WARNE: A Stalkerware Evidence Collection Tool

Philippe Mangeard, Bhaskar Tejaswi, Mohammad Mannan, Amr Youssef

Concordia University, Montreal, Quebec, Canada

Abstract

Intimate partner violence (IPV) is a form of abuse in romantic relationships, more frequently, against the female partner. IPV can vary in severity and frequency, ranging from emotional abuse or stalking to recurring and severe violent episodes over a long period. Easy access to stalkerware apps helps foster such behaviors by allowing non-tech-savvy individuals to spy on their victims. These apps offer features for discreetly monitoring and remotely controlling compromised mobile devices, thereby infringing the victim's privacy and the security of their data. In this work, we investigate methods for gathering evidence about an abuser and the stalkerware they employ on a victim's device. We develop a semi-automated tool intended for use by investigators, helping them to analyze Android phones for potential threats in cases of IPV stalkerware. As a first step towards this goal, we perform an experimental privacy and security study to investigate currently available stalkerware apps. We specifically study the vectors through which vulnerabilities found in stalkerware apps could be exploited by investigators, allowing them to gather information about the IPV services, IPV abusers, and the victims' stolen data. We then design and implement a tool called `WARNE`, leveraging the identified flaws to facilitate the information and evidence collection process. In our experiments, we identified 50 unique stalkerware apps and their corresponding download websites that are still reachable, including one available on the Google Play Store. Among these apps, we found 30 that were free or offered a free trial. We enumerated and experimentally verified several invasive capabilities offered by these apps to clearly identify the severe privacy risks posed by them. We also found that most stalkerware apps store private information locally on the compromised device, potentially giving away information about the abuser. Our evidence-gathering tool found data related to the abuser and/or the stalkerware company, such as account credentials, dashboard URLs, and API tokens in 20 apps out of 30 tested apps. We hope our tool will help IPV victims and investigators against the growing threat of stalkerware abuse.

1. Introduction

Intimate partner violence (IPV) has been recognized by the World Health Organization as a major global public health concern, affecting people of all genders, ages, and backgrounds and causing long-term health, social, and economic consequences [11]. IPV can take various forms, from physical violence to psychological abuse. Stalking is one form of harassment involving privacy invasion and remote monitoring, which has been greatly facilitated by programs called *stalkerware* (a.k.a. *spouseware* or *creepware*). Stalkerware apps are generally mobile applications enabling undetected remote control and activity monitoring of the compromised victim's phone by the abuser.

Previous studies have demonstrated the large size of the stalkerware landscape on mobile platforms, with hundreds of dual-use apps available on the Google Play Store [1] and dozens of companies distributing stalkerware apps outside the official Android marketplace [22]. ESET [20] analyzed 86 stalkerware applications and reported over 18 critical vulnerabilities that could allow an attacker to perform actions such as remotely controlling the victim's device, hijacking an abuser's account, capturing victim's data, and uploading forged data on behalf of the victim. Liu et al. [22] analyzed 14 such Android apps and identified the mechanisms used for spying and hiding from detection, as well as the security failings of these apps (e.g., not using HTTPS). On the other hand, various online services exist to help victims and raise awareness [31]; see also `stopstalkerware.org`. Other apps [17] and open source projects [9] aim at mitigating the direct threat of stalkerware apps by detecting or uninstalling known stalkerware from mobile devices.

In short, existing IPV work mainly focuses on detecting and removing suspicious apps and analyzing their security weaknesses. However, to the best of our knowledge, no technical tools exist to help the victim (or an *investigator* who is helping the victim) to gather information, which could potentially be used as evidence against the abuser.

Contributions. We design and implement `WARNE`[1] to col-

Email addresses: `p_mangea@ciise.concordia.ca` (Philippe Mangeard), `b_tejasw@ciise.concordia.ca` (Bhaskar Tejaswi), `mmannan@ciise.concordia.ca` (Mohammad Mannan), `youssef@ciise.concordia.ca` (Amr Youssef)

[1]The tool is named after Kate Warne (1833–1868), best known as the first female detective in the US (`https://en.wikipedia.org/wiki/Kate_Warne`).

lect possibly incriminating and/or identifying information about an IPV abuser by leveraging a set of common security weaknesses in stalkerware apps, their backends, and the stored data on the victim device. First, we collect 80 well-known, easy-to-find stalkerware apps from the web, and identify 50 unique APKs. Among these 50 apps, 30 are free or offer a free trial. Then, we analyze these free apps and the web dashboards they provide to an abuser against five common vulnerability types (as identified in past work, e.g., ESET [20]). Through a vulnerability analysis of stalkerware apps, we explore ways in which such security flaws give away information about the stalkerware and the abuser. Our tool `WARNE` leverages these flaws to collect evidence from the victim's device (e.g., locally stored app data from the stalkerware app). In this process, the tool searches for identifying information about the abuser (e.g., account information and email addresses) in the collected data. It provides an easy-to-use graphical user interface (GUI) that allows the victim/investigator to navigate through readable files and databases found during the analysis. By default, `WARNE` uses a list of common stalkerware apps' package names for automated analysis. However, any suspicious app from the device can be selected for analysis. We evaluate `WARNE` against 30 stalkerware apps, and for 20 of them, the tool is able to gather useful information about the abuser.

Ethical considerations. Following the guidelines from our university's research ethics unit, all testing performed on stalkerware apps is done with our own (test) accounts and with a dedicated phone as the victim's device. We do not record or save any information from the online dashboard other than our own. The designed tool runs locally on the investigator's machine and no information is shared with any third party. Regarding the ethical aspects of the tool's usage and the admissibility of the evidence gathered in their respective jurisprudence, an investigator is advised to seek legal help. The tool's GUI displays an explicit warning indicating the same. The tool has been uploaded to a private GitHub repository and will be made available to researchers and IPV clinics upon request. A public repository (with some artifacts) is also available here: `https://github.com/PhilippeMangeard/WarnePublic`.

2. Related Work

Over the past years, several studies have been conducted on the stalkerware industry [38, 16], revealing the expanding landscape of spyware apps and keeping track of emerging actors in the field [8, 14, 6, 30, 36]. Chatterjee et al. [2] provided one of the first significant studies of the intimate partner stalking (IPS) spyware ecosystem where they identified several hundreds of such IPS-relevant apps. While they found dozens of overt spyware tools, the majority are "dual-use" apps, i.e., apps that have a legitimate purpose (e.g., child safety or anti-theft), but can be easily and effectively repurposed to spy on a partner. They also show how some dual-use app developers are encouraging their use of IPS via advertisements, blogs, and customer support services. The authors analyze existing anti-virus and anti-spyware tools, which mostly fail to identify dual-use apps as a threat. Given the increasing exposure of intimate partner violence, which is further exacerbated by the growing online presence of various actors in the field (such as the distribution of stalkerware apps and the availability of more help services on the internet) [4, 10, 29, 32, 34, 23, 24, 17, 2], there is a growing need for a more in-depth analysis of the mechanisms employed by these new tools.

Freed et al. [12] provide a qualitative study that focuses on how IPV abusers exploit technology to intimidate, monitor, impersonate, and harass their victims. The authors argue that many forms of IPV are technologically unsophisticated from the perspective of IT/security experts. For example, these attacks are often carried out by a user interface-bounded adversary, i.e., an authenticated adversarial user who can interact with the victim's device or account via standard user interfaces, or by installing a readily available application that enables remote spying on the victim. Still, such attacks not only harm the victims but are also difficult to counteract because they undermine the dominant threat models considered during the design stage of most systems (e.g., attackers not having physical device access). Thomas et al. [33] argue that security, privacy, and anti-abuse protections are failing to address the widespread threat of online harassment.

Our work relates more with the studies of stalkerware apps' technical capabilities [6, 7, 21, 28]. These studies considered one or two specific apps and provided insight regarding the poor security state of these apps, highlighting flaws such as inconsistent encryption usage or hard-coded secrets. More recently, Liu et al. [22] investigated the available features of 14 leading Android spyware apps and provided details about their mechanisms.

Security vulnerabilities in stalkerware systems are abundant, with apps like mspy [19], TheTruthSpy [35], Cerberus [27], spyHuman [5] and LetMeSpy [37],[2] leaking data of hundreds of thousands of users through data breaches. Unprotected databases is just one of many other flaws that are found on stalkerware apps and can be potentially exploited. ESET [20] manually analyzed 86 stalkerware apps and reported over 18 critical vulnerabilities that let an attacker perform actions such as remotely controlling the victim's device, hijacking an abuser's account, capturing victim's data or uploading forged data on behalf of the victim. They reported a substantial growth of stalkerware usage in 2020, which correlates with an increase in IPV reports during the COVID-19 pandemic [23].

Regarding stalkerware apps detection and mitigation, notable studies include comprehensive records of known stalkerware apps [9, 20], often used as a baseline for spyware detection tools. Similar to traditional anti-viruses,

[2] LetMeSpy's website has been taken down after our first tests and is now unavailable.

Android spyware detectors mostly work via package name analysis, therefore requiring thorough and up-to-date spyware package databases. Havron et al. [17] developed a tool (ISDI) to detect spyware on Android and iOS devices. ISDI can help remove tracking apps from the phone but only provides limited information such as static app package data and the phone's resources information. New detection techniques such as those using activity analysis with machine learning [26] and traffic examination through external hardware [18] are also emerging.

An analysis of the stalkerware monetization ecosystem has been conducted by Gibson et al. [13] on over 6000 Android apps, sampled from the Stalkerware Threat List in 2021. They mainly looked for keywords in the apps' code and evaluated the presence of payment/advertisement libraries used by the stalkerware apps.

We prioritize evidence collection through the security analysis of stalkerware apps and websites. We specifically focus on vectors that could be leveraged to get information about the abuser or the stalkerware distributing company.

3. Stalkerware Overview and Dataset

In this section, we first provide an overview of the way stalkerware apps are set up and used. Next, we discuss the set of stalkerware apps we used for analysis, and their privacy-invasive features.

Stalkerware overview. The person (i.e., abuser) who intends to stalk their intimate partner (i.e., victim) first creates an account on the stalkerware's web dashboard. While many stalkerware apps are available for free (e.g., TheTruthSpy), some require a paid subscription (e.g., MobileSpy). The abuser requires physical access to the victim's phone to install the stalkerware app. During installation, the abuser grants the app extensive privacy-invasive permissions. Thereafter, the abuser logs into the app on the mobile phone, with their stalkerware account's login credentials. Unaware of the stalkerware app's installation, the victim continues to use the compromised mobile device as usual. With the app successfully installed and configured, the abuser can log into the stalkerware's web dashboard (e.g., see Fig. 1), and remotely monitor/control the victim's phone.

Stalkerware dataset. Up-to-date lists of stalkerware apps are not readily available; although past work provides multiple (non-exhaustive) lists of such apps [20, 25, 22], some of them became outdated over the years; for example, some apps listed in the ESET paper [20] are no longer available for download or have changed their names (e.g., Xnore, Appmia or AntiFurto Droid Web are all unavailable as of Dec. 2023). Some stalkerware websites have been shut down and/or no longer distribute their apps. Others have been re-opened under modified names with download links. We manually tested the online availability of stalkerware apps listed in the ESET 2021 report [20] as well as AssoEchap's Stalkerware Indicators Of Compromise (IOC)

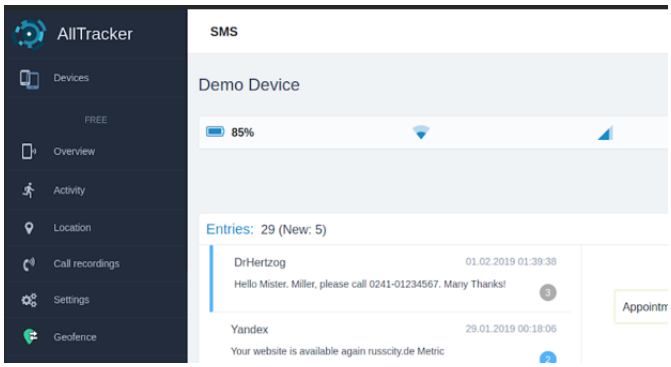

Figure 1: Web dashboard of the AllTracker app

list [9]. We also tried getting access to the Stalkerware Threat List (STL) established by the Coalition Against Stalkerware, which has been used in past studies [13]. Unfortunately, we could not get any account for this service as their registration website was out of order as of July 2023. We contacted them via email and explained our study, but received no response. We manually gathered information about 96 stalkerware apps, the availability of their online websites, their economic model (free, paid subscription, free trial), and whether the terms such as "spouse", "husband" or "wife", along with "spy" or "cheating" are present on the app's download source page (website or Google Play Store page). Checking the latter allows us to separate stalkerware apps from legitimate apps that could be used in an unethical way (we refer to such applications as "dual-use apps").

We found 80 apps with reachable online websites. Many of them were redirected to the same APK download and turned out to be duplicates of other stalkerware apps. In the end, we identified 50 unique ones out of these 80 apps. Notable examples include TheTruthSpy with 6 different websites, Cocospy with 7, and Mspy with 4 (similar duplicates were also reported in the past [9]).

Out of the 50 apps found online, 49 of them explicitly referenced spouseware features like the ability to "monitor cheating spouses", or "verify spouse loyalty". Three stalkerware websites showcased articles promoting such features, links to which are, however, only accessible through search engine results and are unreachable through normal website navigation. Several websites promote features like child monitoring or employee surveillance but showcase functionalities or reviews referring to intimate partners.

We chose to prioritize testing of free apps and the ones offering free trials on account creation (30 out of 50). We also note that very few of these apps were available on the Google Play Store, mainly because of their terms of service change in October 2020 [15], prohibiting the publication of apps "presenting themselves as a spying/secret surveillance solution". Only one app in our list is available on the Play Store,[3] offering features such as GPS tracking and

[3]https://play.google.com/store/apps/details?id=com.phonetrackerofficial1

contact information gathering. It has been available since 2014 and has been downloaded over a million times. Even though it does not advertise itself as a surveillance tool in the app description, many reviews for this app on the Google Play Store feature the terms "spouse" and "spying" and praise the efficiency of the app to secretly spy on someone. We reported this app to Google as it violates Developer Program Policies.[4]

To identify potential vulnerabilities in the stalkerware environment, it is crucial to understand what kind of data these apps gather and through which mechanisms (see e.g. [22]). For each tested stalkerware, we gather information about the features they provide by creating an account and accessing the online dashboard. We also search on the app's website and its online dashboard for a comprehensive list of capabilities that the stalkerware can offer. When possible, we also look at the data packets sent by the phone to upload information to the backend servers during regular use. This step is the base for the rest of our analysis, as testing specific features enables us to understand their mechanisms and their flaws.

4. Identifying Vulnerabilities in Stalkerware apps for Evidence Collection

In this section, we first describe the analysis setup that we used during our testing. Then, we provide a detailed methodology to detect five commonly found vulnerabilities in stalkerwares. Two of these vulnerabilities (i.e., cross-site scripting, and insecure local data storage) are later leveraged in our proposed tool for evidence collection.

4.1. Analysis Setup

We use a Google Pixel 3 phone running Android 12, and a Genymotion virtual device running Android 10, with Google API installed. The Google Pixel 3 phone is rooted to allow superuser rights in the Android Debug Bridge (ADB) shell and certificate pinning bypass for our testing. Genymotion virtual devices are rooted by default. Our setup consists of an analysis device (workstation/laptop) to which the victim's device is connected via ADB (requires enabling developer options on the phone). Whenever possible, we download the stalkerware APK directly on the device; otherwise, we install it from the computer using ADB. In some cases, the website's download link pointed to an installer that must be run first. In this situation, we use ADB to get access to the app's local files once fully installed and pull its APK file from the phone through a superuser shell. Fig. 2 illustrates our analysis setup and shows the different steps composing our approach.

We use Frida's[5] built-in tools for simple native Java function hooking, as well as process listing and information gathering. We also use Burp Suite's[6] proxy tool to intercept/modify HTTPS traffic between the compromised mobile device and the stalkerware's backend server.

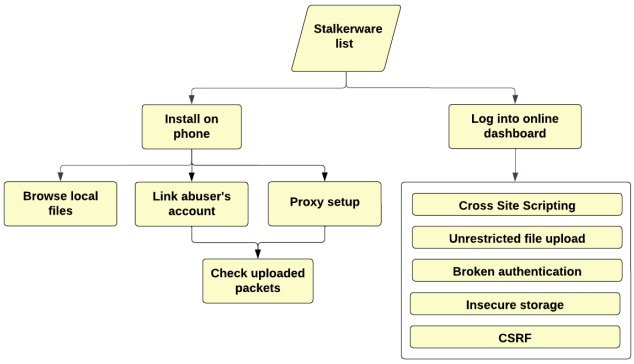

Figure 2: Overview of the stalkerware security analysis methodology

4.2. Stalkerware Vulnerabilities

During our study, we identified and tested five main vulnerability types in the stalkerware app ecosystem, which we either found during our tests or were reported in prior work (e.g., [20]). This ecosystem includes the stalkerware's mobile app, its backend server, and the web dashboard.

Cross-site scripting (XSS). After identifying what data the stalkerware gathers from the victim's phone, we check if the web dashboards of the stalkerware apps lack user-input sanitization, which could lead to XSS vulnerabilities.[7] XSS is possible when uploaded information like contact names, text messages, calendar data, or any other user-input field is not verified by the Android app or the backend system. This allows unrestricted usage of special characters in strings, which when displayed on the abuser's dashboard, can trick the abuser's browser into interpreting the unrestricted input as code. XSS would allow an investigator to inject JavaScript code into the web application through the victim's device. We first compile a list of easy-to-edit inputs in the victim's device that are being reflected on the abuser's dashboard (the most common ones are contacts and text messages). We then use XSS fuzzing payloads from a self-hosted instance of XSShunter-express[8] and manually inject them into our identified inputs. We add a new contact in the victim device's phone book and provide the XSS payload in the contact's name. We also send text messages containing the payload to and from the victim's phone. When the payload is executed on the abuser's machine, the investigator can collect information about the abuser's dashboard, or the machine on which they are logged in.

Note that we need to inject the payloads into the victim's device only once. As soon as the stalkerware app uploads it to the backend server, the custom string will be

[4] https://play.google.com/about/developer-content-policy/
[5] https://frida.re/
[6] https://portswigger.net/burp
[7] https://owasp.org/www-community/attacks/xss/
[8] https://github.com/mandatoryprogrammer/xsshunter-express

displayed on the dashboard (when the abuser browses the corresponding page), even if the XSS payload is deleted from the phone afterward. In this case, each time the abuser opens a web page containing the malicious string, the injected script will be executed.

Unrestricted file upload. One of the key operations performed by stalkerware applications is to regularly upload data from the phone, including photos, videos, and other files from the victim's phone storage to a remote server. We observed that 22 of the tested apps can access the device's internal storage, e.g., downloaded files, SD card storage, and even system files if the app is given admin rights. However, such file uploads lack file content verification[9] during data synchronization, allowing custom files (with any chosen payload by the victim/investigator) to be transferred from the phone to the backend server and later downloaded by the abuser. An investigator could take advantage of this behavior to collect information about the abuser and their environment by uploading files[10] with specifically crafted payloads that would be triggered when the abuser downloads those files from the stalkerware dashboard, and opens them on their machine. Using a similar method as for XSS, they could place files in the victim's phone and wait for them to be uploaded to the dashboard.

Broken authentication and access control. Since the majority of stalkerware apps use a centralized platform to store victim's data, we verify if these platforms are vulnerable to broken authentication and access control.[11] We first create two accounts (one for the abuser and one for the investigator) on the stalkerware dashboard. We install the abuser app on the victim's phone using the abuser's credentials and browse the abuser's dashboard. We run Auth Analyzer[12] in the background while we browse, by configuring it to replay requests using the investigator's account session tokens after logging in from another browser. An access control vulnerability is detected in case a replayed request (from the investigator's session) generates the same response as the browsed request (from the abuser's session). Similarly, to test for broken authentication, we configure Auth Analyzer to replay requests using null or blank sessions; a successful response code indicates the presence of a broken authentication.

Furthermore, we notice that JSON Web Tokens (JWTs) are commonly used for authentication, managing user sessions, and controlling access to resources in stalkerware applications. We check if the JWT signature field is properly verified by logging into the abuser dashboard and collecting all corresponding tokens. We then test for all signature-related flaws by supplying collected tokens to an open-source tool.[13] Note that the signature field allows the server to authorize the request after ensuring that the token has not been tampered with.

For data uploads from the victim phone to the stalkerware backend server, the server needs to identify and authenticate the device and store the transferred information accordingly. To test authentication flaws in this case, we use Burp Suite's built-in proxy to intercept data sent by the mobile device and check the packets sent during data upload. Without proper device authentication, it is possible to upload data to the backend server on behalf of the victim's device. This specific vulnerability can be used by an investigator to send payloads or other information to the dashboard without having to interact with the compromised device.

Insecure (online) storage. The victim's multi-media data (e.g., screenshots, images, videos, call recording audios) are generally stored differently compared to text-based data (e.g., social media chats or text messages). Text-based information is often directly displayed on their corresponding dashboard page, while pictures or videos can be shown through previews on the dashboard since viewing them requires generating a URL. It can be stored either on the cloud (e.g., AWS), or on the stalkerware's server itself, albeit in a different directory. In both cases, we check if access to sensitive multi-media content is protected. If access to such files is possible, the investigator could find evidence of data collection by the stalkerware.

First, we collect and store the list of all relevant multi-media URLs. This is done by first syncing the victim's device data to the stalkerware's server and then manually browsing the abuser's dashboard to identify all such URLs. We then make a curl[14] request to each of the collected URLs without providing any authentication token. A successful response (with `200 OK` status code and content body) indicates the presence of insecure storage of multi-media data. Second, we check if it is possible to guess the URLs to access the multi-media data of other victims. For example, the use of high entropy tokens (e.g., UUID) in the URL makes guessing infeasible, whereas the use of short numeric identifiers makes it possible for an adversary to quickly form and test potential URLs that may contain other victims' sensitive information. In the case of high entropy tokens, we make use of the Wayback Machine[15] to find any leak of such tokens. Lastly, we repeat the process of triggering curl requests on top of the log file, after deleting the abuser's account from the platform and uninstalling the Android app from the victim's device. This helps us to verify the retention status of the victim's multi-media data. This method could be used by the investigator to check whether the victim's information remains available after the abuser's account deletion.

Insecure (local) storage. Stalkerware apps also use the phone's internal storage to cache data such as collected

[9] https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload

[10] https://canarytokens.org/

[11] https://owasp.org/Top10/A01_2021-Broken_Access_Control/

[12] https://github.com/PortSwigger/auth-analyzer

[13] https://github.com/ticarpi/jwt_tool

[14] https://curl.se/

[15] https://archive.org/web/

contact names, text messages, installed apps, keylogger history, and app activity. These internal files may contain credentials used for data uploads to the backend servers, as well as information used to link the phone to the abuser's account. Typically, accessing the content of the internal storage of applications requires root privileges on Android. However, by leveraging Android application backup functionality, the same can be done without rooting the phone. In both cases, we use ADB to pull the stalkerware's internal files. If the app sets the "debug protection" parameter to prevent users from tampering with its local directory, we use the Android backup functionality to fetch the app's data. In other cases, we can directly pull the app directory with ADB pull (with a rooted phone) and browse the SQLite databases with an online tool.[16]

Cross-site request forgery (CSRF). In stalkerware apps, it is possible to induce abusers to perform actions that they do not intend to perform (e.g., sending remote commands to the victim's device) via CSRF.[17] It can be exploited by sending a link to the abuser and luring them to click on it. We detect CSRF vulnerabilities in all of the state-changing HTTP/s requests that are triggered upon browsing the stalkerware dashboard, from an abuser account. First, we check for the presence of any anti-CSRF tokens in the request body. Second, we check if those tokens are tied to the abuser session. Specifically, we test if the request can successfully be processed by supplying any valid anti-CSRF token. To do this, we log into the investigator's account and provide their anti-CSRF token to the abuser's state-changing requests. Successful execution of this request indicates the presence of CSRF.

5. WARNE - Steps for Collecting Evidence

In this section, we describe the workflow of our tool, WARNE, for collecting evidence against the abuser. We leverage possible vulnerabilities in stalkerware apps and the unprotected content of the local storage of these apps on the victim's device (see Sec. 4.2). WARNE generates a report consisting of all the data (e.g., IP address, email address) that can be used to infer the identity of the abuser. See Fig. 3 for an overview of WARNE's workflow. The tool's mechanism is split into eight steps:

1. Preliminary setup. Upon starting, WARNE allows the user to configure the analysis through three separate settings by providing: (a) a custom XSS payload pointing to a preemptively configured server; (b) a suspicious package name that may not be in our known stalkerware list; and (c) the device's backup password/secret if the analyzed phone is rooted or encrypted (i.e., when all backups are password-protected).

2. Package detection. WARNE scans the device to list all installed packages and flags suspicious ones: if the package

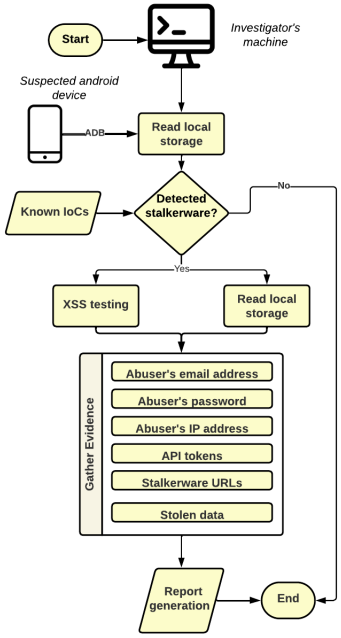

Figure 3: WARNE's evidence collection process

name is present in AssoEchap's stalkerware Indicators of Compromise list (which is fetched from its GitHub repository to ensure that it is always up-to-date), or if the package name matches the one provided by the investigator.

3. Information dump. For each flagged app, WARNE dumps information about the package (this does not require root). This information contains the app's installation date, requested permissions, local storage path, actions, and intents. We also check whether the app requires the phone to be rooted to fetch its local files.

4. Local data extraction. During this step, WARNE fetches all files related to the suspicious app that are stored locally. This is done differently depending on whether the target device is rooted or not. If the app's manifest explicitly states that backups are not allowed, the app's files cannot be extracted on a non-rooted phone.

5. XSS injector and listener. WARNE offers compatibility with XSShunter-express, a self-hosted XSShunter server. We automatically inject a user-given payload in the target device and listen for any payload trigger. WARNE then identifies the trigger source and appends payload information to the related text report. This listener feature requires WARNE to either run in the background on the computer or to be regularly booted up. Note that the XSShunter-express server needs to be configured separately, on an instance with a public DNS record.

6. Report content collection. Information gathered in the previous steps is then stored in a readable format. This includes a dump of the package information, all local file names, and their readable content (including .xml files, databases, and any other files containing textual information). Files that are empty, unreadable, or cannot be opened, are also logged in the report.

[16] https://inloop.github.io/sqlite-viewer/
[17] https://owasp.org/www-community/attacks/csrf

7. Report parsing. Once the full report content has been gathered, WARNE parses the collected data for relevant information. This gives users an initial idea of what can be found in the report, which they can then analyze further.

8. App removal. This optional step allows the user to uninstall the flagged app. Additional instructions are provided if the removal fails (e.g., due to the app being installed as a device admin). It should be noted that attempting to remove stalkerware apps may be noticed by the abuser and could lead to reprisal.

6. WARNE - Implementation

In this section, we first expand on the set of prerequisites required for WARNE to function correctly. Thereafter, we present details on how specific features (e.g., testing local storage) of WARNE are implemented.

6.1. Pre-requisites and Additional Tools

WARNE is designed to run on Linux and analyze Android devices. The analysis code is written in Python and features an in-browser graphical user interface for readability and ease of use. This interface is handled by the Dash library.[18] Communication with the phone is handled by the Android Debug Bridge (ADB), which must be installed on the host machine as well as enabled on the target phone. The latter can be done by turning developer options on and allowing USB debugging. The ADB tool lets us run commands on the target device and is core to most features offered by WARNE, including package detection, data extraction, app removal, and XSS injection.

WARNE uses the Android Backup Extractor[19] to extract data from encrypted backups. WARNE also uses Android Asset Packaging Tool (AAPT)[20] to perform various requests about packages installed on the target phone. AAPT can dump data about a specific app, which is used to fetch information from an app's manifest, such as its common name (i.e., the one displayed on the phone's interface). Upon startup, WARNE automatically checks for AAPT's presence and downloads it if necessary.

The handling of XSS payloads and payload triggers is done by XSShunter-express, a containerized instance of XSShunter with customizable settings and self-hosting capabilities. To allow compatibility with WARNE, we modified the tool to create text reports of triggered payloads and linked them to the machine hosting WARNE using the SSH File System (SSHFS). Our tool then regularly checks for trigger files' creation and updates its corresponding app's text report. To identify the trigger source app, we compare its originating URL to our IOC threat list and find a corresponding package name. In case no related app is found, the payload information is also stored in a separate file with all other recorded triggers. Since this feature requires a substantially complex setup, its setup is not mandatory for WARNE to function. Additionally, as this XSShunter instance is publicly accessible, standard server hardening guidelines (e.g., see [3]) must be followed.

6.2. Implementation of WARNE Features

Stalkerware detection. The tool fetches information about known stalkerware apps from a GitHub repository [9]. Among other details, the repository provides the package names of known stalkerware apps. The tool queries the device to obtain a list of packages installed on the victim's phone and checks if any such packages are installed on the device. If found, the tool records details such as the name of the detected stalkerware, the date of installation, and the permissions given to the app.

Testing local storage. Typically, accessing the content of the internal storage of applications requires root privileges on Android. However, by leveraging Android application backup functionality, the same can be done without rooting the phone. Depending on whether the analyzed phone is rooted or not, WARNE uses the most convenient method to fetch local data related to the identified/suspected stalkerware. For each known malicious app detected, the tool issues a backup command and copies the backup to the analysis system. The tool then parses the backup and reads the content of each file in the application's internal storage. For each SQLite database, it reads all the tables in the database and converts them into a readable text format. Note that the backup functionality may not work on every stalkerware, as it can be disabled via the `allowBackup` flag in the app's manifest.

Report generation. WARNE is designed to provide a thorough and easy-to-read text report of its analysis. An exhaustive report is created by appending all data (app stats and files that can be displayed as text) to a single text file. The report is then parsed with regular expressions to find relevant information. This includes any email address that is not the phone's primary user address, hostnames, and URLs, occurrences of words such as "username" or "password" and recognizable Google API keys/tokens.

XSS reports. WARNE sets the XSS payload by creating a contact on the phone whose name is the payload given by the victim/investigator during setup. The user can also choose to clear all the injected contacts from the device's contact list. Simultaneously, a thread is spawned to monitor the local folder linked to the XSShunter-express server. Information gathered by a triggered payload includes the time of the trigger, the source IP address, the URL of the page the payload has been fired on, its referring page, Non-HTTPOnly cookies, HTML data, and User-Agent.

7. Results

We present the results of both our stalkerware security analysis and our WARNE tests. We first provide a thorough

[18] https://plotly.com/dash/
[19] https://github.com/nelenkov/android-backup-extractor
[20] https://developer.android.com/tools/aapt2

list of all features offered by the stalkerware apps in our dataset before presenting the security weaknesses found in these features that could be leveraged to efficiently gather data about the abuser. Finally, we present the results of testing our tool on our dataset of 30 stalkerware apps.

7.1. Stalkerware App Capabilities

Most stalkerware apps use two separate systems in parallel: a stalkerware app installed on the phone and a web-based dashboard accessible by the abuser. This platform is linked to backend databases where the collected data can be found and also serves as a control panel through which the abuser can manage their subscriptions, enable/disable features, or send remote commands to the phone.

Table 1 compiles a comprehensive list of the data collected by 30 stalkerware apps for Android devices (duplicates excluded). 26 of them gather text messages and phone call logs, 27 of them feature GPS tracking and geo-fencing (triggering alerts whenever the target leaves a specified area), these are the most common capabilities available on such apps. Other noteworthy functionalities include secret live recording with the device's camera (17) or microphone (13), a keylogger collecting keystrokes, therefore potentially disclosing the victim's passwords to the abuser (14), access to file storage like photos, videos, or documents (16) and social media chat services such as Facebook Messenger, Instagram, Whatsapp or Viber (18).

Name	Text. messages	Calls	GPS	Contacts	Camera	Microphone	Notifications	Files	Keylogger	Screen	Apps	Social media	Browsing history	Remote cmd.	Wi-Fi
AllTracker	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓		
Android Monitor	✓	✓	✓				✓		✓		✓	✓	✓		
CallSmsTracker	✓	✓	✓									✓			
CatWatchful	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Cerberus					✓			✓	✓	✓				✓	✓
Couple Tracker	✓	✓	✓												
Easy logger	✓	✓	✓								✓				
Free Android Spy			✓	✓	✓									✓	
i-Monitor	✓	✓	✓	✓	✓						✓		✓		
iKeyMonitor	✓	✓	✓	✓	✓		✓		✓	✓	✓		✓	✓	
LetMeSpy	✓	✓	✓											✓	
Lost Android	✓	✓	✓		✓	✓					✓		✓	✓	✓
Meuspy	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	
MobileTrackerFree	✓	✓	✓	✓	✓					✓	✓	✓	✓	✓	
MobileSpy	✓	✓	✓	✓				✓	✓		✓	✓	✓		✓
Mycellspy	✓	✓	✓	✓	✓	✓		✓	✓			✓	✓		
OwnSpy	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Panspy	✓	✓	✓					✓	✓		✓	✓	✓		✓
Remote Audio Rec.						✓									
Reptilicus	✓	✓	✓			✓		✓				✓		✓	
Shadow SPY	✓	✓	✓	✓			✓	✓	✓		✓			✓	
Snoopza	✓	✓		✓	✓					✓		✓	✓		
SpAppMonitoring	✓	✓	✓			✓		✓			✓	✓	✓		
Spy24	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓
Spyhuman	✓	✓	✓	✓		✓		✓			✓	✓	✓		
Spyic	✓	✓	✓					✓	✓		✓	✓	✓		
Spylive 360	✓	✓	✓		✓	✓		✓			✓	✓	✓	✓	
TheTruthSpy	✓	✓	✓	✓	✓	✓	✓		✓		✓	✓	✓		✓
Tispy	✓	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓		
Track My Phone Rem.			✓		✓										

Table 1: Features available on tested stalkerware apps. A checkmark means that data is collected by the app.

After being collected by the Android app, the victim's data is sent to the abuser in the following ways. (1) In 29 out of 30 cases, data is uploaded to an online database which can be browsed by the abuser through the web dashboard. The stalkerware database therefore stores all pictures, text messages, contact names, and other collected information from the devices monitored by the platform. (2) When not using a centralized database system, data can be directly sent to the abuser's email address via regular reports. Apps using this approach however tend to offer fewer features than the database approach.

7.2. Security Vulnerabilities

The complete list of vulnerabilities found within each app can be found in Table 2.

Cross-site scripting (XSS). We identified 20 different apps whose online dashboards did not conduct any input validation before displaying the collected victim data on the web page. Depending on where the XSS payload was planted (e.g., contact list, text message, file names), the investigator can see related data displayed on the web page on the XSS report. For 18 out of 20 apps, XSS could be performed through text message injection, by either sending or receiving a message containing a payload. Among these 18 apps, 15 of them feature social media compatibility and were therefore also vulnerable to XSS payload injections through social media chat services. 16 out of 20 apps were vulnerable through the contact list and 11 out of 20 through filenames.

XSS payloads can also be used to verify the presence of a stalkerware app on a phone, e.g., by sending text messages or adding a contact name containing a payload to the device and waiting for the potentially compromised phone to upload it to the app's backend servers. It should however be noted that this approach relies on the abuser logging into the online dashboard and loading the page displaying the payload. It is also possible that such an attack could raise an abuser's suspicion, if they notice strangely formatted messages on the dashboard. As XSS payloads are customizable, they could be programmed to send a notification when it is activated. This would make it possible to hijack the abuser's session without delay, with increased odds that the session cookies are still valid to the server.

Broken authentication. We found that 8 stalkerware apps are vulnerable to account hijacking or unauthenticated command transmission due to broken authentication. Two apps (CatWatchful and Shadow Spy) use the Google Identity Toolkit for credential verification and account management. It uses a token to identify the abuser on the victim's device, which is exposed inside the */shared_prefs* directory on the phone. This token can be used to issue commands to the monitored device, but also to request API calls through the Google Identity Toolkit (e.g., to delete the abuser's stalkerware account). Another app (Lost Android) uses Google Cloud Messaging (GCM) to upload collected data (using Google's servers as intermediates for data upload and commands). The GCM key can be found unprotected on the victim's phone and could be used to craft data upload packets.

A button in the CatWatchful app also redirects to the abuser's online dashboard and leaks their credentials in the redirection URL. The dashboard of LetMeSpy was only

accessible through HTTP, therefore exposing the abuser's credentials. The JWTs used for authentication on Spyic's dashboard were vulnerable to null signature attacks, allowing for easy account takeover.

To authenticate the device to the backend platform during data uploads, stalkerware apps can use multiple identifiers, including a license number entered by the abuser, the phone's International Mobile Equipment Identity (IMEI) number, a fixed session ID, or the abuser's credentials. Data uploads from the mobile device are poorly secured in four apps, allowing replay attacks on packets sending information about the phone, installed apps, contacts, messages, or GPS location. Authentication of the device is made with the license entered by the abuser and a session ID that stays unchanged even after multiple data uploads.

We also found that none of the stalkerware apps use certificate pinning. This allows for easy interception of the packets during data upload with only a few configuration steps on the phone. This means that any man-in-the-middle[21] attacker could collect the stalkerware authentication credentials with a proxy.

Insecure data storage. We identified issues regarding insecure data storage in 6 tested stalkerware apps. 3 of them failed to curtail access to files such as pictures, that were requested by the abuser on the online dashboard. Such files were accessible via static URLs, allowing unrestricted access to the file to anyone, regardless of authentication. However, the generated links had a limited period of validity (24 hours on average).

This vulnerability was mostly tested with pictures uploaded from the phone to the stalkerware's backend server and then requested from the dashboard. However, as it is a flaw inherent to the backend database configuration, all other data that can be given a URL on request from the dashboard could potentially be accessed by an unauthorized person. For pictures, generated URLs are made up of a mobile device's identifier along with either the time-stamp at which the picture was taken or uploaded, or seemingly random tokens.

Stalkerware apps also keep sensitive data about the abuser on the mobile device itself. 4 different apps store information in easily accessible locations on the mobile device, such as shared preferences. Data such as the abuser's email address, the stalkerware registration license, the application unlocking PIN code, and even the abuser's password can be found in the internal files. Even though some cases require the phone to be rooted, these pieces of information can be used to identify the abuser or execute commands that would be reserved for the stalker.

Two apps provide functionalities to uninstall the stalkerware application from the phone, either remotely or from the phone itself. These ways of deleting the app differ from manually removing it from the phone's settings, as mechanisms are used to prevent access to such features (automatically redirecting the user to another legitimate app's settings when trying to access the stalkerware settings). These functionalities require authentication to be used, which can be bypassed by looking for the corresponding password/verification token stored on the phone.

In three stalkerware apps, we also found databases containing a summary of all gathered data, as well as credentials like the abuser's email address and password or the device's identifier to the backend server. These databases could be used as evidence of the collected information in case of account deletion on the abuser's side. We also identified 7 websites that use Google Firebase as their online database service, with 3 being misconfigured, leading to partial or complete leakage of all users' information.

Unrestricted file upload. During our analysis, we have not found a stalkerware conducting any kind of file verification when requesting files from the dashboard. This means that sending malicious files to the backend servers for them to be downloaded by the abuser is easily doable. Anyone knowing the victim's phone number could send a malicious file (e.g., via a text message). The stalkerware will then automatically upload it to the online dashboard for it to be downloaded by the abuser.

The protection provided by the abuser's system is the only variable that could influence the gravity of such a vulnerability. Combined with data transfer presenting broken authentication mechanisms, an attacker could send files containing malicious code to the dashboard without having to download them on the phone. Someone could also send the payload directly to a victim unaware that they are being monitored, as the file only needs to stay on the phone for a relatively short amount of time for it to be uploaded to the stalkerware's backend server.

Cross-site request forgery. We found 4 apps that are vulnerable to cross-site request forgery (CSRF) attacks. The change password functionality in LetMeSpy is vulnerable to CRSF, making it possible to take over the abuser's account. In Spapp Monitoring and Panspy, it is possible to change the destination email address of the app's notifications to receive all information in place of the stalker.

7.3. Test Results for Stalkerware Apps

We tested `WARNE`'s effectiveness on a non-rooted Samsung Galaxy M02 phone running Android 11, and then on the same device after rooting it. We chose to use two different setups for our testing conditions to be closer to real-life situations. We tested our tool on 30 free stalkerware apps (or premium ones offering free trials) and successfully flagged 24 of them. It is still possible to analyze them with `WARNE` by manually providing the package name to the tool or by enabling non-trusted source detection (flagging apps not installed from the Play Store).

Out of the 30 apps we tested, 12 of them allowed ADB backups of their local data to be performed. After extracting the files from these apps' local storage and parsing the

[21] `https://www.rapid7.com/fundamentals/man-in-the-middle-attacks`

Vulnerability	Apps
XSS	Meuspy, Message Call tracker, Ownspy, Track My Phone Remotely, Tispy, MobileTrackerFree, Spytomobile, Free Android Spy, iKeyMonitor, Shadow SPY, SpAppMonitoring, Flexispy, Alltracker, A-spy, Mycellspy, Android Monitor, Spy phone labs phone tracker, CallSmsTracker, Lost Android, Reptilicus
Unrestricted File Upload	All apps
Broken Authentication	CatWatchful, Shadow SPY, LetMeSpy, Message Call tracker, Lost Android, SpAppMonitoring, Reptilicus, Spyic
Insecure Storage	Meuspy, CatWatchful, iKeyMonitor, Shadow SPY, AllTracker, Spyic
CSRF	LetMeSpy, OwnSpy, SpAppMonitoring, Panspy

Table 2: Vulnerabilities identified within each tested apps.

generated reports, we found occurrences of the abuser's email address in 15 of them. 5 apps also stored the abuser's password (in cleartext, or as an MD5 hash for OwnSpy) next to the email address. OwnSpy also stores a value called "encryption_key" as an MD5 hash which is used to communicate with the server. The abuser's password for CallSmsTracker's account was also found in a .xml file, but was labeled as "spin".

Keylogger features offered by stalkerware apps can list a large amount of inputs in tables. Tables containing keylogger data are the main trigger cause for the detection of certain elements such as email addresses. We manually went through occurrences to verify their relevance and detect false negatives. Five apps (PanSpy, Meuspy, MobileSpy, SpyLive360, and AllTracker) featured keylogger tables with all collected inputs.

URL detection can be permissive enough to be able to flag links with no "https://" or "www." prefixes but could sometimes result in longer processing time. However, manual verification of such cases is possible (for example, if the flagged text is the package name). 6 occurrences of the stalkerware dashboard's URL were found in `WARNE`'s generated reports. Other kinds of URLs were also found, like in some of Panspy's SQLite tables labeled "webWhite", "WebBlack", "webFilterClass" and "traffic" (the first three actually being web filtering lists with specific domain names. Most of the stored URLs pointed towards domains like youtube.com, amazon.com or aol.com, and one of them pointed to a taobao.com domain.

Finally, two apps (Shadowspy and SpAppMonitoring) triggered the XSS payload we had injected into the phone's contact list. However, we noticed that some online dashboards that displayed the XSS payload without input sanitization were not necessarily firing it back to our XSShunter-express server. It could be possible to avoid this issue by trying other injection media (i.e., text messages, file names, etc.) because some online dashboards use different display mechanisms depending on the data (some of which might be more responsive to XSS payloads). Table 3 shows a summary of items found by `WARNE` in the tested stalkerware apps (unlisted apps provided no relevant result).

App	Abuser's email	Abuser's password	Dashboard URL	XSS trigger	API tokens/creds	Other URLs	Stolen data
AndroidLost	✓			✓			
CallSmsTracker	✓	✓					
CatWatchful	✓	✓	✓				✓
Cerberus				✓			✓
Mobilespy							✓
MyCellSpy	✓		✓				
Panspy	✓		✓		✓	✓	
OwnSpy	✓	✓			✓		✓
Shadow Spy	✓	✓		✓	✓		
Spappmonitoring			✓	✓	✓		✓
Spy24	✓	✓			✓		
AllTracker	✓					✓	✓
EasyLogger	✓				✓		✓
FreeAndroidSpy	✓						✓
iKeyMonitor	✓						
Meuspy	✓				✓	✓	✓
MobileTrackerFree					✓		✓
SpyLive 360			✓		✓		✓
Spyhuman	✓						
TheTruthSpy	✓	✓	✓			✓	

Table 3: `WARNE`-parsed evidence and other relevant values found in tested stalkerware apps. Apps in the second half of the table require the phone to be rooted to fetch files from the phone.

8. Conclusion

We presented `WARNE`, a tool facilitating the gathering of information and evidence on stalkerware apps and the responsible abuser. To develop this tool, we performed a systematic experimental privacy and security analysis of 30 unique stalkerware APKs available online and identified their features and vulnerabilities that could be exploited for this purpose. Many of these invasive capabilities were enumerated and experimentally verified to find ways of using them against the abuser. In this context, we found apps/services vulnerable to various exploitable attacks, including broken authentication mechanisms, insecure storage of sensitive data, and other vectors that could be leveraged to gain information about the IPV perpetrator. These findings were used to develop our tool, which was tested against 30 apps and successfully found relevant data including abuser's credentials, dashboard URLs, and stolen data evidence in 20 of them. We believe that this first step towards stalkerware evidence collection can lessen the threat posed by such apps by giving concrete resources that organizations and investigators could use to support IPV victims.

References

[1] M. Almansoori, A. Gallardo, J. Poveda, A. Ahmed, and R. Chatterjee. A global survey of Android dual-use applications used in intimate partner surveillance apps. In *Proceedings on Privacy Enhancing Technologies Symposium*, 2022.

[2] R. Chatterjee, P. Doerfler, H. Orgad, S. Havron, J. Palmer, D. Freed, K. Levy, N. Dell, D. McCoy, and T. Ristenpart. The spyware used in intimate partner violence. In *IEEE Symposium on Security and Privacy*, pages 441–458, 2018.

[3] CIS. CIS Ubuntu Linux Benchmarks, 2023. `https://www.cisecurity.org/benchmark/ubuntu_linux`.

[4] S. Clevenger and M. Gilliam. Intimate partner violence and the internet: Perspectives. *The Palgrave Handbook of International Cybercrime and Cyberdeviance*, pages 1333–1351, 2020.

[5] J. Cox. Hacker steals customers' text messages from Android spyware company, 2018. `https://www.vice.com/en/article/qvm44m/hacker-steals-text-messages-android-spyware-company-spyhuman`.

[6] J. Dalman and V. Hantke. Commercial spyware-detecting the undetectable, 2015. `https://www.blackhat.com/docs/us-15/materials/us-15-Dalman-Commercial-Spyware-Detecting-The-Undetectable.pdf`.

[7] S. Desai. Why you shouldn't trust "safe" spying apps!, 2018. `https://www.zscaler.com/blogs/security-research/why-you-shouldnt-trust-safe-spying-apps`.

[8] S. Desai. A new wave of stalkerware apps, 2019. `https://www.zscaler.com/blogs/security-research/new-wave-stalkerware-apps`.

[9] Echap. Stalkerware indicators of compromise, Dec. 2022. `https://github.com/AssoEchap/stalkerware-indicators`.

[10] C. El Morr and M. Layal. Effectiveness of ICT-based intimate partner violence interventions: a systematic review. *BMC public health*, 20(1):1–25, 2020.

[11] D. Freed, S. Havron, E. Tseng, A. Gallardo, R. Chatterjee, T. Ristenpart, and N. Dell. "Is my phone hacked?" analyzing clinical computer security interventions with survivors of intimate partner violence. In *Proceedings of the ACM on Human-Computer Interaction*. ACM New York, NY, USA, 2019.

[12] D. Freed, J. Palmer, D. Minchala, K. Levy, T. Ristenpart, and N. Dell. "a stalker's paradise": How intimate partner abusers exploit technology. In *CHI conference on human factors in computing systems*, pages 1–13, 2018.

[13] C. Gibson, V. Frost, K. Platt, W. Garcia, L. Vargas, S. Rampazzi, V. Bindschaedler, P. Traynor, and K. Butler. Analyzing the monetization ecosystem of stalkerware. *Proceedings on Privacy Enhancing Technologies*, 4:105–119, 2022.

[14] R. Gibson. Countering tech abuse together, 2018. `https://www.virusbulletin.com/uploads/pdf/conference_slides/2019/VB2019-ZakorzhevskyG.pdf`.

[15] Google. Developer program policy: September 16, 2020 announcement, 2020. `https://support.google.com/googleplay/android-developer/answer/10065487?hl=en`.

[16] D. Harkin, A. Molnar, and E. Vowles. The commodification of mobile phone surveillance: An analysis of the consumer spyware industry. *Crime, media, culture*, 16(1):33–60, 2020.

[17] S. Havron, D. Freed, R. Chatterjee, D. McCoy, N. Dell, and T. Ristenpart. Clinical computer security for victims of intimate partner violence. In *USENIX Security '19*, pages 105–122, Santa Clara, CA, USA, 2019.

[18] KasperskyLab. Tinycheck, 2021. `https://github.com/KasperskyLab/TinyCheck`.

[19] B. Krebs. For 2nd time in 3 years, mobile spyware maker mSpy leaks millions of sensitive records, 2018. `https://krebsonsecurity.com/2018/09/for-2nd-time-in-3-years-mobile-spyware-maker-mspy-leaks-millions-of-sensitive-records/`.

[20] L. Stefanko. Android stalkerware vulnerabilities, May 2021. `https://www.welivesecurity.com/wp-content/uploads/2021/05/eset_android_stalkerware.pdf`.

[21] A. Langton. Stalking stalkerware: A deep dive into flexispy, 2019. `https://blogs.juniper.net/en-us/threat-research/stalking-stalkerware-a-deep-dive-into-flexispy-2`.

[22] E. Liu, S. Rao, S. Havron, G. Ho, S. Savage, G. M. Voelker, and D. McCoy. No privacy among spies: Assessing the functionality and insecurity of consumer Android spyware apps. *Proceedings on Privacy Enhancing Technologies*, 1:1–18, 2023.

[23] D. N. Moreira and M. P. Da Costa. The impact of the COVID-19 pandemic in the precipitation of intimate partner violence. *International Journal of Law and Psychiatry*, 71, 2020.

[24] B. Palanisamy, S. Sensenig, J. Joshi, and R. Constantino. LEAF: A privacy-conscious social network-based intervention tool for ipv survivors. In *IEEE 15th International Conference on Information Reuse and Integration*, pages 138–146, 2014.

[25] C. Parsons, A. Molnar, J. Dalek, J. Knockel, M. Kenyon, B. Haselton, C. Khoo, and R. Deibert. The predator in your pocket: A multidisciplinary assessment of the stalkerware application industry, 2019. `https://citizenlab.ca/2019/06/the-predator-in-your-pocket-a-multidisciplinary-assessment-of-the-stalkerware-application-industry/`.

[26] M. K. Qabalin, M. Naser, and M. Alkasassbeh. Android spyware detection using machine learning: A novel dataset. *Sensors*, 22(15), 2022.

[27] Rithvik. Cerberus acknowledges data breach, states some usernames and encrypted passwords stolen, 2014. `https://www.droid-life.com/2014/03/26/cerberus-data-breach`.

[28] M. Robinson and C. Taylor. Spy vs spy: Spying on mobile device spyware, 2020. `https://media.defcon.org/DEF%20CON%2020/DEF%20CON%2020%20presentations/DEF%20CON%2020%20-%20Robinson-Spy-vs-Spy.pdf`.

[29] J. M. Schokkenbroek, W. Hardyns, and K. Ponnet. Baby don't hurt me: Victimization and perpetration experiences of offline and online intimate partner violence. In *Annual Meeting of the Belgian Association of Psychological Sciences*, 2021.

[30] S. Sidor. Android: apps can take photos with your phone without you knowing., 2014. `https://rstforums.com/forum/topic/79016-android-apps-can-take-photos-with-your-phone-without-you-knowing/`.

[31] H. L. Storer, E. X. Nyerges, and S. Hamby. Technology "feels less threatening": The processes by which digital technologies facilitate youths' access to services at intimate partner violence organizations. *Children and youth services review*, 139, 2022.

[32] S. Taylor and Y. Xia. Cyber partner abuse: A systematic review. *Violence and victims*, 33(6):983–1011, 2018.

[33] K. Thomas, D. Akhawe, M. Bailey, D. Boneh, E. Bursztein, S. Consolvo, N. Dell, Z. Durumeric, P. G. Kelley, D. Kumar, D. McCoy, S. Meiklejohn, T. Ristenpart, and G. Stringhini. SoK: Hate, harassment, and the changing landscape of online abuse. In *IEEE Symposium on Security and Privacy (SP)*, pages 247–267, 2021.

[34] E. Tseng, R. Bellini, N. McDonald, M. Danos, R. Greenstadt, D. McCoy, N. Dell, and T. Ristenpart. The tools and tactics used in intimate partner surveillance: An analysis of online infidelity forums. In *USENIX Security '20*, pages 1893–1909, online, 2020.

[35] Waqas. Company that sells spyware to domestic abusers hacked, 2018. `https://www.hackread.com/company-that-sells-spyware-to-domestic-abusers-hacked/`.

[36] Z. Whittaker. Xnspy stalkerware spied on thousands of iphones and Android devices, 2022. `https://techcrunch.com/2022/12/12/xnspy-stalkerware-iphone-android/`.

[37] Z. Whittaker. Letmespy, a phone tracking app spying on thousands, says it was hacked, 2023. `https://techcrunch.com/2023/06/27/letmespy-hacked-spyware-thousands/`.

[38] C. Wyburn. The Consumer Spyware Industry: An Australian based analysis of the threats of consumer spyware, 2019. `https://accan.org.au/grants/completed-grants/1435-risks-impacts-and-accountability-in-the-consumer-spyware-industry`.