Using infrastructure-based agents to enhance forensic logging of third-party applications

ICISSP 2023 9th International Conference on Information Systems Security and Privacy

Jennifer Bellizzi, Mark Vella, Christian Colombo and Julio Hernandez-Castro

This work is supported by the DETECTIF (Digital Evidence Targeting covErt Cyberattacks through Timely Information Forensics) project. DETECTIF is funded by the Malta Council for Science and Technology (MCST) under the FUSION R&I: Research Excellence Programme, Grant Agreement No. REP-2022-007.

L-Università ta' Malta

University of Kent

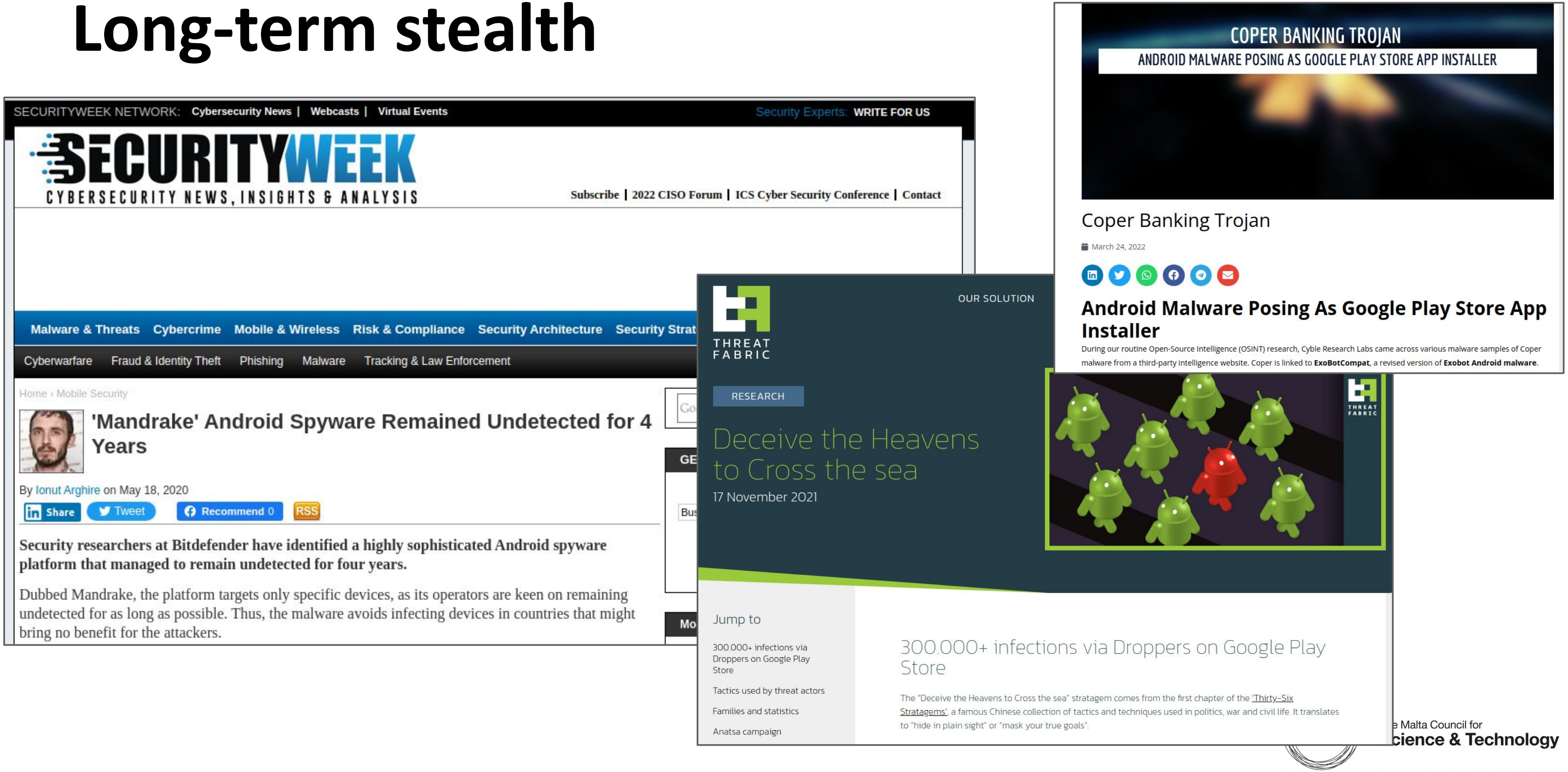
Xjenza The Malta Council for Science & Technology

RESEARCH EXCELLENCE PROGRAMME

Xjenza FUSION The R+I Programme

DETECTIF

Long-term stealth



SECURITYWEEK NETWORK: Cybersecurity News | Webcasts | Virtual Events

Security Experts: WRITE FOR US

SECURITYWEEK
CYBERSECURITY NEWS, INSIGHTS & ANALYSIS

Subscribe | 2022 CISO Forum | ICS Cyber Security Conference | Contact

Malware & Threats Cybercrime Mobile & Wireless Risk & Compliance Security Architecture Security Strat

Cyberwarfare Fraud & Identity Theft Phishing Malware Tracking & Law Enforcement

Home › Mobile Security

'Mandrake' Android Spyware Remained Undetected for 4 Years

By Ionut Arghire on May 18, 2020

Share Tweet Recommend 0 RSS

Security researchers at Bitdefender have identified a highly sophisticated Android spyware platform that managed to remain undetected for four years.

Dubbed Mandrake, the platform targets only specific devices, as its operators are keen on remaining undetected for as long as possible. Thus, the malware avoids infecting devices in countries that might bring no benefit for the attackers.

THREAT FABRIC

OUR SOLUTION

RESEARCH

Deceive the Heavens to Cross the sea

17 November 2021

Jump to

300.000+ infections via Droppers on Google Play Store

Tactics used by threat actors

Families and statistics

Anatsa campaign

300.000+ infections via Droppers on Google Play Store

The "Deceive the Heavens to Cross the sea" stratagem comes from the first chapter of the 'Thirty-Six Stratagems', a famous Chinese collection of tactics and techniques used in politics, war and civil life. It translates to "hide in plain sight" or "mask your true goals".

COPER BANKING TROJAN
ANDROID MALWARE POSING AS GOOGLE PLAY STORE APP INSTALLER

Coper Banking Trojan

March 24, 2022

Android Malware Posing As Google Play Store App Installer

During our routine Open-Source Intelligence (OSINT) research, Cyble Research Labs came across various malware samples of Coper malware from a third-party intelligence website. Coper is linked to **ExoBotCompat**, a revised version of **Exobot Android malware**.

Motivation

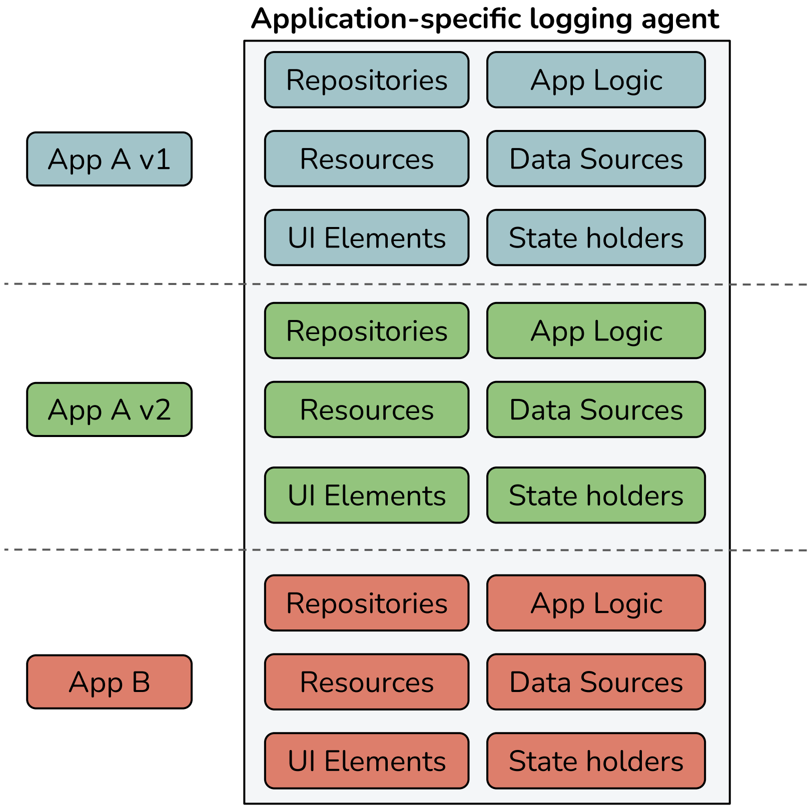
- Logs are the primary data source forensic analysts to:
  - Diagnose faults in distributed systems (VAIF[1])
  - Diagnose attacks in the case of Incident Response[2]
- **BUT** it is difficult to anticipate where logs may be needed, especially in cyber attacks
- Post-deployment application-specific **logging agents** that use instrumentation are needed for endpoint visibility.

[1] Toslali, M., Ates, E., Ellis, A., Zhang, Z., Huye, D., Liu, L., Puterman, S., Coskun, A. K., and Sambasivan, R. R. (2021). Automating instrumentation choices for performance problems in distributed applications with VAIF. In ACM SoCC , pages 61–75

[2] Ma, S., Lee, K. H., Kim, C. H., Rhee, J., Zhang, X., and Xu, D. (2015). Accurate, low cost and instrumentation free security audit logging for windows. In ACSAC, pages 401–410

Xjenza
The Malta Council for
Science & Technology

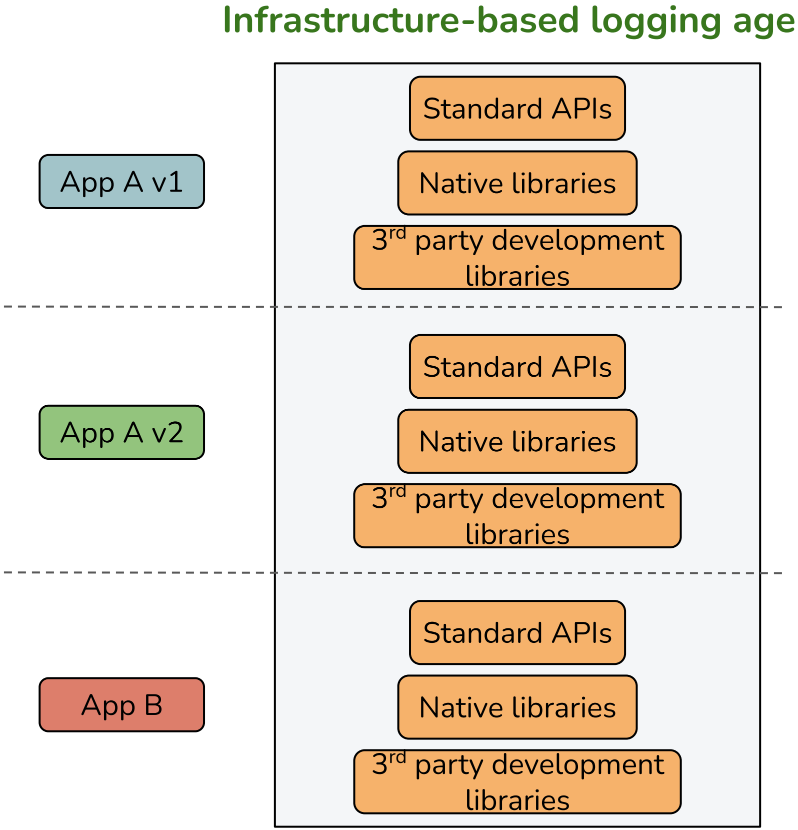
Problem



- Relies on application-specific knowledge and code comprehension effort to determine:
  - Objects of interest
  - Where/when they are used during execution
- Are therefore likely to break compatibility between application versions and across applications, requiring frequent updates

Xjenza
The Malta Council for
Science & Technology

Proposed Solution

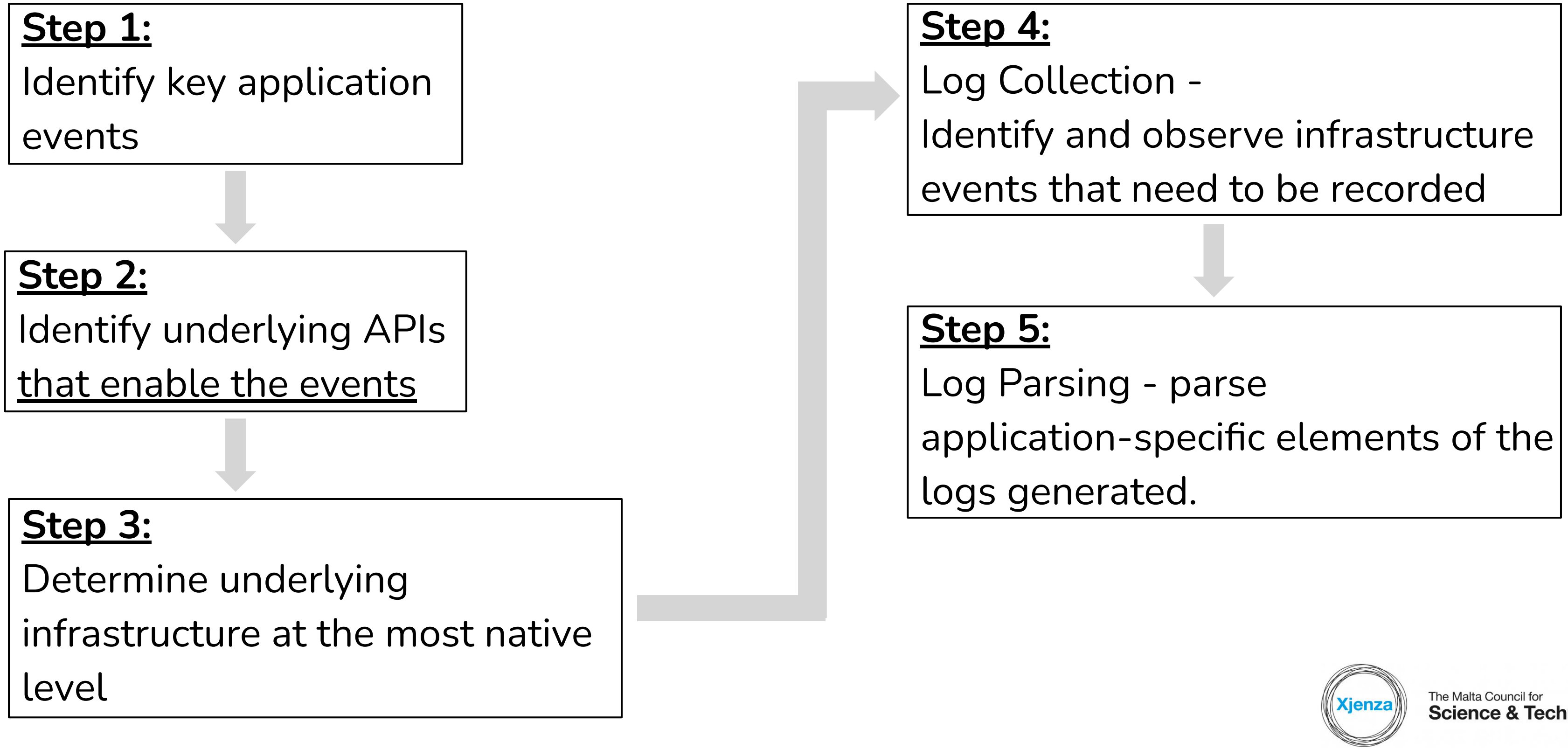

Infrastructure-based logging agent
App A v1
App A v2
App B
Standard APIs
Native libraries
3rd party development libraries
Standard APIs
Native libraries
3rd party development libraries
Standard APIs
Native libraries
3rd party development libraries

Potential benefits:

- More stable than application-specific code
- Backward-compatible
- Publicly-available documentation (reducing app-specific code comprehension efforts)
- Common across applications and versions

Xjenza
The Malta Council for Science & Technology

Methodology



Step 1:
Identify key application events

Step 2:
Identify underlying APIs that enable the events

Step 3:
Determine underlying infrastructure at the most native level

Step 4:
Log Collection -
Identify and observe infrastructure events that need to be recorded

Step 5:
Log Parsing - parse application-specific elements of the logs generated.

Experimentation Context

Experimentation Context

android

JIT-MF

Experimentation Context

Just In Time - Memory Forensics (JIT-MF):

- Timely collection of **critical data objects** in **volatile memory** related to the critical attack steps from victim benign apps
- **Uses JIT-MF drivers**: responsible for establishing the points in time when memory dumps should be triggered and the heap/native memory areas/objects to be included.

Experimentation Context

android

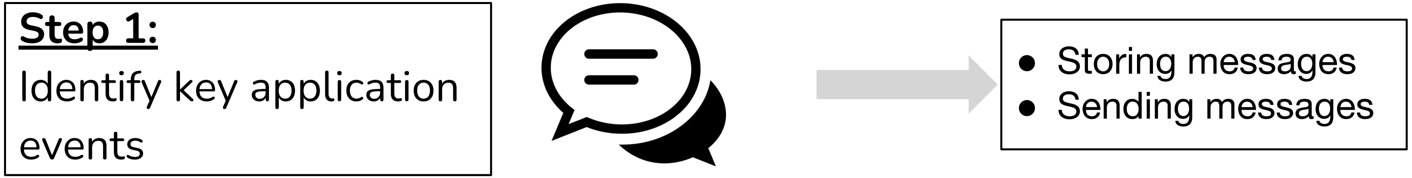
JIT-MF

Experimentation Objectives

<u>RQ1:</u> Is common infrastructure usage prevalent across different versions of a messaging apps ?

<u>RQ2:</u> Can infrastructure-based agents work across different Android messaging apps while maintaining the same accuracy as application-specific agents?

Experiment Setup

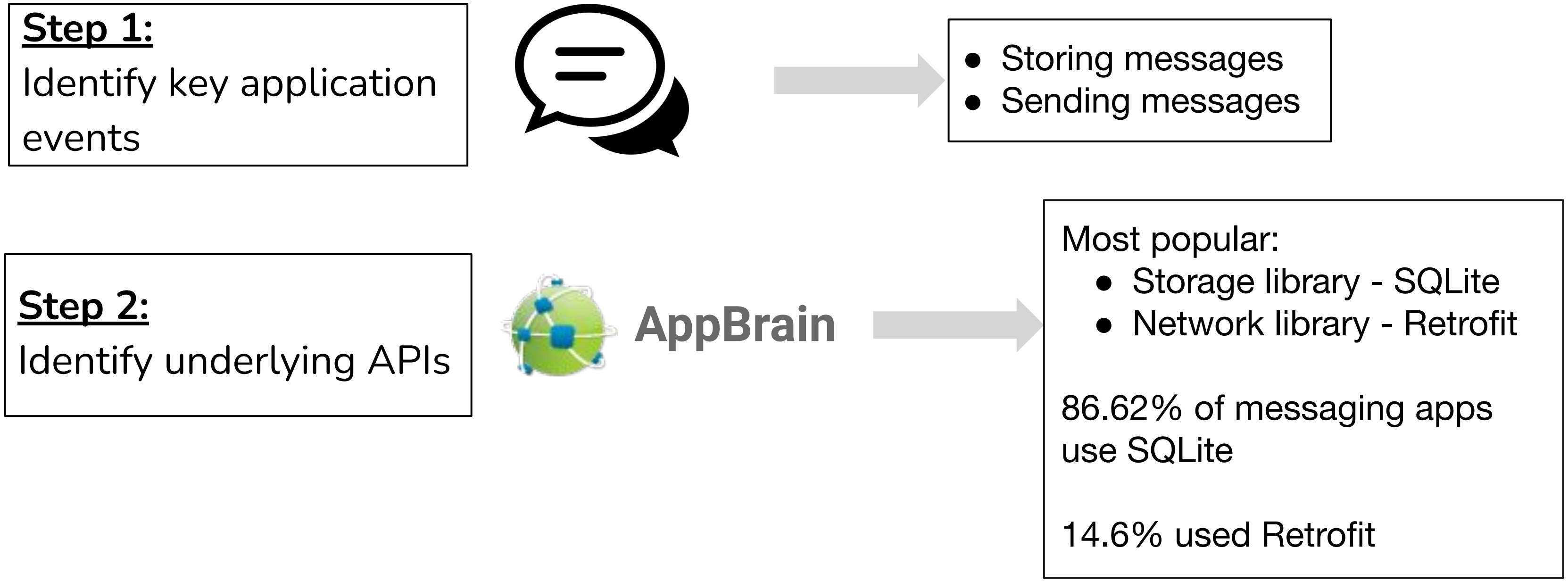


Step 1:

Identify key application events

- Storing messages
- Sending messages

Experiment Setup



Step 1:
Identify key application events

- Storing messages
- Sending messages

Step 2:
Identify underlying APIs

AppBrain

Most popular:

- Storage library - SQLite
- Network library - Retrofit

86.62% of messaging apps use SQLite

14.6% used Retrofit

Experiment Setup

Step 1:
Identify key application events

- Storing messages
- Sending messages

Step 2:
Identify underlying APIs

AppBrain

Most popular:

- Storage library - SQLite
- Network library - Retrofit

86.62% of messaging apps use SQLite

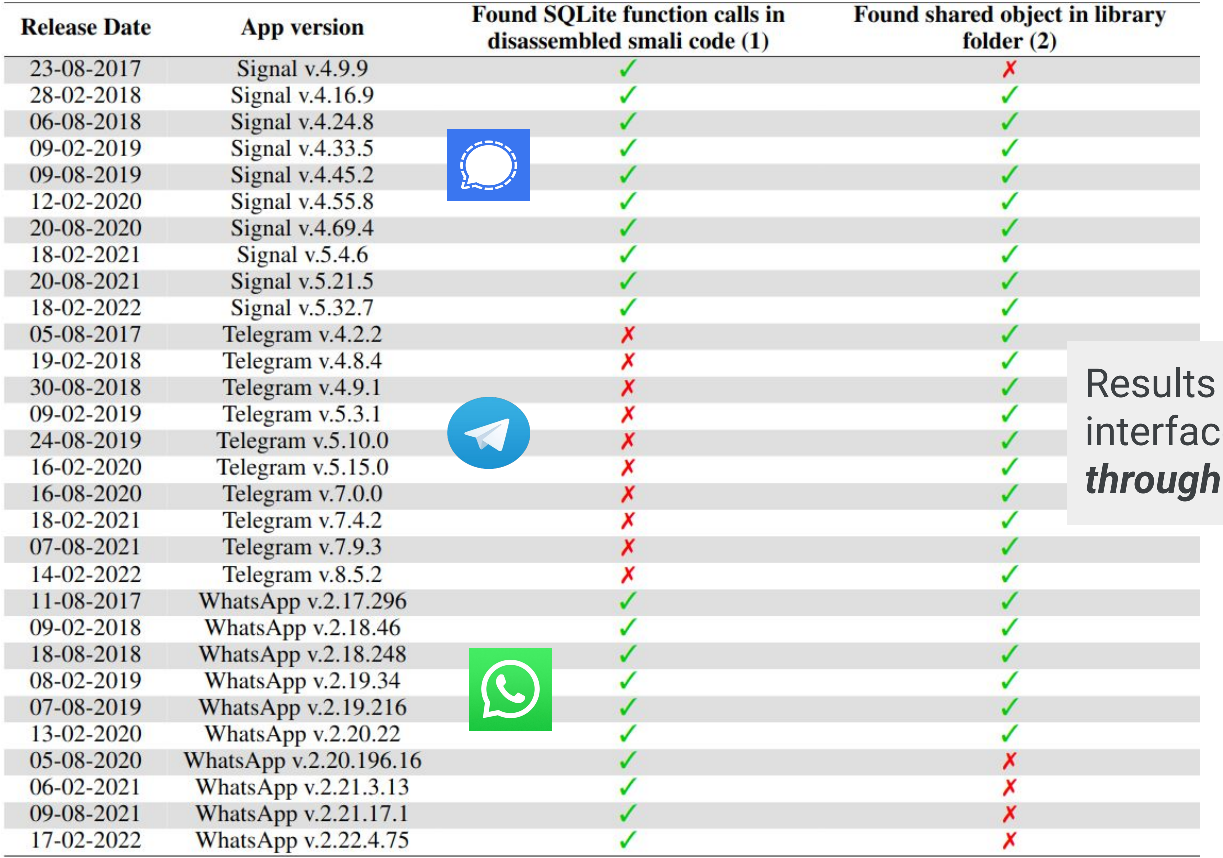
14.6% used Retrofit

Step 3:
Determine underlying infrastructure at the most native level

SQLite

`sqlite.c`

Results: SQLite prevalent across a 5-year span



Release Date	App version	Found SQLite function calls in disassembled smali code (1)	Found shared object in library folder (2)
23-08-2017	Signal v.4.9.9	✓	✗
28-02-2018	Signal v.4.16.9	✓	✓
06-08-2018	Signal v.4.24.8	✓	✓
09-02-2019	Signal v.4.33.5	✓	✓
09-08-2019	Signal v.4.45.2	✓	✓
12-02-2020	Signal v.4.55.8	✓	✓
20-08-2020	Signal v.4.69.4	✓	✓
18-02-2021	Signal v.5.4.6	✓	✓
20-08-2021	Signal v.5.21.5	✓	✓
18-02-2022	Signal v.5.32.7	✓	✓
05-08-2017	Telegram v.4.2.2	✗	✓
19-02-2018	Telegram v.4.8.4	✗	✓
30-08-2018	Telegram v.4.9.1	✗	✓
09-02-2019	Telegram v.5.3.1	✗	✓
24-08-2019	Telegram v.5.10.0	✗	✓
16-02-2020	Telegram v.5.15.0	✗	✓
16-08-2020	Telegram v.7.0.0	✗	✓
18-02-2021	Telegram v.7.4.2	✗	✓
07-08-2021	Telegram v.7.9.3	✗	✓
14-02-2022	Telegram v.8.5.2	✗	✓
11-08-2017	WhatsApp v.2.17.296	✓	✓
09-02-2018	WhatsApp v.2.18.46	✓	✓
18-08-2018	WhatsApp v.2.18.248	✓	✓
08-02-2019	WhatsApp v.2.19.34	✓	✓
07-08-2019	WhatsApp v.2.19.216	✓	✓
13-02-2020	WhatsApp v.2.20.22	✓	✓
05-08-2020	WhatsApp v.2.20.196.16	✓	✗
06-02-2021	WhatsApp v.2.21.3.13	✓	✗
09-08-2021	WhatsApp v.2.21.17.1	✓	✗
17-02-2022	WhatsApp v.2.22.4.75	✓	✗

Static check for presence of SQLite interface usage across versions from last 5 years:

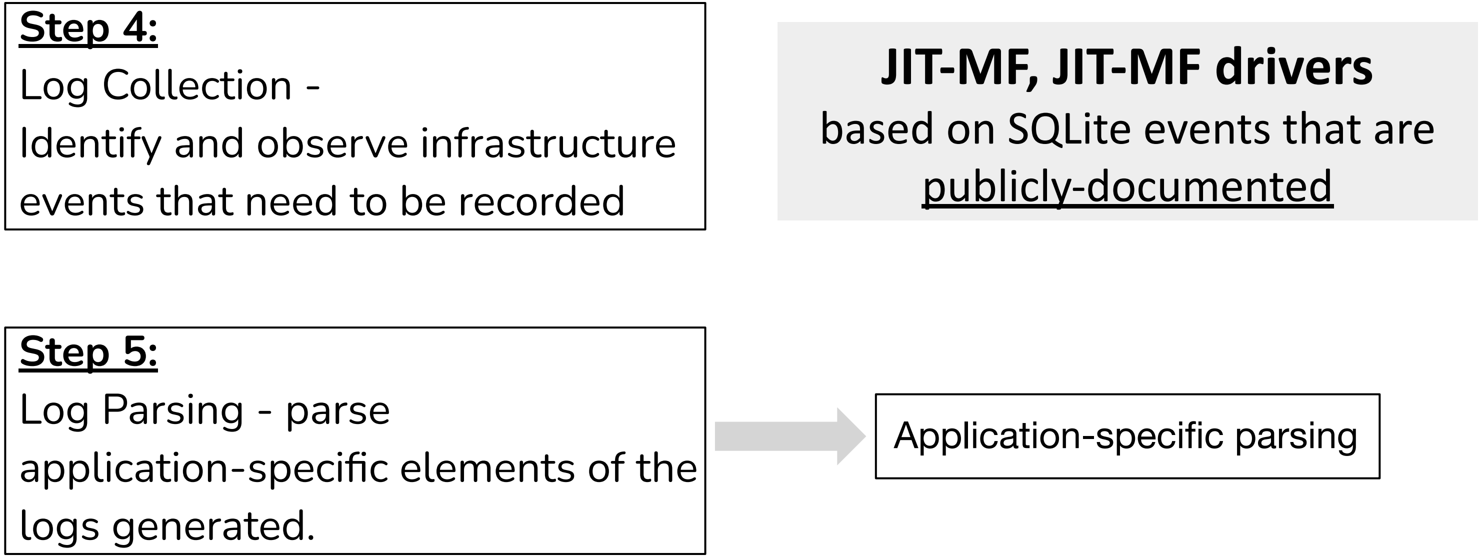
Results show that each version and app interfaced with SQLite in some way ***(either through API or native library or both)***

Results: SQLite prevalent across a 5-year span

- Common across applications and versions
- More stable than application-specific code
- Publicly-available API documentation

Codebase	Average Release time (in days) over the last 5 years
WhatsApp	6.324
Telegram	14.917
Signal	7.319
SQLite	39.48*

Experiment Setup



Step 4:

Log Collection -
Identify and observe infrastructure events that need to be recorded

JIT-MF, JIT-MF drivers
based on SQLite events that are publicly-documented

Step 5:

Log Parsing - parse application-specific elements of the logs generated.

Application-specific parsing

Results: Maintaining accuracy

JIT-MF app-specific driver

```
{"time": "1662485256" , "event": "Telegram Message Sent" , "trigger_point": "recv" ,
"object": {"date": "1662483779" ,"message_id" : "2328" , "text": "Normal_message_1" ,
"to_id": "5181266731" , "to_name": "target_phone ;;;" , "to_phone":"35699626972" ,
"from_id": "1679923803" , "from_name": "contact_phone ;;;" , "from_phone": "35679247196" }}
```

JIT-MF SQLite driver

```
{"time": "1662483789" , "event": "Message Sent" , "trigger_point (s)":
"sqlite3_clear_bindings|sqlite3_prepare_v2|sqlite3_prepare16_v2|sqlite3_bind_int|sqlite3_bind_
int64|sqlite3_bind_text|sqlite3_bind_text16|sqlite3_bind_blob|sqlite3_finalize" , "object": {“
REPLACE INTO messages_v2 VALUES(2328, 1662483779, 2, 0, 1662483779,
n8\"QY[!d\"QY[!dC}cNormal_message_1 , 0, 0, 18446744073709552000, NULL, 0, 0, 0, undefined, 0,
0, 0, undefined)” }}
```

Results: Maintaining accuracy

JIT-MF app-specific driver

```
{"time": "1662485256" , "event": "Telegram Message Sent" , "trigger_point": "recv" ,
"object": {"date": "1662483779" ,"message_id" : "2328" , "text": "Normal_message_1" ,
"to_id": "5181266731" , "to_name": "target_phone ;;;" , "to_phone":"35699626972" ,
"from_id": "1679923803" , "from_name": "contact_phone ;;;" , "from_phone": "35679247196" }}
```

JIT-MF SQLite driver

```
{"time": "1662483789" , "event": "Message Sent" , "trigger_point (s)":
"sqlite3_clear_bindings|sqlite3_prepare_v2|sqlite3_prepare16_v2|sqlite3_bind_int|sqlite3_bind_
int64|sqlite3_bind_text|sqlite3_bind_text16|sqlite3_bind_blob|sqlite3_finalize" , "object": {"
REPLACE INTO messages_v2 VALUES(2328, 1662483779, 2, 0, 1662483779,
n8\"QY[!d\"QY[!dC}cNormal_message_1 , 0, 0, 18446744073709552000, NULL, 0, 0, 0, undefined, 0,
0, 0, undefined)" }}
```

Application-specific parsing

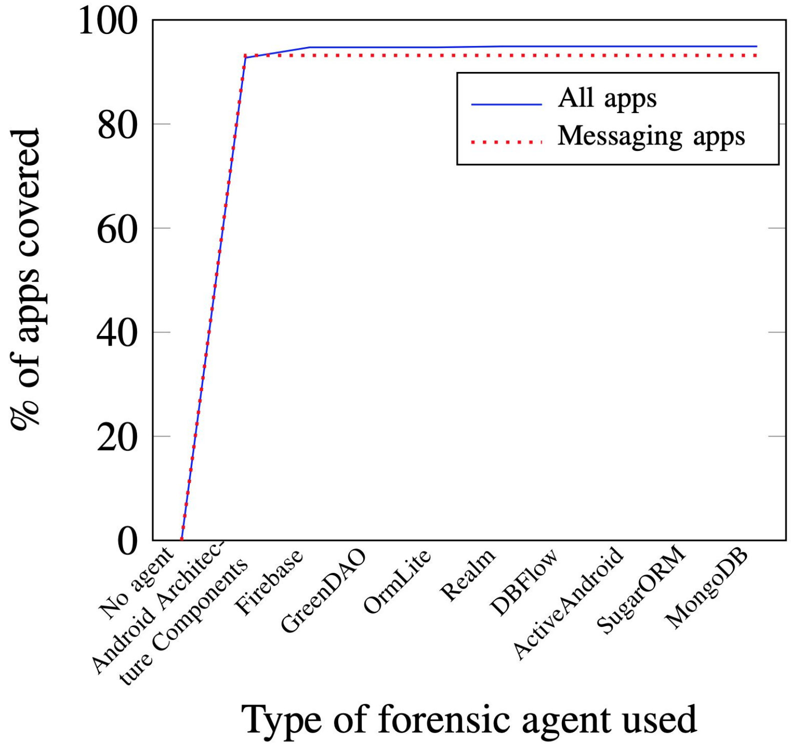
```
{"time": "1662483789" , "event": "Message Sent" , "trigger_point (s)":
"sqlite3_clear_bindings|sqlite3_prepare_v2|sqlite3_prepare16_v2|sqlite3_bind_int|sqlite3_bind_
int64|sqlite3_bind_text|sqlite3_bind_text16|sqlite3_bind_blob|sqlite3_finalize" , "object": {
"message_number" : "2328" , "date": "1662483779" , "text": "Normal_message_1 " , "type"
:"received" , "to_id": "5181266731" , "to_name": "target_phone ;;;" , "to_phone":
"35699626972" , "from_id": "1679923803" ,"from_name": "contact_phone ;;;" , "from_phone":
"35679247196" }}
```

Xjenza The Malta Council for Science & Technology

Results: Reducing code comprehension efforts

Application	Maximum LoC within scope for app-specific JIT-MF driver	Maximum LoC within scope for SQLite JIT-MF driver
WhatsApp	1,515,334	395,076
Telegram	1,025,467	-
Signal	1,552,171	-

Results: Coverage Analysis for storage- based JIT-MF drivers



Xjenza

The Malta Council for Science & Technology

Future Work

- Further applicability of JIT-MF:
  - As used in the context of Endpoint Detection and Response Systems (e.g. GRR, Velociraptor) for mobile devices.
- Towards a less intrusive approach to post-deployment log enhancement of mobile application logging.

Questions