

Received 3 September 2022, accepted 22 October 2022, date of publication 28 October 2022, date of current version 7 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3217795

RESEARCH ARTICLE

Smishing Strategy Dynamics and Evolving Botnet Activities in Japan

RYU SAEKI¹, LEO KITAYAMA¹, JUN KOGA², MAKOTO SHIMIZU³,
AND KAZUMASA OIDA¹, (Member, IEEE)

¹Department of Computer Science and Engineering, Fukuoka Institute of Technology, Fukuoka 811-0295, Japan

²Cyber Crime Control Division, Community Safety Department, Fukuoka Prefectural Police Headquarters, Fukuoka 812-8576, Japan

³Juvenile Section, Kokurakita Police Station, Fukuoka Prefectural Police, Fukuoka 803-8576, Japan

Corresponding author: Kazumasa Oida (oida@fit.ac.jp)

This work was supported in part by the Fukuoka Institute of Technology; and in part by the Fukuoka Prefectural Police, Japan.

ABSTRACT XLoader and FakeSpy, the two major smishing botnets targeting Japan, change their attack strategies over various timescales. Based on recent observations of the botnets and Twitter data, we present empirical facts about their strategies and activity patterns and applied some of these strategic and activity patterns to malware detection and malicious domain detection. All the proposed methods yielded small false positive and negative rates, and are expected to run on user devices owing to their small computational cost. Recent malware detection methods based on traffic analysis extract TCP/IP traffic features if the upper layers of TCP are encrypted. In this study, Frida's hooking capability was employed to decode the upper layers (WebSocket and JSON-RPC) to create a list of all commands flowing over the botnet channel. The command-level traffic analysis presents decisive attack features because commands are transmitted according to strategies developed by the attackers. The proposed malicious domain detection method, on the other hand, exploited the tendency of the attackers to create domains in batches. Previous researchers focused on how benign and malicious domains were registered and used on the name servers. The proposed method, on the other hand, focuses on the arrival rate of SMS messages with URL links. The error rates become significantly small when users do not receive such messages very often.


INDEX TERMS Botnet, domain name registry, FakeSpy, hooking, JSON-RPC, malware, smishing, Twitter, XLoader.

I. INTRODUCTION

Smishing is spreading worldwide. It is an attack whereby the short message service (SMS) of smartphones and other mobile devices are used to direct users to malicious sites to steal personal information and install malware. In Japan, smishing was first reported in 2015 [1], and its damage has increased considerably in recent years. In particular, the Android operating system, with its high market share and lax security, is a target for attackers. As a countermeasure, major Japanese mobile carriers have announced that they would begin detecting smishing in 2022. This paper provides the results of a joint research project between the university

and police that began in 2020, with the goal of investigating smishing in detail in Japan.

There are several high-performance solutions for detecting malicious SMS messages [2], illegitimate uniform resource locators (URLs) [3], and mobile botnets [4], all of which achieve accuracies of more than 95%. However, smishing remains a problem in many countries [5], [6], [7], [8], [9], [10], [11]. Some researchers have pointed out the importance of security education, attributing the current situation to low security awareness [12], [13]. Furthermore, researchers indicate that a massive number of malicious apps and their evolving evasion techniques yield a major gap between malware detection research and practice [14], [15]. To keep up with the rapid evolution of mobile malware, we should use up-to-date malware datasets [14] and consider how to reconfigure

The associate editor coordinating the review of this manuscript and approving it for publication was Zhaojun Steven Li .

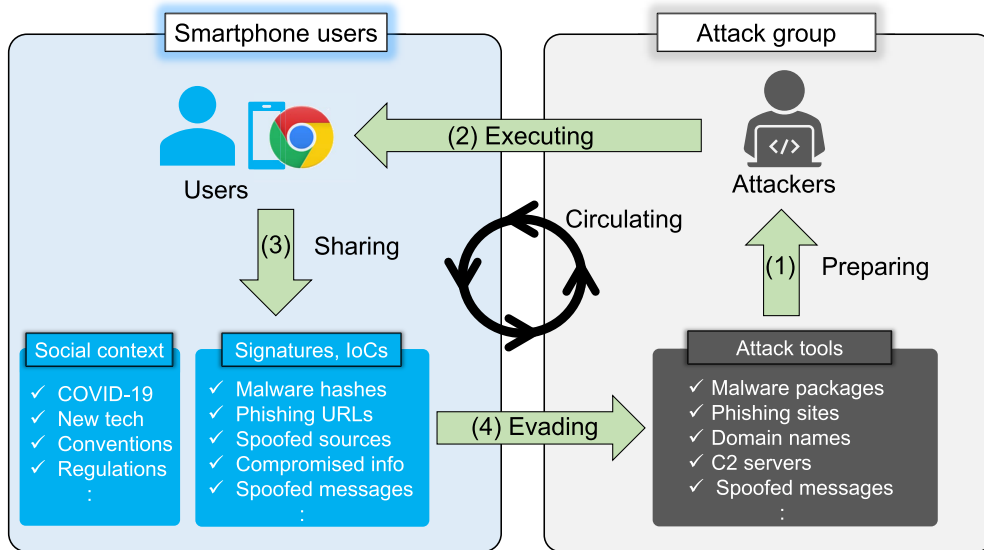


FIGURE 1. Circular dynamical system with processes (1)–(4) as its elements. The Chrome icon represents mobile malware.

machine learning (ML) classifiers [16], [17] and update indicators of compromises (IoCs).

Instead of dealing with all the active mobile botnets in the world, this study tackled major botnets in Japan, XLoader (also known as MoqHao and Wroba) and FakeSpy, whereby their activity patterns and their evasion techniques were sufficiently considered to devise effective countermeasures. According to [14], FakeSpy and XLoader are ranked 11th and 27th among the world’s top-50 Android malware families in terms of the number of apps they use. They have evolved by influencing each other [18] and have expanded their activities to North America and Europe [5], [6], [7].

Fig. 1 illustrates a circular dynamical system consisting of the following four processes (1)–(4) activated by smishing: (1) preparing attack tools, (2) executing attacks using the tools, (3) sharing the signatures and IoCs of the tools, and (4) evading attacks using the shared data relating to the tools. If the attack tools are properly updated, the dynamical system keeps working; otherwise, the system terminates. Smishing is a social engineering attack; thus, the system behavior is influenced by the social context, for example, the Coronavirus Disease-19 (COVID-19) pandemic [19], [20]. We examined how attackers maintain and operate the tools in Fig. 1. Each tool has an inherent lifespan. The command and control (C2) servers typically operate for months or years. Malware packages, on the other hand, are replaced more frequently.

Fig. 2 shows the framework of our work. First, we monitored the execution and preparation processes shown in Fig. 1; then, we collected smishing-related data shared among Twitter users in the past. Subsequently, we proposed the malware and malicious domain detection methods based on attack features extracted from the execution and preparation processes, respectively. Our major contribution to the field of smishing and mobile botnet research is the following empirical evidence.

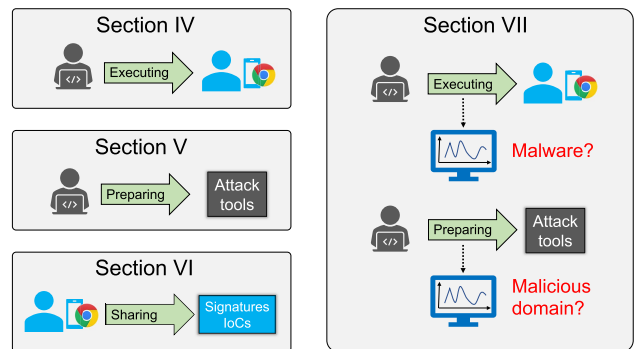


FIGURE 2. The processes in Fig. 1 are detailed in Sections IV, V, and VI. Section VII presents the malware and malicious domain detection methods based on attack features extracted from the execution and preparation processes, respectively.

- *Command-level traffic analysis contributes to malware detection.* In general, C2 servers send commands (C2 commands) to infected devices (bots) to control them. XLoader and FakeSpy enable bots to send commands (bot commands) to C2 servers. Although bot commands are encrypted, traffic features of the commands are closely tied to attack strategies. Some commands notify the C2 server of device state changes in real time, whereby the server can give orders only to bots being in sleep/silent mode, while others periodically report the current device state, such as the need for package updates.
- *Attackers’ everyday routine tasks provide many decisive features.* Our observations showed that 180 commands were delivered to each bot at the same time every day. Phishing domains were mostly created in the morning and were stored in the smishing repositories after a fixed amount of time from their creation. Our malicious

domain detection method exploited the tendency of the attackers to register many domains simultaneously. Their routine tasks can also be exploited in criminal investigations. The findings of our analysis revealed that “the XLoader group operates in the UTC+8 time zone,” which corroborated Trend Micro [18].

The remainder of this paper is organized as follows. Section II describes related studies. Section III outlines the smishing and the XLoader and FakeSpy botnets. Section IV introduces the evolving malware structure and details attack strategies and activity patterns of the botnets. Section V derives various attack features from the access history of smishing repositories. Section VI analyzes Twitter data to visualize the trend of smishing attacks in the past. In Section VII, we describe the application of the observed activity patterns of FakeSpy to malware and malicious domain detection. In Section VIII, the implications of the findings are discussed. Finally, the conclusions and future directions of this study are presented in Section IX.

II. RELATED WORK

Research on smishing can be divided into two research categories: smishing fraud and mobile malware. The former deals with information leakage through smishing, whereas the latter focuses on user device exploitation for smishing message diffusion. We first describe the relationship between our study on smishing fraud and previous studies.

- 1) *The impact of smishing on society.* Damage caused by smishing in North America [5], Europe [6], [7], Australia [8], Arabic-speaking countries [9], and African countries [10], [11] has been reported. The effects of social engineering during the COVID-19 period were discussed in [19] and [20]. This study covers smishing strategy dynamics in Japan from the perspective of various timescales (hours, days, months, and years), and demonstrates that circa 2018 was a turning point in smishing strategies in Japan.
- 2) *Smishing detection.* Smishing detection identifies malicious SMS messages by consulting blacklist/whitelist data [21] or by analyzing message content [3], [22], web page appearance [23], and downloaded Android application packages (APKs) [24]. The authors in [2] compared the performance of 18 existing detection algorithms using nine different datasets and demonstrated that most of the algorithms achieved accuracies of more than 95%. According to our data analysis, “message arrival times,” which was neglected in previous studies, should be considered because bots sent smishing messages at the same time every day. In addition, “sender names” should also be considered because couriers and mobile carriers have been the most used smishing senders for three consecutive years since 2019.
- 3) *Malicious domain detection.* Malicious domains have been detected by various approaches such as blacklisting/whitelisting [25], lexical characteristics [26],

traffic analysis [27], web page inspection [28], web reputation [29], and registration records [30], [31]. The proposed method is similar to [30] and [31] in that domain registration records are used for detection. However, although [30] and [31] infer many malicious domains among newly registered ones, our method infers malicious domains included in SMS messages based on the SMS message arrival rate. The proposed method can improve the domain scan results of VirusTotal [32].

Next, we describe the relationship between our study on mobile malware and previous studies.

- 1) *Malware analysis.* For efficient and accurate malware detection, the authors in [33] investigated code reuse in legitimate and malicious mobile apps, and code similarity measurement techniques were improved in [34]. There are many evasion techniques against static and dynamic malware analyses such as code obfuscation and reflection. A hooking method that bypasses the evasion techniques was proposed in [35]. We detail a case where encrypted communications can be decoded using the hooking functionality of Frida [36].
- 2) *Malware classification.* A meta-learning algorithm [37] and two-dimensional image processing techniques [38] have been proposed for obfuscated malware classification. One of the current issues is timely reconstruction of up-to-date malware datasets [14], ML models [16], and classifiers [17] to keep up with evolving malware such as XLoader. We present an idea that facilitates quick dissemination of new malware packages. Attackers’ routine work pauses for several days before a new package is released; thus, we propose that the pause be used as a signal to update the classifier.
- 3) *Malware/botnet detection.* In a survey paper [39], mobile malware detection methods were classified into local analysis (static, dynamic, and hybrid) and network traffic analysis. The static method achieved 98.9% accuracy using API calls, permissions, and intent settings [4]. Dynamic methods extract behavioral features by executing infected binaries on a sandbox [15], [40]. Network traffic analysis approaches extract malicious signature patterns [41], [42] or HTTP, TCP, or UDP flow features [43] using datasets such as malware-traffic-analysis.net [44] and CICAndMal2020 [45]. The proposed malware detection methods, on the other hand, detect bot/C2 command flows. Bot commands have not been reported, except on McAfee’s blog [46]. Our methods are different from previous ones in the following ways:
 - a) They utilize knowledge on command-level traffic features.
 - b) They not only passively monitor traffic but also actively generate malware traffic for detection.
- 4) *Mobile botnet features.* The author in [47] summarizes the C2 channel characteristics, bots’ abilities, and the main attack motives of a large number of mobile

botnets. We present the capabilities of XLoader and FakeSpy botnets in more detail than they were presented in [47]. For example, we describe their evolutionary history and present new bot commands.

III. BACKGROUND

A. SMISHING

Smishing, a compound word of SMS and phishing, is an attack in which SMS messages are sent to mobile device users to direct them to a website asking for personal information. This attack exploits human instincts such as fear and stress [48], [49]. For example, an attacker impersonates a trusted sender, such as a bank, to send important messages such as a notice that personal information has been stolen. Smishing has affected the electric vehicle transition [50] and sustainable power grid program [51].

Some of the reasons for the recent worldwide prevalence of smishing are as follows: (1) The number of mobile device users has increased. (2) Mobile devices contain sensitive information (e.g., credit card information). (3) Users immediately check incoming SMS messages [52]. (4) Mobile devices are not as secure as ordinary computers [53]. (5) The bring your own device business model, which allows users to run company apps on their personal devices, has become popular [53]. (6) Infected devices contribute to the spread of the infection.

Compared with email phishing, smishing is harder to detect in the following respects. First, email is filtered by Internet service providers, firewalls, and servers; SMS messages, on the other hand, arrive from various carriers and do not have a well-developed filtering environment [53]. Second, mobile devices have fewer resources and cannot run heavy algorithms. Additionally, because of their small displays, URLs are shortened, and messages are shorter. Finally, there are few smishing datasets available [54].

B. XLOADER AND FAKESPY

An overview of XLoader and FakeSpy based on Trend Micro's commentary [18] is provided in this subsection. As at Oct. 2018, both had infected a total of 385,000 devices [18]. They share similarities in their code, domain names, and the method used to obtain the IP addresses of the C2 servers [55], [56].

Let us begin with an overview of XLoader. XLoader malware evolves with each new package released. For example, the first version of the malicious code (payload) was included as a Dalvik executable (DEX) file. In Version 2.0, it was encoded in Base64, and in Version 3.0, both Base64 and ZIP compression were applied. The XLoader malware first spread to Japan when it was Version 3.0 (Aug. 2017 to Apr. 2018), disguised as a legitimate Facebook or Chrome application. In Version 4.0, smishing messages were written in twenty-seven languages, and the target device types were expanded from Android to iOS and PC. Since Version 5.0 (Jun. 2018), XLoader has adopted a strategy of impersonating

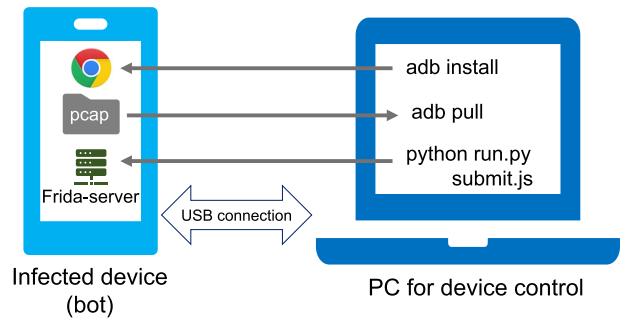


FIGURE 3. The botnet monitoring system. The infected device was controlled using adb commands and Python scripts.

major Japanese home delivery companies. It has spread to North America and Europe [5], [6], [7].

FakeSpy steals information from mobile users using evolving malware in a similar manner. With Version 2.0 (Dec. 2017), FakeSpy switched its target from South Korea to Japan, posing as major Japanese delivery, mobile carrier, credit card, fashion/clothing, and postal companies. The Version 4.0 malware gained the ability to send smishing messages to other mobile users. From Version 5.0 (Jul. 2018) upward, C2 servers send destination phone numbers and smishing messages to the bots. The phone numbers are either generated by the attackers or obtained from the contact lists of the infected devices.

IV. BOTNET ACTIVITY

A. EVOLVING MALWARE

Fig. 3 shows the botnet monitoring system. All the results in this section were obtained from the activity of malware running on the rooted Android 10 device, UMIDIGI BISON. During the monitoring period, the following apps were used: YouTube, web search app, games, banking, and Gmail. The device had a contact list, and the “background data usage” setting was unlimited. A PC was connected to the device via a USB cable to install packages, retrieve log files, execute shell commands, and send JavaScript code `submit.js` to the Frida-server, which can modify malware code in memory.

Two packages were considered in this study: XLoader's *TrojanSpy:AndroidOS/Wroba.D!MTB* (Chrome.apk) and FakeSpy's *TrojanDownloader.AndroidOS/Agent.C* (Yamato.apk). Both have very similar functionalities. Fig. 4(left) shows the directories and files contained in Chrome.apk. When Chrome.apk was installed, an icon almost identical to Google Chrome appeared on the display. When the icon was clicked, `classes.dex` in Fig. 4(left) decrypted payload `1cnqacu` into a DEX file and saved it as a file, thus eliminating the need to create a program to unpack the payload.

The launched malware (Chrome for short) retrieved the address of the C2 server stored in the account profile of Blogger, a free blogging service. Communication between Chrome and the C2 server was performed using JSON-RPC [57] over WebSocket [58], which enabled bidirectional command-and-response communication. Fig. 4(right) shows



FIGURE 4. (Left) Directories, files, and the malware icon after decompressing Chrome.apk. (Right) Twenty-two C2 commands in the payload defined as Java methods.

	Aug. 2017 [46]	May 2018 [59]	Apr. 2019 [60]	Sep. 2020 [61]	Chrome	Yamato
sendSms	○	○	○	○	○	○
setWifi	○	○	○	○	○	○
gcont	○	○	○	○	○	○
lock	○	○	○	○	○	○
bc	○	○	○	○	○	○
setForward	○	○	○	○	○	○
getForward	○	○	○	○	○	○
hasPkg	○	○	○	○	○	○
setRingerMode	○	○	○	○	○	○
setRecEnable	○	○	○	○	○	○
reqState	○	○	○	○	○	○
showHome	○	○	○	○	○	○
getnpki	○	○	○	○	○	○
http	○	○	○	○	○	○
onRecordAction	○	○	○	○	○	○
call	○	○	○	○	○	○
get_apps	○	○	○	○	○	○
show_fs_float_window	○	○	○	○	○	○
ping	○	○	○	○	○	○
getPhoneState	○	○	○	○	○	○
get_gallery	○	○	○	○	○	○
get_photo	○	○	○	○	○	○

FIGURE 5. Evolving XLoader malware. Dates indicate when new C2 commands were first reported. Commands that were actually used by Chrome and Yamato are marked with red circles.

the twenty-two C2 commands implemented in the payload, and Fig. 5 indicates when security vendors first reported them [46], [59], [60], [61]. As shown in Fig. 5, the XLoader malware is evolutionary, in that the number of commands has increased over the years. Specifically, the number increased by more than 70% from Aug. 2017 to May 2018, which is when XLoader began to spread to Japan. Fig. 5 also shows the C2 commands implemented in Chrome and Yamato, where Yamato denotes malware in Yamato.apk, and Chrome.apk and Yamato.apk were obtained in Dec. 2020 and Oct. 2021, respectively. The figure shows that Chrome and Yamato had almost the same kinds of commands. The red circles in Fig. 5 denote the commands they used. It can be observed that they used almost the same kinds of commands and that they used fewer than one-third of all the commands.

TABLE 1. Seven C2 commands used by Chrome and their functions.

Command name	Functionality
sendSms	Send an SMS message
gcont	Send contacts to C2 server
bc	Send SMS messages to contacts
reqState	Send Wi-Fi state to C2 server
showHome	Display the home screen
get_apps	Send app info to C2 server
getPhoneState	Send device id to C2 server

Table 1 lists the commands marked with red circles in Fig. 5 and their functions described in [46], [60], and [62]. Fig. 19 in Appendix A shows how these commands were typically used to diffuse smishing messages. Command sendSms ordered a bot to send a smishing message, where the destination phone number, message content, and phishing URL were sent as parameters of the command. Before sending sendSms commands, the C2 server always sent showHome to display the home screen and getPhoneState to confirm the device identification information. Occasionally, gcont and get_apps were inserted between showHome and getPhoneState to retrieve contact and app lists.

B. MASSIVE SENDSMS ARRIVAL

In the XLoader and FakeSpy botnets, many sendSms commands were delivered from a C2 server to a bot once a day, except on Sundays. Fig. 6 illustrates the number of sendSms commands received by Yamato and its successor malware, JapanPost, per day, from Nov. 2 to Dec. 18, 2021. This massive command influx took place at almost the same time every day; specifically, approximately 180 commands arrived in one hour between approximately 14:00 and 15:00. Yamato and JapanPost cannot send messages; hence, a messaging app, such as Zello or Google Messages, was automatically launched after receiving each sendSms command to send the message on their behalf. Therefore, it is possible to predict when such messages are delivered to other users with a high probability. There were no duplicates in the phone numbers included in the 6,402 sendSms commands that arrived during the period from the first day to the last. These findings suggest that attackers conduct well-planned and routinized tasks.

When the sender’s name was switched from the Yamato Transport Company to the Japan Post on Nov. 16, the message content, phishing site, and malware icon were simultaneously replaced to impersonate the Japan Post. Analysis revealed that JapanPost.apk (the package of JapanPost malware) and Yamato.apk were the same, except for their icons and packing methods. Two months after the switchover date (on Jan. 26, 2022), a new JapanPost.apk was released from the fake Japan Post website. This differs from the previous version of JapanPost.apk only in terms of the packing method. After Feb. 13, Yamato stopped receiving sendSms commands, even though the latest version of JapanPost continued to

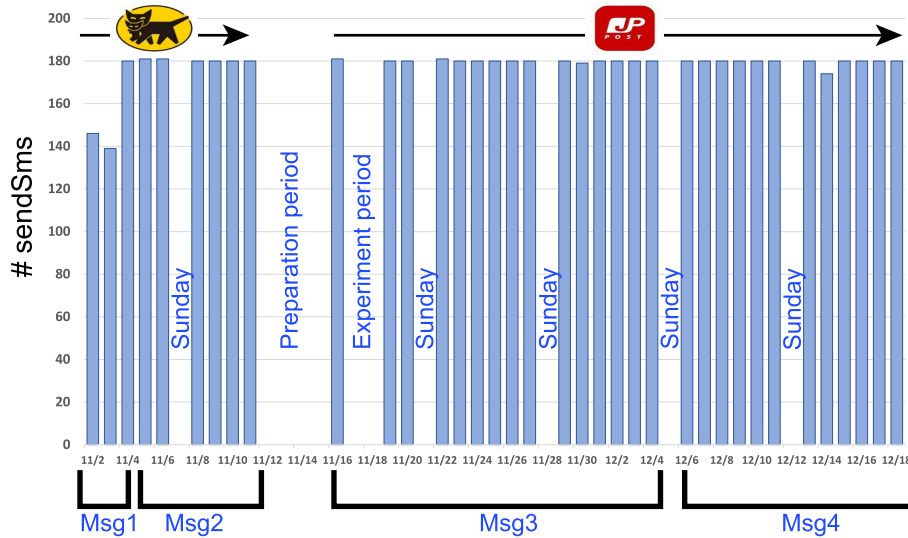


FIGURE 6. The number of `sendSms` commands Yamato and JapanPost received per day and four different smishing messages (msg1–4) observed during the period from Nov. 2 to Dec. 18. The four-day period from Nov. 12 to 15 is a preparation period for switching the sender from the Yamato Transport Company to the Japan Post.

receive them. Because Yamato and JapanPost used the same C2 server, this fact implies that the attackers intentionally discarded the older malware.

Fig. 6 shows that four different messages were used during the observation period. When the sender’s name was changed, the message content also changed; however, the content occasionally changed, even if the name was the same. Accordingly, the lifespan of a sender’s name T_{src} , that of a malware package T_{pac} , and that of a smishing message T_{msg} satisfy the following relationship:

$$T_{src} \geq \max(T_{pac}, T_{msg}). \tag{1}$$

The `sendSms` commands arrived only when the device was in silent mode. This result was verified during the experiment period (Nov. 17-18) in Fig. 6. This implies that the attackers are sensitive to the device mode to prevent users from receiving the notification of transmission results.

C. BIDIRECTIONAL JSON-RPC

XLoader and FakeSpy bots can send commands (bot commands) to the C2 server. However, bot commands cannot be observed because data transmitted by bots are encrypted. This section decodes encrypted bot commands using Frida. JavaScript code `submit.js` in Fig. 3 modifies the encryption function such that the function sends JSON-RPC commands to the Frida-client on the PC before encrypting them (see the JavaScript code in Appendix B).

Fig. 7 shows JSON-RPC commands and responses decoded by Frida. As shown in the figure, the `openbrowser2` command (id 834) was sent with the parameter value `ja_JP`, and the C2 server returned `false` (id 834). The `state` commands were sent to notify the current state of the device, along with the Wi-Fi MAC address. The `state` command can convey

TABLE 2. Bot command functionality, commands described in [46], and commands used by Yamato.

Command name	Functionality	[46]	Yamato
<code>getnotify</code>	Get notification contents	o	o
<code>getPkgMap</code>	Get an APK list for update	o	o
<code>getPkgData</code>	Download an APK requested	o	
<code>onSms</code>	Send SMS messages received	o	
<code>setMyinfo</code>	Send username and birth date	o	
<code>hello</code>	Send device type and info	o	o
<code>state</code>	Send device state and mode	o	o
<code>openbrowser2</code>	Ask whether to open a browser		o
<code>getSmsKW</code>	Under investigation		o
<code>is_call_rec_enable</code>	Ask whether to record calls		o

device states to the C2 server in real time; we verified that the command is transmitted as soon as the power button is pressed.

Table 2 lists the currently known bot commands and those used by Yamato. Although security vendors, such as McAfee, Trend Micro, and Kaspersky, have provided detailed information on smishing malware, there are no research papers or documents that mention bot commands, except for [46]. The last three commands in the table are those reported for the first time. The following can be inferred from the table. (1) When `getPkgMap` and `getPkgData` are used to update (il)legitimate apps, malware can exploit the latest features of the apps. (2) Device information (`hello`, `state`) and user information (`onSms`, `setMyinfo`, `is_call_rec_enable`) can be used to successfully evade botnet detection. (3) The `openbrowser2` command can be used to ask users to input information at the right time. We observed that a browser automatically opened an input form of personal information (name, date of birth, and card number) immediately after pressing the power button.

```

*****time:2022/1/20 13:0:46*****Outbound*****
{"jsonrpc":"2.0","method":"openbrowser2","id":834,"params":["ja_JP"]}
*****time:2022/1/20 13:0:47*****Inbound*****
{"jsonrpc":"2.0","id":834,"result":false}
*****time:2022/1/20 13:1:26*****Outbound*****
{"jsonrpc":"2.0","method":"state","id":835,"params":["","WiFi","02:00:00:00:00:00,1b3a8c0",100,9,false,true,false,0,false]}
*****time:2022/1/20 13:1:27*****Inbound*****
{"jsonrpc":"2.0","id":835,"result":{}}
*****time:2022/1/20 13:1:43*****Outbound*****
{"jsonrpc":"2.0","method":"state","id":836,"params":["","WiFi","02:00:00:00:00:00,1b3a8c0",100,9,false,true,false,0,false]}
*****time:2022/1/20 13:1:44*****Inbound*****
{"jsonrpc":"2.0","id":836,"result":{}}
*****time:2022/1/20 13:2:22*****Outbound*****
{"jsonrpc":"2.0","method":"state","id":837,"params":["","WiFi","02:00:00:00:00:00,1b3a8c0",100,9,false,true,false,0,false]}
*****time:2022/1/20 13:2:22*****Outbound*****
{"jsonrpc":"2.0","method":"getnotify","id":838}
    
```

FIGURE 7. A JSON-RPC command and response sequence decoded by Frida. Inbound: from C2 server to bot. Outbound: from bot to C2 server. The state commands were sent for informing the device status.

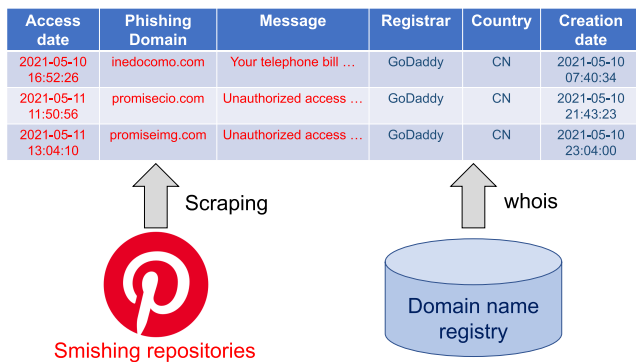


FIGURE 8. The monitoring system collects messages and domains in the smishing repositories, and accesses the domain name registry to retrieve registrars, registrant countries, and creation dates of the collected domains.

V. SMISHING REPOSITORY

XLoader stores smishing messages in social networking service (SNS) account profiles (smishing repositories). Fig. 8 illustrates the system that collected smishing messages in the repositories (three Pinterest accounts) by web scraping at five-minute intervals during the 96-day period from Mar. 4 to Jun. 7, 2021. The system also collected domain information in the domain name registry, such as registrar name and creation date, using the Linux command, whois, or using two websites, DomainBigData [63] and RiskIQ [64].

A. UPDATE HISTORY

During the monitoring period, the message contents were updated 216 times; therefore, the average update frequency was 2.25 times/day. If the URL changes in the messages are ignored, the message update occurrence during the period becomes eleven times; thus, the average lifespan of message content other than URLs was approximately nine days. Fig. 9 shows the histogram of the update intervals of the message contents. Because 95% of the updates were URLs in the messages, the figure indicates that URLs were mostly updated every few hours to every other day. It should be mentioned that updates were actively made in three ranges, 1-9, 17-32, and 43-48 hours, corresponding to the day of domain

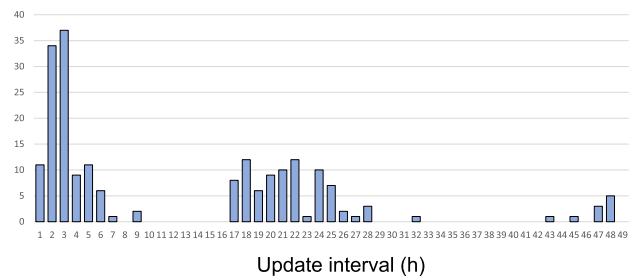


FIGURE 9. Histogram of message content update intervals in three repositories. More than 49 hours (less than 5%) are omitted. Ninety-five percent of the updates were URL updates in the messages.

TABLE 3. Percentages of days of the week of how often the repository data were updated.

Sun.	Mon.	Tue.	Wed.	Thu.	Fri.	Sat.
3%	19%	20%	21%	17%	21%	15%

creation, the day after, and two days later, respectively. This is evidence that human activity, which follows the circadian rhythms, influenced the update process. Table 3 presents another evidence that implies the effect of human activity. The table shows that little work was done on Sunday, which agrees with the results in Fig. 6.

B. PHISHING DOMAINS

According to our analysis, 94% of the top-level domains (TLDs) of URLs in the smishing repositories were .com, and, as shown in Table 4, 83% of the registrars of the domains in the repositories were GoDaddy. According to the Internet Corporation for Assigned Names and Numbers, the registrar that registered the most domains containing TLD .com in Oct. 2021 was GoDaddy (32% of all registrations) [65]. These facts indicate an attack strategy that prolongs the lifespans of phishing URLs by using popular .com and GoDaddy.

Moreover, as shown in Table 4, XLoader used the same registrant country name when applying to GoDaddy for domains in the repositories (this may be a natural consequence because to register a domain with a different country name every

TABLE 4. Percentages of registrars used by XLoader to obtain domains stored in the repositories. XLoader used the same registrant country name for each registrar.

GoDaddy	GKG	DYNADOT	duckdns	NICNIC
83%	5%	4%	4%	2%
China	USA	USA	France	Japan

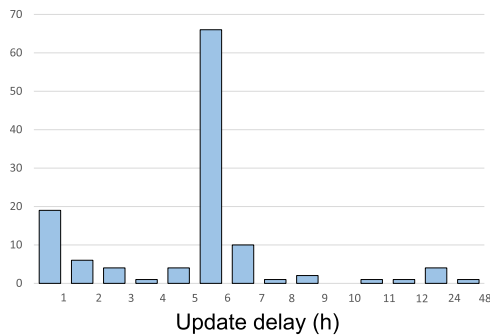


FIGURE 10. Histogram of the update delays (from the creation of a domain until it is written to the repositories). The number of samples is 120.

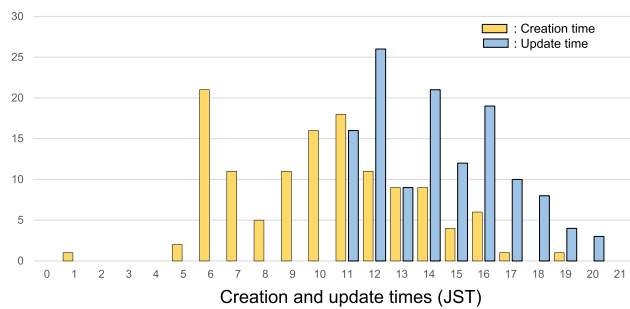


FIGURE 11. Histogram of creation and update times of domains in the repositories. The numbers of samples of creation and update times are 126 and 128, respectively.

day, the registrant needs to enter different payment information and phone number requested by GoDaddy every day). Such strong biases in TLDs, registrars, and registrant country names should be used to detect malicious URLs.

Fig. 10 shows the histogram of the update delays, where the update delay is an interval between the time a domain including .com was created and the time the domain was stored in the repositories. As shown in the figure, there is a sharp peak between 5 and 6 h. This result implies that the process from domain creation to URL replacement followed the same steps every day.

Fig. 11 shows the histogram of the times when domains containing .com were created and updated, where the time zone was Japan Standard Time (JST). As shown in the figure, most domains were created in the morning and updated in the afternoon. The update frequency decreased at 13:00. We attributed this phenomenon to lunch. In other words, the attackers were operating in the UTC+8 time zone, one hour after JST. This inference is consistent with the following

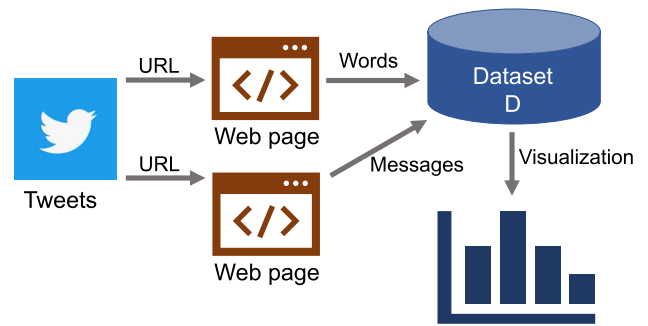


FIGURE 12. Dataset *D* consists of words and messages in the web pages that are referenced in the tweets about smishing attacks. It is used to visualize shared information about smishing.

conclusion in [18]: Based on the time gap between the creation of the certificate and the signature file, we believe that the first version of XLoader was created in the UTC+8 time zone. From the figure, more than 80% of the domains were created in the morning of the UTC + 8 time zone. This fact can be leveraged in detecting phishing URLs. There is another drop at 8:00, which is probably a break period.

VI. TWITTER ANALYSIS

Twitter is a tool for quickly obtaining information related to security incidents, such as IoCs and malware distribution sites, and [66] reported that Twitter captures ongoing malware threats better than public threat intelligence. Based on Twitter data, this section presents shared information about previous smishing attacks.

A. DATA COLLECTION

Fig. 13 shows the monthly change in the number of Japanese tweets containing the term “smishing” collected using Twitter search [67]. As shown in the figure, the first tweet appeared in 2015 and the tweets were active from Apr. 2019 to Dec. 2021. This result reflects the recent prevalence of smishing attacks in Japan.

Each tweet often included links to web pages that provide more detailed information about smishing. Fig. 12 illustrates that dataset *D*, consisting of smishing-related words and messages, was created in two steps: URL and word/message extraction. The following details the two steps.

- 1) Convert all shortened URLs (`http://t.co/`) in the tweets to the original URLs and collect text areas (the areas enclosed by tag `<p>`) on the web pages pointed by the URLs. SNS, summary, duplicate, or unrelated pages were not used for creating dataset *D*. Two pages were considered duplicated if their URLs, titles, or 75% or more of their text areas were the same. Pages were considered unrelated unless they contained “smishing” or unless they contained “SMS” and at least one of the following terms: phishing, fraud, scam, fake, deceive, cheat, threat, spoof, and impersonate.

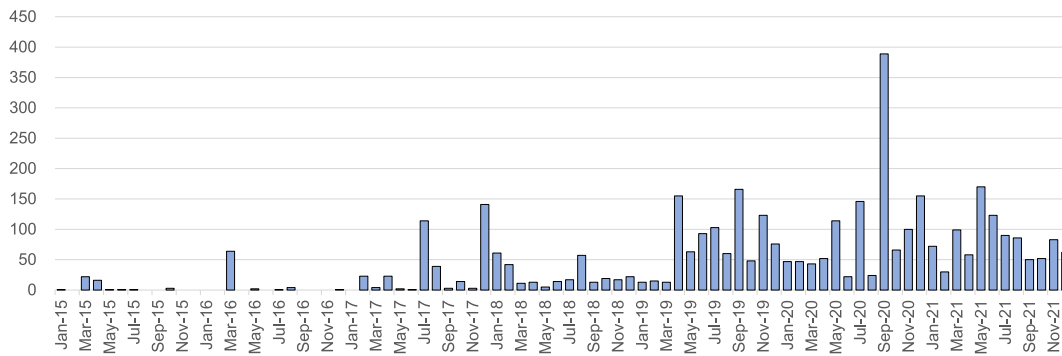


FIGURE 13. Number of tweets in Japanese containing the term “smishing” per month. The total number of tweets is 3,883.

2) Extract smishing messages and words (proper and compound nouns) from the text areas collected above using a morphological analysis library janome [68] and then create dataset D consisting of 3-tuples (t, u, v) , where t , u , and v denote the time the tweet was posted, a URL in the tweet, and a message or a word on the web page URL pointed, respectively. Dataset D satisfied the condition that more than one (t, u, v) with the same u and v pairs did not exist in it.

A total of 173 URLs ($\{|u|\}$) was used to create the dataset. The collected words and messages ($\{v\}$) were grouped into several categories because they contained a variety of words and messages, including synonyms and paraphrases. Consequently, the number of categories for fake sender names, attack motives, and smishing messages were seven, four, and seven, respectively.

B. YEARLY STRATEGIES

Let us first examine the yearly trends in fake sender names and attack motives. Fig. 14(a) shows the yearly frequencies of the fake sender categories, where the frequency $f_{C,Y}$ of category C in year Y is defined as follows:

$$f_{C,Y} = |\{(t, u, v) \in D | t \in Y, v \in C\}|. \quad (2)$$

The figure reveals different trends in sender names between 2017 and 2019 and beyond. The primary categories changed from Google and bank to home delivery and mobile carrier. Home delivery and mobile carrier accounted for a large percentage of the next three years after 2019. The strategy of impersonating these senders for such a long period of time must provide a great opportunity for smishing detection. A large decrease in mobile telecommunication charges in Japan in 2020 and the high usage of home delivery owing to COVID-19 could have had an impact on the sender’s name selection.

Fig. 14(b) shows the yearly changes in the attack motives. Although the figure shows that credit card information was the main target in 2017, the proportion of phone number and authentication information has increased since 2019. In other words, the sender names and attack motives changed

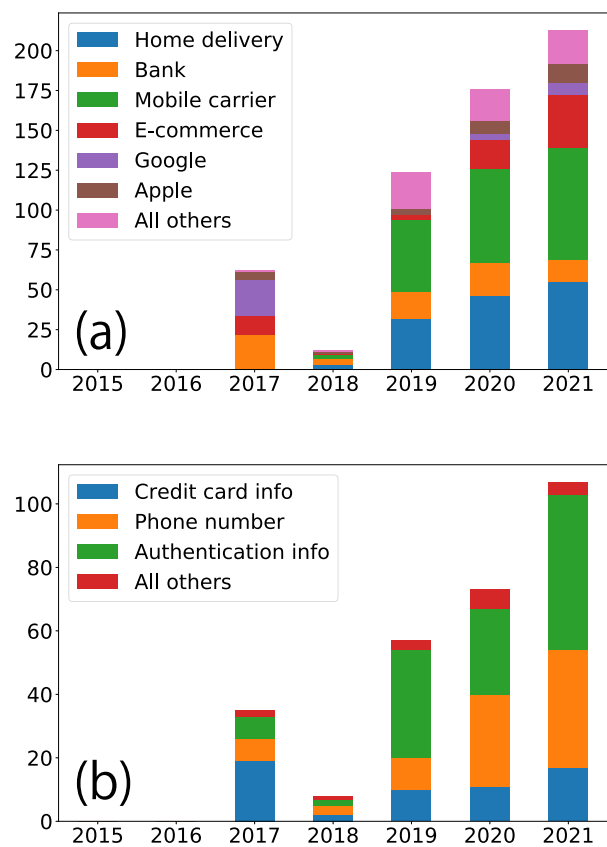


FIGURE 14. Yearly frequencies $f_{C,Y}$ of (a) sender and (b) attack motive categories. “All others” is the categories satisfying $f_{C,Y} < 20$ for all Y .

simultaneously. Accordingly, circa 2018 was a turning point in smishing strategies in Japan. Thus, smishing detection should use smishing data collected after the turning point. Note that the period when the number of C2 commands increased by 70% (Aug. 2017 to May 2018 in Fig. 5) overlaps with the turning point period. Thus, the malware’s capabilities also improved during the turning point.

Fig. 15 shows the yearly frequencies of the message content categories. It can be seen that many categories appear

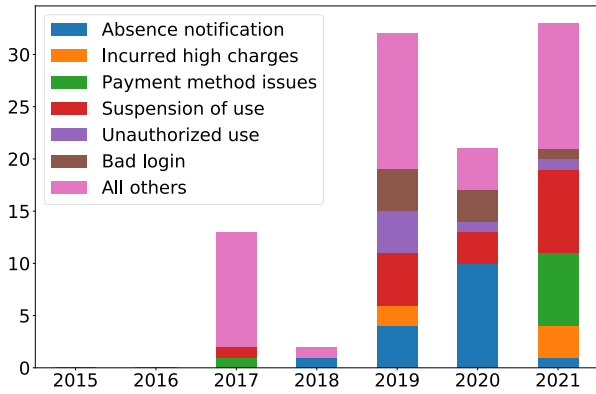


FIGURE 15. Yearly frequencies $f_{C,Y}$ of message content categories. “All others” is the categories satisfying $f_{C,Y} < 5$ for all Y .

TABLE 5. Three commands `state`, `getPkgMap`, and `sendSms` detectable by network traffic analysis.

	Active	Passive	
	Statistic	Statistic	Signature
Command	<code>state</code>	<code>getPkgMap</code>	<code>sendSms</code>
Features	Timestamp Frame length	Send interval Frame length	Phone number URL
Run time	≈ 30 s	Background	≈ 1 h
Device mode	Sleep mode on/off	-	Silent mode

with a low frequency of occurrence each year. Unlike in Fig. 14, the primary categories in Fig. 15 change every year. For example, the percentage of “absence notification” in 2020 was 48%, and 13% and 4% in 2019 and 2021, respectively. We tested the following null hypothesis:

Null Hypothesis: Frequency $f_{C,Y}$ of each message content category C is independent of year Y .

Fisher’s exact test was conducted using $\{f_{C,Y}\}$, $Y = 2019, 2020, 2021$, which had sufficient sample sizes. The test showed that the null hypothesis was rejected because the P-value was 2.4×10^{-4} . This result indicates that the most used message categories vary depending on the year even after the turning point.

VII. APPLICATIONS

This section applies FakeSpy activity patterns to malware and malicious domain detection.

A. MALWARE DETECTION

Network traffic analysis-based malware detection methods are classified into signature-, statistic-, and lexical-feature-based methods [39]. In this paper, the terms *active* and *passive* are introduced to classify these methods further, where active (passive) methods (do not) require human operations for traffic generation.

Table 5 shows three detectable commands transmitted/received by malware JapanPost. Let us first discuss the detection of `state`. Because `state` appears whenever the power button is pressed, it is detectable by checking whether

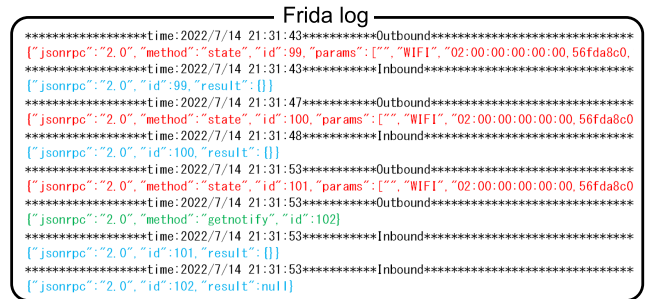
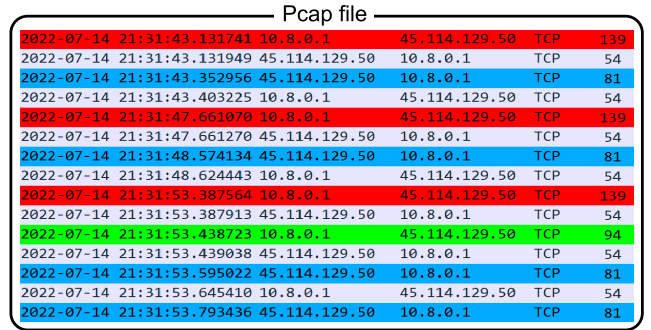


FIGURE 16. Timestamps and frame lengths detect encrypted `state`. Upper: Encrypted commands in the pcap file. Packets colored red, blue, green, and gray denote `state`, responses of `state`, `getnotify`, and TCP acks, respectively. Lower: Decrypted commands in the Frida log file.

TABLE 6. Frequencies of `getPkgMap` transmission intervals in four days.

Interval (s)	500	501
Frequency	640 (96.5%)	23 (3.5%)

the Frida log file includes the command after pressing the button. Note that Frida requires root privilege, which is not available in most devices. Therefore, we identified packets carrying `state` in the packet capture (pcap) file. Fig. 16 illustrates that timestamps can be used to identify those packets. From the figure, for example, the first `state` command in the Frida log was recorded at 21:31:43, which was the same as the timestamp of the first packet in the pcap file. Because the device state does not frequently change, `state` commands have the same frame length (139 bytes in Fig. 16 even though the id value increases from two to three digits); this fact can relax the constraint of measuring the exact times of pushing the button.

The following detection problem verified our approach:

Problem 1: Find five packets carrying `state` in the pcap file after manually turning sleep mode on and off five times every five seconds referring to the timer.

Let $O_k = \{p|s(p) \in [t_b + 5(k - 1), t_b + 5(k - 1) + 2]\}$, where $s(p)$ is the timestamp (in second) of packet p and t_b is the time the power button is to be pressed first. O_k is the set of packets in which k -th `state` is expected to be included.

Algorithm 1: Output the value of function g defined by

$$g(\{O_k\}) := \left| \bigcap_{k=1}^5 \{(\ell(p), f(p)) | p \in O_k\} \right|, \quad (3)$$

TABLE 7. Comparison with recent network traffic analysis-based malware detection methods.

	Hybrid [69]	TCP & HTTP [70]	JSON-RPC (our methods)
Features	Code & TCP	TCP/HTTP	JSON-RPC
Malware	Android malware		FakeSpy
Scheme	Embedding feature vectors	C4.5 decision tree	Command-level analysis
CPU load	Heavy	Heavy	Light
TPR (%)	97	97.89	≈ 100
FPR (%)	-	More than 10	≈ 0
Accuracy (%)	97	-	-

where $\ell(p)$ is the length of p and $f(p)$ is the flow id of p given by source IP and Port, destination IP and Port, and Protocol.

Function g outputs the number of solution candidates to Problem 1. The experiments were performed more than ten times; Algorithm 1 always outputted $g = 1$ and the Frida log files verified that all the candidates were true solutions. At present, there are no benign apps that transmit a packet whenever the power button is pressed. Thus, both the false negative rate (FNR) and false positive rate (FPR) of this method are close to zero.

Commands `getPkgMap` in Table 5, which are used to get the list of packages that need to be updated, can be detected by a passive statistic-based method because they are transmitted by a bot approximately every 500 seconds. Table 6 shows that 96.5% of the transmission intervals of `getPkgMap` measured in four days were 500 s. Moreover, the frame size of the command did not frequently change; it changed only once during the four-day experiment period. Thus, if packets of the same size appear every 500 or 501 s in the pcap file, they are `getPkgMap` commands with a high probability. This method is suitable for running as a background task because it does not require anything from the device user. If candidates of `getPkgMap` appear, the method of detecting state can be used to make the FPR close to zero.

Lastly, we describe the detection of `sendSms` in Table 5. As detailed in Section IV, bots receive more than a hundred of `sendSms` commands per hour every day and each `sendSms` includes an unencrypted phone number and a URL (see Fig. 19), which can be used as signatures. Thus, if more than a hundred of packets including the signatures arrived in an hour, the probability that the malware is running on the device is one. There are no benign apps that receive such packets. This method, however, reduces device usability because `sendSms` arrives only if the device is in silent mode; therefore, it might be used only for final confirmation of malware infection.

The comparison result of the proposed method with existing traffic analysis-based methods is presented in Table 7. As shown in the table, the hybrid system [69] achieved excellent results by analyzing both the malware code and TCP traffic. Current traffic analysis-based methods utilize TCP/IP traffic features, and extract upper layer protocol (e.g., HTTP) features if the TCP payload is not encrypted [70]. Our

analysis is new, in that we revealed encrypted JSON-RPC traffic features. As a result of command-level analysis, all our methods were simple and achieved approximately zero error rates. They can run on a device because they do not need computationally expensive (ML) algorithms.

B. MALICIOUS DOMAIN DETECTION

This section uses the botnet monitoring system in Fig. 3 to collect FakeSpy domains included in `sendSms` commands for malicious domain detection. FakeSpy, like XLoader in Section V, creates domains whose TLD is `.com` using GoDaddy in the morning.

Let us define domain sets G (referred to as batch sets) using collected FakeSpy domains $\{d_1, d_2, \dots\}$ such that $d_i, d_j \in G$ if $|t(d_i) - t(d_j)| < 1$ (hour), where $t(d_i)$ is the creation time (Unix time) of d_i . This definition allowed every domain to belong to a unique batch set G . Let us next define the mean creation time of batch set G as $t(G) = \frac{1}{|G|} \sum_{d \in G} t(d)$. Fig. 17 shows deviations L_i given by

$$L_i = t(d_i) - t(G), \quad (4)$$

where $d_i \in G$ and $|G| \geq 2$. From the figure, the standard deviation σ of L_i was 4.4 s. Fig. 17 also shows the histogram of batch set size $|G|$, which indicates that more than 64% of sets G satisfy $|G| > 1$ and the mean size of $|G|$ is three. The mean should increase if more monitoring systems in Fig. 3 are used. The following detection algorithm is based on the fact that FakeSpy tends to create domains in batches.

Algorithm 2: Domain d_i is malicious, if there is a domain $d_j \in D_{mal}$ satisfying

$$|t(d_i) - t(d_j)| \leq t_h, \quad (5)$$

where t_h is a positive number in second and D_{mal} is a set of known malicious domains.

Algorithm 2 needs seed set D_{mal} . In this study, D_{mal} is the set of domains in `sendSms` collected by the monitoring system in Fig. 3. Thus, unknown FakeSpy domains are inferred from FakeSpy domain samples D_{mal} . Let \mathcal{G} be the set of domains in the SMS messages a user receives from sources other than FakeSpy. The domain creation processes have been proposed as compound Poisson processes [71]; therefore, we assume that $Z(t)$, the number of domains $d_k \in \mathcal{G}$ created in $(0, t]$, also follows the same rule; i.e.,

$$Z(t) = Y_1 + Y_2 + \dots + Y_{N(t)}, \quad (6)$$

where $N(t)$ is a Poisson process with the batch creation rate λ , and Y_1, Y_2, \dots are i.i.d. Poisson random variables with the mean batch size μ . Note that $E(Z(t)) = \lambda\mu t$ and that $\lambda\mu$ is the mean arrival rate of SMS messages including URL links. A Poisson process is a case when Y_1, Y_2, \dots are all one. We also assume that the deviations L_i in each batch set G have a normal distribution with mean zero and variance σ^2 , $\mathcal{N}(0, \sigma^2)$. Because $t(d_i) - t(d_j) = L_i - L_j \sim \mathcal{N}(0, 2\sigma^2)$, the FNR of Algorithm 2 is given by

$$P(|t(d_i) - t(d_j)| > t_h | d_i, d_j \in G) = 1 - \operatorname{erf}\left(\frac{x}{2}\right), \quad (7)$$

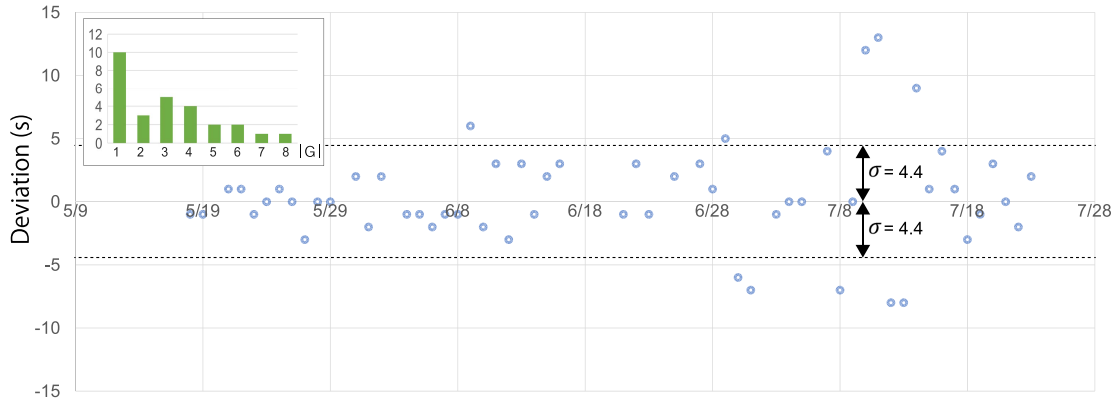


FIGURE 17. Deviations L_i calculated every day from Mar. 18 to Jul. 23, 2022, and the histogram of $|G|$. The standard deviation $\sigma = 4.4$ s.

where $x = t_h/\sigma$ and $\text{erf}(y) = \frac{2}{\sqrt{\pi}} \int_0^y e^{-t^2} dt$. The FPR can be written as

$$\begin{aligned}
 &P(|t(d_k) - t(d_j)| \\
 &\leq t_h | d_k \in \mathcal{G}, d_j \in \mathcal{G}) \\
 &= P(t(d_k) \in [-t_h, t_h] | d_k \in \mathcal{G}) \\
 &= \begin{cases} 1 - \exp(-2\lambda\sigma x) & \text{if } Y_i = 1, \\ 1 - \exp(-2\lambda\sigma x(1 - e^{-\mu})) & \text{if } Y_i \neq 1. \end{cases} \quad (8)
 \end{aligned}$$

From (7) and (8), the FNR (FPR) is a strictly decreasing (increasing) function of x . Thus, there is a unique x at which the FNR agrees with the FPR, and the equal error rate is referred to as the crossover error rate (CER). Fig. 18 shows CERs at $\sigma = 4.4$ s, which was given in Fig. 17. From Fig. 18(a), the CER increases with λ ; thus, the CER becomes significantly small, if users do not receive SMS messages frequently. For example, if $\lambda = 0.1/(24 \times 60^2)$ (domains/s), by using $t_h = 5.9\sigma$, the CER becomes 3×10^{-5} . Fig. 18(b) shows the CERs when the arrival rate $\lambda\mu$ is set at one. The CERs at $\mu = 1, 2$ are approximately the same as that of the Poisson process with $\lambda = 1$; however, when the mean batch size μ increases to 10, the CER falls considerably. Accordingly, a smaller SMS arrival rate or a larger batch size reduces the CER. SMS messages are primarily used for notification and prompting purposes and are not free. Therefore, people generally do not receive many messages. The advantage of this method is that users with low SMS arrival rates are given lower CERs.

Let us estimate how much this method can improve the performance of existing technology. Table 8 shows the sendSms arrival dates, creation dates, and VirusTotal scan results of FakeSpy’s second-level domains (SLDs), where all their TLDs were .com and all their creation times were used to calculate L_i in Fig. 17. A scan result of a domain is the number of vendors that considered the domain malicious, where VirusTotal consisted of the products and datasets from 94 vendors. Let D_{vt} be the set of FakeSpy domains that were consisted malicious by at least one vendor. In Table 8, domains with the same creation date belong to the same batch set G , and the zero is red, if the domain has at least one

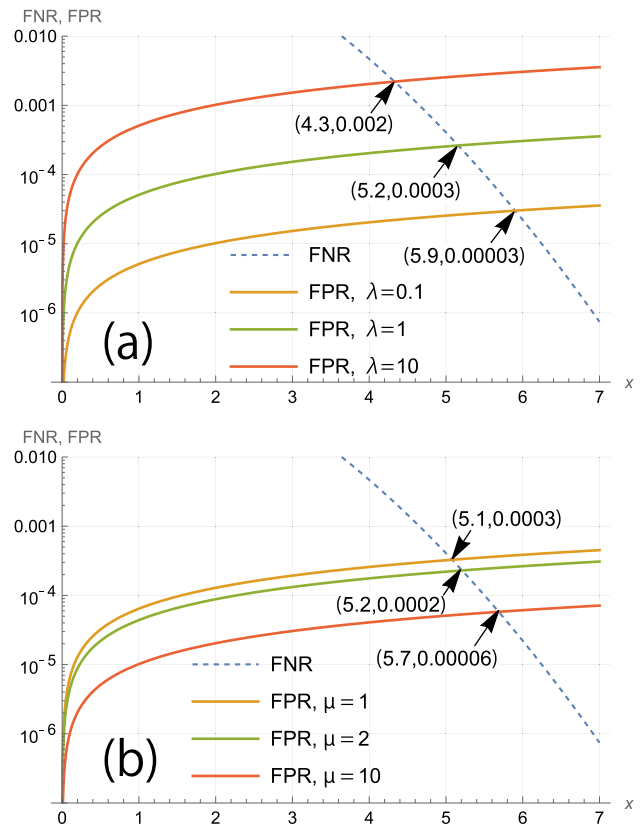


FIGURE 18. CERs for (a) Poisson and (b) compound Poisson processes with $\lambda\mu = 1$. $\sigma = 4.4$ s. (a) CER increases with λ (domains/day). (b) CER is significantly small at a large μ .

effective seed, where d_j is an effective seed of $d_i \notin D_{vt}$ if $d_i, d_j \in G$ and $d_j \in D_{vt}$. Algorithm 2 can change all red zeros to ones with a high probability if $D_{mal} = D_{vt}$. For example, all red zeros become ones if $x = 5.7$, which corresponds to $\mu = 10$ in Fig. 18(b), because $t_h = 5.7\sigma > 25$ and for all L_i in Fig. 17, $-10 < L_i < 15$. In Table 8, 33% of zeros are red. Long-term observations from Jul. 4 to Aug. 15 showed that 31% of zeros were red.

The comparison results of our method with existing ones [30], [31] are given in Table 9. All the methods inferred

TABLE 8. Red zero (0) denotes the case wherein the SLD had at least one effective seed. For example, `hrvxe` was an effective seed of `rwptm` on Jul. 4. All SLDs were made up of five alphabetic letters.

FakeSpy SLD	Arrival date	Creation date	VirusTotal scan dates and results				
			7/4	7/6	7/8	7/10	7/12
hrvxe	6/25	6/21	3	11	12	12	12
rwptm	6/27	6/21	0	4	10	10	10
vvdpf	6/28	6/21	10	14	16	16	16
yengn	6/29	6/21	0	1	10	10	12
yfpzp	6/30	6/21	9	11	14	14	16
zqcne	7/1	6/21	0	0	0	0	3
eauzg	7/2	6/22	0	0	0	0	3
ewybg	7/2	6/26	0	0	2	2	3
mzsac	7/3	6/26	0	0	1	1	2
hyeyu	7/3	6/26	0	0	2	2	3
rdybz	7/4	6/26	0	0	2	2	3
yermu	7/5	6/26	0	2	2	2	3
uzshq	7/5	6/26	0	0	1	1	2
yxmnu	7/6	6/26	0	0	0	0	4
bznrm	7/6	6/27	0	0	0	0	3
qpaqr	7/7	7/3			0	0	0
vsbvu	7/8	7/3			0	0	2
utbtn	7/8	7/3			0	0	5
aoxra	7/9	7/4				0	1
pbyr	7/9	7/4				0	1
dphqa	7/10	7/7				0	1
cavxv	7/10	7/7				0	2
paxgc	7/11	7/7					0
qbtth	7/11	7/7					0
ruhht	7/12	7/7					0
thqee	7/12	7/7					0

TABLE 9. Malicious domain detection methods based on registration information. TPR of [30] was 73% (93%) when the inferred domains were blacklisted (suspect). TPR and FPR of our method were derived from Fig. 18.

Features	Proactive blacklisting		User protection (our method)
	[30]	[31]	
Registrar, creation date, name server usage		22 features of domain profile, registration history, batch correlation	Registrar, Variance σ^2 , arrival rate λ , batch size μ
Attackers	Unrestricted		FakeSpy
Seed set	Blacklist	-	D_{mal}
Scheme	Inference from seed domains	ML systems infer domain reputations	Algorithm 2 checks arrivals
TPR (%)	73 (blacklist) 93 (suspect)	70	99.8 ($\lambda = 10$) 99.97 ($\mu = 1$)
FPR (%)	A few	0.35	0.2 ($\lambda = 10$) 0.03 ($\mu = 1$)

malicious domains based on their registration information. Unlike [30] and our method, [31] does not need seed domains (even though [31] needs many features). The methods [30] and [31] predicted a large number of newly registered bad domains used for spam, phishing, drive-by downloads, etc. In [31], for example, more than a thousand bad domains were detected per day. Our method, in contrast, focused on FakeSpy smishing messages, and presented satisfactory performance even when $\lambda = 10$ in Fig. 18(a) and $\mu = 1$ in Fig. 18(b). The main advantage of our method is to reduce the error rates significantly if the SMS arrival rate $\lambda\mu$ is small. Our method could detect many botnets by preparing their (σ, D_{mal}) because spammers in general employ bulk registration [71].

VIII. DISCUSSION

In this section, we discuss the strength, limitation, impact/significance, the failing conditions of the proposed method, as well as threats to its validity, and the possible improvements to existing systems.

The strength of our work is that all our findings are based on large amounts of XLoader and FakeSpy activity data we collected. Owing to these new data, the proposed methods had precise and reliable attack features and parameter values, which included the `state` behavior in Fig. 16, the transmission intervals in Table 6, and the standard deviation σ in Fig. 17.

However, we focused only on XLoader and FakeSpy, the two major malware families in Japan. Although they have spread globally [5], [6], [7], our objective was originally to curb smishing attacks in Japan. Note that the proposed methods are not limited to Japan. In addition, there are other smishing malware families in Japan that are not addressed in this paper, such as TianySpy and KeepSpy, all of which are not among the top-50 Android malware families globally [14].

The significance of this research is that we not only investigated the lifespans and functions of the attack tools in Fig. 1 but also clarified the activity patterns of attackers who manipulate the tools behind the scenes. More than when only the tools are analyzed, understanding the activity patterns was useful for extracting decisive attack features and inferring the attackers' time zone. This study also demonstrated the importance of reviewing attack history. The Twitter data analysis in Fig. 14 suggests that smishing tools created before 2018 are outdated in Japan.

The proposed methods fail under the following conditions. First, the malware detection methods fail if the transmission conditions of `command state`, `getPkgMap`, or `sendSms` change or if the frame size of `state` or `getPkgMap` varies frequently in a short period of time. Second, the malicious domain detection method fails if it is difficult to obtain seed domains, if malicious domains are not registered in batches, or if device users receive SMS messages with URL links very often.

There are two potential threats to the validity of our work. First, in order not to contribute to the spread of smishing messages, we did not use any subscriber identity module (SIM) cards in our botnet observations. Installing a SIM card must affect the responses of the `sendSms` commands; however, its effect on `sendSms` traffic is unknown. Second, some probability distributions were assumed in (7)–(8) to estimate the error rates. Measuring the precise shapes of the distributions could affect threshold t_h and CER values. However, every distribution satisfies the fact that the CER decreases as the SMS arrival rate decreases.

The possible improvements to the existing systems are as follows. First, the command-level traffic analysis should be incorporated into the existing systems. The analysis could provide decisive attack features because commands are transmitted according to attack strategies. Second, existing

malicious domain detection methods should consider how often users receive SMS messages, specifically because the error rates fall significantly if the users do not receive SMS messages very often.

IX. CONCLUSION

In Japan, smishing is widespread and can cause considerable damage. The attackers dynamically switch strategies based on countermeasures and the social context. To understand their activity patterns and the shift in their strategies, we observed the activities of the XLoader and FakeSpy botnets, revealing a variety of previously unreported findings. The following are our observational facts and their applications:

- The attackers performed routine work every day, except Sunday, and occasionally, they were inactive for several days. A new malware package was released just after the inactive period; therefore, the period can be taken as a signal to perform activities such as updating malware signatures, etc. Their routine work of spreading more than one hundred smishing messages per bot was performed at a fixed daily time period, and this fact was applied to the proposed method.
- Frida’s hooking capability revealed bidirectional command-and-response communication between the C2 server and bots. This communication conveyed the state changes of bots, such as sleep mode changes, to the server in real time. The number of XLoader and FakeSpy commands increased over time, and three new commands were discovered in this study.
- The attackers had repositories for smishing messages. According to the repository data analysis, they primarily used the GoDaddy registry and created domains in the morning. The attackers’ lunch time, inferred from the repository update history, identified their time zone.
- According to Twitter data analysis, circa 2018 was a turning point in smishing strategies in Japan; the primary sender names and attack motives completely changed. Thus, smishing detection should use smishing message samples after the turning point. The main industries used as the sender names during the three-year period from 2019 to 2021 were courier services and mobile carriers.
- The command-level traffic analysis yielded methods for detecting FakeSpy malware with small computational complexity and small error rates. An attackers’ domain registration pattern led us to a malicious domain detection method with small error rates. None of the methods require computationally expensive ML algorithms.

Future research directions of this study are as follows. First, it is necessary to develop Android apps that repel XLoader and FakeSpy malware based on the methods proposed in this paper and measure CPU and memory utilization and user operability. Second, to suppress smishing attacks in Japan, decisive features such as (σ, D_{mal}) for all the major smishing botnets should be measured.

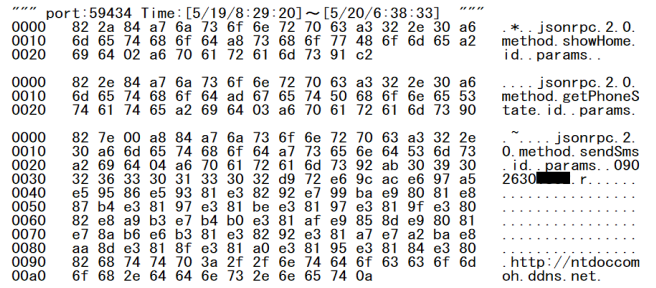


FIGURE 19. Hexadecimal dump of JSON-RPC commands and their ASCII code representation.

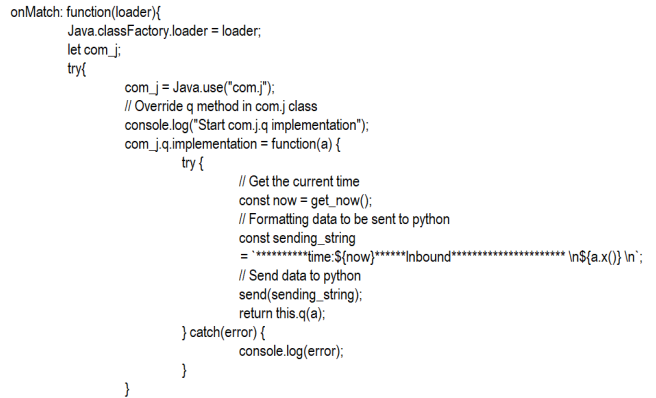


FIGURE 20. JavaScript code submit.js sent to the Frida-server.

**APPENDIX A
JSON-RPC LOG**

Fig. 19 shows JSON-RPC commands from the C2 server captured by tPacketCapture [72]. As shown in the figure, the server first sent a showHome command to make the device screen the home screen. It then sent getPhoneState to confirm the device identification information, which is the International Mobile Subscriber Identity, IC Card Identifier, and Android ID. Subsequently, it began to send many sendSms commands. Each sendSms included a phone number, a smishing message, and a phishing URL as parameters. In the ASCII code representation in Fig. 19, a Japanese smishing message was encoded in MessagePack [73] between phone number 0902630**** and URL http://ntdoccomoh.ddns.net.

**APPENDIX B
JAVASCRIPT CODE**

Fig. 20 shows the JavaScript code submit.js sent to the Frida-server. The code first determines the loader instance because the payload is dynamically loaded. When the bot receives JSON-RPC commands encoded in MessagePack from the C2 server, the code modifies the q() method of class com.j such that com.j.q() sends the commands to the Frida-client on the PC after decoding them. When the bot sends JSON-RPC commands to the C2 server (this case is not included in Fig. 20), the code modifies the com.j.r() method such that com.j.r() sends the commands to the Frida-client before encrypting them.

REFERENCES

- [1] (2015). *Phishing Scam Targeting Mobile Users: Confirmation of Guidance via SMS*. [Online]. Available: <https://blog.trendmicro.co.jp/archives/11599/>
- [2] P. Kalaharsha and B. M. Mehtre, "Detecting phishing sites—An overview," 2021, *arXiv:2103.12739*.
- [3] N. Al-Milli and B. H. Hammo, "A convolutional neural network model to detect illegitimate URLs," in *Proc. 11th Int. Conf. Inf. Commun. Syst. (ICICS)*, Apr. 2020, pp. 220–225.
- [4] S. Y. Yerima and M. K. Alzaylaee, "Mobile botnet detection: A deep learning approach using convolutional neural networks," in *Proc. Int. Conf. Cyber Situational Awareness, Data Analytics Assessment (CyberSA)*, Jun. 2020, pp. 1–8.
- [5] E. Montalbano. *Fakespy Android Malware Spread via 'Postal-Service' Apps*. [Online]. Available: <https://threatpost.com/fakespy-android-malware-spread-via-postalservice-apps/157102/#:text=Android2020>
- [6] T-Seals. (2022). *Roaming Mantis Expands Android Backdoor to Europe*. [Online]. Available: <https://threatpost.com/roaming-mantis-android-backdoor-europe/178247/>
- [7] I. Suguru. (2022). *Roaming Mantis Reaches Europe*. [Online]. Available: <https://securelist.com/roaming-mantis-reaches-europe/105596/>
- [8] S. Sawyer, "Potential threats and mitigation tools for network attacks," *Austral. J. Wireless Technol., Mobility Secur.*, vol. 1, no. 1, pp. 6–11, 2021.
- [9] H. Ahmad and L. Erdodi, "Overview of phishing landscape and homographs in Arabic domain names," *Secur. Privacy*, vol. 4, no. 4, pp. 1–14, Jul. 2021.
- [10] J. Obuhuma and S. Zivuku, "Social engineering based cyber-attacks in Kenya," in *Proc. IST-Africa Conf. (IST-Africa)*, 2020, pp. 1–9.
- [11] P. N. Mangut and K. A. Datukun, "The current phishing techniques—perspective of the Nigerian environment," *World J. Innov. Res.*, vol. 10, no. 1, pp. 34–44, Feb. 2021.
- [12] A. Darem, "Anti-phishing awareness delivery methods," *Eng., Technol. Appl. Sci. Res.*, vol. 11, no. 6, pp. 7944–7949, Dec. 2021.
- [13] Y. A. Younis and M. Musbah, "A framework to protect against phishing attacks," in *Proc. 6th Int. Conf. Eng. (MIS)*, Sep. 2020, pp. 1–6.
- [14] L. Wang, H. Wang, R. He, R. Tao, G. Meng, X. Luo, and X. Liu, "Mal-Radar: Demystifying Android malware in the new era," in *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 6, no. 2, pp. 1–27, 2022.
- [15] H. Cai, "Assessing and improving malware detection sustainability through APP evolution studies," *ACM Trans. Softw. Methodol.*, vol. 29, no. 2, pp. 1–28, Apr. 2020.
- [16] X. Zhan, M. Zhang, Y. Zhang, M. Zhong, X. Zhang, Y. Cao, and M. Yang, "Slowing down the aging of learning-based malware detectors with API knowledge," *IEEE Trans. Dependable Secure Comput.*, early access, Jan. 2022, doi: [10.1109/TDSC.2022.3144697](https://doi.org/10.1109/TDSC.2022.3144697).
- [17] X. Zhang, Y. Zhang, M. Zhong, D. Ding, Y. Cao, Y. Zhang, M. Zhang, and M. Yang, "Enhancing state-of-the-art classifiers with API semantics to detect evolved Android malware," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2020, pp. 757–770.
- [18] L. Wu and E. Xu. (2018). *The Evolution of Xloader and Fakespy: Two Interconnected Android Malware Families*. [Online]. Available: <https://documents.trendmicro.com/assets/pdf/wp-evolution-of-xloader-and-fakespy-two-interconnected-android-malware-families.pdf>
- [19] M. Hijji and G. Alam, "A multivocal literature review on growing social engineering based cyber-attacks/threats during the COVID-19 pandemic: Challenges and prospective solutions," *IEEE Access*, vol. 9, pp. 7152–7169, 2021.
- [20] H. S. Lallie, L. A. Shepherd, J. R. C. Nurse, A. Erola, G. Epiphaniou, C. Maple, and X. Bellekens, "Cyber security in the age of COVID-19: A timeline and analysis of cyber-crime and cyber-attacks during the pandemic," *Comput. Secur.*, vol. 105, Jun. 2021, Art. no. 102248.
- [21] G. Sonowal and K. S. Kuppusamy, "SmIDCA: An anti-smishing model with machine learning approach," *Comput. J.*, vol. 61, no. 8, pp. 1143–1157, Aug. 2018.
- [22] S. Mishra and D. Soni, "Smishing detector: A security model to detect smishing through SMS content analysis and URL behavior analysis," *Future Gener. Comput. Syst.*, vol. 108, pp. 803–815, Jul. 2020.
- [23] M. Sanchez-Paniagua, E. F. Fernandez, E. Alegre, V. Al-Nabki, and V. Gonzalez-Castro, "Phishing URL detection: A real-case scenario through login URLs," *IEEE Access*, vol. 10, pp. 42949–42960, 2022.
- [24] S. Mishra and D. Soni, "A content-based approach for detecting smishing in mobile environment," in *Proc. Int. Conf. Sustain. Comput. Sci., Technol. Manage. (SUSCOM)*. Jaipur, India: Amity University Rajasthan, 2019.
- [25] C. T. Yadav, B. Rajendran, and P. Rajani, "An approach for determining the health of the DNS," *Int. J. Comput. Sci. Mobile Comput.*, vol. 3, no. 9, pp. 442–449, 2014.
- [26] P. Zhang, T. Liu, Y. Zhang, J. Ya, J. Shi, and Y. Wang, "Domain watcher: Detecting malicious domains based on local and global textual features," *Proc. Comput. Sci.*, vol. 108, pp. 2408–2412, 2017.
- [27] L. Bilge, S. Sen, D. Balzarotti, E. Kirde, and C. Kruegel, "Exposure: A passive DNS analysis service to detect and report malicious domains," *ACM Trans. Inf. Syst. Secur.*, vol. 16, no. 4, pp. 1–28, Apr. 2014.
- [28] B. Eshete, A. Villafiorita, and K. Weldemariam, "BINSPECT: Holistic analysis and detection of malicious web pages," in *Proc. Int. Conf. Secur. Privacy Commun. Syst. Berlin, Germany: Springer*, 2012, pp. 149–166.
- [29] I. Mishsky, N. Gal-Oz, and E. Gudes, "A topology based flow model for computing domain reputation," in *Proc. IFIP Annu. Conf. Data Appl. Secur. Privacy*. Cham, Switzerland: Springer, 2015, pp. 277–292.
- [30] M. Felegyhazi, C. Kreibich, and V. Paxson, "On the potential of proactive domain blacklisting," *Int. Comput. Sci. Inst.*, vol. 10, p. 6, Apr. 2010.
- [31] S. Hao, A. Kantchelian, B. Miller, V. Paxson, and N. Feamster, "PREDATOR: Proactive recognition and elimination of domain abuse at time-of-registration," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 1568–1579.
- [32] *VirusTotal*. Accessed: Jul. 12, 2022. [Online]. Available: <https://www.virustotal.com/gui/home/upload/>
- [33] H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Measuring code reuse in Android apps," in *Proc. 14th Annu. Conf. Privacy, Secur. Trust (PST)*, Dec. 2016, pp. 187–195.
- [34] A. Kalysch, O. Milsterfer, M. Protsenko, and T. Müller, "Tackling androids native library malware with robust, efficient and accurate similarity measures," in *Proc. 13th Int. Conf. Availability, Rel. Secur.*, Aug. 2018, pp. 1–10.
- [35] N. Totosis and C. Patsakis, "Android hooking revisited," in *Proc. IEEE 16th Intl Conf Dependable, Autonomic Secure Comput., 16th Intl Conf Pervasive Intell. Comput., 4th Intl Conf Big Data Intell. Comput. Cyber Sci. Technol. Congress (DASC/PiCom/DataCom/CyberSciTech)*, Aug. 2018, pp. 552–559.
- [36] *Frida*. Accessed: Feb. 6, 2022. [Online]. Available: <https://frida.re/>
- [37] J. Zhu, J. Jang-Jaccard, A. Singh, P. A. Watters, and S. Camtepe, "Task-aware meta learning-based Siamese neural network for classifying obfuscated malware," 2021, *arXiv:2110.13409*.
- [38] A. Rahali, A. H. Lashkari, G. Kaur, L. Taheri, F. Gagnon, and F. Massicotte, "DiDroid: Android malware classification and characterization using deep image learning," in *Proc. 10th Int. Conf. Commun. Netw. Secur.*, Nov. 2020, pp. 70–82.
- [39] M. E. Zadeh Nojoo Kamar, A. Esmailzadeh, Y. Kim, and K. Taghva, "A survey on mobile malware detection methods using machine learning," in *Proc. IEEE 12th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2022, pp. 0215–0221.
- [40] J. P. Singh and A. Chauhan, "Detection and prevention of non-Pc botnets," Tech. Rep., 2018.
- [41] K. F. Yu and R. E. Harang, "Machine learning in malware traffic classifications," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Oct. 2017, pp. 6–10.
- [42] H. Fu, P. Hu, Z. Zheng, A. K. Das, P. H. Pathak, T. Gu, S. Zhu, and P. Mohapatra, "Towards automatic detection of nonfunctional sensitive transmissions in mobile applications," *IEEE Trans. Mobile Comput.*, vol. 20, no. 10, pp. 3066–3080, Oct. 2021.
- [43] J. Feng, L. Shen, Z. Chen, Y. Wang, and H. Li, "A two-layer deep learning method for Android malware detection using network traffic," *IEEE Access*, vol. 8, pp. 125786–125796, 2020.
- [44] *malware-traffic-analysis.net*. Accessed: Aug. 1, 2022. [Online]. Available: <https://www.malware-traffic-analysis.net/>
- [45] D. S. Keyes, B. Li, G. Kaur, A. H. Lashkari, F. Gagnon, and F. Massicotte, "EntropLyzer: Android malware classification and characterization using entropy analysis of dynamic characteristics," in *Proc. Reconciling Data Analytics, Autom., Privacy, Security, Big Data Challenge (RDAAAPS)*, 2021, pp. 1–12.
- [46] (2017). *Android Banking Trojan Moqhao Spreading via SMS Phishing in South Korea*. [Online]. Available: <https://www.mcafee.com/blogs/other-blogs/mcafee-labs/android-banking-trojan-moqhao-spreading-via-sms-phishing-south-korea/>
- [47] R. Nigam, "A timeline of mobile botnets," *Virus Bull.*, vol. 1630, pp. 1–23, Mar. 2015.
- [48] M. Clasen, F. Li, and D. Williams, "Friend or foe: An investigation into recipient identification of SMS-based phishing," in *Proc. Int. Symp. Human Aspects Inf. Secur. Assurance*. Cham, Switzerland: Springer, 2021, pp. 148–163.

- [49] R. Griffin, "A demographic analysis to determine user vulnerability among several categories of phishing attacks," M.S. dissertation, DIT, 2018.
- [50] E. U. Soykan, M. Bagriyanik, and G. Soykan, "Disrupting the power grid via EV charging: The impact of the SMS phishing attacks," *Sustain. Energy, Grids Netw.*, vol. 26, Jun. 2021, Art. no. 100477.
- [51] E. Ustundag Soykan and M. Bagriyanik, "The effect of SMiShing attack on security of demand response programs," *Energies*, vol. 13, no. 17, p. 4542, Sep. 2020.
- [52] M. Shadkam, "Consumer's attitude to receive and response to SMS advertising," *Int. J. Bus. Inf. Syst.*, vol. 24, no. 1, pp. 69–90, 2017.
- [53] A. Eshmawi and S. Nair, "The roving proxy framewrok for SMS spam and phishing detection," in *Proc. 2nd Int. Conf. Comput. Appl. Inf. Secur. (ICCAIS)*, May 2019, pp. 1–6.
- [54] S. Mishra and D. Soni, "DSmishSMS—A system to detect smishing SMS," *Neural Comput. Appl.*, pp. 1–18, Jul. 2021.
- [55] (2018). *Roaming Mantis Part 3*. [Online]. Available: <https://blog.kaspersky.co.jp/roaming-mantis-new-methods/21749/>
- [56] L. Wu and E. Xu. (2018). *Examining Xloader, Fakespy, and the Yanbian Gang*. [Online]. Available: https://www.trendmicro.com/en_us/research/18/k/a-look-into-the-connection-between-xloader-and-fakespy-and-their-possible-ties-with-the-yanbian-gang.html/
- [57] *JSON-RPC*. Accessed: Feb. 6, 2022. [Online]. Available: <https://www.jsonrpc.org/>
- [58] A. Melnikov and I. Fette, *The WebSocket Protocol*, document RFC 6455, Dec. 2011.
- [59] (2018). *Evolving Roaming Mantis*. [Online]. Available: <https://blog.kaspersky.co.jp/roaming-mantis-update/20383/>
- [60] (2019). *Roaming Mantis Part 4*. [Online]. Available: <https://blog.kaspersky.co.jp/roaming-mantis-part-iv/22949/>
- [61] I. Tomoomi. (2010). *About Fake Short Messages*. [Online]. Available: <https://jpn.nec.com/cybersecurity/blog/200925/>
- [62] (2018). *Xloader Android Spyware and Banking Trojan Distributed via DNS Spoofing*. [Online]. Available: https://www.trendmicro.com/en_us/research/18/d/xloader-android-spyware-and-banking-trojan-distributed-via-dns-spoofing.html/
- [63] *Whois domainbigdata.com*. [Online]. Available: <https://www.whois.com/whois/domainbigdata.com>
- [64] *RiskIQ*. [Online]. Available: <https://www.riskiq.com/>
- [65] *Icann Monthly. Com Monthly Registry Reports*. Accessed: Feb. 6, 2022. [Online]. Available: <https://www.icann.org/resources/pages/com-2014-03-04-en/>
- [66] H. Shin, W. Shim, S. Kim, S. Lee, Y. G. Kang, and Y. H. Hwang, "#Twit: Social listening for threat intelligence," in *Proc. Web Conf.*, Apr. 2021, pp. 92–104.
- [67] *Explore/Twitter*. Accessed: Feb. 6, 2022. [Online]. Available: <https://twitter.com/explore/>
- [68] *Japanese Morphological Analysis Engine Written in Pure Python*. Accessed: Feb. 6, 2022. [Online]. Available: <https://github.com/mocobeta/janome/>
- [69] M. R. Norouzian, P. Xu, C. Eckert, and A. Zarras, "Hybrid: Toward Android malware detection and categorization with program code and network traffic," in *Proc. Int. Conf. Inf. Secur.* Cham, Switzerland: Springer, 2021, pp. 259–278.
- [70] S. Wang, Z. Chen, Q. Yan, B. Yang, L. Peng, and Z. Jia, "A mobile malware detection method using behavior features in network traffic," *J. Netw. Comput. Appl.*, vol. 133, pp. 15–25, May 2019.
- [71] S. Hao, M. Thomas, V. Paxson, N. Feamster, C. Kreibich, C. Grier, and S. Hollenbeck, "Understanding the domain registration behavior of spammers," in *Proc. Conf. Internet Meas. Conf.*, Oct. 2013, pp. 63–76.
- [72] *tPacketCapture for Android*. Accessed: Feb. 6, 2022. [Online]. Available: <https://tpacketcapture.en.uptodown.com/android/>
- [73] *MessagePack*. Accessed: Feb. 6, 2022. [Online]. Available: <https://msgpack.org/>



RYU SAEKI received the bachelor's degree in computer science and engineering from the Fukuoka Institute of Technology, in 2022, where he is currently pursuing the master's degree with the Graduate School of Engineering of Course of Computer Science and Engineering. His research theme is cyber security. He is currently conducting research on decoding Tor traffic. In the future, he hopes to work as a Cyber Security Specialist.



LEO KITAYAMA received the bachelor's degree in computer science and engineering from the Fukuoka Institute of Technology, in 2022. He then joined a Japanese security company, DIT Corporation, in 2022 as a new hire. He plans to work in the field of networking or security. His research interests include malware analysis, kernel architecture, cyber security, and windows malware analysis.



JUN KOGA received the bachelor's degree from the Department of Electrical Engineering, Faculty of Engineering, Fukuoka Institute of Technology, in 1996. He was a System Engineer at a private sector for seven years. He has been appointed as a Specialized Investigator of information technology at Fukuoka Prefectural Police, since 2003. He is currently in-charge of cyber-crime investigations and countermeasures at the Cyber Crime Control Division, Fukuoka Prefectural Police Headquarters. His main interests include cybersecurity and blockchains.



MAKOTO SHIMIZU received the advanced professional degree from the Kyushu Computer College Kitakyushu, and the bachelor's degree from the Department of Systems and Informatics, Faculty of Business Administration and Information Science, Hokkaido Information University as a combined degree of the Kyushu Computer College Kitakyushu, in 2012. He has been appointed as a Specialized Investigator for information technology with Fukuoka Prefectural Police, since 2012. He was in-charge of cybercrime investigations and countermeasures at the Cyber Crime Control Division, Fukuoka Prefectural Police Headquarters, until March 2022. Currently, he is an Investigator at the Juvenile Section of Kokurakita Police Station, Fukuoka. His main interests include cybersecurity and blockchains.



KAZUMASA OIDA (Member, IEEE) received the bachelor's degree in information science from the University of Tsukuba, in 1983, the M.E. degree from Hokkaido University, in 1985, and the Dr.Sc.Inf. degree from Kyoto University, in 2002. He worked at Nippon Telegraph and Telephone Corporation for twenty years as an Engineer, where he participated in the development of private network systems. He is currently a Professor with the Department of Computer Science and Engineering, Fukuoka Institute of Technology, Japan. He is currently working with the Fukuoka Prefectural Police to analyze smishing and Emotet malware and to trace attack traffic in the Tor network. His research interests include cybersecurity, social network analysis, blockchain applications, secure UAV systems, and complex networks.

...