

How Many Hands in the Cookie Jar? Examining Privacy Implications of Popular Apps in India

Koustuv Kanungo, Rahul Khatoliya, Vishrut Arora, Aairah Bari,
Arani Bhattacharya, Mukulika Maity, Sambuddho
Indraprastha Institute of Information Technology Delhi
New Delhi, India

{koustuvk, rahul19265, vishrut19399, aairah19003, arani, mukulika, sambuddho}@iiitd.ac.in

Abstract—Smartphone app usage has steeply risen in India in the past decade. But limited efforts in the past assess the privacy aspects of these smartphone apps. Many of these are used for common utilities and handle sensitive user data. Such sensitive data leaks can have a wide variety of consequences when exposed to untrusted players (e.g., repressive governments, data/content hosting companies, and other third parties). These could range from mere embarrassment to personal targeting and surveillance.

This paper presents a measurement study on the data collection and privacy considerations of some of the most popular apps on the Indian Google Play Store. We selected 24 apps, and analyzed their data collection behavior on phones as well as the security of the servers to whom they send the data. We also obfuscated the data being collected and sent to the backend servers. Interestingly, for the non-government apps we found that extensive (mostly personally identifiable information), often “unnecessary”, data collection is being performed. In other words, we observed that a lot of these apps work fine even without these pieces of sensitive information. We then classified the data collected as *necessary* or *not necessary* based on this information. Furthermore, we found that often such sensitive data may be available in plaintext to the intermediate players managing/deploying the hosting infrastructure. We also found that while the governmental services-based apps collect fewer such unnecessary data points, they often store the data on web-fronted back-end databases with little to no user authentication mechanisms enabled. We expect our study to enable better understanding among both users and app developers about the privacy implications of these data collection practices.

1. Introduction

Mobile apps have been dominating the lives of modern (often urban) Indians [78]. People use them for all sorts of applications, from mobile banking [40] and digital payments [36], [100] to tracking COVID transmission [38] and even investing in portfolios [42] (sometimes even cryptocurrencies [50]). Obviously, a lot of these applications involve handling sensitive private data. This naturally increases the risk of privacy violations— either due to security mishaps, collusion of the associated companies, or by coercion of repressive governments.

Several security mishaps, involving breaches of mobile app data have occurred before, both in India [43] and abroad [86], [104]. E.g., there can be cases where an app

can take more data than what is needed to function. This can lead to much more damage when there is a breach, as was the case of the multiplayer games [104]. Data leaks could also be facilitated by insiders. For example, app companies could themselves succumb to authoritarian coercion [87], [103] of repressive regimes. Alternatively, such companies could deliberately share such data to third-parties (governmental or otherwise) for financial gains [26]. In either case, data is leaked to third parties that the user was not aware of or had no desire to share.

Given such risks, smartphone users would like to learn what data is collected by apps and furthermore, whether all of it is necessary for the app to perform the operations that the user requires. This is especially true for apps that transport sensitive (or personally identifiable) information. Sometimes such apps present misleading information on their app store pages. Further, the download pages of the apps on app stores have a section detailing the data collected. However, this may be misleading in some cases¹. Hence, it is necessary to quantify app data collection practices, with a further bifurcation between what is necessary vs unnecessary for the apps to work, and the attack vectors they are subjected to.

Our work presents a privacy-related measurement study of apps in India that are popular, facilitate ease of necessary everyday operations (e.g. online payments), and require sensitive information to operate. We attempted to expose the risks a user faces when using such apps and whether all the data collected by the apps is essential or not. Our study included apps such as payment apps, government utilities, cryptocurrency exchanges, and netbanking apps. For each of these categories, we have chosen those apps that have a very large number of downloads (possibly the most popular ones).

For our research, we enumerated threats arising from three different kinds of adversaries that could obtain and exploit users’ private data. The first is an external attacker that compromises a company’s servers and gains access to sensitive data of users. Such adversaries may misuse the data for their own gains through identity thefts [37], [85], learning about users’ online spending habits, their precise geo-location [44], etc. In India, this has taken the form of increased ransomware attacks [47], [97], data breaches

1. E.g., the Data safety section on PayTM (a popular payment app in India [14]) contradicts our observations, elaborated in § 6.1. Firstly, PayTM needs at least the transactors’ details to facilitate transactions. Not only that, but we also observed that the device details are transmitted to third-parties using tracking APIs.

to sell data to brokers [110], and cyber attacks on critical infrastructure [39].

The second is a surveilling government that encroaches on users' privacy by coercing private companies (and possibly through collusion with public agencies under its aegis) [103]. Companies like Apple, Facebook, Google, and Reddit publish transparency reports on government requests for user data. There have been many such requests, particularly to Google, for reasons not stated [41], [46], [48], [51]. Such extensive data collection can lead to users being unfairly targeted, either via discriminatory factors, or false classification due to weak inferences from the data collected.

The third type of adversary could be the app company itself which could deliberately share sensitive user data with third-parties for financial gains [26], [33]. The latter could in turn use such sensitive data to bolster their own analytics [110], or even use it to affect services offered. The user may not have given consent for either their data to be shared, or the terms of the service may have changed [76].

Given such risks to users' privacy, we analyze twenty-four popular apps in India that require sensitive user data. For our research, we look at payment apps, government utility apps, cryptocurrency exchanges, and banking apps because these apps compulsorily require sensitive user data to operate and cannot be used anonymously. In addition, each of these apps also have more than a million downloads on the Play Store, which indicates their popularity and widespread use.

We inspected the network traffic generated by these apps. Since the apps internally rely on HTTPS, we relied on tools to decrypt their traffic, so as to analyze them. We observed the data being sent to the apps' servers and tried to correlate them to their appropriate privacy policies (often listed online) to see the transmitted data adhere to the said policies or not. We also performed static analysis on the apps' APKs in order to discover any additional user tracking or other misconfigurations or vulnerabilities. We further tried to see if suppressing or obfuscating the users' personally identifiable data renders the app dysfunctional.

Further, we also analyzed the security postures of the servers that the apps connect to. This included searching for open and vulnerable services, directory searches, and DNS subdomain availability to identify other vulnerable services in the subdomains corresponding to these apps, exploiting known vulnerabilities on these servers used, and input validation checks. We also looked for obvious vulnerabilities in the application's logic. Further, we inspected the available web interfaces and the API endpoints for issues like faulty user authentication or unintended access to other services on the server.

Finally, we scrutinized the privacy policies of the apps and compared them to multiple data regulations in India [27], [28], [73], to check for consonance and legal permissibility. This was motivated by the fact that the Digital Personal Data Protection Act [73] was passed in the third quarter of 2023, and it gives us a basis to judge the data collection behaviour of apps. We do note that the Act itself has not been enforced yet. We have also raised multiple appeals (Application number RBIND/R/E/22/04501, see Appendix A.2), under the Right to Information Act (RTI) 2005 [67] to the Reserve Bank of India (RBI) (the

financial regulations enforcing agency) to understand what measures are taken by the app companies to comply with the RBI regulations. We search for threat vectors based on mismatches between the apps' privacy policies, the data regulations and their implementations.

From our analysis, we discovered extensive data collection prevalent across most non-government apps. Interestingly, our experimental analysis revealed that the apps function correctly even when a lot of these are either obfuscated or simply dropped. Payment, cryptocurrency, three banking apps, and one government app also had built-in trackers that transmit users' app usage and system analytics to third parties. From our analysis of the server endpoints contacted by the apps, we found that all the apps utilized CDNs, however, the corresponding HTTPS encryption terminated at the edge servers. This is further elaborated in Section 7.2 This means that the CDN hosting company has the ability to snoop on the network traffic. On the other hand, this reliance on CDNs also ensured that there are no obvious vulnerabilities that could be used to escalate privileges and cause unauthorized access to user data. Some apps also mitigate the encryption issue by employing another layer of encryption inside the HTTPS connection. We also found via a written request to the regulatory authorities that there are no mechanisms to check for app compliance with the data regulations that are present in India. Thus, we expect this study to bring a better understanding of the state of digital privacy among privacy advocates and users of apps in India.

Briefly, our contributions are as follows:

- We obtained and categorized the data collected by the selected apps dealing with sensitive information in India. We achieved this through APK analysis and inspecting network traffic. We sorted the collected data into various categories, some of which include personally identifiable information. This novel effort quantitatively shows the amount of data that the apps collect.
- We observe that several payment apps, viz. PayTM, Mobikwik, and PhonePe, and cryptocurrency apps, viz. WazirX, Binance and CoinDCX collect "unnecessary" data beyond what is required for the app to work. Further, government apps such as Digilocker, several banking apps (Canara, ICICI and Axis), and cryptocurrency apps, use third-party trackers that expose users' app usage analytics (to the tracking companies). Interestingly, we observe that even after the *suppression* of "unnecessary" data, most apps work without any perceptible problems.
- We looked for vulnerabilities in the servers the apps contacted, using service scans, input validation checks, and statically analyzing the pages of the corresponding sites and the APKs.
- We point out that the data collection may not adhere to the guidelines set by the Digital Personal Data Protection Act and other data regulations, and the lack of supervision on such matters.

2. Related Work

We classify existing works into 4 major categories – vulnerabilities of apps, violations of privacy policies/regulations by apps, existing studies on applications'

TABLE 1: Coverage of related works.

Name	Client Side				Server Side	Cert. Pin.	Country	Attack Vector			
	Payment Apps	Govt. Apps	Banking Apps	Crypto Apps				External attacker	App company	3rd party collusion	Surv. govt.
Kumar et al. [90]	✓	×	×	×	×	×	India	✓	×	×	×
Mahmud et al. [95]	✓	×	✓	✓	×	×	Global	✓	×	×	×
Agarwal et al. [75]	✓	✓	✓	✓	×	×	India	✓	×	×	×
Mahmud et al. [96]	×	×	×	×	✓	✓	Global	✓	×	×	×
Chen et al. [81]	×	×	✓	×	×	×	Global	✓	×	×	×
Bosu et al. [79]	×	✓	✓	✓	×	×	Global	✓	×	×	×
Calciati et al. [80]	✓	✓	✓	✓	×	×	Global	×	✓	×	×
Nguyen et al. [101]	✓	✓	✓	✓	×	×	Global	×	✓	×	×
Samarasinghe et al. [109]	×	✓	×	×	×	×	Global	×	×	✓	×
Leith et al. [92]	×	×	×	×	×	×	Global	×	✓	×	×
Leith et al. [93]	×	✓	×	×	×	✓	EU	×	×	✓	×
Nguyen et al. [102]	✓	✓	✓	✓	×	✓	Global	×	✓	×	×
Du et al. [82]	✓	✓	✓	✓	×	×	Global	×	✓	×	×
Leith et al. [94]	×	×	×	×	×	×	EU	×	✓	✓	×
Our work	✓	✓	✓	✓	✓	✓	India	✓	✓	✓	✓

privacy considerations, and automated frameworks to assess privacy issues.

Vulnerabilities of apps: Kumar *et al.* [89] analyze the security of UPI, a framework offered by the Indian government to facilitate digital payments. They identified threats from third-party apps that may be able to intercept the SMS messages pertinent to payment apps, *e.g.* those bearing the registration OTP. Permissions to read SMS messages are commonly (and often unsuspectingly) granted to many apps. However, they note that recent versions of the UPI framework have fixed this problem. While their study primarily focuses on security aspects of UPI, ours looks at what kind of information is leaked by popular apps (mentioned earlier).

Others focus on security aspects of communication between user apps and their servers [75], [95], [96]. However, they do not comment on privacy issues beyond the security issues. Finally, Chen *et al.* [81] survey the type of vulnerabilities seen in Android banking apps. They look for data transmission-related security issues. While they analyzed over fifty apps, they look for only one kind of vulnerability, *i.e.*, misconfigurations that can leak sensitive plaintext data during transmission.

Violation of privacy policies and/or data regulations: We also surveyed several papers that try to identify cases where apps violate their its own privacy policies. Some such as Bosu *et al.* [79] identify techniques of inter-app collusion to leak user data without being easily detected. They found that several popular apps on the Play Store engaged in sharing data with one another via inter-app channels. This happens stealthily, without the users’ knowledge (hence violating their privacy). Others, *e.g.* Calciati *et al.* [80], identify the privacy leakages due to the automatic granting of permissions to Android apps. They found 17% of apps in their dataset (with > 1M apps) requesting permission in ways that would not notify the users. These apps then, without the users’ knowledge, exfiltrate data to backend servers and trackers. While meaningful, their findings are not directly relevant to the privacy leakages of popular Indian apps. We study popular Indian apps that deal with sensitive private data.

Studies on apps’ privacy: There are prior efforts in a similar vein as ours. *E.g.*, Nguyen *et al.* [102] compare

the users’ perception of apps’ data collection behavior, to what is actually being collected. They found a mismatch between the two for over 75% apps. About 38% of those shared the collected data using third-party analytics and advertising libraries.

A number of other works also study the privacy implications of using specific service-oriented apps. *E.g.*, Samarasinghe *et al.* [109] present a measurement study of government apps all over the world that involve third-party tracking. They do consider certain Indian apps, but those are mostly obscure and not very popular, *e.g.* those that have under 20 downloads. This is likely because their methodology does not involve bypassing certificate pinning, a mechanism commonly employed in most apps these days (including the most popular ones, and those that deal with sensitive user data).

Similar work has been carried out by Leith *et al.* [92], [93]. They look at privacy aspects of web browsers, contact tracing apps, and even the privacy of phones’ OSes themselves. Their work is the most similar to our work. However, theirs did not involve Indian apps. It is even more selective in the type of applications they analyze. *E.g.*, they focus primarily on contact tracing apps, web browsers, smartphone OSes *etc.*, but do not cover the kind of apps our work does, *i.e.* those that are popularly used by average smartphone users and often involve the collection and transmission of sensitive data. In addition, we could not replicate their analysis methodologies, as their tools for automated removal of certificate pinning, no longer work with current Indian apps. Such automated tools find one certificate pinning check to disable. The apps usually have several more which go undetected by such tools. Finally, Nguyen *et al.* [101] show how Europe’s GDPR consent requirements are frequently violated by apps.

Automated frameworks to assess privacy issues: Frameworks like FlowCog [82], use static analysis (on the APKs) to identify the various code-paths, corresponding to the user’s data collection. Next, such code-paths are classified as legitimate or otherwise, depending upon what is presented in the user interface. On the contrary, our work involves correlating app execution (*e.g.* which feature of the app is activated, what page the user visits *etc.*) to dynamic network traffic analysis and the clauses laid out

in the apps’ privacy policies. We try to spot the violations in such policies, based on the observed network traffic and the apps’ execution events. Additionally, `FlowCog` has only been tested on a set of randomly collected apps from Play Store and some Chinese app stores. It does not provide any insights regarding the privacy aspects of apps used in various countries (including India).

A quick overview of the literature surveyed is provided in Table 1. We specify the attack vectors being analyzed, the scope of apps of the work, the country focused on in the work and whether apps with certificate pinning are included or not.

3. Background and Threat Model

In this section, we describe the current situation in India regarding app privacy, and we state our assumptions regarding definitions and threat model.

3.1. Privacy in the Context of Mobile Apps

Privacy as a concept has been discussed and debated upon for years and can vary in meaning from person to person, or situation to situation [83], [92], [99], [111]. However, a common threat across these cases is that privacy often entails the ability of a person to reveal or withhold information regarding themselves, instead of other people or parties being able to do that without the person’s consent, and be able to remain anonymous if they wish to do so. Hence, in this work we refer to privacy as the users’ choice/ability to control what data is to be revealed, and to which party, in the entire chain of transactions. Others have adopted similar connotations [83], [92], [93], [111].

3.1.1. State of app usage in India. App usage in India has soared [78], particularly after a big drop in mobile data plan prices by cellular service providers [45], [98]. On top of that, events with nation-wide effects like demonetization [36], [100], and those with global ones, like COVID pandemic lockdowns, have driven up app usage to cater to the situation [36], [38], [49]. Many of these mobile apps help with day-to-day activities and utilities, such as payments, banking, government / municipal services, *etc.* While these apps are very convenient for users, they often require linking user credentials with personal identifiers in order to perform their tasks – *e.g.* bank account numbers and transactors’ data, when using payments apps as well as location information and medical histories, when using contact tracing apps.

3.1.2. Implications and motivation. Any leak of sensitive personally identifiable information could be abused. Unnecessary data collection puts users and/or their data at greater risk. There are multiple ways through which the collected data could be abused. Unauthorized leaks that may irritate or embarrass the user [23], further exacerbated through social/physical targeting (including, by repressive governments) [25], [34]. Such dangers are compounded by the fact that novel sensitive insights can be derived from already existing data [88], revealing information the user may not want to share with others. Moreover, the information published by app developers themselves could

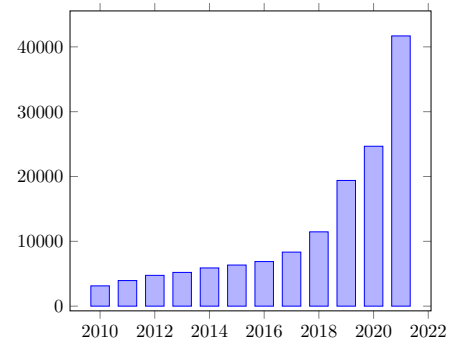


Figure 1: Number of legal disclosure requests to Google over 10 years.

be misleading and contradictory to the actual policies and behavior of the app, as already mentioned in § 1.

Further, once the data reaches the app servers, it is still vulnerable to other attackers. These include other companies that might collude with the app company to monetize the data. It may also include surveillance agencies that could coerce the company into giving up the data.² These may also include external attackers exploiting the systems and their databases’ vulnerabilities to leak users’ data.

Therefore, in this work we focus on what kind of data is collected by the apps, what kind of security mechanisms are employed and if their privacy policies conflict against the current regulations.

3.2. Adversaries and Their Capabilities

In order to contextualize our research, we first take a look at the potential threat vectors for apps’ users. Most apps internally rely on web-based protocols [56] and thus often the connections to their cloud storages terminate on CDN edge servers. In the case of most apps, the user utilizes the app, the app makes a connection to the CDN used by the app, which then communicates with the origin server of the app. We do not consider attacks where the adversary needs to approach or manipulate the user or user’s device specifically, like compromising the user device’s OS, installing malware on user’s device, or phishing the user for sensitive information. We only consider attacks which apply to all the app users in general without having to target any user specifically. This leaves us with attack vectors on the communication between users and CDN, CDN and app server, as well as the data storage on CDN and app server.

Thus, we surmise three adversaries that can affect the CDN and the app server: (1) the external attacker, (2) a motivated third-party attacker that can influence the CDN or the app company, or (3) a coercive government that can force the CDN or the app company to hand over the data.

3.2.1. Surveillance. A government that uses surveillance could compel any organization within its jurisdiction into revealing users’ data. There are two ways for this to happen. Firstly, through collusion between government departments where data collected by one government department gets shared across several others. Secondly, by

² In fact, the user may also wonder whether all the sensitive information is an imperative for the app to function correctly or not.

coercing private parties to give up customer data to not face negative consequences to their business. The coercive government is the strongest type of attacker. The user may or may not be alerted regarding this data collection. A company might be unable to deny the request as it may affect their business. This attacker can coerce any participant in the network, including the app company and any hosting services, making its reach the widest [24].

3.2.2. Motivated third party. *Motivated attackers* include attackers that can coerce or persuade the app (and/or its hosting) companies to share data for financial gains [26], [30]. If the data protection policies of the country are not strong enough, then it is possible for these attackers to influence and extract data from the companies, thereby making privacy violations much easier.

3.2.3. External attacker. An external attacker is a malicious party that has no correspondence with the user or any company involved in the network communication. This attacker tries to breach user privacy by attacking the network infrastructure being used for the communication, which includes the origin server used by the app company, and any CDN edge servers being utilized by the company. This is achieved by probing the infrastructure through network reconnaissance to spot exploitable vulnerabilities. Further, the attacker can also analyze the public-facing facilities like apps and websites, to find misconfigurations in authentication processes. If this attacker is successful, then they can access the user data stored on the servers, and leak them to the world in the form of data breaches, utilize them for nefarious purposes like identity theft, or illegally sell it, unbeknownst to the app company or the user.

4. Methodology

Our work deals with the possible avenues of data leaks by the popular apps, *viz.* the client app running on the smartphone, and their backend servers. We selected a total of 24 apps across four categories summarised in Table 2, particularly those with large download counts, or those that are indispensable for the user’s needs. These include payment apps, government utility apps, crypto exchanges’ apps, and netbanking apps. We believe that the potential privacy leaks in apps, owing to their high user count and the sensitive information they handle, are a good representation of how vulnerable Indian customers are, in general. We conduct our experiments following ethical guidelines as per prior research [105], [106].

4.1. Selection of Apps

In order to analyze the privacy threats to users in India, we required selecting apps that deal with sensitive user-identifiable information. Hence, we have selected apps that would be hard to use with fake accounts and would need real user-identifying information to function properly. Such apps include online payments, government services, online banking, and cryptocurrency exchanges. Apps in these categories use information like monetary transactions, bank account details, location, medical history, Aadhaar ID, *etc.* Further, we selected apps with at

least a million downloads on the Play Store. Our observation concerns a significantly large number of users.

TABLE 2: A list of apps that we analyzed along with their services

Category	App	Use of App
Payment	PayTM - 10.23.0 (net.one97.paytm) Mobikwik - 22.54.4 (com.mobikwik_new) Freecharge - 15.1.0 (com.freecharge.android) Google Pay 171.1.4 (com.google.android.apps.nbu.paisa.user) PhonePe - 4.1.47 (com.phonepe.app) BHIM - 3.1.1 (in.org.npci.upiapp)	Online payment
Govt services	MCD - (com.mcd.mcsuite) DJB mSeva - 2.2.7 (com.tcs.djb.mseva) CESC - 2.1.4 (cesc.co.in) MahaDiscom - 9.8 (com.msedcl.app) BSES Rajdhani - 10.1 (com.bses.bsesapp) mPassport Seva - 6.5 (gov.mea.psp) Aarogya Setu - 2.0.3 (nic.goi.aarogyasetu) Digilocker - 7.3.2 (com.digilocker.android)	Municipal Corporation of Delhi Delhi water service Kolkata power distribution service Maharashtra power distribution service Delhi power distribution service Passport service Government COVID contact tracing Government certification service
Crypto exchanges	CoinSwitch - 4.9.4 (com.coinswitch.kuber) WazirX 2.35.2 (com.wrx.wazirx) CoinDCX - 5.09.004 (com.coindcx.btc) Binance - 2.65. (com.binance.dev)	Cryptocurrency exchange
Netbanking apps	Canara Bank - 3.1.52 (com.canarabank.mobility) HDFC Bank 11.1.7 (com.snapwork.hdfc) SBI Bank - 1.23.63 (com.sbi.lotusintouch) IDBI Bank - 2.6 (com.snapwork.IDBI) ICICI Bank - 16.1 (com.csam.icici.bank.imobile) Axis Bank - 8.3 (com.axis.mobile)	Banking service

4.2. Client App Analysis

Our goal is primarily to identify data leaks that would affect all users using the app. Thus, we inspect the network traffic to spot sensitive user information like details of financial transactions, app usage, sensitive medical information *etc.* We also statically analyzed the apps’ APK to find potential data leaks due to vulnerabilities or third-party trackers.

4.2.1. Decrypting network traffic. Our analysis involves network inspection of app traffic on Android mobile phones. The app traffic is encrypted with TLS and so inspection requires decrypting it. We pass the traffic via a MITM proxy, `mitmproxy` [63] (under our control), to decrypt and inspect the contents. However, these days

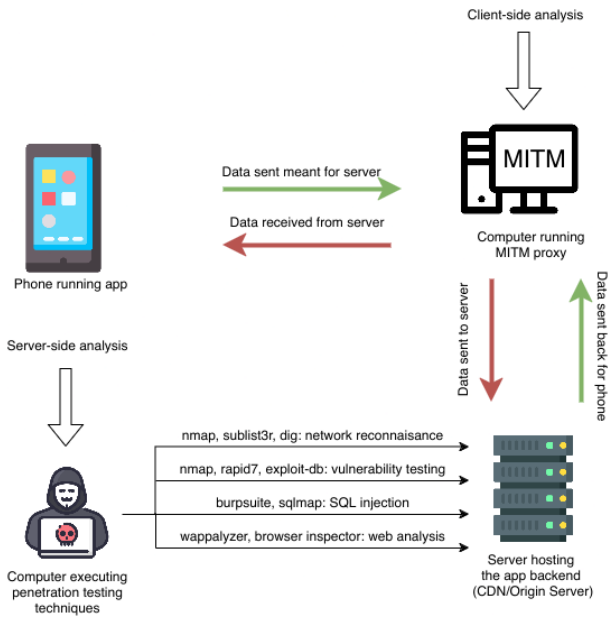


Figure 2: Experimental setup for snooping on app traffic (with mitmproxy).

apps often come bundled with their own certificate stores and verification processes. Thus, the MITM proxy’s certificate is not accepted, even when the root certificate was manually installed on the phone. To get around this, we perform several techniques, which apply as needed. The techniques are:

1. *Certificate unpinning*: To do this, we modify the app APKs manually, using `apk-mitm` [54]. This tool decompiles APKs into an intermediate Java representation called `smali`. In these `smali` files, we search for procedures that invoke the certificate pinning functionality and disable it. For this, we searched the `smali` code for references to classes that are likely used for performing certificate pinning (e.g. `CertificatePinner`, `CertPinner` etc.). Next, we identified the `check()` methods and programmatically bypassed them. Finally, we recompiled the `smali` files and installed them onto the target device.

Alternatively, applications like `Frida` [61] and `apk-mitm` automate the process of finding such `check()` methods. However, they disable *only* the first one they identify.

2. *Disabling integrity checks*: In the cases of banking apps and BHIM UPI app, disabling certificate pinning alone was insufficient. In these cases, the apps have in-built integrity checks, that detect if they have been signed by someone other than the developer, and report appropriate error messages. Thus we needed to disable such integrity checks as well. We used two methods to achieve this. In the first method, we searched the `smali` files for procedures that report these error messages, and disabled them one at a time to find the procedure that actually performs the integrity check. In the second method, we used the Zygisk functionality of Magisk rooting framework [74] to manipulate the process memory directly at runtime, hence bypassing the integrity check, without requiring to reverse engineer the APKs at all.

For this, we used phones running different versions of the Android OS (Android 6.0.1, Android 10, and An-

droid 12). Depending upon the OS, different features of the app may be available which in turn would lead to sharing/leaking different data in each case. As described later, we did not see much diversity in the kind of data that is shared/leaked (with respect to the Android versions). In order to verify the prevalent privacy threats, we manually activated the primary functions of the apps, and then observed the data sent during the working of the app.

4.2.2. *APK static analysis*. Besides analyzing the network traffic of the apps, we also looked for suspicious permissions requests, API-level trackers, and code-snippets. In particular, we look for trackers, which are libraries that provide functions for facilitating the collection and transmission of various metrics, from app usage to user habits. Such libraries can be included for the app developer to collect metrics to improve their app, such as `Firebase`, and does not count as a privacy violation. However, if this library is provided by a third party to transmit data to their servers, this reveals a vulnerability in the form of collusion of the app company and the third party. For this purpose, we use `MobSF` [64] and `WebXRy` [71]. `MobSF` analyzes the APKs and identifies their various components. In these, we looked for:

- Trackers revealing transmission of data between the app and third parties.
- Permissions giving insight into the data that can be exfiltrated from the user.
- Hardcoded strings, that might contain API keys for accessing the database on the app server.

We supplemented our findings using the *Exodus Privacy Project* [59]. It is an online database of analysed APKs (derived from the Play Store), that reports embedded third-party trackers. We list the trackers belonging to tracking companies in Tables 4, 5, 6 and 7.

4.3. App Server Security Analysis

The external attacker, according to our threat model, attempts to exploit the apps’ servers to compromise the users’ privacy. Hence we looked into the security measures employed by the servers to thwart such threats.

4.3.1. *Network service discovery*. At the beginning of the analysis, we noted the IP addresses contacted by the app during use to find the servers. After we obtained the list of IP addresses, we performed our analysis:

- We checked the ownership of the IP address using `whois` [72] in order to determine if it is a CDN, or potentially an origin server.
- We tried to obtain more domains associated with the server, and thus discover a larger possible attack surface, using `sublist3r` [70]. We used the tool to enumerate the possible subdomains related to the apps’ domain, and also tried performing DNS zone transfers to find hidden subdomains.
- From the list of obtained domains, we checked if the discovered domains were active by running them through `dig` [57], and discarding the domains that returned `NXDOMAIN`, and retrieved the IP address of the active domains.
- On the collected set of IP addresses, we attempted to identify the open TCP and UDP ports using port

scanning tools like `nmap` [66]. From the list of open ports we also attempted to check if there were some that were not usually accessed by the apps.

4.3.2. Vulnerability discovery. We tried to connect to the open ports found on the servers, by using common network tools like `telnet` and `ftp`, or simply connecting to them through a web browser. Through these efforts, we tried to see if some of these ports are using HTTP (or similar) protocols. A misconfigured server can leak data to an unauthorised individual who connects to these ports. We tried to discover services that could be vulnerable to recent high-profile vulnerabilities, like `log4j` [31]. To that end, we checked if the discovered services bore known vulnerabilities listed in various source [60], [62], [68], [69]. We immediately stopped the experiment if we found any successful method to obtain access. Since this carries the risk of privacy violation of oblivious parties, we used our own accounts created in the apps for this purpose. We attempted to access data only for these accounts.

4.3.3. Input sanitization. Next, we used `Burp Suite` to intercept the communication between the app and the app server. We intercepted the packets being sent during actions that request data and modified the data fields with random SQL syntax strings. We checked if these modified packets returned any SQL messages or outputs as responses. We also used the tool `sqlmap` to automate common SQL print sequences on the app server IP address, and to check if the database is encrypted. We only wanted to verify whether the inputs are sanitized or not so that modified inputs do not cause unwanted database actions that leak data. Hence our manual inputs were simply SQL `PRINT` statements, and we set `sqlmap` to not dump any data. This allows us to test for misconfigurations without actually launching overt SQL injection attacks that would leak private user data.

4.3.4. Web analysis. We accessed the corresponding web pages available for the apps using a desktop browser, and analyzed the HTML and JavaScript scripts used by the web page for any possible data leaks and/or vulnerabilities that could be exploited. To accomplish this, we searched the code sections that handled logging in and transmission of other user data, and looked for any potential missteps that compromised the security of the login process, or exposed keys to access the database on the app server. As earlier, we used our own accounts and data in the apps for this experiment. Hence, data leaks would be limited to our accounts.

5. Ethical Considerations

A part of our analysis involved performing various types of scans to assess the security postures employed by the hosting servers. While we undertook these for research purposes, we recognize that these experiments might cause harm to such infrastructures. These experiments also have the potential to reveal sensitive personal information of other users of the app. Hence, we designed the experiments keeping in mind the ethical guidelines put down in the Belmont [84] and Menlo [77] reports.

The first requirement to keep in mind is *respect for persons*. The rights of humans as autonomous decision-makers need to be respected. This can take the form of informed consent of all participants, but given the scope of the experiment, this is not really feasible. However, Salganik [108] explains that this isn't always necessary, and the principle can still be adhered to by obtaining some consent for most of the activities. Therefore, we first sought consent to conduct the experiments from the respective companies hosting the services. We read the company policies of our chosen apps, and found that some carry either bug bounties or responsible security vulnerability disclosure policies. These include `PayTM`, `Mobikwik`, `Aarogya Setu`, `Google Pay`, `PhonePe`, `CoinSwitch`, `Binance`, `WazirX` and `Axis Bank`. So we initiated the experiments for these apps intending to report any vulnerabilities found immediately according to their policies. For the remaining apps, we tried to contact the companies for consent through available public channels. For the banking apps, we have contacted the respective managers of the regional branches, who have forwarded our request for consent to the respective infrastructure maintenance team of the company. For the apps who do not have any explicit contact to forward infrastructure-related queries to, we have sent requests for consent through their available customer helplines. It must be noted, most of these apps that do not present means of reporting vulnerabilities, are often hosted on hosting platforms like `CloudFlare` [32] and `Amazon Cloudfront` [29]. These services also have their own bug bounties and vulnerability disclosure schemes. Had we discovered vulnerabilities in these hosting platforms, we would have directly reported it to them via the said schemes.

In addition, we used our (specifically assigned) public IP address for our experiments, making it easier for the relevant companies to reach out to us in case of any issues.

To make up for the lack of consent from all entities involved, we then use the second principle of *beneficence* to design the experiments. This principle seeks to minimize risk and maximize the benefits of performing the experiments. In this case, we wish to learn about the possibility of privacy violations and have no need to view the private information of other users. To abide by the principle, we created our own accounts on the apps and focused our experiments on these accounts alone. Thus, the privacy of other users of the app wasn't violated, even if an attack on privacy was successful.

Further, to minimize harm to the infrastructure and the operators, we limited our service scans as well as DNS requests, to the slowest possible speeds and as infrequently as possible. We also conducted the scans at late nights, so that the additional load does not impact the fewer number of users. These measures would ensure that the operation and performance of the infrastructure remains unaffected. In addition, while we also looked for vulnerabilities, we did not actively exploit them. We simply conducted service scans to check for vulnerable versions. We did analyze weaknesses in the authentication mechanisms of the app and the browser page, but as mentioned prior, we used our own accounts in the apps for this purpose.

Finally, for checking input validation, we modified the data fields to create benign database commands to

prevent dumping sensitive information. We simply used SQL PRINT statements instead of SELECT, and set the `sqlmap` tool to just check the database if it is encrypted, instead of dumping any data.

The third principle of *justice* states that the beneficiaries of the experiments are the same people that bear their risks. In our case, the risks are faced by the users of the app, as well as the operators of the infrastructure. The network analysis experiments do not pose any new risks, as it is just documenting existing behaviour without exploiting any vulnerabilities. However, the server side analysis exposes some vulnerabilities and misconfigurations, which could be exploited by an attacker to leak sensitive information. Hence, we reported our findings to the respective organizations as well as CERT-IN. By the time of our reporting, we found that the power distribution companies had fixed their misconfigurations. The authentication schemes cannot be trivially bypassed, and no internal structure is being broadcast in public. In the case of MCD App, the property tax details are being considered as publicly available data by the government, hence no steps were taken as this was not considered as a breach of privacy or security. Hence, any attacker planning to misuse personal information will not be gaining access to data they were unable to access prior because of our work. We hope that the results we obtained through the experiments serve to enlighten the population regarding the privacy risks of using the selected apps, as well as the infrastructure operators so that they have the capability to mitigate these risks.

Finally, experimental designs also comply with the guideline regarding *respect for law and public interest*. Our experiments have minimal effect on the performance of the infrastructure, as well as the public IP is visible for easy contact in case of issues. We believe that benefits associated with our findings outweigh the risks and is in the interest of preserving the privacy of citizens.

We have also obtained approval from our IRB to move forward with our experiments. The details are being withheld to safeguard our anonymity. An extended discussion of the ethical considerations is provided in Appendix ??.

6. Data and Vulnerabilities Observed

Through our analysis, we found that the apps require a lot of user data to operate. We enumerate the data being sent to the app server after categorizing them, and list the vulnerabilities found in the app servers.

6.1. Client-side Data Leaks

Using the MITM proxy, we are able to observe the network traffic generated by the apps and gather the data they sent. We collect and classify the data into several categories (ref. Table 3).

- 1) *Device information (DI)*, *i.e.* device relevant information that may be used to identify the type of device the app is running on, like the OS, device model details, *etc.* This can be used for troubleshooting or optimizing the apps, but may also be used to profile the users' preferences and means (*i.e.* depending upon its price).
- 2) *Device usage attributes (DU)* are the data points that describe the condition and environment in which the

app is being used, like the time, system locale, battery percentage *etc.* While this data can be used to profile resource usage of the app and the common conditions in which the app is being run, it can also be used to profile the user's interactions.

- 3) *Network usage attributes (NU)* are used to characterize the network that the app uses for communication, like WiFi status or cellular connection type (3G or 4G). As with the previous categories, this can be used to troubleshoot communication issues and optimize the user experience, but it can also be used to figure out the user's communication patterns.
- 4) *App-related info (AI)* are related to the installation, updates, and usage of the app. These are usually benign as they only reveal the user's basic engagement with the app, without profiling the actual usage.
- 5) *User personal data (UP)* are used to identify the user themselves. These can range from simple identifiers like email and phone numbers, to extremely invasive identifiers like location, Aadhar number *etc.*
- 6) *User finance details (UF)* are used to characterize the financial status of the user, like payer/payee identities and account numbers or transaction details, *etc.* These also include IFSC (Indian Financial System Code, used to identify a bank and its branch), account balances, and UPI IDs ⁴.
- 7) *User health details (UH)* are the data points used to track the physical health of the user. These also include data to track the user's history of illnesses, precautions taken, and registration for health checkups.
- 8) *Miscellaneous (MS)* are the data points that do not fit the other categories and have app-specific implementational relevance. The use for such data is usually unique to the app.

There were some apps (all banking apps, PhonePe, BHIM and Google Pay) whose traffic we were unable to decrypt, due to utilization of end-to-end encryption (ref.§7.5).

6.1.1. Data collected by apps: Every app collects some subset of categorized data described in Table 3. Also, Tables 4, 5, and 7 highlight the data collected by the individual apps⁵. We also report the third-party trackers found in the APKs in the tables.

We find that payment apps collect a lot of user data while operating. These apps also include third-party trackers known for advertising and collecting user data for monetization [107]. Almost all payment apps use SMS-based authentication (*i.e.* using OTPs), and need permissions to read them to automate the process. However, we observed that Mobikwik reads and transmits multiple SMSes of the user, instead of only the one with the OTP.

We conducted a similar investigation for PayTM. PayTM tracks the app usage pattern of the user. Whenever a user opens the PayTM app and browses the available features, the app regularly sends updates to the server regarding the page the user visits, and what the user is browsing *etc.*, at any given time. Unlike Mobikwik,

4. Unified Payments Interface, an Indian framework for instant payments between people and merchants

5. For easy referencing, we index the data using the category initials and corresponding data point identifier.

TABLE 3: Classification of data obtained from app traffic analysis.

Category	Attribute
Device information (DI)	01- Local IP, 02- Advertisement ID 03- Device ID, 04- Number of cores 05- Device model, 06- Device manufacturer 07- OS update time, 08 - OS version
Device usage attributes (DU)	01- Date, 02- Time, 03- Timezone 04- System locale, 05- Battery percentage 06- Charging status, 07- Root status 08- Process state, 09- RAM status 10- Storage status
Network usage attributes (NU)	01- Hotspot status, 02- Network carrier 03- WiFi status, 04- Number of SIMs 05- Cellular connection status 06- Cellular connection type 07- SIM slot occupied, 08- Round trip time
App related info (AI)	01- App build number 02- App install time 03- App update time 04- App previous update time 05- App joining date
Users' personal data (UP)	01- Phone number, 02- Email, 03- Date of birth 04- Aadhar number, 05- PAN, 06- Location 07- Personal SMS, 08- Contacts, 09- Login ID 10- Phone number circle ³ 11- Phone number operator, 12- Country code 13- Prepaid/postpaid, 14- Travel history 15- App usage metrics, 16- Phone plan details 17- Electricity Board CA number, 18- Gender

TABLE 4: Data collection and third-party trackers observed for payment apps.

App name	Data (red: necessary)	Trackers
PayTM	UP-01, UP-02, UP-03, UP-05 UP-06, UP-08 UP-09, UP-10, UP-11, UP-13 UP-14, UP-15 UF-01, UF-03, UF-04, UF-08 UF-09, DI-01, DI-02, DI-03 DI-04, MS-01, MS-02, MS-03 DU-01, DU-04, DU-05, DU-07 DU-08, DU-09, DU-10 NU-01, NU-02, NU-03, NU-05 NU-06, NU-07, NU-08	AppsFlyer
Mobikwik	UP-01, UP-02, UP-06, UP-07 UP-08, UF-01, UF-02, UF-05 UF-06, UF-07, UF-09 DI-01, DI-02, DI-03 MS-07 DU-02, DU-03 NU-02, NU-03 AI-02, AI-03, AI-04	Branch, CleverTap, Google AdMob, Google Analytics, Tune
Freecharge	UP-01, UP-02, UP-03 UF-01 DI-03	Branch, Demdex, Facebook Analytics, Facebook Share, Google AdMob
Google Pay	UP-01, UP-02, UP-06 DI-05, DI-06 MS-06 DU-03 NU-03, NU-04 AI-01	
PhonePe	encrypted	AppsFlyer

the app only communicates its own domain with all the data. After obfuscating the data, as with Mobikwik, we observed that the functionality also works with PayTM, showing that all the sensitive information being gathered is superfluous to the functionality as well. We attempted this method on PhonePe as well. We did not observe the exact data being transmitted to the PhonePe domains, due to the use of end-to-end encryption, however, we observed that a lot of data was being transmitted to

Category	Attribute
User finance details (UF)	01- Bank account number, IFSC, type, VPA 02- Bank balance, 03- UPI MPIN 04- Credit report, 05- Payment amount 06- Payment recipient, 07- Payment reason 08- Phone recharge amount 09- Wallet balance, 10- Debit card number 11- Cryptocurrency name, 12- Crypto amount
User health details (UH)	01- Symptoms, 02- Beneficiary ID 03- Certificate URL, 04- Vaccine name 05- Vaccine status, 06- Vaccine status code 07- OPD registration - institution 08- OPD registration - state 09- Vaccine dose number 10- Vaccine dose date
Miscellaneous (MS)	01- Customer ID, 02- Profile type 03- Account reference ID 04- Sequence number 05- Password, 06- PIN, 07- OTP SMS 08- Document type, 09- typeOfUser 10- record_from (MSTL) 11- Hint question, 12- Hint answer 13- DCDR location, 14- Complaint type 15- citizenGuid, 16- eventType

multiple URLs of the domain. After blocking all of them except one, we observed that the payment functionality still works. Thus, PhonePe also transmits a lot of data unnecessarily.

In the case of banking apps, we were not able to observe the payloads being delivered during operation. Compared to other apps under investigation, we found that banking apps employ much more stringent security measures to prevent data snooping. This makes it hard to decrypt and observe the network flows. On one hand, this is good news, as this means that banks are cognizant of data exfiltration threats. However, this also means that the behavior of the app is hard to quantify, and this is concerning because we found embedded trackers in some of the banking apps (both private and government), as shown in Table 6. We surmise that much like payment gateways, these apps might function even without the third-party trackers.

Government apps on the other hand do not collect a comprehensive set of user data while operating. They only transmit just the essential information required to authenticate users. Also, they do not contain third-party trackers. The only notable exception is Digilocker [58], which is used to store and conveniently access government documents. It uses Google's trackers for advertising purposes.

Cryptocurrency apps also transmit users' personal, device, and financial data. Among the apps that we examined, we found that CoinDCX, WazirX, and Binance send data to third-party analytics APIs. This data consists of the device hardware details, the current page on the app, and the user's interactions with the app. WazirX also sends crypto purchase data to a third-party tracker. The apps continue to operate when all the trackers are blocked, but the operation is impeded if the data to the app API itself is obfuscated. Binance and WazirX, in particular, communicate frequently to saasexch [9] and MoEngage [12] trackers respectively.

6.1.2. Obfuscating network traffic: To determine the necessity of the data being transmitted, we attempted to obfuscate some of it using traffic interception. First, we blocked all outgoing traffic to domains except for the primary domain of the app. These blocked domains correspond to the trackers present in the apps. Second, we identify a minimal necessary set of data that would be needed to use the primary functionality of the app, and replace or block all other data contained in outgoing packets. If we find any data fields that cause the functionality to fail when obfuscated, we add it to the set of necessary data. We mark the other data as not necessary for the functionality. This is evident through the red and green data attributes respectively in Tables 4, 5, 6, and 7.

In the case of `Mobikwik`, we found that it sends data to two separate domains, `https://in.wzrkt.com` and `https://api.mobikwik.com`. After blocking the former domain, we found that the app runs unhindered. In addition, we intercepted the packets to `api.mobikwik.com`, and deleted all the fields other than the user’s financial details, as listed in Table 4. This caused the connection to close from the server end. So on the second attempt instead of deleting the fields, we replaced the values with junk data (after certificate unpinning). We found that transactions succeed without issues, with such junk data.

We conducted a similar experiment on `PayTM`. Here, there is only one domain being contacted. We performed a standard payment transaction, and replaced all non-financial data with junk values. Here also, replacing all non-financial with junk information, works fine.

In the case of `Freecharge`, we were able to observe data sent to other tracking domains, which could be blocked without hindering the app’s functionality. We also saw some personally identifiable data that could be deleted without affecting functionality, but bulk of the data was still encrypted (much like `Google Pay` and `PhonePe`).

In the case of government apps, the data collected is already minimal. Any blocking or deleting caused the app functionality to glitch and/or fail.

In the case of cryptocurrency apps, the data collected is quite large. However, for all four apps we found that blocking or obfuscating any data field leads to the app retrying to send the original data.

6.1.3. Traffic confidentiality. We observed that most apps merely rely on TLS for data confidentiality and implement certificate pinning, other than `Google Pay`, `PhonePe` and all banking apps. These apps rely on their own encryption inside of the app, hence the payloads are unreadable even after an MITM on the TLS layer.

6.2. Server-side Vulnerabilities

In order to find weaknesses in the security postures of the app hosting infrastructure, we have conducted IP lookups and service scans. We observed that all apps utilize CDNs. We have listed the CDNs hosting the apps as well as any weaknesses or vulnerabilities, in Table 8.

We then proceeded to analyze the CDN edge servers that the apps communicate with. This makes them the prime target for the external attacker as per our threat model. Hence, we checked the services running on those

TABLE 5: Data collection and third-party trackers in government apps.

App name	Data (red: necessary)	Trackers
MCD App	UP-01, UP-11, MS-07 MS-09, MS-15, MS-16	
DJB mSeva	UP-01, UP-02, MS-14	
BSES Rajdhani	UP-01, UP-16, UF-05	
mPassport Seva	UP-02, UP-03, UP-09 MS-05, MS-11, MS-12 MS-13	
Aarogya Setu	UP-01, UP-04, UP-06 UP-14, MS-07, UH-01, UH-02, UH-03, UH-04 UH-05, UH-06, UH-07, UH-08, UH-09, UH-10,	
Digilocker	UP-01, UP-02, UP-03 UP-05, UP-09, UP-18, MS-06, MS-07, MS-08 MS-10	Google Analytics Google Tag Manager

TABLE 6: Third-party trackers observed for banking apps.

App name	Trackers
HDFC Bank	
SBI	
IDBI Bank	
ICICI Bank	AppsFlyer, CleverTap, Google Ad-Mob, Google Tag Manager, MixPanel
Axis Bank	Appdynamics, AppsFlyer, CleverTap
Canara Bank	CleverTap, Huawei Mobile Services Core

servers to spot vulnerable/unpatched versions or misconfigurations. We used port scanning to discover the services running on the edge servers, and searched for available exploits on `exploit-db` [60], `metasploit` [62], `Vulners` [68] and `MITRE` [69]. In the case of `Cloudflare`, `Amazon` and `Akamai`, we also searched for the vulnerabilities for their customized `Linux` distributions as advertised [53], [55]. We could not accurately verify their claims since port scanning did not accurately reveal the OS being used. We did not find any known vulnerabilities, and the services are all up-to-date.

We also looked for misconfigurations and tried uncovering vulnerable behaviors. *E.g.*, among the government apps, we found that for some of their servers had misconfigurations that could exposure sensitive customer data.

In the case of `MahaDiscom`, the user authentication is carried out by the browser itself. It is handled by a JavaScript function. The script transmits the customer ID for validation, to the server. If the customer ID is valid, then the server responds with the corresponding password. The JavaScript function caches it and matches it against the one the user inputs (when prompted). This can be trivially exploited by presenting a valid ID and observing the password the server responds with. It can then be used as is to authenticate the said ID. In addition, multiple accounts can be created against a valid ID. The server performs no checks for such cases. This vulnerability is also there in `CESC`’s user authentication portal.

Finally for `BSES`, the database schema and query structure is exposed on their unauthenticated testing server. This can be used to formulate SQL injection queries to extract data from their production servers.

Aside from the power distribution companies, we also found a misconfiguration in the `MCD` app server, that could be exploited. Residents’ private data (*e.g.* property

TABLE 7: Data collection and third-party trackers in cryptocurrency apps.

App name	Data (red:necessary)	Trackers
CoinSwitch	UP-01, UP-02, UP-03 UP-06, UP-04, UP-05 UP-09, AI-05, UF-01 UF-05, UF-09, UF-11 UF-12	AppsFlyer, CleverTap Google AdMob New Relic
WazirX	AI-01, AI-02, AI-03 DI-02, DI-03, DI-04 DI-05, DI-06, DI-07 DI-08, NU-02, NU-03 DU-01, DU-02, DU-03 DU-04, UP-01, UP-02 UP-03, UP-05, MS-05 UF-01, UF-05, UF-07 UF-09, UF-11, UF-12	AppsFlyer, MoEngage OneSignal
CoinDCX	AI-01, AI-02, AI-03 DI-02, DI-03, DI-04 DI-05, DI-06, DI-07 DI-08, NU-02, NU-03 DU-01, DU-02, DU-03 DU-04, UP-01, UP-16 UF-05, UF-09, UF-11, UF-12	Adjust, MoEngage Google AdMob
Binance	AI-01, AI-02, AI-03 DI-02, DI-03, DI-04 DI-05, DI-06, DI-07 DI-08, NU-02, NU-03 DU-01, DU-02, DU-03 DU-04, UP-01, UP-02 UP-03, UP-05, MS-05 UF-01, UF-05, UF-07, UF-09	AppsFlyer, saasexch Google AdMob Sentry, Amplitude

withholdings, tax payments, *etc.*) are trivially visible as long as their names and localities are known. No other user authentication is employed before exposing the data. Thus, an adversary can easily get sensitive data as long as he/she can find the victim’s residential locality, often available via OSINT sources. We did not find any vulnerabilities in the payment apps, banking apps and cryptocurrency apps, *w.r.t* security postures and input validations.

7. Experimental Data Analysis

We now discuss our experimental observations, their implications, and their relationship to our threat model.

7.1. Analysis: Client-side Data Leaks

7.1.1. Data encryption. We find that most apps only rely on TLS for assuring the confidentiality and integrity of the data. Once decrypted, the data packets are available in plain text. Very often the TLS connection to the apps’ servers terminates at the CDN edge-servers/end-points. This is evident from the TLS handshake messages (SNI and server certificate), that correspond to the CDN infrastructure provider itself. This means that the CDNs can observe the users’ decrypted data. The users are thus forced to implicitly trust the CDNs’ privacy policies. We observe that apps do not mention this upfront through terms and conditions, and clauses. Neither is this obvious from the apps’ download page. The exceptions were Google Pay and Canara Bank where the applications use an additional layer of encryption for data confidentiality.

7.1.2. User data tracking depth. (A) *Payment apps:* Among the apps, whose traffic we were able to decrypt and inspect, the non-governmental ones gather and send

TABLE 8: Hosts and vulnerabilities discovered during server-side analysis.

App name	CDN	Vulnerabilities
PayTM	Akamai	-
Google Pay	Google	-
Mobikwik	Cloudflare, Akamai, Netmagic	-
Freecharge	Akamai, Amazon	-
PhonePe	Cloudflare, Amazon	-
MCD App	NIC	Property tax details visible by entering name and locality in SDMC website.
BSES Rajdhani	NIC	Beta server open to public with no access controls, database structure and query formation listed in plain text
CESC	Self-hosted	No authentication to associate customer ID number and account created online
Maha Discom	Amazon	User authentication in website happens in the browser itself, hence the password of any user can be intercepted easily
mPassport Seva	NIC	-
Aarogya Setu	Amazon, Akamai	-
Digilocker	Amazon	-
CoinSwitch Kuber	Cloudflare, Amazon, Fastly	-
WazirX	Cloudflare	-
CoinDCX	Cloudflare	-
Binance	Amazon	-

a lot of user data. In particular, Mobikwik and PayTM gather very sensitive information about the user. In case of Mobikwik, we find that the app reads multiple unrelated personal SMSes, and transmits them to the server. This is a clear breach of privacy.

PayTM sends users’ behavioral information, *e.g.*, the page the user is browsing, the hotspot status of the phone, *etc.*, to its servers and trackers. This kind of comprehensive data collection can help profile a user very accurately. Needless to mention, no application layer confidentiality preserving mechanism is used, besides the TLS tunnel terminating at the CDN edge server. Thus, instead of coercing the app companies, the surveilling government can directly obtain all user-related data from the CDN providers. This further exposes the users’ privacy to all third-party adversaries that may (potentially) be colluding with the CDN, as mentioned in our threat model.

We also observe that payment and cryptocurrency apps gather a lot of information regarding the users’ devices, the network operators, app usage patterns, and the user’s financial data. In many cases, some of the data seem unnecessary. *E.g.*, according to the National Payments Corporation of India, UPI transactions need only four pieces of information, *viz.* UPI virtual private address, the authentication PIN, the IFSC code of the bank, and the transaction amount [52]. Yet, for UPI transactions, we find that the users’ bank account numbers and their types are also collected. Similarly, payment apps enforce sharing device location, even though UPI does not [35].

(B) *Government services:* Government apps, collect a lot less data, compared to private apps. They usually collect

user credentials for authentication and financial data for utility payment purposes. *Aarogya Setu* is an exception here. The app requires continuous user location data, and previous travel and health history. Such data may be essential for COVID-19 contact tracing. However, this exposes it to potentially colluding third-party attackers, as it is hosted on *Amazon Cloudfront*.

(C) *Banking apps*: We find that banking apps, both private and government, use third-party CDNs. They also secure their communication with an application layer encryption, besides TLS. This defends against snooping by CDNs.

(D) *Cryptocurrency apps*: In the case of cryptocurrency apps, we find that along with users' personal and financial data, the crypto-wallet identifier (owned by the user) is also sent to the server. This creates a link between the user identity and the crypto-wallet identifier, which defeats the pseudonymity granted by cryptocurrencies. In addition, *WazirX*, *Binance* and *CoinDCX* send user device and network/usage patterns to third parties. In the case of *WazirX*, this even includes the amount of cryptocurrency traded and owned by the user.

We tried pinpointing the data that is actually needed for the app's primary workings. We started by blocking any third-party domains contacted by the apps. We saw that all apps continue working unaffected. We next obfuscated the outgoing data to the app servers themselves. We replaced the fields for user personal data, app usage patterns, device hardware details and network details, with random meaningless strings. We find that the primary functions of the payment apps work unhindered. Government and cryptocurrency apps generally share limited information to their servers. Obfuscating any of these often results in their dysfunction.

7.2. Analysis: Server-side Vulnerabilities

We find that almost all the non-government apps rely on third-party CDN to host their backend servers, with the exception of *Google Pay*, which is hosted on their own CDN. We also find that some government apps utilize third-party CDNs as well, even when there are government CDN available for use [65]. *Digilocker* and *Aarogya Setu* are examples of such apps. Other than banking apps and *BHIM*, TLS alone is used to encrypt communication between app and the server. This encrypted channel terminates at the edge server, which exposes sensitive data to the administrator of the edge server. Given the capabilities of the various adversaries (see §3.2), using third-party CDNs without end-to-end encryption can open up additional threat vectors.

Firstly, the attack surface for external attackers increases, as the security now depends on the app developer and the hosting services. Secondly, the CDN may comply with the coercion of surveilling governments, and reveal the data of the apps they serve. Finally, the CDN itself can collude with other third parties to monetize the data.

We did not find many apps with vulnerabilities in their configuration or hosting servers, aside from several government and power distribution app servers. At the time of submission, even these vulnerabilities have been patched. This suggests that basic precautions are taken regarding authentication and data protection, which helps deter external attackers.

7.3. Privacy Regulations

As of the third quarter of 2023, a new law called Digital Personal Data Protection Act has been enacted in India. To ascertain how the data collection practices fare with the new law, we looked at the privacy policies offered by the apps under consideration. We find that in India, the terms and conditions of most apps do not specify exactly what user data they require. Instead their privacy policies often allow for accessing users' data beyond what is required. The privacy policies also include ambiguous agreements about sharing the data collected, involving clauses pertinent to actors whose identities cannot be ascertained. In Table 9 we highlight how some of the important clauses of the apps' respective privacy policies seemingly contradict with the guidelines of the Digital Personal Data Protection Act (2023) [73].

Other than the Act, the Reserve Bank of India has also issued guidelines that pertain to the security and storage of user data [27], [28]. Hence, we also checked how the apps under consideration complied with the guidelines. To obtain this information, we filed a Right To Information [67] request to learn the status of compliance of the companies. In the response, we found that there seems to be no mechanism enabled to assess compliance status, even when the regulations require their periodic reporting [28]. The details of the responses are present in Appendix A.1.

7.4. Takeaways – Vulnerabilities at a Glance

From our results, we relate each app with each of the three adversaries mentioned in §2.3. Accordingly, in Table 10, we classify the apps as “vulnerable” (V) or “not vulnerable” (NV) to each attacker. “Vulnerable” implies those that can directly be impacted by one of the adversaries. For example, if coerced, *PayTM* can trivially reveal data to the higher authorities, especially since the data is not end-to-end encrypted. “Not vulnerable” implies that the app's confidentiality cannot be easily compromised (especially when it uses end-to-end encryption)⁶.

7.5. Limitations and Future Work

We analyzed the most popular apps (that involve sensitive data) on Indian Play Store. But, our work does not cover several aspects that we explain below.

7.5.1. App coverage. We now explain our limitations related to coverage of apps.

- 1) In our research, we focused on Android apps. We studied their data collection behavior, but we did not study the corresponding ones that are used on Apple iOS devices. Not many people in India can afford an Apple iPhone, as a result, the proliferation of such devices (and in-turn their apps) is rather low [2]. Not surprisingly, some of the apps are not even available for the iOS platform (e.g. *mPassport Seva*). Hence, we believe that the apps' behaviour on iOS would not be representative of the data collection that many smartphone users in India are subjected to.
- 2) Additionally, we only studied the most popular apps on the Play Store. These were selected through the

6. For servers, it implies that they are not vulnerable to known threats.

TABLE 9: Potential contradictions between apps’ privacy policies and Digital Personal Data Protection Act 2023

App Name	Privacy Policy Clause	Potential Non-Compliance
PayTM	I-1 and III-12 [15]	Consent is sought for providing the services, but data is also used for ads. (7-a)
Mobikwik	Use and Storage of Information - 12 [11]	Same as PayTM. (7-a)
Freecharge	8-d and 7-a [21]	Broad consent to multiple purposes is sought, instead of specific ones. (6-1)
Google Pay	Develop new services [19]	User data is not removed from storage after required action is served. (7-a)
PhonePe	Purpose and Use of Information - 2 [16]	Personal information is processed for business partners services, not just PhonePe services. (7-a)
MCD App	N/A [18]	
DJB mSeva	N/A [17]	
BSES Rajdhani	Personal Information - 9 [4]	Same as that for Freecharge. (6-1)
CESC	Terms of Use - 9 and 12 [5]	The Terms of Use mentions that the data processing limits and durations are as per the privacy policy, but the latter mentions no such thing.(7-a)
MahaDiscom	Personal Information - 9 [10]	Same as CESC. (7-a)
mPassport Seva	N/A [13]	
Aarogya Setu	N/A [1]	
Digilocker	N/A [8]	
CoinSwitch	Information Collected [7]	Personal data can be collected without explicit consent from the user. (4-1-a)
WazirX	2.3 [22]	Personal data is obtained from third parties instead of the user. (4-1-a)
CoinDCX	Information We Collect - 1 - b [6]	Personal identifiers are collected “automatically”, without explicit consent. (5-1-i)
Binance	Privacy Notice - 9 [3]	Personal data can be used until an objection is raised, instead of ceasing when original purpose is fulfilled. (7-a)

TABLE 10: Potential privacy vulnerabilities to various attackers, denoted by either vulnerable (V) or not vulnerable (NV).

App	External attacker	Coercive government	Colluding third-party
PayTM	NV	V	V
Mobikwik	NV	V	V
Freecharge	NV	V	V
Google Pay	NV	V	NV
MCD App	V	V	NV
DJB mSeva	NV	V	NV
BSES Rajdhani	NV	V	V
CESC APPS	NV	V	NV
MahaDiscom	NV	V	V
mPassport Seva	NV	V	NV
Aarogya Setu	NV	V	V
Digilocker	NV	V	V
CoinSwitch	NV	V	V
CoinDCX	NV	V	V
Binance	NV	V	V
WazirX	NV	V	V

top apps list, with downloads exceeding a million. But we admit that this is not a comprehensive list. In the future, we plan to analyze many more Android apps as well as iOS apps to present a more comprehensive report on the privacy-related issues of such apps.

- 3) Finally, we tried similar exercises for popular apps in US and EU, however, we did not have access to credentials required to access such services in those countries. For example, in order to access payment apps in the US and EU, a US and EU phone number and bank account are necessary.

7.5.2. Shortcomings of Client-side Analysis. Our client-side analysis is limited by the fact that we had to bypass a number of checks imposed by the apps and Android, as well as the variety of ways in which they can be used. We enumerate them in detail as follows:

- 1) For the payment and government apps in our study, we were able to disable certificate pinning and observe the network traffic. While we were able to bypass the integrity checks, we could not decrypt the traffic of those that used end-to-end encryption.

- 2) Some apps also use end-to-end application layer encryption, besides TLS. This is likely a good practice to protect against snooping by the servers’ administrators. But, this also prevented us from analyzing the data being sent to the apps’ servers. In the future, we intend to reverse engineer the APK in order to find such additional keys to decrypt the traffic.
- 3) For the apps whose traffic we were able to analyze, we used them to perform the most obvious/common activities (while their traffic was being monitored). Thus, the captured network traffic may not be representative of all the possible actions. In addition, the apps’ data collection strategies might change based on contextual cues. For example, some apps may suddenly collect more data if they sense movement through the built-in accelerometers in the devices, or upon detecting changes in GPS coordinates. The app could also be surreptitiously exfiltrating data, while they run in the background. The network traffic corresponding to all such actions was not monitored. In the future, we plan to develop and test strategies to capture and observe traffic for such scenarios.

7.5.3. Shortcomings of Server-side Analysis. Our server-side analysis is limited by the tools available and known vulnerabilities. We enumerate them in detail as follows:

- 1) We mostly relied on various types of network scanners to identify the services running on the hosting servers. However, popular OS fingerprinting tools often provided unreliable results in this respect and we observed a lot of erroneous results. *E.g.* Amazon servers were erroneously reported as running Linux 2.6.32. They use and maintain a customized distribution with an up-to-date kernel [53].
- 2) Further, we only searched for available exploits for the software/service versions, in various public databases. We did not look for zero-day exploits that could, for instance, rely on reverse engineering the apps to identify the API functions used and exploit the potential bugs therein.

Because of these limitations, we are not aware of zero-day vulnerabilities that might be present in the hosting

servers. In the future, we intend to improve upon the identification of software signatures (identified through service scans).

7.5.4. User Surveys. This measurement study covers the data collection and transmission behaviour of popular apps used in India. However, it is yet to be seen how Indian users perceive the results obtained in this study. There have been prior studies done on that look into this subject. *E.g.* Kumaraguru *et al.* [91] show that the Indian users, in general, are not aware of the data collection practices of the apps, or their ramifications. In India, the prevailing notion of privacy pertains to personal space, and not to digital information. Further, users also exercise limited caution against identity theft or collection of sensitive data. However, the authors do not consider real-life examples of threat vectors. In future, we intend to conduct another survey with a greater sample size, where participants will be interviewed for their privacy awareness. Thereafter, the results of this study would be shared with them to check if they become more vigilant.

7.5.5. Policy and Suggestions for Technical Improvement. Currently, a new data regulation law, the *Digital Personal Data Protection Act* [20], has been legislated by the Parliament of India. This law covers the details of how much data should an app be allowed to collect, and under what circumstances. We propose that a system should be developed that condenses the regulations into an easy to parse format, and makes it easy to understand the kind of data collection any app is performing. This will enable the users to identify any potential violations regarding the data collection, and act accordingly. To facilitate this process, we also suggest a more fine-grained permissions system in the Android OS, so that data collection permissions can be granted on specific uses only. For example, enabling SMS permissions for specific messages, instead of access to all SMS messages.

8. Concluding Remarks

The rise of mobile app usage in India, fuelled by events like nationwide demonetization and COVID-19, is not without pitfalls. A lot of these apps, used by millions, involve handling sensitive personally identifiable data. There have been virtually no past efforts that assess what kind of information is collected by the apps.

We audit the network traffic of these apps to determine what kind of data is collected and transmitted. We chose a diverse set of apps, ranging from online digital payments, to those pertaining to government services, to crypto exchanges. We also assessed the security postures of the hosting services that ran the apps' backend services. We observed that several non-governmental apps (*e.g.*, PayTM, Mobikwik *etc.*), covertly collect a comprehensive set of personally identifiable user data points. Interestingly, for several apps, these do not seem necessary for their workings. Obfuscating or dropping them had no impact on how the apps work. Several others even use third-party trackers to track users' behavior. Nevertheless, in most such cases the backend cloud servers are relatively secure against known threats. We did not observe such extensive data collection in apps pertinent to government services. But in several cases, we observed misconfigurations, that could lead to unintentional users' data exposure.

The Digital Personal Data Protection Act sets guidelines regarding the data storage and processing of personal user data. We found that many apps seemingly do not abide by those guidelines, and store and process user data beyond the use for which the consent was provided. In addition, there are a few data security and safety regulations to protect against third-party attackers⁷. To understand the extent to which the app developers adhere to such regulations, we filed a request under the Right To Information Act [67] to the appropriate agencies maintaining such regulations. Unfortunately, we received no meaningful responses that could provide us insights. We hope that our work motivates the formulation of stronger guidelines and mechanisms to assess (and enforce) compliance.

Acknowledgment

The authors would like to thank Apar Gupta and his team at the Internet Freedom Foundation for their help and support. Their legal expertise facilitated the proper process to apply for a request under the Right to Information Act.

The authors are also grateful to Vitaly Gorbachev and Freepik from flaticon.com for graciously allowing usage of their artwork for the diagrams.

References

- [1] Aarogya Setu. <https://aarogyasetu.gov.in>.
- [2] Apple likely to garner 5.5% market share in india in 2022. <https://tinyurl.com/ubsdxhpj>.
- [3] Binance Privacy Policy. <https://www.binance.com>.
- [4] BSES Rajdhani Power Limited Privacy Policy. <https://www.bsesdelhi.com/web/brpl/privacy-policy>.
- [5] CESC Privacy Policy. <https://www.cesc.co.in/privacyPolicy>.
- [6] CoinDCX Privacy Policy. <https://coindcx.s3.amazonaws.com/static/documents/privacy-policy.pdf>.
- [7] CoinSwitch Privacy Policy. <https://coinswitch.co/privacy-policy>.
- [8] Digilocker Privacy Policy. <https://www.digilocker.gov.in/about/privacy-policy>.
- [9] Easyprivacy third-party trackers. https://github.com/easylist/easylist/blob/master/easyprivacy/easyprivacy_thirdparty.txt.
- [10] MahaDiscom Privacy Policy. https://wss.mahadiscom.in/wss/images/MSEDCL_Privacy_Policy_Service.pdf.
- [11] Mobikwik Privacy Policy. <https://www.mobikwik.com/privacypolicy>.
- [12] Moengage - insights-led customer engagement platform. <https://www.moengage.com/>.
- [13] mPassport Seva Privacy Policy. <https://coindcx.s3.amazonaws.com/static/documents/privacy-policy.pdf>.
- [14] Paytm, google pay or phonepe: Which one has biggest share in upi transactions? <https://tinyurl.com/bdfzwnwc>.
- [15] Paytm.com - Terms & Conditions. <https://paytm.com/company/terms-and-conditions?company=one97&tab=privacy>.
- [16] PhonePe - Privacy policy. <https://www.phonepe.com/privacy-policy/>.
- [17] Privacy Policy | Delhi Jal Board. <https://delhijalboard.delhi.gov.in/jalboard/privacy-policy>.
- [18] Privacy Policy for Mobile App. <https://mcdonline.nic.in/privacy-policy.html>.
- [19] Privacy Policy – Privacy & Terms – Google. <https://policies.google.com/privacy>.

7. Including data exfiltration to foreign networks




- [20] Salient Features of the Digital Personal Data Protection Bill, 2023.
- [21] Terms and Conditions | FreeCharge. <https://www.freecharge.in/termsandconditions>.
- [22] WazirX Privacy Policy. <https://wazirx.com/legal/privacy>.
- [23] Big-box breach: The inside story of wal-mart's hacker attack. <https://www.wired.com/2009/10/walmart-hack/>, 2009-10-13.
- [24] Officials say us may never know extent of snowden's leaks. <https://www.nytimes.com/2013/12/15/us/officials-say-us-may-never-know-extent-of-snowdens-leaks.html>, 2013-12-14.
- [25] Yahoo says all three billion accounts hacked in 2013 data theft. <https://www.reuters.com/article/us-yahoo-cyber-idUSKCN1C82O1>, 2017-10-04.
- [26] Cambridge analytica and facebook: The scandal and the fallout so far. <https://www.nytimes.com/2018/04/04/us/politics/cambridge-analytica-scandal-fallout.html>, 2018-04-04.
- [27] Storage of payment system data. <https://www.rbi.org.in/commonman/English/Scripts/FAQs.aspx?Id=2995>, 2018-04-06.
- [28] Guidelines on regulation of payment aggregators and payment gateways. <https://www.rbi.org.in/scripts/NotificationUser.aspx?Id=11822>, 2020-03-17.
- [29] Amazon vulnerability research program. <https://hackerone.com/amazonvrp?type=team>, 2020-04-01.
- [30] Krafton admits sharing data of battlegrounds mobile players with non-indian servers, read full statement. <https://tinyurl.com/53tsa59w>, 2021-06-21.
- [31] Cve-2021-44228. <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-44228>, 2021-12-09.
- [32] Announcing the public launch of cloudflare's bug bounty program. <https://blog.cloudflare.com/cloudflare-bug-bounty-program/>, 2022-01-02.
- [33] Rbi punished paytm payments bank for data leaks to chinese firms: Report. <https://tinyurl.com/mr3xsuk4>, 2022-03-14.
- [34] New aadhaar data leak exposes 11 crore indian farmers' sensitive info. <https://tinyurl.com/rkwmdcru>, 2022-06-14.
- [35] Guidelines on capturing customer location on upi apps. <https://tinyurl.com/mryj5t8h>, 2022-07-05.
- [36] 5 years of Demonetisation: How digital payments and lending picked up post note ban. <https://tinyurl.com/36v4ette>, 2022-10-18.
- [37] Aadhaar Data breach: UIDAI finds multiple transactions done with the same fingerprint- Technology News, Firstpost. <https://tinyurl.com/mry5864wk>, 2022-10-18. Section: Firstpost Tech.
- [38] Aarogya setu most downloaded healthcare app in world: Amitabh kant. *The Economic Times*, May 2022-10-18.
- [39] Assam: Cyberattack in Oil India's headquarters, attackers demand over Rs 57 cr as ransom. <https://tinyurl.com/2p8sc9m8>, 2022-10-18. Section: National.
- [40] Eight in ten bank account holders in India's metros now use mobile banking apps. <https://www.businessinsider.in/tech/news/8-in-10-indians-using-mobile-banking-apps-in-pandemic/articleshow/89821333.cms>, 2022-10-18.
- [41] Government Requests for User Data | Transparency Center. <https://transparency.fb.com/data/government-data-requests/country/in/>, 2022-10-18.
- [42] Groww: Stocks & Mutual Fund – Apps on Google Play. <https://play.google.com/store/apps/details?id=com.nextbillion.groww>, 2022-10-18.
- [43] Hackers hit India's No.2 broker Upstox, company says ramped up security. *The Times of India*, April 2022-10-18.
- [44] Hackers launch DDoS attack on security blogger's site, send SWAT team to his home. <https://nakedsecurity.sophos.com/2013/03/17/swat-ddos-brian-krebs/>, March 2022-10-18.
- [45] How the 'Jio effect' brought millions of Indians online and is reshaping Silicon Valley and the internet. <https://tinyurl.com/5ckceztu>, 2022-10-18.
- [46] Privacy - Government Information Requests - Apple (IN). <https://www.apple.com/legal/transparency/in.html>, 2022-10-18.
- [47] Ransomware is the biggest cybersecurity pain point in India: IBM Security's Chris Hockings. *The Economic Times*, April 2022-10-18.
- [48] Requests for user information – Google Transparency Report. <https://transparencyreport.google.com/user-data/overview?hl=en&user-requests-report-period=series:requests,accounts;authority:IN;time:&lu=user-requests-report-period>, 2022-10-18.
- [49] Statistics of NPCI - National Payments Corporation of India. <https://www.npci.org.in/statistics>, 2022-10-18.
- [50] The top 5 Crypto trading apps for seamless and reliable Crypto transactions. <https://tinyurl.com/mr2kpy47>, 2022-10-18.
- [51] Transparency Report 2021 - Reddit. <https://www.redditinc.com/policies/transparency-report-2021-2>, 2022-10-18.
- [52] Upi faqs for the customer. <https://www.npci.org.in/what-we-do/upi/faqs>, 2022-10-18.
- [53] Amazon linux - amazon elastic compute cloud. <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/amazon-linux-ami-basics.html>, 2022-12-12.
- [54] A cli application that automatically prepares android apk files for https inspection. <https://github.com/shroudedcode/apk-mitm>, 2022-12-12.
- [55] Cloudflare global network — data center locations — cloudflare. <https://www.cloudflare.com/network/>, 2022-12-12.
- [56] Connect to the network. <https://developer.android.com/training/basics/network-ops/connecting>, 2022-12-12.
- [57] Dig. <https://gist.github.com/githubutilities/a4aeaf3fb80519cb2d59>, 2022-12-12.
- [58] Digilocker: An initiative towards paperless governance. <https://www.digilocker.gov.in/>, 2022-12-12.
- [59] Exodus privacy project. <https://exodus-privacy.eu.org/en/>, 2022-12-12.
- [60] Exploit database - exploits for penetration testers, researchers, and ethical hackers. <https://www.exploit-db.com/>, 2022-12-12.
- [61] Frida: A world-class dynamic instrumentation toolkit. <https://frida.re/>, 2022-12-12.
- [62] Metasploit — penetration testing software, pen testing security. <https://www.metasploit.com/>, 2022-12-12.
- [63] mitmproxy – an interactive https proxy. <https://mitmproxy.org/>, 2022-12-12.
- [64] Mobile security framework. <https://mobsf.github.io/docs>, 2022-12-12.
- [65] National informatics centre. <https://www.nic.in/>, 2022-12-12.
- [66] nmap. <https://github.com/nmap/nmap>, 2022-12-12.
- [67] Right to information act. <https://rti.gov.in/>, 2022-12-12.
- [68] Search engine for security intelligence. <https://vulners.com/>, 2022-12-12.
- [69] Solving problems for a safer world — mitre. <https://www.mitre.org/>, 2022-12-12.
- [70] Sublist3r. <https://github.com/aboul31a/Sublist3r>, 2022-12-12.
- [71] webxray is a tool for analyzing webpage traffic and content, 2022-12-12.
- [72] whois. <https://github.com/weppos/whois>, 2022-12-12.
- [73] Digital Personal Data Protection Act. <https://www.meity.gov.in/writereaddata/files/Digital%20Personal%20Data%20Protection%20Act%202023.pdf>, 2023.
- [74] Magisk: The Magic Mask for Android. <https://github.com/topjohnwu/Magisk>, 2023-01-01.
- [75] R. Agarwal. A Trusted-Hardware Backed Secure Payments Platform for Android. October 2021-10-20.
- [76] Marshall Allen. Health Insurers Are Vacuuming Up Details About You — And It Could Raise Your Rates. <https://tinyurl.com/yx2jujfp>, 2022-10-18.

- [77] Michael Bailey, David Dittrich, Erin Kenneally, and Doug Maughan. The menlo report. *IEEE Security & Privacy*, 10(2):71–75, 2012.
- [78] Prasad Banerjee. Indians’ app usage at new peak. <https://www.livemint.com/technology/apps/indians-app-usage-at-new-peak-11636658259700.html>, November 2022-10-18. Section: Technology.
- [79] Amiangshu Bosu, Fang Liu, Danfeng (Daphne) Yao, and Gang Wang. Collusive data leak and more: Large-scale threat analysis of inter-app communications. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, ASIA CCS ’17*, page 71–85, 2017.
- [80] Paolo Calciati, Konstantin Kuznetsov, Alessandra Gorla, and Andreas Zeller. Automatically granted permissions in android apps: An empirical study on their prevalence and on the potential threats for privacy. In *Proceedings of the 17th International Conference on Mining Software Repositories, MSR ’20*, page 114–124, 2020.
- [81] Sen Chen, Lingling Fan, Guozhu Meng, Ting Su, Minhui Xue, Yinxing Xue, Yang Liu, and Lihua Xu. An empirical assessment of security risks of global android banking apps. In *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, pages 1310–1322, 2020.
- [82] Xuechao Du, Xiang Pan, Yinzi Cao, Boyuan He, Gan Fang, Yan Chen, and Daigang Xu. Flowcog: Context-aware semantic extraction and analysis of information flow leaks in android apps. *IEEE Transactions on Mobile Computing*, pages 1–17, 2022.
- [83] Steven Englehardt, Jeffrey Han, and Arvind Narayanan. I never signed up for this! Privacy implications of email tracking. *Proceedings on Privacy Enhancing Technologies*, 2018(1):109–126, January 2022-10-17.
- [84] United States. National Commission for the Protection of Human Subjects of Biomedical and Behavioral Research. *The Belmont report: ethical principles and guidelines for the protection of human subjects of research*, volume 1. Department of Health, Education, and Welfare, National Commission for the . . . , 1978.
- [85] Shayan Ghosh. The disturbing facts about India’s know your customer data leaks. <https://www.livemint.com/industry/banking/know-thy-customer-through-data-leaks-11648485306647.html>, March 2022-10-18. Section: Industry.
- [86] Dan Goodin. Hackers who breached T-Mobile stole personal data for 49 million accounts. <https://tinyurl.com/47bste8x>, August 2022-10-18.
- [87] Glenn Greenwald and Ewen MacAskill. NSA Prism program taps in to user data of Apple, Google and others. *The Guardian*, June 2022-10-18.
- [88] Kashmir Hill. How Target Figured Out A Teen Girl Was Pregnant Before Her Father Did. <https://www.forbes.com/sites/kashmirhill/2012/02/16/how-target-figured-out-a-teen-girl-was-pregnant-before-her-father-did/>.
- [89] Renuka Kumar, Sreesh Kishore, Hao Lu, and Atul Prakash. Security analysis of unified payments interface and payment apps in india. In *Proceedings of the 29th USENIX Conference on Security Symposium, SEC’20*, 2020.
- [90] Renuka Kumar, Sreesh Kishore, Hao Lu, and Atul Prakash. Security analysis of unified payments interface and payment apps in india. In *Proceedings of the 29th USENIX Conference on Security Symposium, SEC’20*, pages 1499–1516, August 2022-10-17.
- [91] Ponnurangam Kumaraguru and Niharika Sachdeva. Privacy in India: Attitudes and Awareness V 2.0, November 2012.
- [92] Douglas J. Leith. Web Browser Privacy: What Do Browsers Say When They Phone Home? *IEEE Access*, 9:41615–41627, 2021.
- [93] Douglas J. Leith and Stephen Farrell. Contact Tracing App Privacy: What Data Is Shared By Europe’s GAEN Contact Tracing Apps. In *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pages 1–10, May 2021. ISSN: 2641-9874.
- [94] Haoyu Liu, Paul Patras, and Douglas J Leith. Android Mobile OS Snooping By Samsung, Xiaomi, Huawei and Realme Handsets. page 12.
- [95] Samin Yaseer Mahmud, Akhil Acharya, Benjamin Andow, William Enck, and Bradley Reaves. Cardpliance:{PCI}{DSS} compliance of android applications. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1517–1533, 2020.
- [96] Samin Yaseer Mahmud, K. Virgil English, Seaver Thorn, William Enck, Adam Oest, and Muhammad Saad. Analysis of payment service provider sdks in android. In *Proceedings of the 38th Annual Computer Security Applications Conference, ACSAC ’22*, page 576–590, 2022.
- [97] Ashwin Manikandan and MC Govardhana Rangan. India is fast becoming the global ransomware capital, says NPCI CEO. *The Economic Times*, September 2022-10-18.
- [98] Nandita Mathur. Due to Jio, India is home to world’s 2nd largest internet user base: Report. <https://tinyurl.com/ydmyu5vt>, June 2022-10-18. Section: Industry.
- [99] Lew McCreary. What Was Privacy? *Harvard Business Review*, October 2022-11-01. Section: Technology and analytics.
- [100] Shreya Nandi. Demonetization 3rd anniversary: How digital payments picked up post note ban. <https://tinyurl.com/yc3dhcvs>, November 2019. Section: Politics.
- [101] Trung Tin Nguyen, Michael Backes, and Ben Stock. Freely given consent? studying consent notice of third-party tracking and its violations of gdpr in android apps. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS ’22*, page 2369–2383, 2022.
- [102] Trung Tin Nguyen, Duc Cuong Nguyen, Michael Schilling, Gang Wang, and Michael Backes. Measuring user perception for detecting unexpected access to sensitive resource in mobile apps. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security, ASIA CCS ’21*, page 578–592, 2021.
- [103] Jack Nicas, Raymond Zhong, and Daisuke Wakabayashi. Censorship, Surveillance and Profits: A Hard Bargain for Apple in China. *The New York Times*, May 2022-10-18.
- [104] Charlie Osborne. Chinese developers expose data belonging to Android gamers. <https://tinyurl.com/2467uv9r>, 2022-10-18.
- [105] Paul Pearce, Roya Ensafi, Frank Li, Nick Feamster, and Vern Paxson. Augur: Internet-wide detection of connectivity disruptions. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 427–443, 2017.
- [106] Paul Pearce, Ben Jones, Frank Li, Roya Ensafi, Nick Feamster, Nick Weaver, and Vern Paxson. Global measurement of DNS manipulation. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 307–323, Vancouver, BC, August 2017. USENIX Association.
- [107] Abbas Razaghpanah, Rishab Nithyanand, Narseo Vallina-Rodriguez, Srikanth Sundaresan, Mark Allman, Christian Kreibich, and Phillipa Gill. Apps, Trackers, Privacy, and Regulators: A Global Study of the Mobile Tracking Ecosystem. In *Proceedings of NDSS*, 2018.
- [108] Matthew J Salganik. *Bit by bit: Social research in the digital age*. Princeton University Press, 2019.
- [109] Nayanamana Samarasinghe, Aashish Adhikari, Mohammad Mannan, and Amr Youssef. Et tu, brute? privacy analysis of government websites and mobile apps. In *Proceedings of the ACM Web Conference 2022, WWW ’22*, page 564–575, 2022.
- [110] Aritra Sarkhel and Neha Alawadhi. How data brokers are selling all our personal info for less than a rupee to whoever wants it. *The Economic Times*, February 2022-10-18.
- [111] Ting-Fang Yen, Yinglian Xie, Fang Yu, Roger Peng Yu, and Martín Abadi. Host fingerprinting and tracking on the web: Privacy and security implications. In *19th Annual Network and Distributed System Security Symposium, NDSS 2012, San Diego, California, USA*, 2012.

A. Appendix

A.1. Request Filed Under Right to Information (RTI) Act.

In order to assess compliance of app company security postures with the data security regulations in India, we filed a request under the RTI Act to ask seeking details of the same. The requested queries and the responses can be seen in Figure 3.

  	
भारतीय रिज़र्व बैंक RESERVE BANK OF INDIA <small>www.rbi.org.in</small>	
CO.DPSS.ADM.No.S791/01.06.006/2022-23 04 अगस्त 2022	
RTI Portal [Redacted]	
महोदय, सूचना का अधिकार अधिनियम 2005 के अंतर्गत आवेदन- आरबीआई/एनडी/आर/ई/22/04501 कृपया सूचना का अधिकार अधिनियम, 2005 के अंतर्गत सूचना प्राप्त करने के लिए प्रस्तुत अपने दिनांक 05 जुलाई 2022 के आवेदन का संदर्भ लें। 2. संबंधित जानकारी अनुबंध में प्रस्तुत है।	Madam, Application under Right to Information Act, 2005 - RBIND/R/E/22/04501 Please refer to your application dated July 05, 2022 seeking certain information under the Right to Information Act, 2005. 2. The information is furnished in the Annex.
3. हम आपको सूचित करना चाहते हैं कि भारतीय रिज़र्व बैंक में प्रथम अपील अधिकारी का नाम, श्री आर. एस. रथ, कार्यपालक निदेशक, भारतीय रिज़र्व बैंक, भुवनेश्वर और निपटान प्रणाली विभाग, 14 वीं मंजिल, केंद्रीय कार्यालय भवन, शाहीद भगत सिंह मार्ग, फोर्ट, मुंबई-400001 है। उपरोक्त उत्तर के संबंध में आप यदि कोई अपील करना चाहें तो उसे दस घंटे की प्रतिक्रिया के 30 दिनों के भीतर प्रथम भारतीय प्राधिकारी को भेज सकते हैं।	3. We would like to inform that the First Appellate Authority in Reserve Bank of India is Shri R. S. Ratho, Executive Director, Reserve Bank of India, Department of Payment and Settlement Systems, 14 th Floor, Central Office Building, Shahid Bhagat Singh Road, Fort, Mumbai - 400001. Appeal, if any, in respect of the above reply, should be preferred within 30 days to the First Appellate Authority.
भवदीप/ Yours faithfully बाभ्रदेवन - पी (पी बाभ्रदेवन) केंद्रीय जन सूचना अधिकारी / Central Public Information Officer Encl: As above	
<small>भुवनेश्वर और निपटान प्रणाली विभाग, केंद्रीय कार्यालय, 14 वीं मंजिल, केंद्रीय कार्यालय भवन, शाहीद भगत सिंह मार्ग, फोर्ट, मुंबई - 400001 फोन: (01) 2210149/50; फेक्स: (01) 2210150; ई-मेल: rti@rbi.org.in Department of Payment and Settlement Systems, Central Office, 14th Floor, Central Office Building, Shahid Bhagat Singh Marg, Fort, Mumbai - 400001 सिद्धि अन्वयन, सार्वजनिक प्रयोग</small>	

Annex		
Sr. No.	Information Sought	Reply
The information sought herein pertains to the system audit report of payment and cryptocurrency companies done by RBI. Please provide information on the queries below:		
1.	Please provide an exhaustive list of companies whose system audit reports have been done by RBI.	No such information is available with Reserve Bank of India (RBI).
2.	Please provide an exhaustive list of companies whose system audits have been done by the companies themselves and reports provided to RBI.	Information in the desired manner is not available with RBI.
3.	Please provide detailed information on the technical audit of data management practices that may have been conducted by RBI or by the companies and the report submitted to RBI.	No such information is available with RBI.
4.	In case the system audits have been done by the companies themselves, please provide whether any cross-verification has been carried out by RBI.	
5.	Please state whether system audit reports of payment and cryptocurrency companies are done by RBI internally or by any third parties. If so, please provide the manner in which it was carried out.	The query is not clear.
6.	Please provide information on the Standard Operating Procedure for conducting technical system audits that may have been issued by RBI. Please provide a copy of the same.	
7.	In case any third-party company has been engaged, please provide information on the payments made for such services rendered along with a copy of the contract entered into.	No such information is available with RBI.
8.	Please provide copies of the system audit reports as sought in query number 3 and 4.	

Figure 3: Reply to RTI request from RBI.