary. Logged timestamps are only precise to *full seconds*. Further, the *AppChk* app minimizes the recorded data before it leaves the device. Absolute time information is replaced with relative time offsets based on a start date, which has a precision of *calendar weeks* only. Moreover, users can configute the *AppChk* app to automatically delete logs after a specified amount of time. This reduces the risk of inferring too much personal information from historical data that is kept on the device. Further, *AppChk* minimizes third party dependencies. The only used dependency is NEKit which is used for DNS resolving and packet transfer.

Transparency & User Control The *AppChk* app allows its users to contribute app recordings. As recordings may include sensitive pieces of information, we have to handle them diligently. The *AppChk* app explains what data is shared and how it is used. Users are not obliged to contribute recordings, nor are they nudged to do so. We also give users the choice to exclude individual requests, on a subdomain level, from their contributions. As an additional defense-in-depth measure, we remove unique domains on the server side if a user sends them mistakingly. This filtering happens automatically by cross-correlation between different recordings of the same app. Domains that appear only in a single recording are removed. Users can also configure *AppChk* to ignore or block specific domains (DomFilter). If a domain is *ignored*, it will not be logged by the *AppChk* app. If a domain is *blocked*, *AppChk* will disconnect all network connections to that domain. Thus, *AppChk* can be used as an on-device content blocker. The filters give users greater control over their data; namely, what data is persisted and what data is shared with other parties. Lastly, the *AppChk* app offers users the option to export their recordings for independent analysis. *AppChk* exports the database as is, with everything ever recorded (and not deleted). Using data exports, users can use the app without ever sharing information with us.

3.2 App Recordings

AppChk can not differentiate between traffic from one application and another due to technical limitations of iOS. Instead, app recordings capture the domain names of all outgoing network requests during a particular time frame. A single

recording may include requests from multiple apps and system background processes. Therefore, we urge users to quit all running applications before starting a new recording. App recordings temporarily disable any user set DomFilters. This might violate a user's decision to control how their data is processed. However, we made this decision for comparability reasons, to have an unaltered view of "what happened." Filters are mainly used to block third-party tracking – which is exactly what we want to detect with *AppChk*. Users are notified of the deactivation of filters when a recording is started.

3.3 Continuous Monitoring

Apart from using *AppChk* for on-the-spot app recordings, users can keep the app running in the background as an always-on network monitor (and tracking blocker). As long as *AppChk*'s on-device VPN service is active, *AppChk* will log network requests in the background, independently of recordings.

Co-Occurrence Analysis One problem of looking at network requests alone is the sheer quantity of requests; some apps issue up to 445.7 requests per minute (cf. Sect. 4). This would overwhelm users if they would have to analyze each request separately. Therefore, we provide an in-app context analysis mechanism that relies on the time correlation of requests. This *co-occurrence analysis* feature helps users to attribute seemingly unrelated requests over a longer period of time. The analysis can also uncover new tracking domains, as many tracking requests happen in close proximity to one another.

Given a domain name X, we look for all domain names $Y_{1..n}$ which frequently appear simultaneously. Users can choose between different time windows of up to 30 seconds. With a time window of 0, the correlation function will consider requests which happened precisely at the same time (exact to the second). If the window size is greater than 0, results are sorted by close temporal proximity; requests that occur closer to the request(s) of the selected domain are preferred.

Co-occurences are displayed in a ranked list, which relies on the weighted score $(\overline{\Delta t}^2 + \frac{T}{2} + 1)/N$, where $\overline{\Delta t}$ is the mean temporal distance to the selected request entry, T the window size in seconds, and N the number of requests found within the time window. Δt can be at most T for window sizes greater than zero and is always 0 for a 0-second window. The ranking score strikes a balance between favoring domains with many requests and favoring domains with very close proximity. $\overline{\Delta t}^2$ prefers temporally closer results, while $\frac{1}{N}$ prefers results with higher occurrence counts. The weighting factor $\frac{T}{2}$ favors temporally nearby entries if the window size is small. We add $+1$ for numerical stability, otherwise a window size of zero will nullify the numerator.

Figure 1 (right) shows the co-occurrences of the domain `app.adjust.com` for a time window of $T = 5$ seconds. The domain `gum.criteo.com` is ranked second, even though the domain's requests are temporally closer to the requested domain (0.79 s vs. 1.14 s on average). The top-ranked domain has over twice as many intersections (405 vs. 181). The orange-colored bar indicates strong correlation, e. g., for the fifth rank, more requests balance the higher time divergence.

4 Evaluation

In this chapter, we look at some exemplary use cases for *AppChk*. In particular, we show what kind of information we can extract from the collected data. We conclude this chapter with an evaluation of 75 apps, including a comparison *before* and *after* the release of iOS 14, which was announced to introduce changes to acceptable data uses. We start by introducing the two datasets used for these analyses.

Dataset D1 In one of our evaluation use cases, we compare regional differences between app developers. For that we consider three geographic regions: Americas, Europe, and Other. We randomly sample 25 apps per region as follows. First, we obtain monthly top charts of July, August, and September 2020 for 11 countries from the app analytics provider *App Annie*.[2] We consider the Top 20 free apps for each list and month, yielding 660 apps. Second, we filter this list by removing all duplicates. We also remove apps that are not available in Germany and apps that have no company location attached. From the remaining 138 apps we sample 25 apps per region.

Each of the 75 apps is analyzed separately as explained in the following. First, a tester (one of the authors of this paper) quits all running applications and waits five seconds for background processes to finish. Then, the tester launches an app and uses it extensively to cover as much of the functionality as possible. Afterward, the tester quits the app and stops the recording. Whenever an app requires a user login, the recording only includes data up to the login screen. If present, register, login, and help buttons are tapped in any case.

Our D1 dataset holds 75 apps, 1093 recordings, 1062 unique domains, and 102 316 individual requests. 45.0 % of all requests and 26.5 % of all domains are tracking-related. Each app was recorded seven times before and seven times after the iOS 14 release. All iOS 13 recordings were recorded in the week before the release (Sept 12 to Sept 15). All iOS 14 recordings were recorded between Sept 28 and Oct 24.

Dataset D2 The second dataset includes all apps from D1, plus an additional 64 apps that not one of the testers but other users (unknown to us) submitted to the *AppChk* website. Most of the additional apps were tested only a single time (+76 recordings), limiting the validity of the results. We can use these recordings, however, to detect additional tracking domains.

4.1 Use Case: Tracker Detection

To detect previously unknown trackers, we can cross-correlate data of different apps and find domains that appear exceptionally often. Even with our limited dataset of 139 apps in dataset D2, we detect seven domains related

[2] AU, CA, CN, DE, ES, FR, GB, IT, JP, RU, UK, and US

to tracking that are not present in the commonly used tracking lists EasyList, EasyPrivacy list, Peter Lowes Ad and tracking server list, and εxodus ETIP list. We found `app-measurement.com` (in 77 apps), `ocsp.sectigo.com` (16), `inner-active.mobi` (10), `in.appcenter.ms` (7), `track.atom-data.io` (7), `liftoff.io` (7), and `taobao.com` (5). Further, we found that 7 out of 15 subdomains of `unity3d.com` (found in 17 apps) are not marked as trackers even though they should be. Some of these trackers seem to be exclusively designed for specific mobile operating systems. One of the most-used trackers, `app-measurement.com`, is not present in the εxodus tracker list. Thus, findings obtained with *AppChk* can be used to supplement existing lists of trackers.

Table 1. Tracker usage in apps and total network requests (in percent).

	AppChk		**Kurtz et al. [11]**	
Domain	**Apps**	**Requests**	**Apps**	**Requests**
apple.com	84.17	4.92	6.76	1.51
app-measurement.com	55.40	2.77		
crashlytics.com	48.92	0.56	5.68	0.58
facebook.com	46.76	2.67	13.96	3.47
doubleclick.net	33.09	1.02		
appsflyer.com	23.74	1.87		
adjust.com	22.30	5.47		
googleadservices.com	9.35	0.13	3.33	1.36
amazonaws.com	5.76	0.14	3.60	1.30
ioam.de	5.04	0.05	5.59	1.60
tapjoyads.com	4.32	0.12	5.86	1.99
flurry.com	2.16	0.06	23.15	5.41
chartboost.com	0.72	0.05	3.33	0.84
admob.com	0.72	<0.01	11.44	1.44

Table 1 compares the results of our study to the trackers found by Kurtz et al. [11]. 22.3 % of the apps in D2 contact `adjust.com` at least once. Considering all apps, about 5.47 % of the network requests in D2 are routed to `adjust.com`. We observe that the tracker landscape has changed drastically since 2014. Previously dominant trackers such as `flurry.com` and `admob.com` have a smaller market share than before. Other trackers have grown rapidly and taken their place. In contrast to our dataset, the dataset by Kurtz et al. did directly attribute the network traffic to a specific application. In our case, we also detect requests that originate from other apps or system services. That could explain why our study detects so much more domain calls to `apple.com`.

4.2 Use Case: Comparing apps and app groups

App Comparison Maas et al. show that, at least for websites, adding a comparison of different providers on privacy and security practices creates a competition

Table 2. Regional differences; each with 25 apps and 219 recordings (min—avg—max).

Region	Total Req.	Req. / min	Domains	Subdomains	Tracker
America	9198	3—32—75	4—10—21	4—20—60	0—29—64 %
Europe	37735	1—75—446	1—38—184	1—65—302	0—44—92 %
Other	13806	0—52—169	1—13—29	0—27—92	0—34—75 %

between these providers [14]. This competition sets incentives to reduce third party tracking. *AppChk* follows suit by allowing direct comparisons between apps: Users are able to compare two similar apps and decide which of the two respects their privacy better.

Consider the following example. *Viber* is an instant messaging app, which was hyped a few years ago as a secure and privacy-friendly alternative to Skype and WhatsApp. Their website states: "Our mission is to protect your privacy so that you never have to think twice about what you can or can't share when you're using Viber." Figure 2 shows the evaluation results for the Viber app as displayed on the *AppChk* website. Users can see, at a glance, whether an app uses tracking at all and to what extent. Further, users can see how many domains are contacted and what proportion of requests are known trackers (red color). In this example, the app connects to 12 different domains, eight of which are known trackers (`crashlytics.com`, `app-measurement.com`, `mopub.com`, `googlesyndication.com`, `doubleclick.net`, `appboy.com`, `mixpanel.com`, and `adjust.com`). 66 % of all network requests go to tracking providers.

App Comparison Lists On the *AppChk* website, users also have the ability to compare lists of apps with each other. For that, we tabulate key metrics for each app, e. g., the number of tracking domains, the percentage of tracking domains, and the number of requests per minute. The results are presented in a configurable and sortable table.

Group Comparison Lists Comparison groups are similar to app comparison lists but compare groups of apps against each other, such as *free* versus *paid* applications. Han et al. [6] find that paying for an app does not guarantee an app to be free of trackers. Most paid apps even reuse the same tracking libraries and permissions as the free version. With *AppChk*, we can set up a continuous evaluation process. For now, groups can only be configured in the backend of the website, but we plan to allow users to do that on their own as well.

Our dataset D2 does not include any paid applications yet. We can, however, consider regional differences (cf. Table 2). Our classification depends on the location of the companies' headquarters. *Europe* is doing significantly worse than the other two regions, if comparing the amount and frequency of contacted domains. European apps contact, on average, three to four times as many domains as apps from other regions; simultaneously, the proportion of tracking domains increases by 10 %.

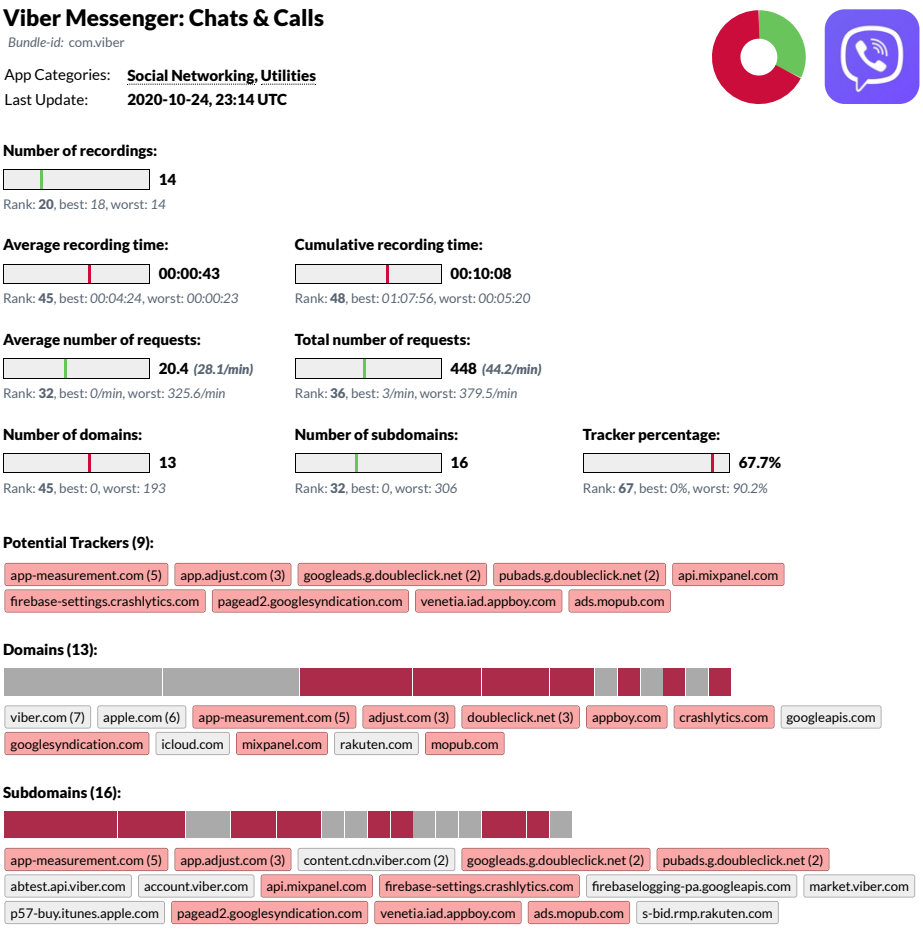

Fig. 2. App results overview with potential trackers highlighted in red.

This result, however, is biased by the skewed distribution of app categories over regions. The number of games is much higher in the group *Europe* (eight games whereas the other regions only have four games each). If we compare by category (cf. Table 3), we see that the *Games* category is one of the worst in terms of tracking. Additionally, the four worst apps overall are in the *Europe* region. All four apps (three games, one weather app) connect to at least 138 different domains each.

4.3 Comparison: iOS 13 vs. iOS 14

Our last evaluation example is a comparison study on differences between major iOS versions. The evaluation is performed on Dataset D1. This study intends to evaluate Apple's newly introduced "App tracking controls and transparency"

Table 3. App categories (in avg or avg—max). An app can have up to three categories.

Category	Req. / app	Req. / min	Domains	Tracker
Books (2)	52.8	49.3—57.2	15.0—16	33—50 %
Business (10)	23.1	27.3—60.0	9.2—18	**44**—75 %
Education (8)	24.3	19.5—54.5	8.6—14	27—60 %
Entertain. (23)	49.0	32.3—108.5	12.5—46	37—**92** %
Finance (10)	47.2	47.3—88.4	10.8—25	35—67 %
Food & Drink (5)	38.3	38.5—60.7	14.2—21	**44**—65 %
Games (24)	**164.7**	**59.3—417.3**	**36.2—184**	**53—81** %
Health & Fit. (14)	22.8	29.9—147.7	7.9—19	19—66 %
Lifestyle (25)	49.2	55.1—168.8	13.6—46	34—68 %
Medical (11)	39.3	18.9—42.9	9.0—20	18—55 %
Music (7)	53.6	40.0—59.4	12.4—26	42—**92** %
Navigation (8)	35.7	38.4—108.8	8.4—21	27—54 %
News (7)	67.2	31.6—48.4	18.9—73	42—67 %
Photo & Vid. (10)	25.9	27.9—53.4	8.7—18	30—55 %
Productivity (18)	40.5	36.0—95.0	10.6—34	23—53 %
Reference (5)	34.1	24.8—41.4	8.8—14	19—50 %
Shopping (12)	59.4	54.9—156.3	13.0—25	**45**—65 %
Social Netw. (19)	24.6	32.0—168.8	7.9—28	27—75 %
Sports (3)	42.8	53.9—82.5	15.7—23	30—66 %
Travel (9)	**69.9**	**84.5—445.7**	**24.6—138**	34—60 %
Utilities (17)	23.8	31.9—71.5	9.9—34	30—66 %
Weather (2)	**171.8**	**229.7—445.7**	**70.0—138**	22—44 %

Table 4. Comparison between iOS 13 and iOS 14 (in total, or avg—max).

OS	Rec.	Req.	Req. / min	Domains	Subdomains	Tracker
iOS 13	549	51714	53.6—459.2	19.9—193	37.3—314	35.6—90.8 %
iOS 14	543	50581	51.2—302.6	20.1—206	38.9—351	34.0—89.2 %

feature. Our assumption is that the introduction of that feature incentivized developers to make changes to their apps' tracking functionalities. We hypothesize that these changes will reduce the number of connections to tracking domains.

Our results, which show only negligible differences, do not support this assumption. Table 4 suggests that iOS 14 recordings did contact slightly more unique domains – on average, an additional 0.2 domains (+ 1.0 %) and 1.6 subdomains (+ 4.3 %). Meanwhile, the tracker percentage dropped by 1.6 %. We took a closer look at individual apps and chose four high-credibility commercial apps, and three tracking-intensive games (cf. Table 5). These results demonstrate that there is no clear trend towards less tracking. Some apps seem to use more tracking on iOS 14 than on iOS 13. With the exception of Google Chrome, big companies seem to have reduced tracking. We expect to see a more drastic change once the announced privacy features are in effect.

Table 5. App comparison between major iOS versions (iOS 13 ± difference in iOS 14).

App	Req. / min	Domains	Subdomains	Tracker
IKEA	$18.3 - 4.9$	$6 + 0$	$11 + 1$	$35.4 - 8.3\,\%$
McDonalds - Non-US	$59.3 + 3.2$	$21 - 4$	$35 - 9$	$44.4 - 14.4\,\%$
Microsoft Teams	$47.5 - 25.7$	$10 - 3$	$22 - 10$	$13.1 - 2.0\,\%$
Cube Surfer!	$415.5 - 199.1$	$193 - 38$	$314 - 43$	$45.8 + 7.1\,\%$
Spiral Roll	$150.8 + 151.8$	$170 + 36$	$272 + 79$	$48.0 - 0.3\,\%$
Stack Colors!	$117.4 + 158.4$	$175 + 10$	$282 + 38$	$48.0 + 0.8\,\%$
Google Chrome	$25.8 + 25.5$	$8 + 33$	$17 + 67$	$12.1 + 35.4\,\%$

We added Google Chrome to highlight a potential caveat when conducting user studies with *AppChk*. Chrome is a browser that displays user-content. Most of what a user does in Chrome should not be assigned to the application itself but the requested website. Everything a user does during a recording, will influence the evaluation results. Even though user-content centered apps, such as Google Chrome, are more prone to error to a user's actions, other apps may experience similar traits. For example, network request can be triggered by many different environmental factors, such as daytime, location, WiFi connection, or individual system preferences. This shortcoming can be mitigated with more recordings as these would filter out outliers.

5 Discussion

Meaningful recordings depend on a high coverage. In our case studies, the recordings span 69 sec (D2: 141 sec) on average. In cases where an app presents a login screen, the average recording time drops to 30 seconds. Previous studies tested apps for 4 min [12] (normal usage) or 5 min [11] (random execution), i. e., which suggests that we should instruct app testers to use apps for longer periods of time. On the other hand, Kurtz et al. found that 77.7 % of apps communicate within the first 30 sec after launch [11].

AppChk can not detect whether communication with a third party resulted in an actual privacy violations (personal data being exposed). This kind of analysis requires in-depth inspection with specialized tools such as a MitM proxy (`https://mitmproxy.org`). The limitation to focus on uncovering connection attempts is a conscious design choice balancing the utility of the recordings and the privacy of the users.

Further, the *AppChk* app can not detect or prevent deliberately hidden or malicious information sharing. For example, data can be exfiltrated by hiding the request in an innocuous first-party domain requests. Resolving the destination of CNAME records is currently not supported but will be added in future work.

AppChk can not determine the origin of a network request. A system process or background app may interfere and inject wrong domains into the recording. Long-term recordings, which are not discussed in this paper, allow users

to capture background activity. Background recordings could be used to reduce attribution errors by establishing ground truth for device-specific anomalies.

Tracker detection is currently done manually but could be automated to provide an always up-to-date tracking providers list. Further research is needed to compare the results of *AppChk* (iOS) to the tracking list of εxodus (Android).

We have shown that *AppChk* can be used to compare regional differences. Other interesting comparisons such as free vs. paid apps, correlation between app ratings and privacy, or changes in tracking after the introduction of privacy features can be introduced in the future. Further, we consider to integrate temporal analyses to detect trends, for instance, to find apps that improved recently. Uncovering such trends could help users choose one of multiple related apps.

6 Conclusion

AppChk is an easy-to-use tool to improve the transparency of iOS applications. Our platform allows users and privacy advocates to analyze mobile network traffic on the device (*AppChk* app) and share the results with our evaluation website (*appchk.de*). The *AppChk* app does not rely on deep packet inspection, TLS interception, a Jailbreak, or external servers, and it uses only well-documented APIs to be future-proof for upcoming iOS updates. This allows users to conduct a study immediately after a major OS update.

The *AppChk* website is built on the premise to provide comparable results. The website allows users to rank and compare apps and trackers.

AppChk can be used for app-group comparisons, highlighting systematic deficits, such as in the gaming category. The games considered in our study contact on average 36.2 domains with 53 % of the traffic being directed to tracking domains. Moreover, during the course of our experiments we identified seven new trackers which are not present in current tracking lists such as *EasyList*.

AppChk fosters the idea that research can be an ongoing citizen science project, with enthusiastic people who are willing to contribute recordings on a regular basis. More people can test more applications in less time and keep the data up to date which results in better privacy for everyone. The results aid users in making an informed decision about whether an app respects their privacy and leads to public visibility and increased transparency. Ultimately, the improved transparency may create a competition between app vendors and incentivize them to reduce tracking.

Acknowledgements

This work received grant support from the German Federal Ministry of Education and Research (BMBF).

References

1. Amrein, S.: Does your phone spy on you? Master's thesis, ETH Zurich (2016)
2. Claesson, A., Bjørstad, T.E.: "Out of Control" – A review of data sharing by popular mobile apps. Tech. rep., Norwegian Consumer Council (January 2020)
3. Döbelt, S., Halama, J., Fritsch, S., Nguyen, M.H., Bocklisch, F.: Clearing the Hurdles: How to Design Privacy Nudges for Mobile Application Users. In: HCI International Conference. pp. 326–353. Springer (2020)
4. Exodus Privacy: εxodus (2017), https://exodus-privacy.eu.org/
5. Fowler, G.A.: I checked Apples new privacy nutrition labels. Many were false. (2021), https://www.washingtonpost.com/technology/2021/01/29/apple-privacy-nutrition-label/
6. Han, C., Reyes, I., Feal, Á., Reardon, J., Wijesekera, P., Vallina-Rodriguez, N., Elazari, A., Bamberger, K.A., Egelman, S.: The Price is (Not) Right: Comparing Privacy in Free and Paid Apps. PoPETs pp. 222–242 (2020)
7. He, Y., Yang, X., Hu, B., Wang, W.: Dynamic privacy leakage analysis of Android third-party libraries. JISA pp. 259–270 (2019)
8. Kollnig, K.: TrackerControl (2019), https://github.com/OxfordHCC/tracker-control-android
9. Kurtz, A., Gascon, H., Becker, T., Rieck, K., Freiling, F.: Fingerprinting Mobile Devices Using Personalized Configurations. PoPETs pp. 4–19 (2016)
10. Kurtz, A., Troßbach, M., Freiling, F.: SNOOP-IT: Dynamische Analyse und Manipulation von Apple iOS Apps. Sicherheit 2014 (2014)
11. Kurtz, A., Weinlein, A., Settgast, C., Freiling, F.: DiOS: Dynamic Privacy Analysis of iOS Applications. Tech. Rep. CS-2014-03, FAU Erlangen-Nürnberg (June 2014)
12. Leung, C., Ren, J., Choffnes, D., Wilson, C.: Should You Use the App for That? Comparing the Privacy Implications of App- and Web-based Online Services. In: Proceedings of the 2016 IMC. pp. 365–372 (2016)
13. Liu, X., Liu, J., Zhu, S., Wang, W., Zhang, X.: Privacy Risk Analysis and Mitigation of Analytics Libraries in the Android Ecosystem. IEEE Trans. Mob. Comput. pp. 1184–1199 (2019)
14. Maass, M., Walter, N., Herrmann, D., Hollick, M.: On the Difficulties of Incentivizing Online Privacy through Transparency: A Qualitative Survey of the German Health Insurance Market. International Conference on Wirtschaftsinformatik (2019)
15. Maass, M., Wichmann, P., Pridöhl, H., Herrmann, D.: PrivacyScore: Improving Privacy and Security via Crowd-Sourced Benchmarks of Websites. In: Annual Privacy Forum. pp. 178–191. Springer (2017)
16. Papadopoulos, E.P., Diamantaris, M., Papadopoulos, P., Petsas, T., Ioannidis, S., Markatos, E.P.: The Long-Standing Privacy Debate: Mobile Websites Vs Mobile Apps. In: Proceedings of the 26th International Conference on World Wide Web. pp. 153–162 (2017)
17. Perta, V.C., Barbera, M.V., Tyson, G., Haddadi, H., Mei, A.: A Glance through the VPN Looking Glass: IPv6 Leakage and DNS Hijacking in Commercial VPN clients. PoPETs pp. 77–91 (2015)
18. Yang, Z., Yue, C.: A Comparative Measurement Study of Web Tracking on Mobile and Desktop Environments. PoPETs pp. 24–44 (2020)