

ZHENHUA DATA LEAK

PUBLIC REPORT

**Internet
2.0** 

AUTHORISED BY:
ROBERT POTTER co-CEO
DAVID ROBINSON co-CEO

WWW.INTERNET2-0.COM
CONTACT@INTERNET2-0.COM

Security



OUR STATEMENT

On Monday the 14th of September, a consortium of international news outlets released reports of a leaked database from China's Zhenhua Data Information Technology Co. that included significant amounts of data pertaining to over 2.4 million people. The database included the details of prominent figures and their families from around the world including key national decision makers from Australia and the United States of America. Internet 2.0's contribution was to restore the corrupted data, and in doing so uncovered the details of over 52,000 American, 35,000 Australian and nearly 10,000 British citizens.

We assess that around eighty percent of the information collected by Zhenhua Data was collected from open-source, publicly available information collated from social media profiles, the Factiva and Crunchbase aggregation platforms as well as Criminal records. As a result of the leak and subsequent investigation by Internet 2.0, Facebook has stated to the Washington Post that there was no data sharing agreement with Zhenhua Data.

For the vast majority of individuals whose information has been compiled by Zhenhua Data, the only information available in the database is information those same individuals had published and made available online.

KEY FACTS

Data summary and analysis.



WHERE IS THE DATA FROM

Through our analysis, we have assessed that 80% of the data is available from online public and open sources. The other 20% of data is from offline or non-public sources that is hard to access but still provided to advertisers or social platforms by users

VALUE TO CHINA

According to Dr Samantha Hoffman from the Australian Strategic Policy Institute GTCOM's Big Data Director, Liang Haoyu, claimed that '90% of military-grade intelligence data can be obtained from open data analysis'. GTCOM is a partner of Zhenhua Data



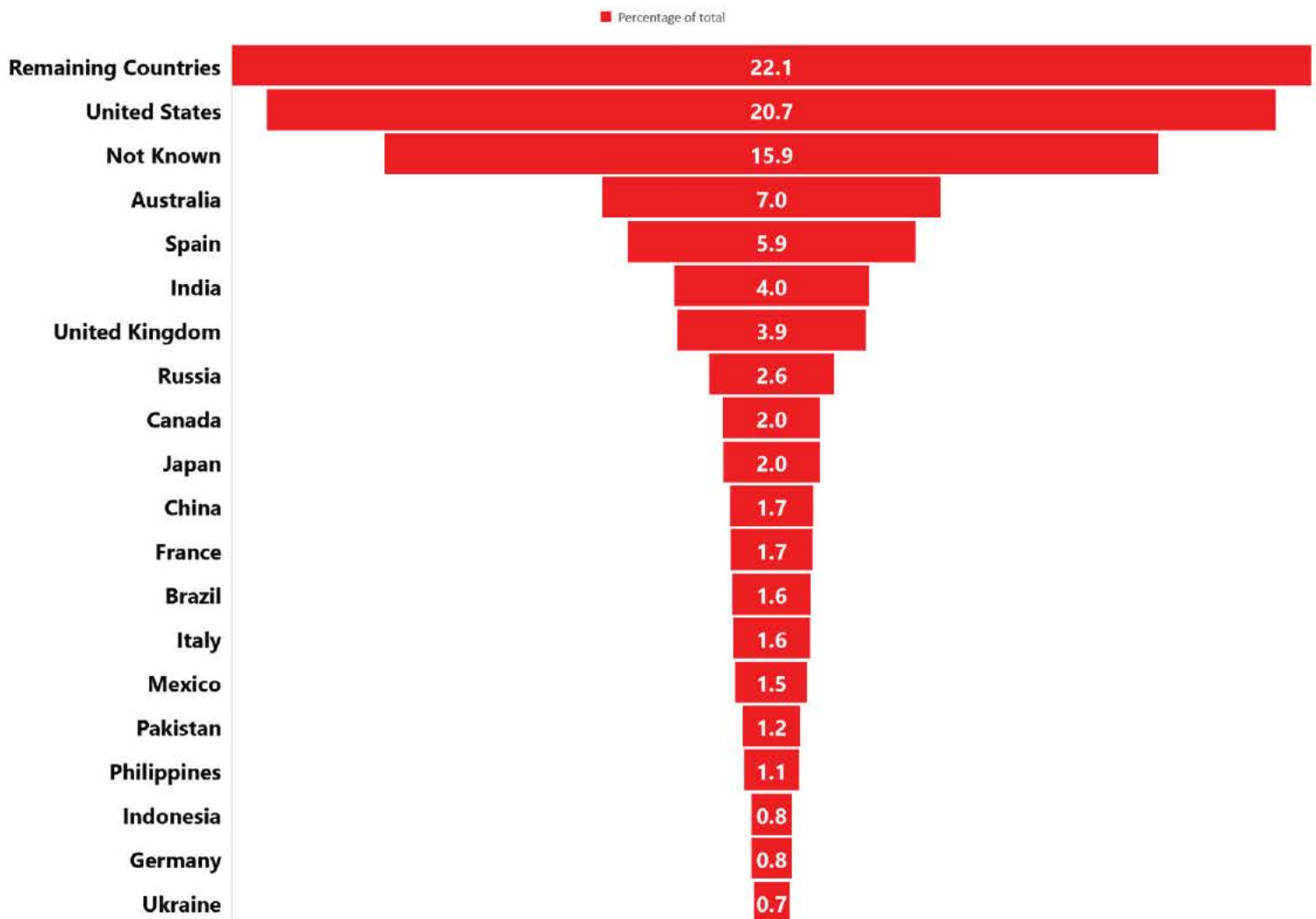
CONSENT



According to the Washington Post, Facebook and other social media platforms have stated that there was no data sharing agreements between them and Zhenhua Data. This appears to be in breach of their terms of service.

KEY FACTS

The demographics of the data.



2.4 MILLION



Number of people
Zhenhua Data
claimed
to track

650K



Number of
organizations
Zhenhua Data
claimed
to track

10% RESTORED



Amount of data
Internet 2.0 was
able to restore

It is important to mention that some of the information that was shared by users was through private accounts, and was never published publicly. Legal advice quoted by The Global Times, a Chinese state-affiliated media outlet, stated that under Chinese law, the ‘implied consent’ given by a user to social media makes recording this information legal.

This statement alone is fundamentally problematic for those of us interested in the development and preservation of personal privacy. Legal advice sought by Internet 2.0 during the investigation found that this data used in China in this way likely violates a range of national privacy laws in the jurisdictions of the users.

In the past, employees and individuals alike have had a laissez faire approach to internet privacy. Most users rely solely on the belief that their information and data will be protected by service providers and online security teams. What this leak demonstrates is that there is a misunderstanding as to what hostile actors deem as valuable information.

Our online activity and the data it produces is more valuable to companies and governments than ever, and protecting ourselves online is a responsibility that begins with the user. There is no amount of cybersecurity technology that can guard an individual or business from the collection of data they have previously authorised to be sold or used by advertisers. We must make wholesale change to our online habits and better understand the potential consequences of the material that we share with others.

“The reason Cambridge Analytica was scandalous wasn’t because they were accessing information on people’s private messages on Facebook. It was because they were misusing the permissions that were given by users to those platforms.”

Robert Potter, Co-CEO of Internet 2.0, to the Guardian.

It is clear that some of this information acquired by Zhenhua Data did not come from the public domain. It is vital that as a democracy we no longer send the message that behaviour like this is acceptable, and the easiest way to act is by adding further protections and safeguards against this malicious behaviour in the future. Laws are important, but laws are not effective without enforcement.

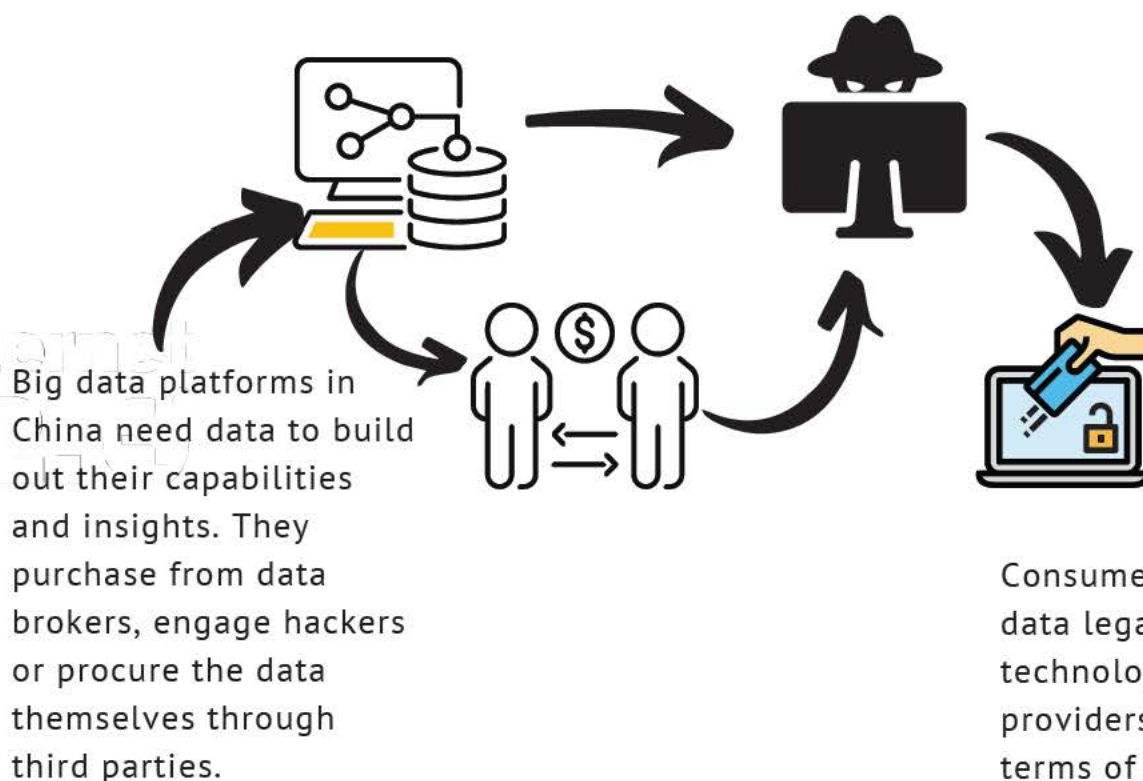
It is a reminder to be aware of the potential for our information and online presence to have a much further reach than we expect. The audience you get isn't always the one you expect. The hacking and data broker market for your personal information exists because there is market demand.

ILLEGAL DATA MARKET

This case study is important because it is a window to an illegal data market. Internet 2.0 was only able to restore 10% of the database so our statistics may be less severe than reality. But the list of Zhenhua Data's partners and their capabilities demonstrates the big data industry that is growing in China. In the end there are market forces inherently built into this story. The complex question is which forces are driving market demand.

SUPPLY & DEMAND

Corporations with rich consumer data, specifically personal information, are a high value targets for cyber crime.



MILITARY OUTPUTS AND CAPABILITIES

Understanding the wider possible military capability outputs for China, in this specific case study, is also important. Big Data Platforms have numerous applications and it must be noted that for these platforms are just a snapshot in time of the wider aspirations of a growing capability.

It appears the data held by Zhenhua Data is a compilation of personal, military and organisational data that was utilised to derive the following information and capability outputs: In-real time tracking of military platforms, financial reports and ownership structures of key entities and the ability to map the architecture of a national decision making network.



ZHENHUA DATA PARTNERS

By analysing the partner network of Zhenhua Data we can observe a larger networked industry with many possible implications for the extent of other data collected and military capability outputs that exist in China.



Taken from Zhenhua Data website



internet2.0

MILITARY-GRADE

CYBER PROTECTION

Influence Botnets: 2021 Myanmar Coup

Author

David Robinson

Table of Contents

Executive Summary	1
Data Collection and Intent	1
Introduction.....	2
Influence Botnet Indicators.....	5
User Names, User IDs, and Comment Generation	5
Algorithmic Nature	8
Character Perfect Messaging.....	9
Research Limitations and Implications.....	11

Executive Summary

US Army's Facebook page comment section was the target of a sophisticated Influence Botnet attack from 16 to 17 February 2021, with over 3000 comments per minute (exponential algorithmic growth), calling for US military intervention into Myanmar in support of "President Aung San Suu Kyi" and "Pro-Democracy Forces", who were imprisoned by the Myanmar military after a recent landslide election victory. The Influence Botnet used 337,463 Sender User Ids and 203,032 Sender Screen Names to post images/videos (436,247; 57%), and text (327,845; 43%), overwhelming the US Army's Facebook page comment section. All 764,092 comments were classified as authentic behavior (not bots) by Facebook and the top ten USER IDs are still active today on Facebook (some accounts are private to the public). Only a handful of entities and individuals in the world maintain and operate Influence Botnets. In this paper we have chosen to make no attribution assessment due to the limited forensic evidence available at this time. Research and experts believe this capability will soon be indistinguishable from authentic behavior online and will be used as a tool of massive influence. Further research and training in this field is needed.

Data Collection and Intent

This study used data collected from commercially available social media data of the US Army's Facebook Page comment section (<https://www.facebook.com/USArmy>) from 0500 13 February 2021 to 0400 18 February 2021. Variables of Interest: Created Time (time of post), Message (message content), Sender User Id (unique social media message sender user id), Sender Screen Name (social media message sender screen name), and Bot (message detected as spam or not spam). The dataset is composed of 764,092 observations with 15 variables. The opinions and facts in this report are those of the authors only and nothing in this report is the official position of the US Military. It is the target in this public example of an attempted influence operation. We believe this is a good example to learn about such capabilities and to generate discussion on effective countermeasures.

Introduction

On 01 February 2021, the Myanmar military seized power and detained recently elected Aung San Suu Kyi, President Win Myint and other senior officials – declaring a “one-year state of emergency”. Protests erupted across Myanmar leading to civil unrest, with the junta ordering

Twitter, Instagram and eventually the internet shutdown. On 11 February the United States imposed sanctions on Myanmar’s acting president and several other military officials as nationwide pro-democracy demonstrations continued across Myanmar. On 13 February the military ordered the arrest of key leaders in the pro-democracy protests. The civil disobedience movement spread, and police continued to escalate crackdowns on the protestors. On 15 February the US Army’s Facebook page experienced the first spike in their comments section related to the Myanmar Coup. 24 hours later, on 16 February the US Army’s Facebook page comment section was targeted by hundreds of thousands of comments, posted by hundreds of thousands of users. This report is an exploratory analysis into the tactics, techniques, and procedures of the social media activity on the US Army’s Facebook page from 15 February to 17 February.

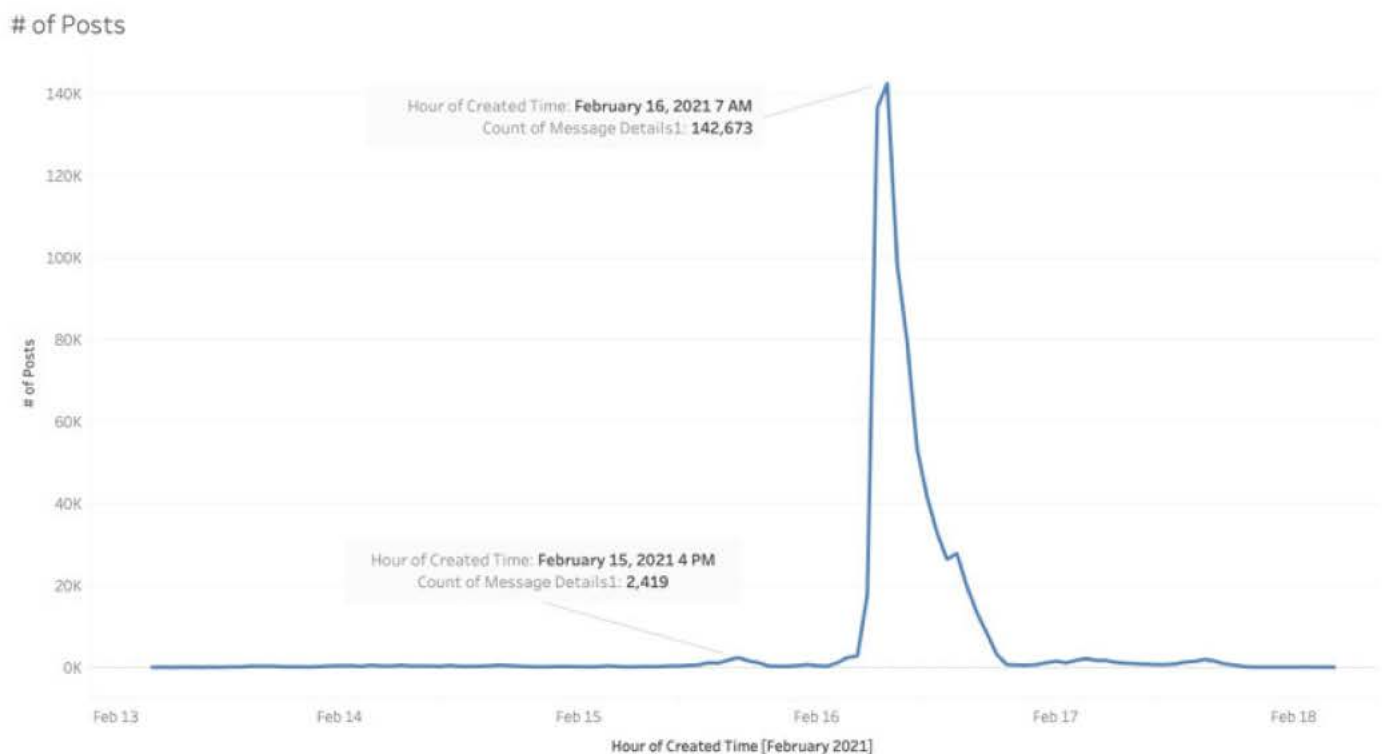


Figure 1. US Army’s Facebook page comment section activity.

Figure 1 depicts the comment section activity on the US Army's Facebook page. The comment section activity initially started around 1600 on February 15 with comments peaking at 2,419 (per hour) only to subside and return on February 16 the following day, where the comments peaked at 142,673 (per hour) at 0700. Below are screen captures depicting the comment section activity on the US Army's Facebook page.

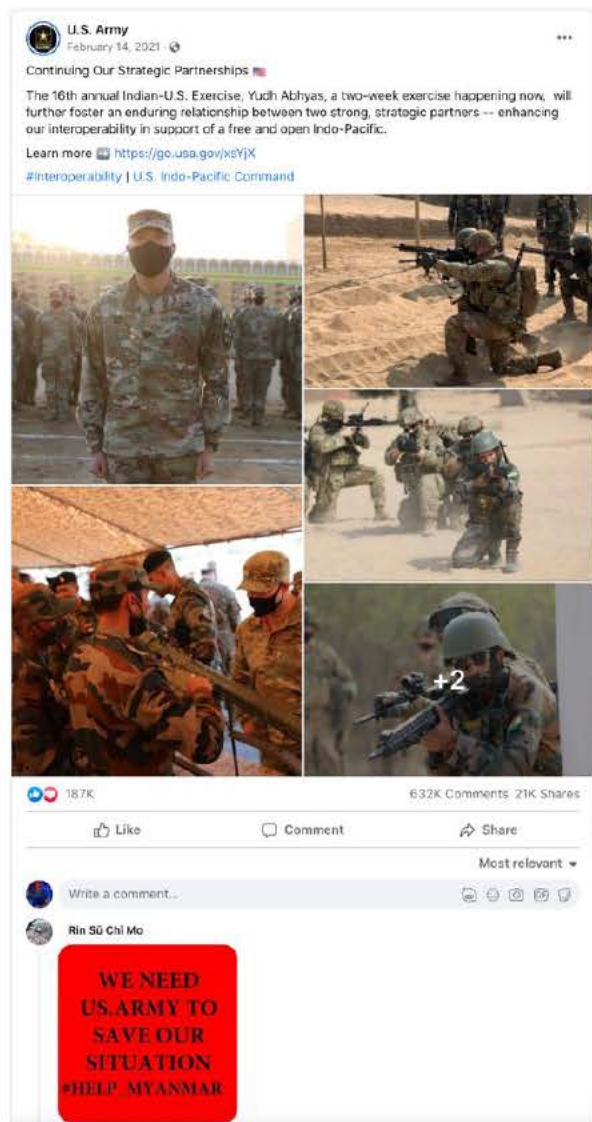


Figure 2. US Army's Facebook page comment section activity.

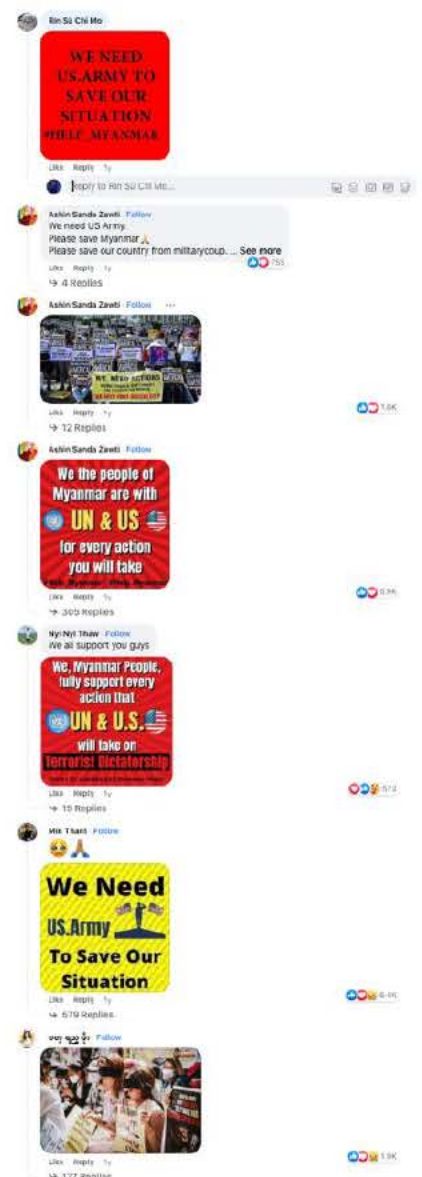


Figure 3. US Army's Facebook page comment section activity.

Images, videos, and text messages in the comment section stated: “WE NEED US.ARMY TO SAVE OUR SITUATION #HELP_MYANMAR”, “WE NEED US.ARMY To Save Our Situation”, “We, Myanmar People, fully support every action that UN and US take on Terrorist Dictatorship”, and “We need US Army. Please save Myanmar? Please save our country from militarycoup. Welcome for your support #RejectMilitaryCoup”. Below is a depiction of the Message variable overlayed by number of posts across time. Notably, the “NULL” message significantly dominates all other Message types. This is because the “NULL” input represents an image or video file that was posted without a text comment.

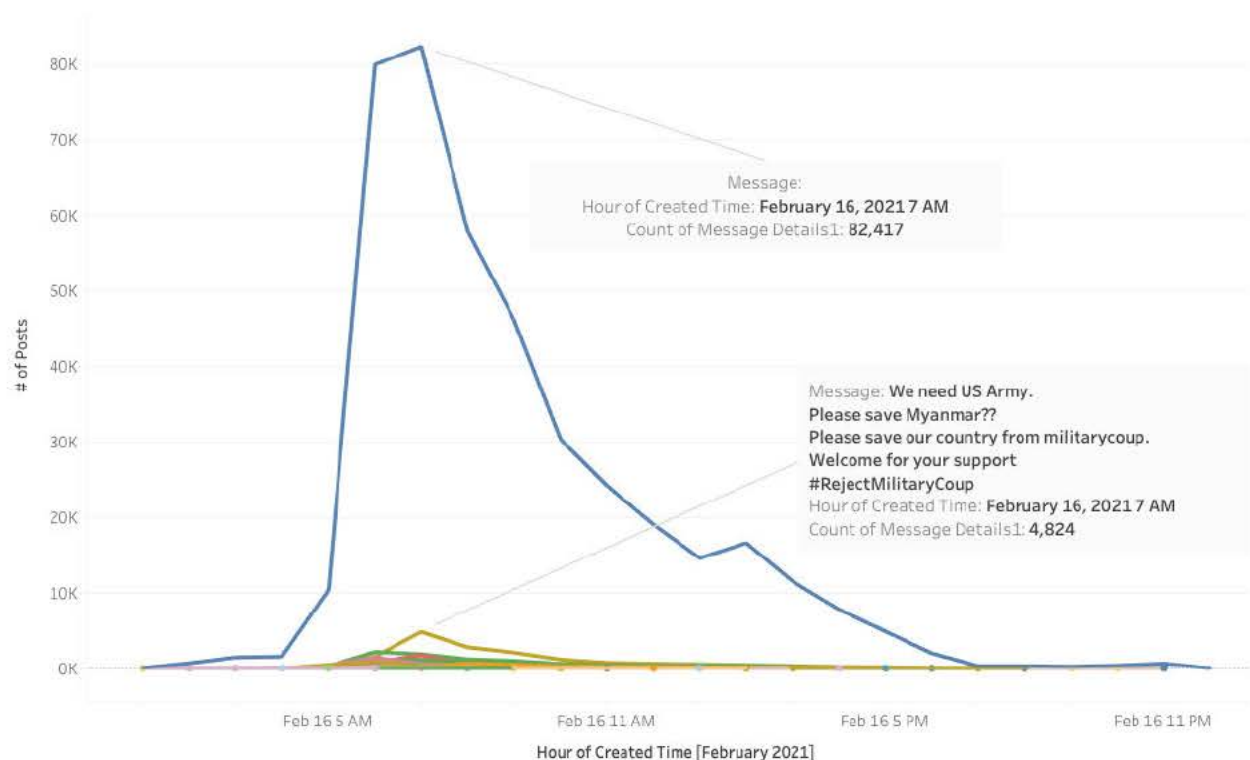


Figure 4. US Army's Facebook page comment section activity by # of Messages across Time.

In the graphic above, at the peak of the spike, on 0700 February 16, 2021, the count was 82,417 comments (per hour) and the Message was blank (NULL). Under the blue spike the count of the next most posted comment (4,824 comments per hour) was, “We need US Army. Please save Myanmar?? Please save our country from militarycoup. Welcome for your support #RejectMilitaryCoup.” The comment section activity is unique and possibly the activity of an Influence Botnet. In the next section, we will identify and describe Influence Botnet Indicators. This is not the first time Myanmar has been associated with “inauthentic behavior” on Facebook. Reuters reported on the issue on 6 November 2020: “Facebook said it had taken down a network of 36 accounts and six pages run by a Myanmar public relations agency, Openmind,

because it said they were using fictitious people to promote the military-backed Union Solidarity and Development Party (USDP).”¹

Influence Botnet Indicators

User Names, User IDs, and Comment Generation

Bot detection is not based on just one indicator, rather it’s the combination of indicators that provides a degree of confidence to make the assessment on botnet identification. In the visualizations below, we overlayed the variables, Sender User ID, Sender Screen Names and Message to better understand the comment section activity. In Figure 5, the blue color represents “NULL”, and all other colors represent various types of other text messages.

Total # of Posts by Sender User ID and Message

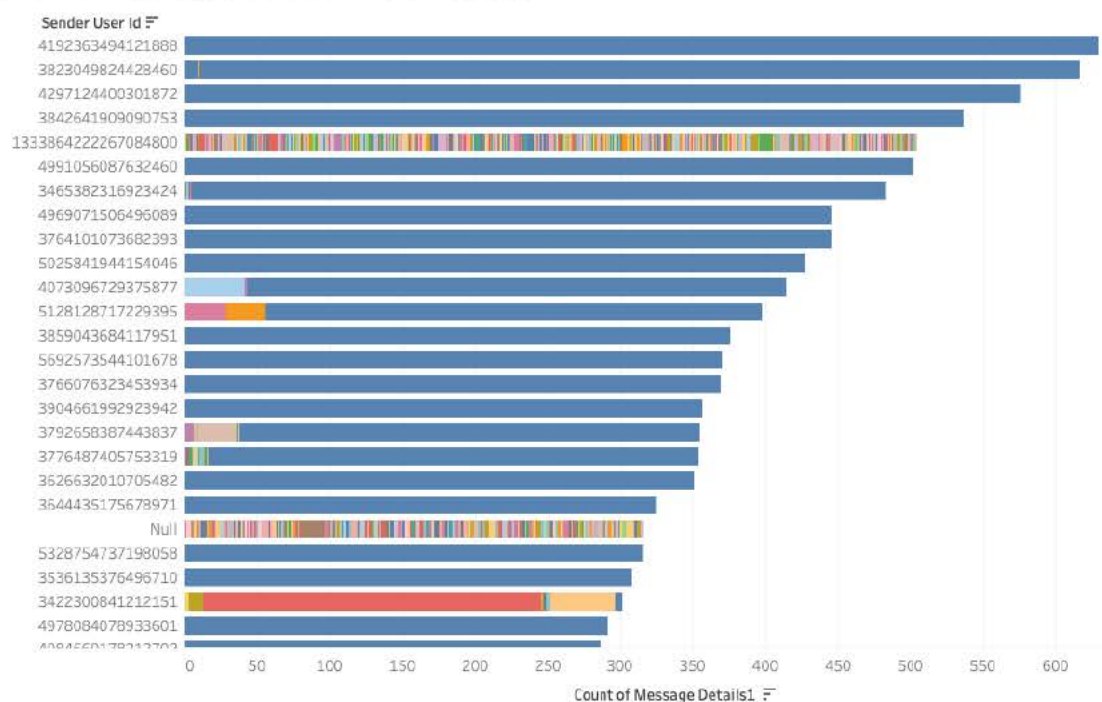


Figure 5. Total number of Posts by Sender User Id and Message.

The top Sender User Ids are as follows: 4192363494121888, 3823049824428460, 4297124400301872, 3842641909090753, 1333864222267084800, and 4991056087632460.

¹ <https://www.reuters.com/article/us-facebook-myanmar/facebook-shuts-dozens-of-myanmar-pages-over-inauthentic-behaviour-idUSKBN27M1EH>

The top Sender User Names are as follows: Wai Yan, Aung Aung, Ko Ko, Htet Htet, Kyaw Kyaw, Min Min, Min Khant, Kaung Kaung, Zaw Zaw, Nilar Soe Aung, and Jolly Jolly.

Sender User ID, Sender Screen Name, and # of Posts



Figure 6. Total number of posts by Sender User Id, and Sender Screen Name.

Figure 6 on the previous page, clearly shows Sender User IDs were using the same exact Sender Screen Name to possibly appear as the same Facebook User. It is logical that as Facebook blocked these User IDs for spamming the US Army comment section, the Botnet created new accounts with the same Sender Screen Names to appear as the same user. Because Bot accounts are often suspended by Facebook, these accounts are often newly created, low in followers, and lacking a complete profile: all common bot indicators.

The social media activity appears to be a combination of actual user accounts and influence bot accounts, acting in coordination to manipulate Facebook's engagement algorithm and bypass its spam countermeasures. For example, in Figure 6, 395 unique Sender User Ids posted 1,311 comments under the Sender Screen Name, 'Wai Yan'; 508 unique Sender User Ids posted 981 comments under the Sender Screen Name, 'Aung Aung'; and 447 unique Sender User Ids posted 966 comments under the Sender Screen Name, 'Ko Ko'. However, Sender Screen Names 'Nilar Soe Aung', 'Jolly Jolly', 'Kyaw Zaw Hlaing', 'NayAung61146884', and 'Sai Lin K' posted all of their comments under their own unique Sender User Id. Of the top ten Sender User Id accounts by number of comments posted, all are still live and active on Facebook and Twitter.

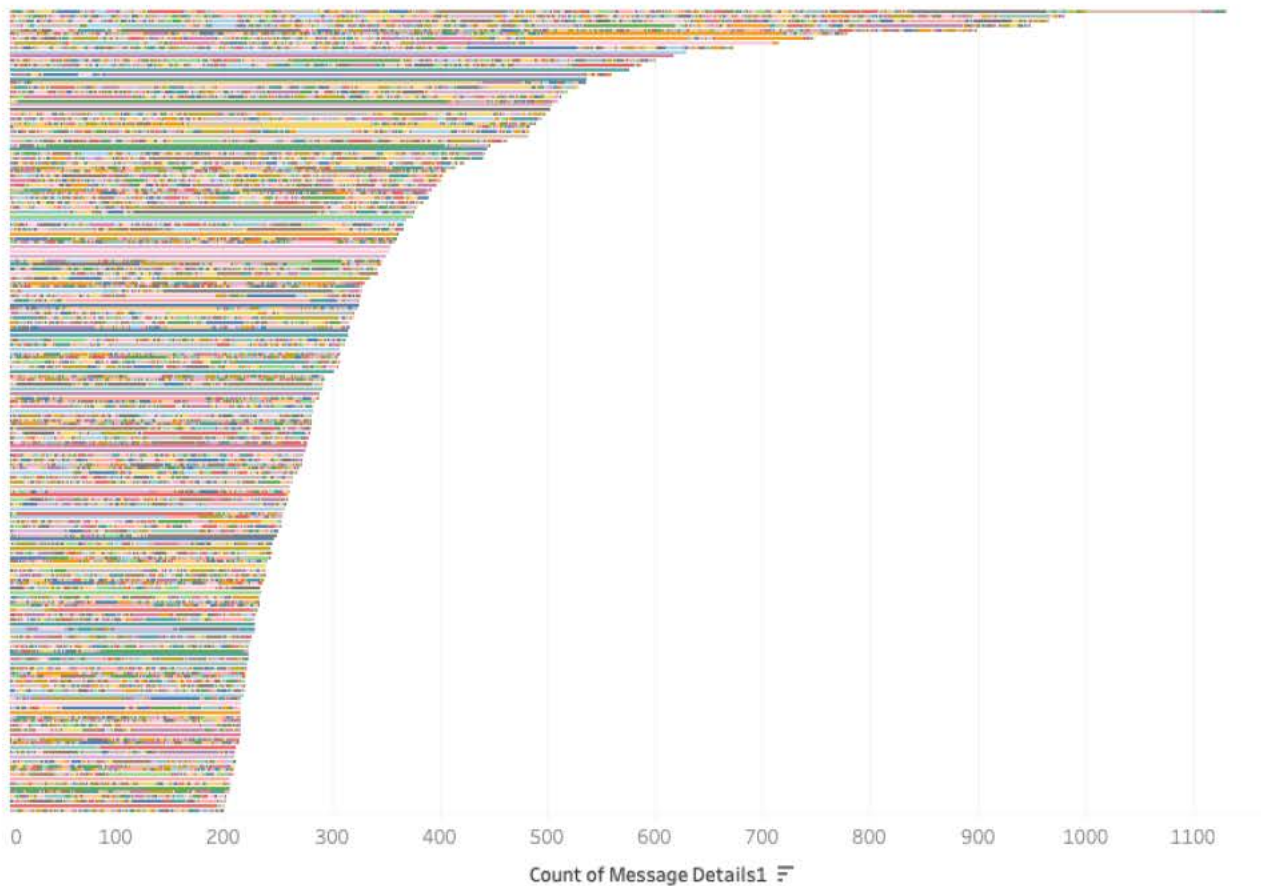


Figure 7. Number of posts by Sender Screen Name and Sender User Id.

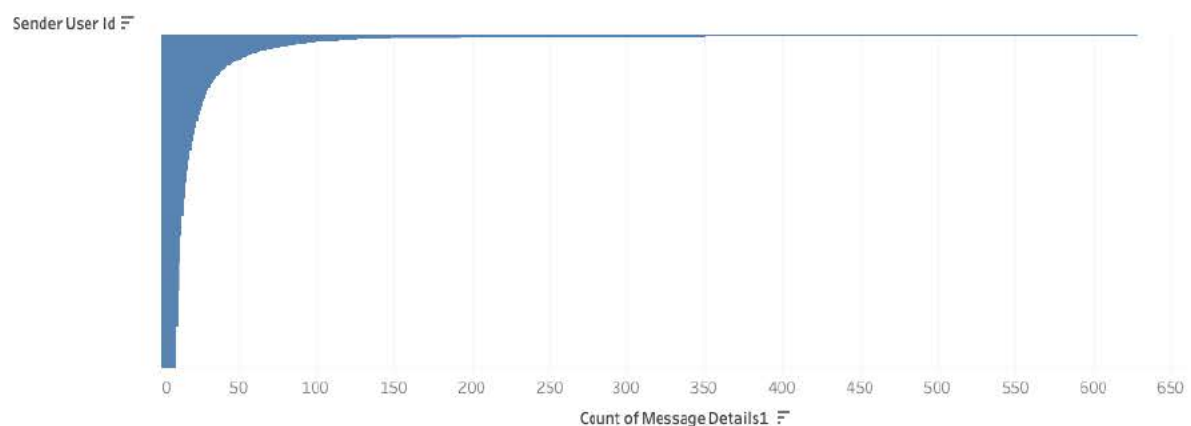
Above, Figure 7 colorfully demonstrates the software generating User IDs against a scripted number of Sender Names, with a scripted number of unique Messages, that was possibly generated using something similar to OpenAI's GPT-3. GPT-3 (Generative Pre-trained Transformer 3) is an autoregressive language model that uses deep learning to produce human-like text.² GPT-3 is an example of the type of capability required to create an influence botnet. In this report we have not made any attribution assessment due to the limited digital forensics information available.

² <https://openai.com/>

Algorithmic Nature

The Influence Botnet built engagement that is in our assessment inauthentic behavior. The method in which it was built is one of the indicators that contribute to our assessment that this is inauthentic. Firstly, the number of posts per user was algorithmic. Figure 8 is a side-by-side comparison of the total comments made by Sender Screen Names and Sender User IDs. The highest message count by user is 629 and the highest message count by username is 1130. Both numbers are outside of the standard range for authentic human behavior. 629 unique comments are a lot of comments to copy and paste into a single post manually. Of the largest user count the account at some points was creating a comment every 3 seconds.

Total # of Posts by Sender User ID



Total # of Posts by Sender Screen Name

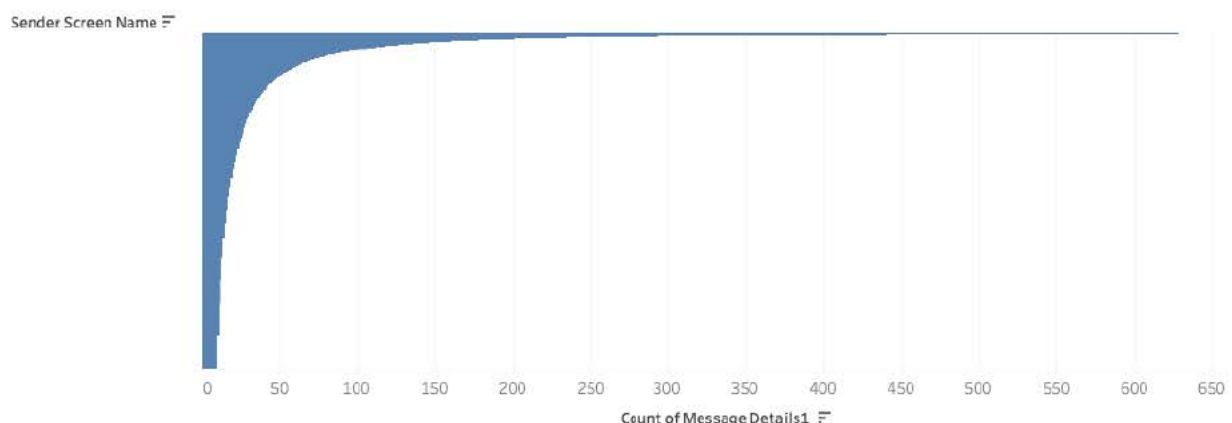


Figure 8. Distribution of Sender Screen Name and Sender User Id by Message count.

Another feature is that every single account we marked as inauthentic at the time of the content release on 16 February 2021 had zero followers while they were commenting on the Myanmar Coup. This showed that the accounts were probably made for this specific campaign. A

sophisticated feature of this capability, which gives us insight into the source code, is the foundational software characteristic of a mathematical algorithm that takes an exponential curve in its deployment of content. Genuine authentic behavior has a limiting factor defined by human limitations. For this reason, authentic behavior would be limited to the number of posts a user could create manually. Authentic behavior is generally seen as an organic growth curve or a logarithmic curve in a dataset. This is because these curves have a ceiling determined by human characteristics in this case. When we analyze the dataset we see an algorithm that deploys content in an exponential way. This is true for how it created usernames; how it manipulated usernames amongst multiple user identities to defeat Facebook's defences; and how it generated character similar content. The software is trying to mimic an authentic viral event but is immature. Figure 9 is an example of the concept of exponential curves compared to logarithmic curves. In this case the exponential curve is seen in Figures 1 when deploying content over time; in figure 7 when generating names across multiple account identities and in figure 8 when deploying similar character content.

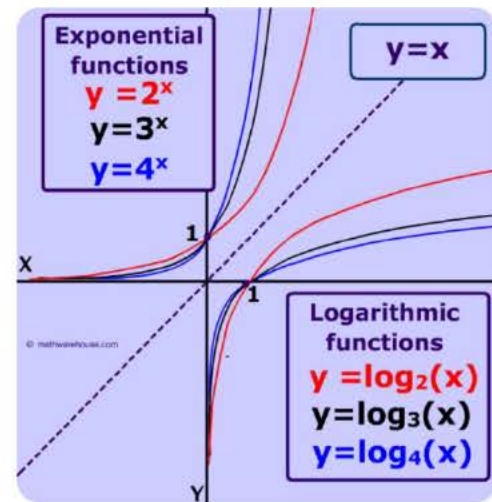


Figure 9. Theoretical curves

Character Perfect Messaging

The Messages in the comment section, along with the rate of posting strongly indicate bot activity. Figure 10 illustrates that 57% of the posts (436,247) were image or video posts without an accompanying text message. The top text post in the comment section was, "We need US Army. Please save Myanmar?? Please save our country from militarycoup. Welcome for your support #Reject Military Coup" This comment was posted 15,621 times as seen in Figure 11. Of note many of the messages had grammar and spacing errors and all were replicated thousands of times. Interesting to note the 5th, 7th and 8th highest message by count was nearly identical except for variations of the number and placement of the "??" character. This limitation of variety of content the software could produce is what we based part our assessment on in that it is what we believe to be inauthentic behavior. The replication of messages that are all character perfect (even the grammatical and spacing errors were exactly the same) and posted at an exponential rate until the attack ceased strongly indicates bot activity. As opposed to a viral growth explanation this feature indicates an influence botnet.

Comment Section Messages

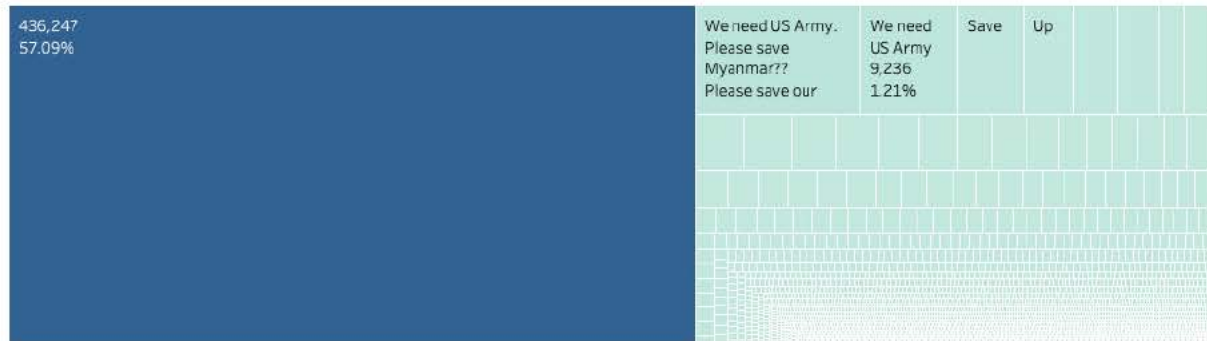


Figure 10. US Army Facebook page comment section activity, number of comments by Message.

In a recent article Harvard Professor Joseph Nye Jr. recognized the use of OpenAI’s GPT-3 for Influence Botnets stating, “Generative neural networks can also create new images or texts. In 2019, the company developed a language model that trains itself by consuming freely available texts from the internet.”³ In this case, the programmers could have easily provided a working script for the AI to generate message content. He continued, “Given a few words, it (GPT-3) can extrapolate new sentences and paragraphs by detecting patterns in sequential elements...displaying intelligent behavior indistinguishable from that of a human being.”⁴

³ <https://www.project-syndicate.org/onpoint/age-of-ai-and-our-human-future-review-kissinger-schmidt-by-joseph-s-nye-2021-11?barrier=accesspaylog>

⁴ <https://www.project-syndicate.org/onpoint/age-of-ai-and-our-human-future-review-kissinger-schmidt-by-joseph-s-nye-2021-11?barrier=accesspaylog>

Count of Unique Messages		
Message		
We need US Army. Please save Myanmar?? Please save our country from militarycoup. Welcome for your support #RejectMilitaryCoup	15,576	Nobel prize winner, our national leader AungSanSukyi was detained and put under house arrest by the Military. We are against the military dictatorship which ignore human rights. We citizens, who are uncertain whether the internet will be cut off again but we are still fighting for Democracy even if we disappear on social media. #CivilDisobedienceMovement .. 2,328
Please, Save our leader Daw Aung San Su Kyi and Our President U Win Myint We Want justice We want Democracy the leaders of China & Russia support the military diktatorship and are evil neighbors who are destroying Democracy we want UN and US Action urgent Help US Army #savemyanmar	4,059	We need US Army. Please save Myanmar???? Please save our country from militarycoup. Welcome for your support #RejectMilitaryCoup 2,228
Up	3,972	We need US Army. Please save Myanmar Please save our country from militarycoup. Welcome for your support #RejectMilitaryCoup 1,927
We need US Army	3,949	Where is our Democracy? Where is our Freedom? Where is justice for Us? 1,809
We need US Army. Please save Myanmar???? Please save our country from militarycoup. ?? Welcome for your support #RejectMilitaryCoup	3,930	#WhatsHappeningInMyanmar #JusticeForMyanmar #WhereIsHumanity #Feb16Coup
Save Myanmar	2,983	We need help please helps us #Savemyabmar 1,627
		We need US army 1,543
		We need you, US Army! We all citizens support for every action you will take! Please help us! 1,429

Figure 11. Count of Unique Messages

Research Limitations and Implications

This influence botnet is impossible for us to attribute due to the lack of forensic information available. We have chosen not to make any hypothesis on attribution in this paper. It is hard to assess intent on the actor behind this inauthentic event without attribution. The influence botnet also used many emoji, gif and other visual media in the comments and messages. We were unable to provide specific detailed big data analysis on the trends to these visual social media messages due to the restrictions in the dataset and the data ingest process. The standard assessments on who is behind these types of capabilities focus on either influence operations or advertising fraud. For this reason, this topic should be of significant interest to both the military in the fact they were the target as part of the US Government but also the Department of Justice. As these capabilities increase in sophistication the ease for actors to perpetrate advertising fraud is likely to increase. This is a governance and compliance risk for any social media platform.



This cyber weapon was built to sway discourse and influence the information environment. It was a huge number of bots pulled together to push this message in comments across the US Military's social pages. It is backed by software which we believe is in its infancy and when matured will become a serious threat to democratic speech online. This threat will have capability to drown out genuine political speech and make most forms of media redundant. It is a threat to both democratically established institutions and politicians as it could sway the public discourse by amplifying a voice in the majority. This type of capability can be wielded by an individual or an autocracy with the intent to harm democracy.

It is important to be aware of such capabilities because in our opinion the strategic advantage that controlling the dominant narrative has in military operations has only been demonstrated yet again in the conflict between Russia and Ukraine. Often cyber weapons have been pigeonholed in terms of intelligence collection or their technical characteristic as a "Zero Day". As we have experienced during disinformation campaigns over the Covid-19 pandemic and now the Russian invasion of Ukraine is that in our assessment a capable influence botnet is in fact strategically equal to the most dangerous categorized cyber weapons reserved for malware designed to wreck a catastrophic disruption on a critical infrastructure target. This is because high morale is a prerequisite for any victory in conflict and the influence botnet is designed to target morale. In conclusion in our opinion it is important to track and identify such capabilities. Collectively we must strive to share identifying features of such capabilities as they become more prevalent in use.

internet20

MILITARY-GRADE

CYBER PROTECTION

Australia

Level 1

18 National Circuit

Barton ACT 2600

ABN: 17 632 726

United States

Suite 100

211 N Union St

Alexandria 22314

EIN: 86-1567068

contact@internet2-0.com

internet2.0

MILITARY-GRADE

CYBER PROTECTION

TikTok Analysis

Author

Thomas Perkins

Editors

David Robinson

Michael Lammбраu, Ph.D

Robert Potter

Table of Contents

Executive Summary	1
Introduction.....	2
User Permissions and Third-Party Data Access.....	3
Device and user data harvesting	8
IOS connects to mainland China.....	13
Conclusion	14

Executive Summary

This report is a technical analysis of the source code of TikTok mobile applications Android 25.1.3 as well as IOS 25.1.1. Analysis of the Android application was performed using a Galaxy S9 cell phone. Internet 2.0 conducted static and dynamic analysis of the source code between 01-12 July 2022. This report aims to analyse TikTok device and user (customer) data collection. Prepared by Internet 2.0, this report is for policy makers and legislators to make evidence-based decisions. TikTok is a dominant social media application and is the 6th most used application globally with forecasted advertising revenues in 2022 expected to be USD12 billion. In our analysis the TikTok mobile application does not prioritise privacy. Permissions and device information collection are overly intrusive and not necessary for the application to function. The following are examples of excessive data harvesting.

- **Device Mapping.** The application retrieves all other running applications on the phone. TikTok also gathers all applications that are installed on the phone. In theory this information can provide a realistic diagram of your phone.
- **Location.** TikTok checks the device location at least once per hour.
- **Calendar.** TikTok has persistent access to the calendar.
- **Contacts.** TikTok has access to contacts and if the user denies access, it continuously requests for access until the user gives access.
- **Device information.** TikTok has code that collects the following device detailed information on Android.
 - Wi-Fi SSID
 - Device build serial number
 - SIM serial number
 - Integrated Circuit Card Identification Number (this is global unique serial number that is specifically tailored to your SIM card)
 - Device IMEI
 - Device MAC address
 - Device line number

- Device voicemail number
- GPS status information (updates on the GPS location)
- Active subscription information
- All accounts on the device
- Complete access to read the clipboard (dangerous as password managers use clipboards)

Also of note is that TikTok IOS 25.1.1 has a server connection to mainland China which is run by a top 100 Chinese cyber security and data company Guizhou Baishan Cloud Technology Co., Ltd.

Introduction

TikTok is currently one of the dominant social media application in the market. It is the 6th most used application. As at September 2021 TikTok has over 1 billion active users globally with 142.2 million users in North America.¹ It has been downloaded over 3.5 billion times as of January 2021, with 43.7% of users 18-24 years old and 31.9% 25 to 34 years old. TikTok's projected advertising annual revenue in 2022 will hit USD12 billion, up from USD1.41 billion in 2020.²

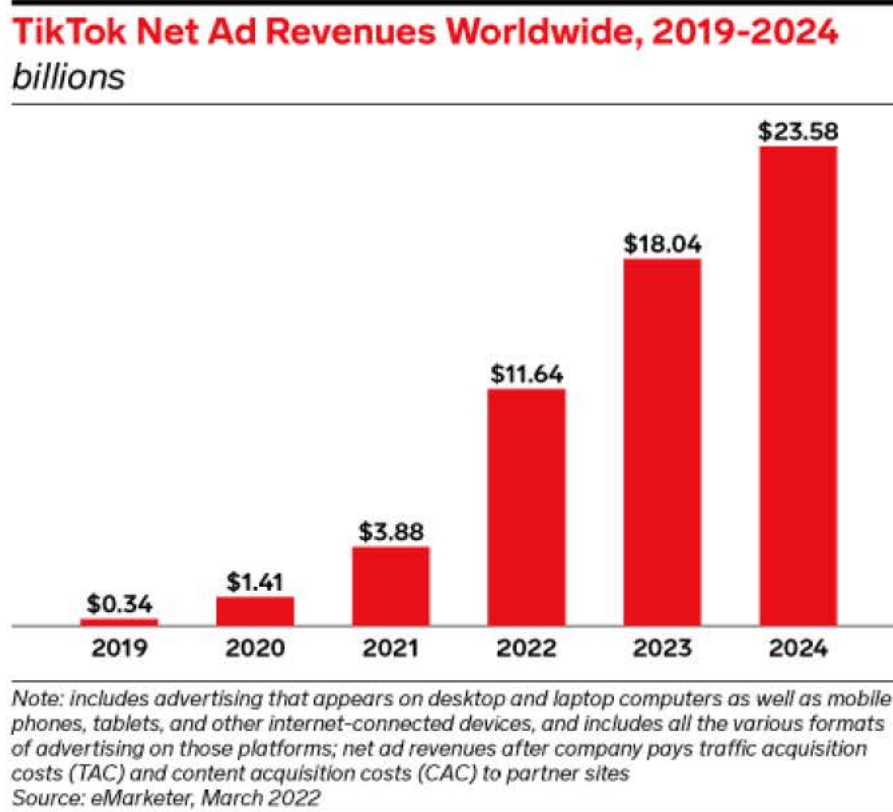


Figure 1. Projected TikTok advertising revenue (see footnote 2)

¹ <https://www.shopify.com/blog/tiktok-statistics>

² <https://www.insiderintelligence.com/content/tiktok-douyin-digital-ad-spend>

Internet 2.0 conducted static and dynamic analysis of the TikTok mobile application Android 25.1.3 as well as static analysis of the TikTok mobile application IOS 25.1.1 to understand user and device data collection.³ The analysis also seeks to confirm the existence of any malicious code or features of the application. We decompiled the source code of the application available on the app stores and analysed it through multiple systems (including multiple sandbox services) and manual source code reviews. This is divided into the following sections: user permissions and third-party data access; device and user data harvesting; and conclusion.

User Permissions and Third-Party Data Access

There are certain permissions that the Android documentation considers to be “dangerous”. They are considered dangerous due to the permission providing additional access to restricted data. For example, the ability to read all SMS messages could be considered dangerous because an application could send all your texts to a server and save the information for future use (such as a malware). Unfortunately, TikTok makes use of a lot of these dangerous permissions. We noted the Android version had many more than the IOS version. IOS has a justification system where to gain a permission the developer must justify why this permission is required before it is granted. We believe the justification system IOS implements systematically limits a culture of “grab what you can” in data harvesting. The fact that TikTok had far more permissions for Android over IOS is a good demonstration of their culture when it comes to privacy.

android.permission.WRITE_EXTERNAL_STORAGE	dangerous	read/modify/delete external storage contents	Allows an application to write to external storage.
android.permission.ACCESS_COARSE_LOCATION	dangerous	coarse (network-based) location	Access coarse location sources, such as the mobile network database, to determine an approximate phone location, where available. Malicious applications can use this to determine approximately where you are.
android.permission.AUTHENTICATE_ACCOUNTS	dangerous	act as an account authenticator	Allows an application to use the account authenticator capabilities of the Account Manager, including creating accounts as well as obtaining and setting their passwords.
android.permission.CAMERA	dangerous	take pictures and videos	Allows application to take pictures and videos with the camera. This allows the application to collect images that the camera is seeing at any time.
android.permission.GET_TASKS	dangerous	retrieve running applications	Allows application to retrieve information about currently and recently running tasks. May allow malicious applications to discover private information about other applications.
android.permission.READ_CALENDAR	dangerous	read calendar events	Allows an application to read all of the calendar events stored on your phone. Malicious applications can use this to send your calendar events to other people.
android.permission.READ_CONTACTS	dangerous	read contact data	Allows an application to read all of the contact (address) data stored on your phone. Malicious applications can use this to send your data to other people.
android.permission.READ_EXTERNAL_STORAGE	dangerous	read external storage contents	Allows an application to read from external storage.
android.permission.RECORD_AUDIO	dangerous	record audio	Allows application to access the audio record path.
android.permission.SYSTEM_ALERT_WINDOW	dangerous	display system-level alerts	Allows an application to show system-alert windows. Malicious applications can take over the entire screen of the phone.
android.permission.WRITE_CALENDAR	dangerous	add or modify calendar events and send emails to guests	Allows an application to add or change the events on your calendar, which may send emails to guests. Malicious applications can use this to erase or modify your calendar events or to send emails to guests.

Figure 2a. TikTok Android access permissions rated as dangerous.

³ This analysis provides impartial advice for users to evaluate the extent to which their data is collected for privacy reasons. It allows policy advisors and legislators to make evidence-based decisions when discussing privacy concerns with vendors. This report was written for a global audience and does not include any legal or jurisdiction based regional assessments.

PERMISSIONS	STATUS	INFO	REASON IN MANIFEST
NSAppleMusicUsageDescription	dangerous	Access Apple Media Library.	To select sound from your library, allow us to access your Apple Music.
NSCalendarsUsageDescription	dangerous	Access Calendars.	Add events to your device calendar to get reminders when events start.
NSCameraUsageDescription	dangerous	Access the Camera.	You'll be able to record video.
NSContactsUsageDescription	dangerous	Access Contacts.	Sync your contacts to easily find people you know on TikTok. Your contacts will only be used to help you connect with friends.
NSLocationWhenInUseUsageDescription	dangerous	Access location information when app is in the foreground.	If location services are enabled, TikTok will collect, store and use your device's approximate location to improve your experience.
NSMicrophoneUsageDescription	dangerous	Access microphone.	To record audio, allow us to access your Microphone.
NSPhotoLibraryUsageDescription	dangerous	Access the user's photo library.	To upload from or download to your device, allow access to your photos in your phone settings.

Figure 2b. TikTok IOS access permissions rated as dangerous.

Device mapping

The Android application collects all other running and installed applications on the phone (this is an unnecessary function), see figure 3. Theoretically, this information can provide a realistic diagram of your phone.

```

L1212: .append(100909, new a(100909, "android.telephony.TelephonyManager", "getAllCellInfo", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION", 0, C17260jk.L1212("location")}));
L1212: .append(100910, new a(100910, "android.telephony.TelephonyManager", "requestCellInfoUpdate", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION", 0, C17260jk.L1212("location")}));
L1212: .append(100911, new a(100911, "android.telephony.PhoneStateListener", "onCellLocationChanged", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION", 0, C17260jk.L1212("location")}));
L1212: .append(100912, new a(100912, "android.telephony.PhoneStateListener", "onCellInfoChanged", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION", 0, C17260jk.L1212("location")}));
L1212: .append(100908, new a(100908, "android.net.wifi.WifiInfo", "getSSID", "", "", new String[0], 0, C27831lx.L1212("location", "wifi", "device_info")));
L1212: .append(100901, new a(100901, "android.net.wifi.WifiManager", "getConfiguredNetworks", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION", "android.permission.ACCESS_WIFI_STATE", 0, C27831lx.L1212("location", "wifi")}));
L1212: .append(101208, new a(101208, "android.net.wifi.WifiInfo", "getSSID", "", "", new String[0], 0, C27831lx.L1212("location", "wifi", "device_info")));
L1212: .append(101209, new a(101209, "android.os.Battery", "getLevel", "", "", new String[]{"android.permission.READ_PHONE_STATE", 0, C17260jk.L1212("device_info")}));
L1212: .append(101208, new a(101208, "android.app.ActivityManager", "getRunningTasks", "", "", new String[0], 0, C17260jk.L1212("application")));
L1212: .append(101201, new a(101201, "android.app.ActivityManager", "getRunningServices", "", "", new String[0], 0, C17260jk.L1212("application")));
L1212: .append(101204, new a(101204, "android.content.pm.PackageManager", "getInstalledApplications", "", "", new String[0], 0, C17260jk.L1212("application")));
L1212: .append(101205, new a(101205, "android.content.pm.PackageManager", "getInstalledApplications", "", "", new String[0], 0, C17260jk.L1212("application")));
L1212: .append(101206, new a(101206, "android.view.accessibility.AccessibilityManager", "getInstalledAccessibilityServiceList", "", "", new String[0], 0, C17260jk.L1212("application")));
L1212: .append(101207, new a(101207, "android.view.accessibility.AccessibilityManager", "getEnabledAccessibilityServiceList", "", "", new String[0], 0, C27731lx.INSTANCE));
L1212: .append(101208, new a(101208, "android.content.pm.PackageManager", "getInstalledPackagesAsUser", "", "", new String[0], 0, C17260jk.L1212("application")));
L1212: .append(101208, new a(101208, "android.content.pm.PackageManager", "getInstalledPackages", "", "", new String[0], 0, C17260jk.L1212("application")));
L1212: .append(101210, new a(101210, "android.content.pm.PackageManager", "getPackageForId", "", "", new String[0], 0, C17260jk.L1212("application")));
L1212: .append(101211, new a(101211, "android.content.pm.PackageManager", "queryIntentActivities", "", "", new String[0], 0, C17260jk.L1212("application")));
L1212: .append(101208, new a(101208, "android.telephony.TelephonyManager", "getSimSerialNumber", "", "", new String[]{"android.permission.READ_PHONE_STATE", 0, C27831lx.L1212("network", "device_info")}));
L1212: .append(101208, new a(101208, "android.telephony.SubscriptionInfo", "getIcdId", "", "", new String[]{"android.permission.READ_PHONE_STATE", 0, C27831lx.L1212("network", "device_info")}));
L1212: .append(101208, new a(101208, "android.telephony.TelephonyManager", "getDeviceId", "", "", new String[]{"android.permission.READ_PHONE_STATE", 0, C27831lx.L1212("network", "device_info")}));
L1212: .append(101201, new a(101201, "android.telephony.TelephonyManager", "getNetType", "", "", new String[]{"android.permission.READ_PHONE_STATE", 0, C27831lx.L1212("network", "device_info")}));
L1212: .append(101202, new a(101202, "android.telephony.TelephonyManager", "getNetType", "", "", new String[]{"android.permission.READ_PHONE_STATE", 0, C27831lx.L1212("network", "device_info")}));
L1212: .append(101202, new a(101202, "android.net.wifi.WifiInfo", "getMacAddress", "", "", new String[0], 0, C27831lx.L1212("location", "wifi", "device_info")));
L1212: .append(101201, new a(101201, "java.net.NetworkInterface", "getHardwareAddress", "", "", new String[0], 0, C17260jk.L1212("device_info")));
L1212: .append(101208, new a(101208, "android.content.ClipboardManager", "clearPrimaryClip", "", "", new String[0], 0, C17260jk.L1212("clipboard")));
L1212: .append(101201, new a(101201, "android.content.ClipboardManager", "addPrimaryClipChangeListener", "", "", new String[0], 0, C17260jk.L1212("clipboard")));
L1212: .append(101202, new a(101202, "android.content.ClipboardManager", "removePrimaryClipChangeListener", "", "", new String[0], 0, C17260jk.L1212("clipboard")));
L1212: .append(101203, new a(101203, "android.content.ClipboardManager", "getPrimaryClip", "", "", new String[0], 0, C17260jk.L1212("clipboard")));
L1212: .append(101204, new a(101204, "android.content.ClipboardManager", "getText", "", "", new String[0], 0, C17260jk.L1212("clipboard")));
L1212: .append(101205, new a(101205, "android.content.ClipboardManager", "hasPrimaryClip", "", "", new String[0], 0, C17260jk.L1212("clipboard")));
L1212: .append(101206, new a(101206, "android.content.ClipboardManager", "hasText", "", "", new String[0], 0, C17260jk.L1212("clipboard")));
L1212: .append(101207, new a(101207, "android.content.ClipboardManager", "setPrimaryClip", "", "", new String[0], 0, C17260jk.L1212("clipboard")));
L1212: .append(101208, new a(101208, "android.content.ClipboardManager", "setText", "", "", new String[0], 0, C17260jk.L1212("clipboard")));
L1212: .append(101209, new a(101209, "android.content.ClipboardManager", "getPrimaryClipDescription", "", "", new String[0], 0, C17260jk.L1212("clipboard")));
L1212: .append(101209, new a(101209, "android.telephony.TelephonyManager", "getSubscriberId", "", "", new String[]{"android.permission.READ_PHONE_STATE", 0, C27831lx.L1212("network", "device_info")}));
L1212: .append(102009, new a(102009, "android.telephony.TelephonyManager", "getLineNumbers", "", "", new String[]{"android.permission.READ_PHONE_STATE", "android.permission.READ_SMS", 1, C27831lx.L1212("network", "device_info")}));
L1212: .append(102091, new a(102091, "android.telephony.TelephonyManager", "getVoiceMailNumber", "", "", new String[]{"android.permission.READ_PHONE_STATE", 0, C27831lx.L1212("network", "device_info")}));

```

Figure 3: Get all applications and running tasks on the device (green highlight).

GPS and Locations requests

The Android application queries the device GPS location at least once per hour while running. This command is seen in figures 4 and 5.

```
public BDLocation(Location location) {
    super(location.getProvider());
    this.LJIIIJ = 2;
    this.LJIIIZ = "wgs84";
    LIZ(location);
    this.LJIIIJ = location.getLatitude();
    this.LJIIIIL = location.getLongitude();
    this.LJIII = location.getTime();
    this.LJIIIIJ = LIZ(location.getProvider());
    this.LJIIIJ = location.getBearing();
    this.LJIIIJ = location.getSpeed();
}
```

Figure 4: Get location code.

Queries the phones location (GPS)	
Source: com.bytedance.bdlocation.BDLocation;-><init>:6	API Call: android.location.Location.getLatitude
Source: com.bytedance.bdlocation.BDLocation;-><init>:7	API Call: android.location.Location.getLongitude
Source: com.bytedance.bdlocation.BDLocation;-><init>:34	API Call: com.bytedance.bdlocation.BDLocation.getLatitude
Source: com.bytedance.bdlocation.BDLocation;-><init>:36	API Call: com.bytedance.bdlocation.BDLocation.getLongitude
Source: com.bytedance.bdlocation.BDLocation;->LIZ:70	API Call: android.location.Location.getLatitude
Source: com.bytedance.bdlocation.BDLocation;->LIZ:72	API Call: android.location.Location.getLongitude

Figure 5: TikTok get longitude and latitude data requests.

Contacts

The Android application requests access to user contacts. If the user denies access the application will continuously ask for access. TikTok does this as it runs its code in a loop that if a Boolean (true or false) is stored as false, it will keep prompting until given a true value (see figure 6). It is normal for an application to initially request access to contacts but TikTok's persistent, endless harassment for user contacts access is abnormal. It reflects a culture that does not prioritize privacy or a user's preferences for privacy.

```
package X;

import com.bytedance.covode.number.Covode;
import com.bytedance.keva.Keva;
import kotlin.g.b.n;
/* loaded from: /mnt/c/Users/rando/bin/python/maltree/temp_files/extracted_apks/Qksfgib0ovyBEyNPuFP5JHACrokQZd/classes4.dex */
public final class AW7 {
    public static final AW7 LIZ = new AW7();

    static {
        Covode.recordClassIndex(28546);
    }

    private boolean LIZ() {
        return Keva.getRepo("FriendsSharePreferences").getBoolean("read_contact_denied", false);
    }

    private void LIZIZ() {
        Keva.getRepo("FriendsSharePreferences").storeBoolean("read_contact_denied", true);
    }

    public final boolean LIZ(String str) {
        0HG.LIZ(str);
        if (n.LIZ(str, "android.permission.READ_CONTACTS")) {
            return LIZ();
        }
        return Keva.getRepo("permission_store").getBoolean(str, false);
    }

    public final void LIZIZ(String str) {
        0HG.LIZ(str);
        if (n.LIZ(str, "android.permission.READ_CONTACTS")) {
            LIZIZ();
        } else {
            Keva.getRepo("permission_store").storeBoolean(str, true);
        }
    }
}
```

Figure 6: The source code for Contacts information.

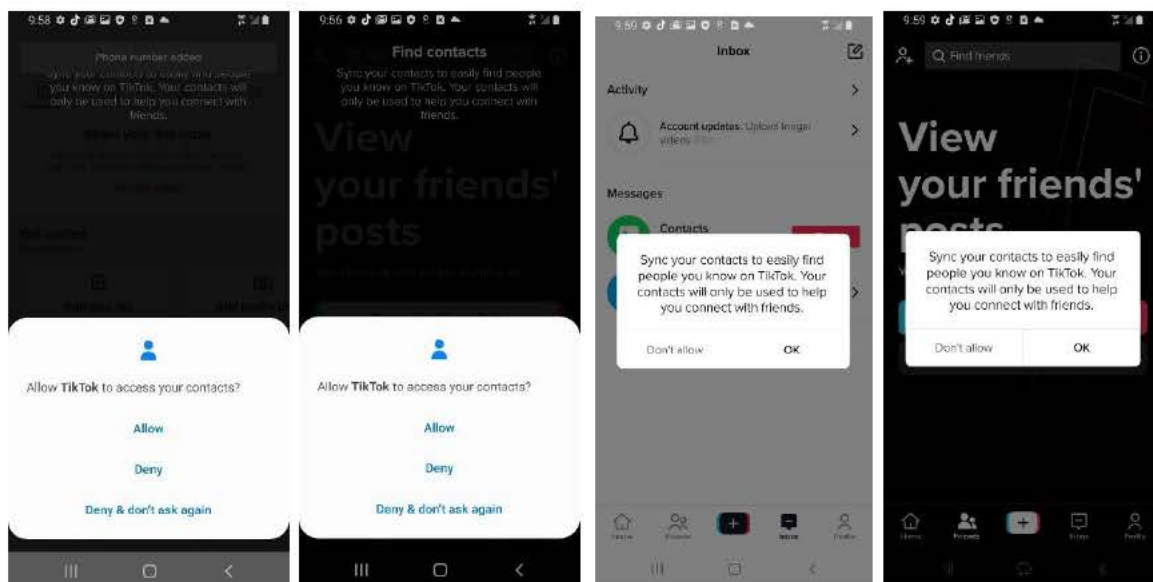


Figure 7: TikTok Contacts access request prompts while in application.

Calendar

The Android application has persistent access to read and modify calendar, see figure 8. TikTok only uses the calendar for special circumstances, for example when there is a TikTok LIVE event, based on our analysis. The persistency of access to the calendar is excessive in our opinion.

```
package X;

import com.bytedance.covode.number.Covode;
import com.ss.android.ugc.effectmanager.common.BuildConfig;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Locale;
import java.util.TimeZone;
import kotlin.g.b.n;
/* renamed from: X.53d */
/* loaded from: /mnt/c/Users/rando/bin/python/maltree/temp_files/extracted_apks/Qksfgib00vyBEyNPuFPSJHAcrokQZd/classes12.dex */
public final class C017153d {
    public static final C017153d LIZ = new C017153d();

    static {
        Covode.recordClassIndex(116083);
    }

    public static final String LIZ() {
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyyMMddHHmmss", Locale.US);
        simpleDateFormat.setTimeZone(TimeZone.getTimeZone("GMT"));
        Calendar calendar = Calendar.getInstance();
        n.LIZIZ(calendar, BuildConfig.VERSION_NAME);
        String format = simpleDateFormat.format(calendar.getTime());
        return 0dC.LIZIZ.LIZ().LJJI().LIZ() + format;
    }
}
```

Figure 8: Persistent calendar access.

External storage

TikTok Android application requests access to external storage. This is a standard command for a social media application to store video and images. The aspect we list as excessive is TikTok doesn't just retrieve the ability to see folders it retrieves a list of everything available in the external storage folder where the application has the access to place files, see figure 9.

```
public static File LIZ(Context context, String str) {
    if (!TextUtils.isEmpty(str)) {
        return context.getExternalFilesDir(str);
    }
    if (C11550aX.LIZLLL != null && C11550aX.LJ) {
        return C11550aX.LIZLLL;
    }
    File externalFilesDir = context.getExternalFilesDir(str);
    C11550aX.LIZLLL = externalFilesDir;
    return externalFilesDir;
}
```

Figure 9: List everything in external storage.

Device and user data harvesting

Device Data

TikTok also has potential to harvest an excessive amount of data about the device, it is important to note that due to limitations with dynamic analysis it is not currently possible to determine if any of this data is ever taken from the device, however, the Android application has code that can gather the following additional device details. See figures 10-12

- Wi-Fi SSID
- Device build serial number
- SIM serial number
- Integrated Circuit Card Identification Number (this is global unique serial number that is specifically tailored to your SIM card)
- Device IMEI
- Device MAC address
- Device line number
- Device voicemail number
- GPS status information (updates on the GPS location)
- Active subscription information
- All accounts on the device
- Complete access to read the clipboard (dangerous as password managers use clipboards)



```
public final class C0N7 {
    public static SparseArray<a> LIZIZ;
    public static final C0N7 LIZLLL;
    public static HashMap<String, Integer> LIZ = new HashMap<>();
    public static List<a> LIZJ = C278311x.LIZIZ(new a(200000, "android.location.LocationManager", "getLastKnownLocation", "LocationManager.getLastKnownLocation", C68462k8.LIZ, C68462k8.LIZLLL, (String[]) null, C17260jk.LIZ("location"), 448), new a(
    public static AtomicBoolean LJ = new AtomicBoolean(false);

    static {
        Covode.recordClassIndex(25449);
        C0N7 r11 = new C0N7();
        LIZLLL = r11;
        LIZIZ = new SparseArray<>(0);
        SparseArray<a> sparseArray = new SparseArray<>(108);
        LIZIZ = sparseArray;
        sparseArray.append(ImagePreloadExperiment.PRIORITY_DEFAULT, new a((int) ImagePreloadExperiment.PRIORITY_DEFAULT, "android.location.LocationManager", "getLastKnownLocation", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION",
        LIZIZ.append(100001, new a(100001, "android.location.LocationManager", "requestLocationUpdates", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION", "android.permission.ACCESS_COARSE_LOCATION"}, 1, C17260jk.LIZ("location"))));
        LIZIZ.append(100002, new a(100002, "android.location.LocationManager", "requestSingleUpdate", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION", "android.permission.ACCESS_COARSE_LOCATION"}, 1, C17260jk.LIZ("location"))));
        LIZIZ.append(100003, new a(100003, "android.webkit.WebChromeClient", "onGeolocationPermissionsShowPrompt", "", "", "", new String[0], 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100004, new a(100004, "android.location.LocationManager", "getCurrentLocation", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION", "android.permission.ACCESS_COARSE_LOCATION"}, 1, C17260jk.LIZ("location"))));
        LIZIZ.append(100005, new a(100005, "android.location.LocationManager", "addGpsStatusListener", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION"}, 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100006, new a(100006, "android.location.LocationManager", "addNmeaListener", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION"}, 0, C17260jk.LIZ("location"))));
        LIZIZ.append(BuildConfig.VERSION_CODE, new a((int) BuildConfig.VERSION_CODE, "android.location.LocationManager", "addProximityAlert", "", "", "", new String[0], 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100008, new a(100008, "android.location.LocationManager", "registerAntennaInfoListener", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION"}, 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100009, new a(100009, "android.location.LocationManager", "registerGnssMeasurementsCallback", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION"}, 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100010, new a(100010, "android.location.LocationManager", "registerGnssNavigationMessageCallback", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION"}, 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100011, new a(100011, "android.location.LocationManager", "registerGnssStatusCallback", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION"}, 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100012, new a(100012, "android.webkit.GeolocationPermissions$Callback", "invoke", "", "", "", new String[0], 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100100, new a(100100, "android.hardware.Camera", "open", "", "", "", new String[]{"android.permission.CAMERA"}, 0, C17260jk.LIZ("video"))));
        LIZIZ.append(100101, new a(100101, "android.hardware.Camera", "release", "", "", "", new String[]{"android.permission.CAMERA"}, 0, C17260jk.LIZ("video"))));
        LIZIZ.append(100106, new a(100106, "android.hardware.Camera", "release", "", "", "", new String[]{"android.permission.CAMERA"}, 0, C17260jk.LIZ("video"))));
        LIZIZ.append(100200, new a(100200, "android.hardware.camera2.CameraDevice$StateCallback", "onOpened", "", "", "", new String[]{"android.permission.CAMERA"}, 0, C17260jk.LIZ("video"))));
        LIZIZ.append(100201, new a(100201, "android.hardware.camera2.CameraDevice", "close", "", "", "", new String[]{"android.permission.CAMERA"}, 0, C17260jk.LIZ("video"))));
        LIZIZ.append(100205, new a(100205, "android.hardware.camera2.CameraDevice", "close", "", "", "", new String[]{"android.permission.CAMERA"}, 0, C17260jk.LIZ("video"))));
        LIZIZ.append(100400, new a(100400, "android.media.AudioRecord", "startRecording", "", "", "", new String[]{"android.permission.RECORD_AUDIO"}, 0, C17260jk.LIZ("audio"))));
        LIZIZ.append(100401, new a(100401, "android.media.AudioRecord", "stop", "", "", "", new String[]{"android.permission.RECORD_AUDIO"}, 0, C17260jk.LIZ("audio"))));
        LIZIZ.append(100403, new a(100403, "android.media.AudioRecord", "release", "", "", "", new String[]{"android.permission.RECORD_AUDIO"}, 0, C17260jk.LIZ("audio"))));
        LIZIZ.append(100404, new a(100404, "android.media.AudioRecord", "stop", "", "", "", new String[]{"android.permission.RECORD_AUDIO"}, 0, C17260jk.LIZ("audio"))));
        LIZIZ.append(100405, new a(100405, "android.media.AudioRecord", "release", "", "", "", new String[]{"android.permission.RECORD_AUDIO"}, 0, C17260jk.LIZ("audio"))));
        LIZIZ.append(100500, new a(100500, "android.media.MediaRecorder", "prepare", "", "", "", new String[]{"android.permission.RECORD_AUDIO"}, 0, C278311x.LIZIZ("audio", "video"))));
        LIZIZ.append(100501, new a(100501, "android.media.MediaRecorder", "release", "", "", "", new String[]{"android.permission.RECORD_AUDIO"}, 0, C278311x.LIZIZ("audio", "video"))));
        LIZIZ.append(100502, new a(100502, "android.media.MediaRecorder", "start", "", "", "", new String[]{"android.permission.RECORD_AUDIO"}, 0, C278311x.LIZIZ("audio", "video"))));
        LIZIZ.append(100503, new a(100503, "android.media.MediaRecorder", "stop", "", "", "", new String[]{"android.permission.RECORD_AUDIO"}, 0, C278311x.LIZIZ("audio", "video"))));
        LIZIZ.append(100702, new a(100702, "android.hardware.SensorManager", "getSensorList", "", "", "", new String[0], 0, C17260jk.LIZ("motion"))));
        LIZIZ.append(100703, new a(100703, "android.hardware.SensorManager", "getDefaultSensor", "", "", "", new String[0], 0, C17260jk.LIZ("motion"))));
        LIZIZ.append(100900, new a(100900, "android.telephony.TelephonyManager", "getCellLocation", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION"}, 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100901, new a(100901, "android.telephony.cdma.CdmaCellLocation", "getBaseStationId", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION"}, 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100902, new a(100902, "android.telephony.cdma.CdmaCellLocation", "getBaseStationLatitude", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION"}, 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100903, new a(100903, "android.telephony.cdma.CdmaCellLocation", "getBaseStationLongitude", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION"}, 0, C17260jk.LIZ("location"))));
        LIZIZ.append(100904, new a(100904, "android.telephony.cdma.CdmaCellLocation", "getSystemId", "", "", "", new String[0], 0, C17260jk.LIZ("location"))));
```

Figure 10: TikTok Data harvest image.



```
LIZIZ.append(182008, new a(182008, "android.telephony.SubscriptionManager", "getActiveSubscriptionInfoList", "", "", "", new String[]{"android.permission.READ_PHONE_STATE"}, 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(182009, new a(182009, "android.telephony.SubscriptionManager", "getOpportunisticSubscriptions", "", "", "", new String[]{"android.permission.READ_PHONE_STATE"}, 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(182010, new a(182010, "android.telephony.SubscriptionManager", "getSubscriptionsInGroup", "", "", "", new String[]{"android.permission.READ_PHONE_STATE"}, 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(182011, new a(182011, "android.telephony.SubscriptionManager", "isActiveSubscriptionId", "", "", "", new String[]{"android.permission.READ_PHONE_STATE"}, 0, C17260jk.LIZ("network"));
LIZIZ.append(182012, new a(182012, "android.telecom.TelecomManager", "getLineNumber", "", "", "", new String[]{"android.permission.READ_PHONE_STATE", "android.permission.READ_PHONE_NUMBERS"}, 1, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(182013, new a(182013, "android.telephony.TelephonyManager", "getNetworkType", "", "", "", new String[]{"android.permission.READ_PHONE_STATE"}, 0, C17260jk.LIZ("network")));
LIZIZ.append(182014, new a(182014, "android.telephony.TelephonyManager", "getSubscriptionId", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(182100, new a(182100, "android.media.projection.MediaProjectionManager", "createScreenCaptureIntent", "", "", "", new String[0], 0, C17260jk.LIZ("screen_record")));
LIZIZ.append(182101, new a(182101, "android.media.projection.MediaProjectionManager", "getMediaProjection", "", "", "", new String[0], 0, C17260jk.LIZ("screen_record")));
LIZIZ.append(182102, new a(182102, "android.media.projection.MediaProjection", "stop", "", "", "", new String[0], 0, C17260jk.LIZ("screen_record")));
LIZIZ.append(182300, new a(182300, "android.net.wifi.WifiManager", "getScanResults", "", "", "", new String[]{"android.permission.ACCESS_FINE_LOCATION", "android.permission.ACCESS_WIFI_STATE"}, 0, C278311x.LIZIZ("wifi", "location")));
LIZIZ.append(182301, new a(182301, "android.net.wifi.WifiManager", "getConnectionInfo", "", "", "", new String[0], 0, C278311x.LIZIZ("wifi", "location")));
LIZIZ.append(182302, new a(182302, "android.net.wifi.WifiManager", "startScan", "", "", "", new String[]{"android.permission.CHANGE_WIFI_STATE"}, 0, C17260jk.LIZ("wifi")));
LIZIZ.append(182500, new a(182500, "android.accounts.AccountManager", "getAccounts", "", "", "", new String[]{"android.permission.GET_ACCOUNTS"}, 0, C17260jk.LIZ("account")));
LIZIZ.append(182501, new a(182501, "android.accounts.AccountManager", "getAccountsByType", "", "", "", new String[]{"android.permission.GET_ACCOUNTS"}, 0, C17260jk.LIZ("account")));
LIZIZ.append(182502, new a(182502, "android.accounts.AccountManager", "getAccountsByTypeAndFeatures", "", "", "", new String[]{"android.permission.GET_ACCOUNTS"}, 0, C17260jk.LIZ("account")));
LIZIZ.append(182600, new a(182600, "android.app.Activity", "requestPermissions", "", "", "", new String[0], 0, C277311n.INSTANCE));
LIZIZ.append(182601, new a(182601, "android.app.Fragment", "requestPermissions", "", "", "", new String[0], 0, C277311n.INSTANCE));
LIZIZ.append(182604, new a(182604, "android.webkit.WebChromeClient", "onPermissionRequest", "", "", "", new String[0], 0, C277311n.INSTANCE));
LIZIZ.append(182605, new a(182605, "android.webkit.PermissionRequest", "grant", "", "", "", new String[0], 0, C277311n.INSTANCE));
LIZIZ.append(182606, new a(182606, "android.webkit.PermissionRequest", "deny", "", "", "", new String[0], 0, C277311n.INSTANCE));
LIZIZ.append(118000, new a(118000, "java.lang.reflect.Method", "invoke", "", "", "", new String[0], 0, C277311n.INSTANCE));
LIZIZ.append(240004, new a(240004, "android.content.ContentResolver", "query", "", "", "", new String[0], 0, C278311x.LIZIZ("album", "calendar", "contact")));
LIZIZ.append(240015, new a(240015, "android.content.ContentResolver", "applyBatch", "", "", "", new String[0], 0, C278311x.LIZIZ("album", "calendar", "contact")));
r11.LIZIZ();
```

Figure 11: TikTok Data harvest image.

internet2.0

TikTok Analysis

```

LIZIZ.append(100904, new a(100904, "android.telephony.cdma.CdmaCellLocation", "getSystemId", "", "", "", new String[0], 0, C17260jk.LIZ("location")));
LIZIZ.append(100905, new a(100905, "android.telephony.cdma.CdmaCellLocation", "getNetworkId", "", "", "", new String[0], 0, C17260jk.LIZ("location")));
LIZIZ.append(100906, new a(100906, "android.telephony.gsm.GsmCellLocation", "getCid", "", "", "", new String[0], 0, C17260jk.LIZ("location")));
LIZIZ.append(100907, new a(100907, "android.telephony.gsm.GsmCellLocation", "getLac", "", "", "", new String[0], 0, C17260jk.LIZ("location")));
LIZIZ.append(100908, new a(100908, "android.telephony.gsm.GsmCellLocation", "getPsc", "", "", "", new String[0], 0, C17260jk.LIZ("location")));
LIZIZ.append(100909, new a(100909, "android.telephony.TelephonyManager", "getAllCellInfo", "", "", "", new String[0], 0, C17260jk.LIZ("location")));
LIZIZ.append(100910, new a(100910, "android.telephony.TelephonyManager", "requestCellInfoUpdate", "", "", "", new String[0], 0, C17260jk.LIZ("location")));
LIZIZ.append(100911, new a(100911, "android.telephony.PhoneStateListener", "onCellLocationChanged", "", "", "", new String[0], 0, C17260jk.LIZ("location")));
LIZIZ.append(100912, new a(100912, "android.telephony.PhoneStateListener", "onCellInfoChanged", "", "", "", new String[0], 0, C17260jk.LIZ("location")));
LIZIZ.append(101000, new a(101000, "android.net.wifi.WifiInfo", "getSSID", "", "", "", new String[0], 0, C278311x.LIZIZ("location", "wifi", "device_info")));
LIZIZ.append(101001, new a(101001, "android.net.wifi.WifiManager", "getConfiguredNetworks", "", "", "", new String[0], 0, C278311x.LIZIZ("location", "wifi", "device_info")));
LIZIZ.append(101100, new a(101100, "android.net.wifi.WifiInfo", "getBSSID", "", "", "", new String[0], 0, C278311x.LIZIZ("location", "wifi", "device_info")));
LIZIZ.append(101200, new a(101200, "android.os.Build", "getSerial", "", "", "", new String[0], 0, C17260jk.LIZ("device_info")));
LIZIZ.append(101300, new a(101300, "android.app.ActivityManager", "getRecentTasks", "", "", "", new String[0], 0, C17260jk.LIZ("application")));
LIZIZ.append(101301, new a(101301, "android.app.ActivityManager", "getRunningTasks", "", "", "", new String[0], 0, C17260jk.LIZ("application")));
LIZIZ.append(101302, new a(101302, "android.app.ActivityManager", "getRunningServices", "", "", "", new String[0], 0, C17260jk.LIZ("application")));
LIZIZ.append(101304, new a(101304, "android.content.pm.PackageManager", "getInstalledApplications", "", "", "", new String[0], 0, C17260jk.LIZ("application")));
LIZIZ.append(101305, new a(101305, "android.content.pm.PackageManager", "getInstalledApplicationsAsUser", "", "", "", new String[0], 0, C17260jk.LIZ("application")));
LIZIZ.append(101306, new a(101306, "android.view.accessibility.AccessibilityManager", "getInstalledAccessibilityServiceList", "", "", "", new String[0], 0, C17260jk.LIZ("application")));
LIZIZ.append(101307, new a(101307, "android.view.accessibility.AccessibilityManager", "getEnabledAccessibilityServiceList", "", "", "", new String[0], 0, C277311x.INSTANCE));
LIZIZ.append(101308, new a(101308, "android.content.pm.PackageManager", "getInstalledPackagesAsUser", "", "", "", new String[0], 0, C17260jk.LIZ("application")));
LIZIZ.append(101309, new a(101309, "android.content.pm.PackageManager", "getInstalledPackages", "", "", "", new String[0], 0, C17260jk.LIZ("application")));
LIZIZ.append(101310, new a(101310, "android.content.pm.PackageManager", "getPackagesForUid", "", "", "", new String[0], 0, C17260jk.LIZ("application")));
LIZIZ.append(101311, new a(101311, "android.content.pm.PackageManager", "queryIntentActivities", "", "", "", new String[0], 0, C17260jk.LIZ("application")));
LIZIZ.append(101400, new a(101400, "android.telephony.TelephonyManager", "getSimSerialNumber", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(101500, new a(101500, "android.telephony.SubscriptionInfo", "getIccId", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(101600, new a(101600, "android.telephony.TelephonyManager", "getDeviceId", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(101601, new a(101601, "android.telephony.TelephonyManager", "getImei", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(101602, new a(101602, "android.telephony.TelephonyManager", "getMeid", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(101700, new a(101700, "android.net.wifi.WifiInfo", "getMacAddress", "", "", "", new String[0], 0, C278311x.LIZIZ("location", "wifi", "device_info")));
LIZIZ.append(101701, new a(101701, "java.net.NetworkInterface", "getHardwareAddress", "", "", "", new String[0], 0, C17260jk.LIZ("device_info")));
LIZIZ.append(101800, new a(101800, "android.content.ClipboardManager", "clearPrimaryClip", "", "", "", new String[0], 0, C17260jk.LIZ("clipboard")));
LIZIZ.append(101801, new a(101801, "android.content.ClipboardManager", "addPrimaryClipChangedListener", "", "", "", new String[0], 0, C17260jk.LIZ("clipboard")));
LIZIZ.append(101802, new a(101802, "android.content.ClipboardManager", "removePrimaryClipChangedListener", "", "", "", new String[0], 0, C17260jk.LIZ("clipboard")));
LIZIZ.append(101803, new a(101803, "android.content.ClipboardManager", "getPrimaryClip", "", "", "", new String[0], 0, C17260jk.LIZ("clipboard")));
LIZIZ.append(101804, new a(101804, "android.content.ClipboardManager", "getText", "", "", "", new String[0], 0, C17260jk.LIZ("clipboard")));
LIZIZ.append(101805, new a(101805, "android.content.ClipboardManager", "hasPrimaryClip", "", "", "", new String[0], 0, C17260jk.LIZ("clipboard")));
LIZIZ.append(101806, new a(101806, "android.content.ClipboardManager", "hasText", "", "", "", new String[0], 0, C17260jk.LIZ("clipboard")));
LIZIZ.append(101807, new a(101807, "android.content.ClipboardManager", "setPrimaryClip", "", "", "", new String[0], 0, C17260jk.LIZ("clipboard")));
LIZIZ.append(101808, new a(101808, "android.content.ClipboardManager", "setText", "", "", "", new String[0], 0, C17260jk.LIZ("clipboard")));
LIZIZ.append(101809, new a(101809, "android.content.ClipboardManager", "getPrimaryClipDescription", "", "", "", new String[0], 0, C17260jk.LIZ("clipboard")));
LIZIZ.append(101900, new a(101900, "android.telephony.TelephonyManager", "getSubscriberId", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(102000, new a(102000, "android.telephony.TelephonyManager", "getLine1Number", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(102001, new a(102001, "android.telephony.TelephonyManager", "getVoiceMailNumber", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(102002, new a(102002, "android.os.Build", "SERIAL", "", "", "", new String[0], 0, C17260jk.LIZ("device_info")));
LIZIZ.append(102003, new a(102003, "android.provider.Settings$System", "getString", "", "", "", new String[0], 0, C17260jk.LIZ("device_info")));
LIZIZ.append(102004, new a(102004, "android.provider.Settings$Secure", "getString", "", "", "", new String[0], 0, C17260jk.LIZ("device_info")));
LIZIZ.append(102005, new a(102005, "android.telephony.SubscriptionManager", "getActiveSubscriptionInfo", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));
LIZIZ.append(102006, new a(102006, "android.telephony.SubscriptionManager", "getActiveSubscriptionInfoCount", "", "", "", new String[0], 0, C17260jk.LIZ("network")));
LIZIZ.append(102007, new a(102007, "android.telephony.SubscriptionManager", "getActiveSubscriptionInfoFor$slotIndex", "", "", "", new String[0], 0, C17260jk.LIZ("network")));
LIZIZ.append(102008, new a(102008, "android.telephony.SubscriptionManager", "getActiveSubscriptionInfoList", "", "", "", new String[0], 0, C278311x.LIZIZ("network", "device_info")));

```

Figure 12: TikTok Data harvest image.

Of note: Joe's Sandbox rated the Android application as malicious for Spyware and Evader categories as seen in Figure 13 because of device and user data collection by the application and evasive techniques the application uses to block any type of analysis. Many applications have anti-sandbox run commands now to inhibit automatic analysis, the sandbox identifies these and categorizes it in the evader category.

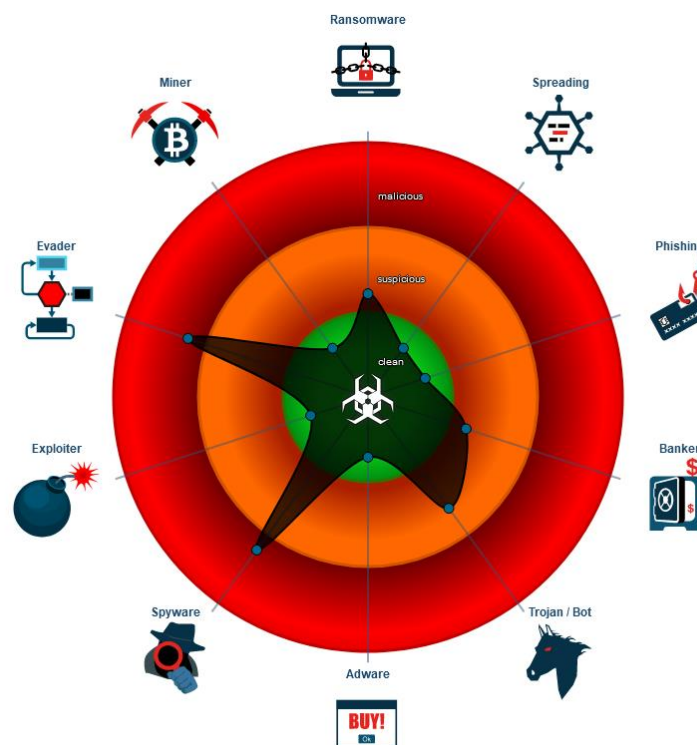


Figure 13. TikTok rating as per Joe's sandbox (<https://www.joesandbox.com/>).

IOS connects to mainland China

TikTok are specific in their statement that TikTok user data is stored in Singapore and the US. However, we found many subdomains in the IOS application resolving all around the world including: Sydney, Adelaide and Melbourne (Australia); New York City, Las Vegas, San Francisco, San Jose, Monrovia, Cambridge, Kansas City, Dallas, Mountain View (USA); Utama and Jakarta (Indonesia), Kuala Lumpur (Malaysia), Paris (France), Singapore (Singapore) and Baishan (China). During analysis we could not determine with high confidence the purpose for the China Server connection or where user data is stored. The China server connection is run by 贵州白山云科技股份有限公司 Guizhou Baishan Cloud Technology Co., Ltd a cloud and cyber security company. The subdomain connected to the "China server connection" resolved in multiple locations around the world including in China. The IP address resolving to China regularly

changed locations, however, connectivity to Baishan Guangxi China was visible across a number of different IP addresses over time. This was confirmed through the use of a number of security products and methods, including virus total, Metasploit, security trails and sandboxing. Interestingly, this company has been rated a top 100 Chinese cyber security company and in 2021 established a joint big data laboratory with Guizhou University.⁴ Of note only the IOS version had this mainland direct server connection. We could not find any direct server connections with mainland China in the Android version of the application.

Conclusion

For the TikTok application to function properly most of the access and device data collection is not required. The application can and will run successfully without any of this data being gathered. This leads us to believe that the only reason this information has been gathered is for data harvesting. It is also notable that the device only needs to ask the user for permission to perform each of these actions once and then follow the user's preferences. The application however has a culture of persistent access or continuously asking for a decision reversal by the user. The hourly checking of location is also unnecessary. Finally, device mapping, external storage access, contacts and third-party applications data collection allows TikTok the ability to reimage the phone in the likeness of the original device.

⁴ https://baike.baidu.com/reference/23443686/44e44NXRiOexRZo-8rbRsVSmZL-hjxLfaZVO4i748emXOcfv_uNtLc1yLac09EyZEBsnmwIHmEiKgrSKyJafiRJXffvnMrZx3fjvd7KgfZXHQTJqcQiSTTzNcYs12v7vcNN
https://baike.baidu.com/reference/23443686/cc63DG_6ZWBsyHhiqR45OVCvsMnuyzIROgdcmvvuXilWB48sb7YhfKhpeWv0ZpsePYpHI2EMcS8LdZe2yWIZPp3rLCUtoQfy96e5-_uuvbQ

internet20

MILITARY-GRADE

CYBER PROTECTION

Australia

Level 1

18 National Circuit

Barton ACT 2600

ABN: 17 632 726

United States

Suite 100

211 N Union St

Alexandria 22314

EIN: 86-1567068

contact@internet2-0.com

Mobile Applications Industry analysis

13 Feb 2022

Summary

TikTok Scores 63.1 - Designed to Collect Data with highest Malcore score in Industry Malcore, by Internet 2.0, will publish analysis results on all popular social media mobile apps. Malcore is an automated analysis tool to scan files and programs to detect malware & assess risk.

The Malcore team at Internet 2.0 have been releasing an industry analysis of social media mobile applications. We have been doing this daily through blog posts. This post is the release of TikTok.

Malcore scored TikTok 63.1. This was the highest (worst) score relative to all other applications we tested. The only score close was VK, the Russian app on 62.7. The industry standard was all other major social media applications scored 34 and below with the average score being 28.8 over 21 applications. TikTok got this score because it had 9 trackers and a lot of permissions and code severity warnings. One of the biggest flags for us was the presence of the Russian VKontakte SDK.

TikTok Discovered Trackers (SDKs)

- Facebook Share
- Bolts
- AppsFlyer
- Google Firebase Analytics
- VKontakte SDK
- Facebook Analytics
- Facebook Login
- Pangle
- Google CrashLytics

VK is a Russian based app which was banned globally on all IOS (Apple) for 1 month last year over questions about ownership. Ukraine banned it in 2017 and it still unable to be accessed by Ukraine. The founder of VK and Telegram, Pavel Durov, accused the lack of independence as “it became increasingly difficult to run the social network after ownership changes put pressure on the company preserving its freedom of speech ethic.” VK is basically impossible to access outside of Russia since 2022 as they now require a Russian phone number to access it. Considering TikTok scored higher than VK this again raised our concerns about TikTok. In 2022 Internet 2.0 published reporting, covered globally, on our concerns about TikTok.

We stated it was their word against their source code concerning the data harvesting and privacy concerns. TikTok stated to the AFR

The TikTok app is not unique in the amount of information it collects, which is less than many popular mobile apps. In line with industry practices, we collect information that users choose to provide to us and information that helps the app function, operate securely, and improve the user experience.

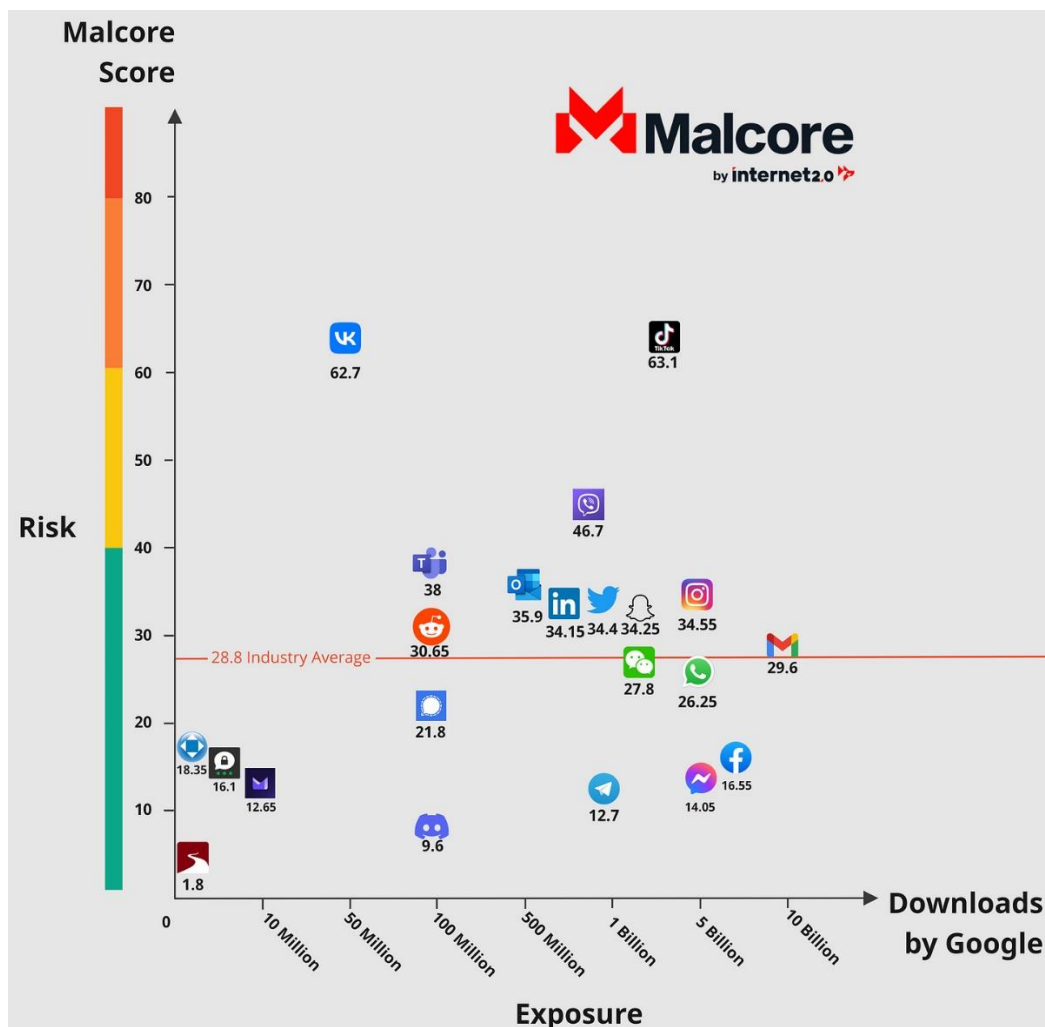
In our opinion this controlled experiment where we conduct the exact same analysis and scoring system across industry reinforces our 2022 report's conclusions as they are more than double the industry average at 63.1.

For a detailed view of this reverse engineering on TikTok view it directly on Malcore or the [blog](#).

[View Full results of TikTok analysis](#)

Industry Analysis Results

The social media industry analysis project shows us the relative Malcore risk score for each application. It is a comparative process where the process controls keep the same Malcore scoring algorithm, all are done on Android APKs, and the time of analysis is the same for all applications. We use these controls to baseline the process and help us determine relative risk.



For the graph there is inconsistent public information on active users and downloads per mobile application. We estimated placement by using the downloads on google play store and bracketed by reported monthly active users. We would be happy to adjust this data if applications send us official numbers.

We conducted this project because many clients and readers ask us basic data collection questions about all mobile applications. Because of our product Malcore we are now able to highlight how apps are built and give a relative scoring system for consumers to make decisions with.

Over the course of the project we have noted that

- The industry average was 28.8.
- Most of the major social media applications scored in a tight group around 34.
- There was a large gap between the privacy marketed email applications of Proton and Tunanota with Gmail and Outlook.
- Most messaging applications came in under 20 except WhatsApp.
- WeChat scored between the messaging and social media applications with 27.8.
- In our assessment a score between 10 and 35 are within a normal range for an app to score based on current industry practices. This shows how bad VK and TikTok scored relative to industry.

Of note having looked into SDKs specifically we found it interesting how these larger companies cooperate in the data ecosystem.

- VKontakte and Telegram have Huawei Mobile Services and Google SDK
- VKontakte also has Facebook and other third parties SDK
- Wechat has Wechat, Baidu and Google SDK
- TikTok has VKontakte, Facebook and Google SDK
- LinkedIn has Facebook, Google and Microsoft SDK
- Facebook and Instagram only have Facebook (Meta) SDK
- WhatsApp only has Google Analytics SDK
- Twitter has Google SDK

We must note this analysis process is not an conclusive code review. It is a static analysis with automated code review using Malcore. A detailed manual source code review, where you manually view app activity during dynamic analysis is considered the most conclusive method to assess risk. A manual code review tends to find a lot more information but costs significant time. Most applications try their best to block dynamic analysis to protect their intellectual property.

All Malcore research is self funded which means we are limited by time. For example our TikTok technical analysis report at [Internet 2.0](#), which included dynamic analysis and manual code review, drew far more conclusive insights into TikTok.

Industry Detailed Reports

Tutanota = 1.8 (Lowest email score due to very few code warning, 0 trackers and suspicious warning, as well as low permissions)

Discord = 9.6 (Lowest social media score due to very few code and device access warnings)

ProtonMail = 12.65 (Higher than Tutanota due trackers, suspicious warnings and higher code warning)

Telegram = 12.7 (Analyzed twice and reduced from 17.2, It has Huawei Mobile Services only for Huawei build phones)

Facebook Messenger = 14.05 (Only has Meta Facebook tracking and not connected to Google ecosystem)

Threema Work = 16.1 (Second lowest score for messenger apps, Internet 2.0 preferred messenger app)

Facebook App = 16.55 (One of the lowest social media scores due to very few code warnings, despite that the Facebook app has a high amount of permissions)

Lamchat = 18.35 (Posted by Rory Chapman, Australian Start-up)

Signal Messenger = 21.8 (Third lowest score for messenger apps, Internet 2.0 preferred messenger app)

WhatsApp Messenger = 26.25 (Unlike Meta Facebook WhatsApp has Google Analytics)

WeChat = 27.8 (Slightly higher than WhatsApp, WeChat has 5 trackers in total, including Baidu Maps and WeChat Location)

Gmail = 29.6 (The highest of all the email clients, due to a high amount of permissions)

Reddit = 30.65 (Reddit falls within the average of other social media apps but has 6 total trackers)

LinkedIn = 34.15 (LinkedIn falls within the average score of other social media apps but has a high amount of trackers, with 9 in total)

Snapchat = 34.25 (Snapchat had only 4 trackers but had many permissions)

Twitter = 34.4 (The first in our series of social media apps, Twitter's score is the result of high permissions)

Instagram = 34.55 (The second highest in our series of social media apps so far, Instagram's score is the result of 2 suspicious warning, several trackers in the Facebook Ecosystem and a high amount of permissions)

Outlook = 35.9 (Outlook has 7 trackers which accounts for the high score)

Microsoft Teams = 38 (Microsoft teams has 4 trackers but a high amount of permissions)

Viber Messenger = 46.7 (Has 11 trackers which accounts the higher score)

VK.com = 62.7 (The highest of any app so far, VK has a total of 13 trackers and 28 dangerous permissions)

How the Malcore scoring system works

Scores are assigned by the following

- Dangerous permission = 0.25
- Suspicious permission = 0.075
- High severity warning for code analysis results = 0.15
- Severity warning for code analysis results = 0.05
- Per tracker or token = 2.5

During code analysis Malcore unzips the APK file and decompiles the compiled .dex files. Malcore runs through each file and uses indicators to determine issues within the code. These code severity warnings are based off Java coding best practices.

Next Malcore parses the AndroidManifest.xml file and determines the device permission requests the app has. These permission levels are graded in severity based on the Android manifest website: <https://developer.android.com/reference/android/Manifest.permission>.

A tracker is a piece of software with the task to gather information on the person using the application. A tracker can be used to monitor usage and engagement, for example in analytics or advertising. Trackers normally are a legitimate software development kit (SDK) designed to help developers understand how their apps are being used, resolve potential issues and improve their software. Importantly for privacy though there is a large market buying the data collected by these SDKs to improve advertising spend and to better understand user's behaviour. An example of one of the most sophisticated SDK in the market is Facebook. This post on their developer forum is a good example how the Facebook SDK works <https://developers.facebook.com/community/threads/278044280345820/>

AN INTERNET 2.0 PAPER

Internet 2.0

RELENTLESS SECURITY

DIGITAL SURVEILLANCE IN CHINA
JANUARY 2022

Authors:

David Robinson
Thomas Perkins

Introduction

With the upcoming 2022 Winter Olympics in China the team at Internet 2.0 saw the need to publish case studies demonstrating the sophisticated and broad surveillance culture that exists in China. All Chinese companies are compelled to follow the national security legislation while operating in China. All athletes and visitors to China for the Olympics will be exposed to such laws and surveillance culture. In this paper we show how these laws manifest in terms of surveillance of mobile phones through mobile applications and internet browsers through desktop software. Part 1 is an analysis of the QI-ANXIN VPN. Part 2 is an analysis of the Kingsoft's anti-virus and WPS Office software. We must state that this is not a criticism or endorsement of QI-ANXIN and Kingsoft. Internet 2.0 does not allege inappropriate conduct by QI-ANXIN or Kingsoft, rather we see it as case studies to understand exactly how the Chinese Communist Party imposes its national security legislation and its implementation in the commercial market from a technical standpoint.

Part 1 QI-ANXIN

Executive Summary

Part 1 is a technical analysis of QI-ANXIN Technology Group Inc's (QI-ANXIN) Virtual Private Network (VPN) software product. QI-ANXIN is an official sponsor to the Beijing 2022 Olympic and Paralympic Winter Games, see Figure 1.¹ The analysis is based on QI-ANXIN's publicly available Winter Olympics mobile protection software, as described on their website.² The Internet 2.0 intelligence team analyzed the VPN and discovered a significant amount of user data being collected by the software.

In our opinion QI-ANXIN's VPN provides a limited degree of privacy and security to its users, this assessment is based on the device and network data collected by the software. We recommend that visitors and athletes travelling to the 2022 Winter Olympics in China are aware of the risks in taking and using personal devices during the event, particularly that China's national data security laws are not designed with western values of privacy and liberty and do not offer the same level of protections. This is true for all digital communications in China and not just while using VPN software.

¹ <https://en.qianxin.com/news/detail/205>

² <https://fanghu.qianxin.com/web/index/index.html>

To mitigate the risk of sensitive information and personal data being collected on personal phones during the 2022 Beijing Winter Olympics, we recommend:



Athletes and visitors to the games buy and take a new phone with them to use only while inside China. This will protect their sim information of their devices that they use in their home country.



Creating a new email address and browser account and using these on the 'burner' Phone. This mitigates the risk of cloud accounts such as google, apple or internet browsers connecting all your personal information with this new and isolated 'burner' phone.



Not using this device or account upon leaving China again as these details are likely collected and stored. Using this device outside of China poses the same risk as taking your personal devices and accounts into China.

QI-ANXIN produces a VPN that is capable of being hosted on Windows, Mac, Linux operating systems, and Apple or Android phones. The QI-ANXIN VPN harvests all available network information on Apple and Android phones, including SIM, MAC Addresses, IMEI, IMSI, Operating Systems information and telephone network information. The VPN software also collects previous network interface information on the device.

Equipped with current and historical device information available to the VPN provider, it is possible to easily identify the user with limited privacy. The VPN's software design has facilitated all the required information on the user's device information and historical network information, which could be provided to Chinese authorities, if requested, under the country's national security laws. Hypothetically, if this information was also combined with telecommunications metadata records, it would be possible to correlate both telecommunications metadata and the information provided by the VPN software to gain a complete picture of the users' internet usage history, network history and location history. We note that the network architecture crossover for the VPN between Legendsec, QI-ANXIN and 360.net is an integrated network design, which can be seen in the VPN Log file uploads between Android and IOS mobile applications. This is problematic, as Qihoo 360 (with the same branding as 360.net) was placed on the Entities list for the Department of Commerce and is considered a separate company to QI-ANXIN.³

³ <https://2017-2021.commerce.gov/news/press-releases/2020/05/commerce-department-add-two-dozen-chinese-companies-ties-wmd-and.html>

Who is QI-ANXIN?

QI-ANXIN is one of largest cyber security companies in the China domestic market. QI-ANXIN markets itself as the cybersecurity provider of choice for 90 percent of the central government, state-owned enterprises, and major banks.⁴ QI-ANXIN also advertises branded subsidiaries on its website, key of which is 网神 'Net God' (domain name is legendsec.com), the primary operator of the VPN according to our metadata analysis. We note that QI-ANXIN is one of the key companies that operates and is heavily invested in China's National Cybersecurity Centre according to Dakota Cary writing for CSET at Georgetown. China's National Cybersecurity Centre serves the Chinese Communist Party's strategic interests and supports the People's Liberation Army (PLA) Strategic Support Force's hacking teams.⁵ Qi Xiangdong is QI-ANXIN's major shareholder. Qi Xiangdong is also a major shareholder, and president and co-founder, of 360 Enterprise Security Technology (Beijing) Group Co. Ltd.⁶



北京2022年冬奥会官方赞助商
Official Sponsor of the Olympic Winter Games Beijing 2022

Figure 1: QI-ANXIN logo.

⁴ <https://en.qianxin.com/news/detail/205>

⁵ <https://cset.georgetown.edu/wp-content/uploads/CSET-Chinas-National-Cybersecurity-Center.pdf>

⁶ <https://www.bloomberg.com/profile/person/6035447>

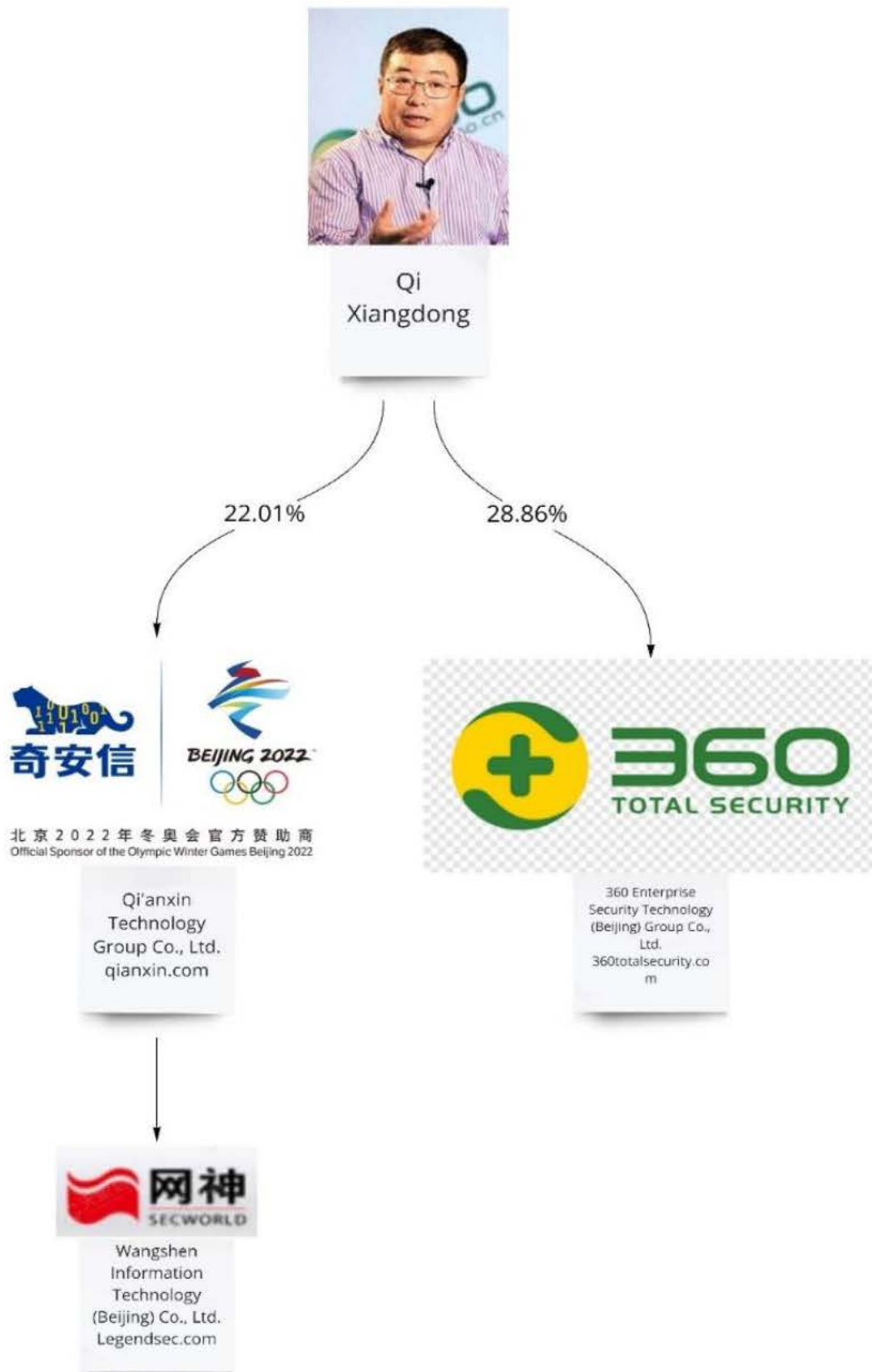


Figure 2: Qi Xiangdong's shareholding in QI-ANXIN and 360 Enterprise Security Technology (Beijing) Group Co., Ltd. As at 3 January 2022.

Multiple open-source reporting suggests that in May 2019 Qi Xiangdong separated his segment of the business from Qihoo 360 and Zhou Hongyi's. 360 Enterprise Security Technology (Beijing) Group Co. Ltd, which is owned by Qi Xiangdong, was now primarily operating under the name of QI-ANXIN.⁷ This was hard to verify as we documented over 61 member companies in the QI-ANXIN Group and 13 member companies in the Qihoo 360 Group, not including 360 Enterprise Security Technology (Beijing) Group Co. Ltd. We could not verify if this is the complete operating structure and saw the complex overlapping company structure, website and branding between QI-ANXIN /360 Enterprise Security Technology (Beijing) Group and the Qihoo 360 Group as confusing to any retail investor. This does raise interesting questions for the United States Government concerning their policy towards close or previous associated entities of US Government sanctioned entities.

Qihoo 360 says they offer internet and mobile security products and services to over 400 million internet users.⁸ We note that Qihoo 360 is recorded as participating in PLA cyber operations through training,⁹ and was placed on the Entities list by the US Department of Commerce.¹⁰

It is important to note that the major shareholder Qi Xiangdong, as seen in Figure 2 above, is the primary shareholder in 360 Enterprise Security Technology (Beijing) Group Co. Ltd and not the higher Qihoo 360 entity. The operating agreement and split between Qihoo 360 and 360 Enterprise Security Technology (Beijing) Group Co. Ltd is not clear. They still share the same branding on their websites between 360.net and 360.cn and when Qihoo 360 is advertised on western sites such as Bloomberg Qi Xiangdong is included in the Qihoo 360 Group.¹¹ 360.net was registered in China as being connected in website to 360.cn, the filing date was 04 Aug 2020.¹² On balance we assess they are still related entities.

⁷[https://baike.baidu.com/item/%E5%A5%87%E5%AE%89%E4%BF%A1%E7%A7%91%E6%8A%80%E9%9B%86%E5%9B%A2%E8%82%A1%E4%BB%BD%E6%9C%89%E9%99%90%E5%85%AC%E5%8F%B8/23546207#reference-\[11\]-24068412-wrap](https://baike.baidu.com/item/%E5%A5%87%E5%AE%89%E4%BF%A1%E7%A7%91%E6%8A%80%E9%9B%86%E5%9B%A2%E8%82%A1%E4%BB%BD%E6%9C%89%E9%99%90%E5%85%AC%E5%8F%B8/23546207#reference-[11]-24068412-wrap)

⁸ <http://www.360.cn/about/englishversion.html>

⁹ <https://cset.georgetown.edu/wp-content/uploads/CSET-Chinas-National-Cybersecurity-Center.pdf>

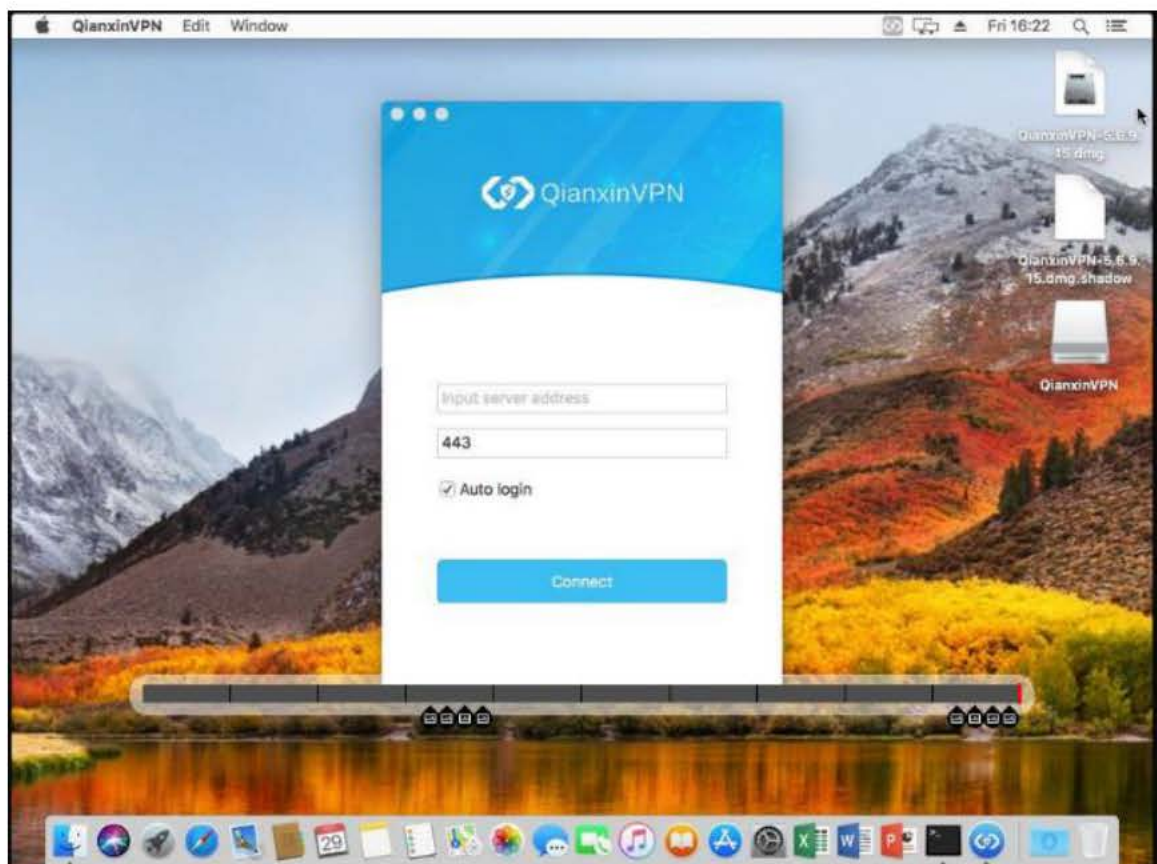
¹⁰ <https://2017-2021.commerce.gov/news/press-releases/2020/05/commerce-department-add-two-dozen-chinese-companies-ties-wmd-and.html>

¹¹ <https://www.bloomberg.com/profile/company/QIHOZ:CH>

¹² <http://www.beian.gov.cn/portal/registerSystemInfo>

QI-ANXIN Apple IOS VPN

The VPN applications for both Android and Apple are similar and the Apple version collects all network information that Android collects. While analysing the source code we noted that the Apple IOS VPN software allows access to the camera and photo library as standard permissions. These third-party application permissions for the camera and photo library are not functional for the use of VPN software.



PERMISSIONS	STATUS	DESCRIPTION	REASON IN MANIFEST
NSCameraUsageDescription	dangerous	Access the Camera.	扫描二维码图片
NSFaceIDUsageDescription	normal	Access the ability to authenticate with Face ID.	人脸识别验证
NSPhotoLibraryUsageDescription	dangerous	Access the user's photo library.	选择二维码图片

Showing 1 to 3 of 3 entries

Previous 1 Next

Figure 3: IOS VPN Camera and Photo permissions.

Both IOS and Android software upload log files back to the VPN Provider. One major difference in this process was the Apple IOS software uploaded the log file to a QI-ANXIN domain and the Android software uploaded the log file to a 360.net domain running the Qihoo 360 branding. This is important to note as they are marketed as separate companies but have overlapping networks. 360 Enterprise Security Technology (Beijing) Group Co., Ltd would be considered a related entity as the major shareholder of both QI-ANXIN and 360 Enterprise Security Technology (Beijing) Group Co., Ltd is Qi Xiangdong. Figure 4 is a diagram separating the difference in log file uploads for the QI-ANXIN VPN across the available platforms.

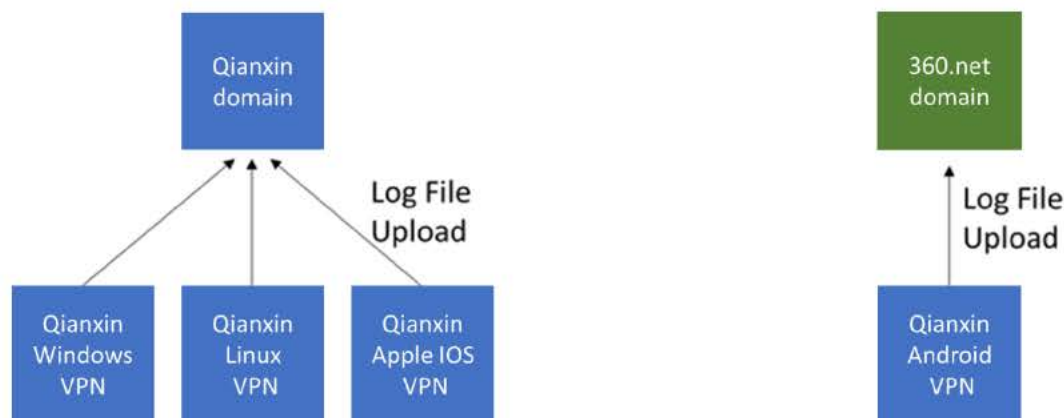


Figure 4: Log File Upload Diagram.

Figure 5 is the source code for the Apple IOS VPN showing the exact line of code where the log file is uploaded. As seen below the VPN log is uploaded to qianxin.com through port 8443.

```
https://%@:%@/fw/images/apk.png  
https://%@:%ld/client/custom_lang.json  
https://%@:%ld/ios_mdm/ios-setup.php?devid=%@  
https://log.aag.qianxin.com:8443/upload.php  
https://appstore.qianxin.com/api/app/lookup  
http://itunes.apple.com/lookup  
https://itunes.apple.com/cn/app/secmobi/id%ld?mt=8  
https://%@:%ld/
```

Figure 5: Example of Apple IOS Log file upload.

Figure 6 below is the source code for the Android VPN showing the exact line of code where the log file is uploaded. As can be seen the Android VPN log is uploaded to 360.net through port 8443. 360.net is running a 360 Government and Enterprise Security Group webpage using the same logo branding depicted in Figure 1 and is also the same logo for Qihoo 360. According to Internet Whois registrant records 360.net is owned by Sun CHANG XIN of BeiJing Qi AnXin KeJi Co.,Ltd, address ChaoYangQu JiuXianQiao Lu 6 Hao Yuan 2 Hao Lou Beijing China. POC: its@360.net +86.1052448735.

```
try {
    String str7 = "https://log.aag.360.net:8443/upload.php";
    Log.v(TAG, "Upload log to " + str7);
    HttpURLConnection httpURLConnection = (HttpURLConnection) new URL(str7).openConnection();
    httpURLConnection.setConnectTimeout(8000);
    httpURLConnection.setReadTimeout(8000);
    httpURLConnection.setRequestProperty("Connection", "Keep-Alive");
    if (str7.startsWith(UriUtil.HTTPS_SCHEME)) {
        httpsURLConnection httpsURLConnection = (HttpsURLConnection) httpURLConnection;
        TrustManager[] trustManagerArr = [new SPHttpClient.SPXS09TrustManager()];
        SSLContext instance = SSLContext.getInstance(Build.VERSION.SDK_INT < 9 ? "TLS" : "SSL");
        instance.init(null, trustManagerArr, new SecureRandom());
        httpsURLConnection.setSSLSocketFactory(instance.getSocketFactory());
        httpsURLConnection.setHostnameVerifier(new SPHttpClient.SPHostnameVerifier());
    }
    httpURLConnection.setDoOutput(true);
    httpURLConnection.setRequestMethod("POST");
    httpURLConnection.setRequestProperty("Content-Type", "multipart/form-data;boundary=*****");
    SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyyMMdd_HHmss");
    Date date = new Date(System.currentTimeMillis());
    if (!TextUtils.isEmpty(str2)) {
        str5 = !TextUtils.isEmpty(str3) ? "vpn-android-" + str2 + "-" + str3 + "-" + simpleDateFormat.format(date) + ".txt" : "vpn-android-";
    } else if (!TextUtils.isEmpty(str3)) {
        str5 = "vpn-android-" + str3 + "-" + simpleDateFormat.format(date) + ".txt";
    } else {
        str5 = "vpn-android-" + simpleDateFormat.format(date) + ".txt";
    }
    httpURLConnection.connect();
    httpURLConnection.getOutputStream().write(uploadLogHeader(str5, str4).getBytes());
    httpURLConnection.getOutputStream().flush();
    writeLog(str, httpURLConnection.getOutputStream());
    httpURLConnection.getOutputStream().write(uploadLogTail().getBytes());
    httpURLConnection.getOutputStream().close();
}
```

Figure 6: Example of Android Log file upload.

QI-ANXIN Android VPN

The following data are only specific to the Android version of the VPN software provided by QI-ANXIN. As seen below in Figure 7, this VPN is grabbing a significant amount of information off this phone, including screen height and width, as well as Bluetooth MAC addresses.

```
public class SPUniqueIDUtil {
    private static void getBuildPropFP(JSONObject jsonObject) {
    }

    private static void getUserFP(Context context, JSONObject jsonObject) {
    }

    public static JSONObject getDeviceFP(Context context) {
        JSONObject jsonObject = new JSONObject();
        SPJSONUtil.put(jsonObject, "os.name", "Android");
        SPJSONUtil.put(jsonObject, "os.version", Build.VERSION.RELEASE);
        SPJSONUtil.put(jsonObject, "os.rooted", SPDeviceUtil.isRooted() ? "1" : "0");
        byte[] readFile = SPFileUtil.readFile("/proc/version");
        if (readFile != null) {
            SPJSONUtil.put(jsonObject, "os.kernel.version", new String(readFile).trim());
        } else {
            SPJSONUtil.put(jsonObject, "os.kernel.version", "");
        }
        SPJSONUtil.put(jsonObject, "dev.name", SPDeviceUtil.getDeviceName(context));
        SPJSONUtil.put(jsonObject, "dev.model", Build.MODEL);
        SPJSONUtil.put(jsonObject, "dev.product", Build.PRODUCT);
        SPJSONUtil.put(jsonObject, "dev.android_id", Settings.Secure.getString(context.getContentResolver(), "android_id"));
        SPJSONUtil.put(jsonObject, "dev.imei", "[md5]" + SPStringUtil.md5(SPDeviceUtil.getIMEI(context)));
        SPJSONUtil.put(jsonObject, "dev.imsi", SPDeviceUtil.getIMSI(context));
        SPJSONUtil.put(jsonObject, "dev.serial", Build.SERIAL);
        SPJSONUtil.put(jsonObject, "dev.manufacturer", Build.MANUFACTURER);
        SPJSONUtil.put(jsonObject, "dev.wifi.mac", SPNetUtil.getWifiMac(context));
        SPJSONUtil.put(jsonObject, "dev.bluetooth.mac", SPDeviceUtil.getBluetoothAddress(context));
        Point screenSize = SPViewUtil.screenSize(context);
        SPJSONUtil.put(jsonObject, "dev.screen.width", String.valueOf(screenSize.x));
        SPJSONUtil.put(jsonObject, "dev.screen.height", String.valueOf(screenSize.y));
        getBuildPropFP(jsonObject);
        getStorageFP(context, jsonObject);
        SPJSONUtil.put(jsonObject, "net.hostname", SPSystemUtil.getProperty("net.hostname"));
        getNetInFP(jsonObject);
        String packageName = context.getPackageName();
        SPJSONUtil.put(jsonObject, "app.packagename", packageName);
        try {
            PackageInfo packageInfo = context.getPackageManager().getPackageInfo(packageName, 0);
            SPJSONUtil.put(jsonObject, "app.version.name", packageInfo.versionName);
            SPJSONUtil.put(jsonObject, "app.version.code", packageInfo.versionCode);
        } catch (Exception unused) {
        }
        String signature = SPSystemUtil.getSignature(context, context.getPackageName());
        SPJSONUtil.put(jsonObject, "app.signature", "[md5]" + SPStringUtil.md5(signature));
        SPJSONUtil.put(jsonObject, "app.uid", "" + Process.myUid());
        getUserFP(context, jsonObject);
        return jsonObject;
    }
}
```

Figure 7: Android VPN device information gathering.

Figure 8 below is an example of the source code where the software records all network information on all previously connected network interfaces. For example, if you have three network interfaces on your phone that your phone has connected to, the above for-loop will run three times to gather each interface IMSI, and carrier information. This is not a functional process as previous network information is not standard or needed for a VPN to work.

```

/* JAVA WARNING: REMOVED DUPLICATED REGION FOR BLOCK: DIRECT ALLOCATION, OBTAINING */
public static int getActiveNetInfo(Context context) {
    int i;
    Exception e;
    int i2 = 0;
    try {
        NetworkInfo[] allNetworkInfo = ((ConnectivityManager) context.getSystemService("connectivity")).getAllNetworkInfo();
        if (allNetworkInfo != null) {
            i = 0;
            for (int i3 = 0; i3 < allNetworkInfo.length; i3++) {
                try {
                    if (allNetworkInfo[i3].getState() == NetworkInfo.State.CONNECTED || allNetworkInfo[i3].getState() == NetworkInfo.State.CONNECTING) {
                        if (allNetworkInfo[i3].getType() == 1) {
                            i |= 1;
                        } else {
                            if (allNetworkInfo[i3].getType() == 0) {
                                try {
                                    TelephonyManager telephonyManager = (TelephonyManager) context.getSystemService("phone");
                                    String subscriberId = telephonyManager.getSubscriberId();
                                    PLog.v("IMSI=%s", subscriberId);
                                    int parseCarrier = subscriberId != null ? SPNetworkInfo.parseCarrier(subscriberId) : 256;
                                    if (parseCarrier == 256) {
                                        parseCarrier = SPNetworkInfo.parseCarrier(telephonyManager.getSimOperator());
                                    }
                                    i |= parseCarrier;
                                } catch (Exception e2) {
                                    PLog.v(e2);
                                }
                            }
                        }
                    }
                } catch (Exception e3) {
                    e = e3;
                    PLog.v(e);
                    i2 = i;
                    if (i2 == 0) {
                        i2 = 1;
                    }
                }
            }
        }
    } catch (Exception e4) {
        e = e4;
        i = 0;
        PLog.v(e);
        i2 = 1;
        if (i2 == 0) {
            i2 = 1;
        }
    }
}

```

Figure 8: Android VPN information gathering previous network connections.

The VPN absorbs even more information into its system by grabbing as much information on the hosting device as possible, including, but not limited to, IMEI, system software version, phone number, network information (provider, country, etc.), SIM information, seen in red in Figure 9.

```
public static String getPhoneInfo(Context context) {
    StringBuilder sb = new StringBuilder();
    try {
        TelephonyManager telephonyManager = (TelephonyManager) context.getSystemService("phone");
        sb.append("\nDeviceId(IMEI) = " + telephonyManager.getDeviceId());
        sb.append("\nDeviceSoftwareVersion = " + telephonyManager.getDeviceSoftwareVersion());
        sb.append("\nLineNumber = " + telephonyManager.getLineNumber());
        sb.append("\nNetworkCountryIso = " + telephonyManager.getNetworkCountryIso());
        sb.append("\nNetworkOperator = " + telephonyManager.getNetworkOperator());
        sb.append("\nNetworkOperatorName = " + telephonyManager.getNetworkOperatorName());
        sb.append("\nNetworkType = " + telephonyManager.getNetworkType());
        sb.append("\nPhoneType = " + telephonyManager.getPhoneType());
        sb.append("\nSimCountryIso = " + telephonyManager.getSimCountryIso());
        sb.append("\nSimOperator = " + telephonyManager.getSimOperator());
        sb.append("\nSimOperatorName = " + telephonyManager.getSimOperatorName());
        sb.append("\nSimSerialNumber = " + telephonyManager.getSimSerialNumber());
        sb.append("\nSimState = " + telephonyManager.getSimState());
        sb.append("\nSubscriberId(IMSI) = " + telephonyManager.getSubscriberId());
        sb.append("\nVoiceMailNumber = " + telephonyManager.getVoiceMailNumber());
    } catch (Exception unused) {
    }
    return sb.toString();
}
```

Figure 9: Android VPN information gathering network connections.

It is important to compare two VPNs from different distributors to show that this level of data collection from QI-ANXIN VPN is not normal for a VPN. We used ProtonVPN as the secondary VPN for comparison due to its reputation in the market as a privacy first VPN based in Switzerland, which is underpinned by Switzerland's robust privacy and data laws.¹³ Figures 10 and 11 (on the next page) compare the software's commands depicting the difference in data collected between the two VPNs. QI-ANXIN collects a lot more data that is personal to the user than Proton VPN does.

¹³ <https://www.dataguidance.com/notes/switzerland-data-protection-overview>

ANDROID API		
Common	Only in ch.protonvpn.android - 2.9.0.77	Only in com.legendsec.sslvpn - v771
Local File I/O Operations HTTP Connection Base64 Decode Message Digest TCP Socket Content Provider Starting Activity Inter Process Communication Starting Service Java Reflection Get System Service Sending Broadcast Crypto Get Installed Applications Base64 Encode Execute OS Command Loading Native Code (Shared Library) HTTPS Connection Query Database of SMS, Contacts etc Certificate Handling Android Notifications	TCP Server Socket URL Connection to file/http/https/ftp/jar WebView JavaScript Interface UDP Datagram Packet UDP Datagram Socket	Get Subscriber ID Get Network Interface information Set or Read Clipboard data WebView GET Request Get Software Version, IMEI/SV etc Get SIM Serial Number Get SIM Provider Details Get SIM Operator Name Get Phone Number Get WiFi Details Kill Process

Figure 10: Comparison of Application Programming Interface calls between ProtonVPN and QI-ANXIN VPN

	ANTI-VM	COMPILER	OBFUSCATOR	PACKER	DROPPER	MANIPULATOR	ANTI-ASSEMBLY	ANTI-DEBUG
Common	possible Build.SERIAL check Build.BOARD check Build.TAGS check Build.PRODUCT check Build.FINGERPRINT check Build.MODEL check possible VM check Build.MANUFACTURER check	r8						
ch.protonvpn.android - 2.9.0.77		r8 without marker (suspicious)						Debug.isDebuggerConnected() check
com.legendsec.sslvpn - v771	subscriber ID check network operator name check SIM operator check device ID check							

Figure 11: Comparison of Anti-Virtual Machine and Anti-Debug between ProtonVPN and QI-ANXIN VPN.

Part 2 KINGSOFT

Executive Summary

Part 2 is a technical analysis of Beijing Kingsoft Office Software Co., Ltd's (Kingsoft) anti-virus software product. Kingsoft through the "WPS Office" suite of software is an Official Supplier to the Beijing 2022 Olympic and Paralympic Winter Games.¹⁴ This analysis is based on Kingsoft's publicly available anti-virus software product. On 5 January 2021 the Whitehouse released Executive Order 13873 effectively banning the use of or transaction with WPS Office.¹⁵ As seen in Figure 12 "WPS Office" is a product of the company Beijing Kingsoft Software Co. Ltd which runs the Office software. The "WPS Office" suite is a pillar of the software company Kingsoft.

The Internet 2.0 intelligence team found the anti-virus installer file flagged as potentially containing malicious behaviours or properties. The installer also runs a file that potentially accesses data from internet browsers that are also running on the user's desktop computer. This file potentially copies all browser cookies as well as personal information and credentials. In our opinion the use of this anti-virus carries risk on use as the software provider could attain access to the internet usage history of the user. In the Android version of the Office suite the team found data collection and upload functions from iciba.com to Kingsoft.com which included GPS location, MAC address, installed applications, the phone number and sim information of the device, operating system information and other features including screenshots and clipboard access.

Who is Kingsoft?

Kingsoft also known as Beijing Kingsoft Software Co Ltd or Zuhai Jinshan Software Co Ltd is a Chinese Technology company that has four main business lines according to their website.¹⁶ These are Cheetah Mobile, Xishanju (entertainment and online games), Kingsoft Cloud and WPS Office Software. See figure 12 for a company structure overview. 邹涛 (Zou Tao) is the executive director and CEO for the conglomerate. He is also the listed legal representative or company director on all company branches. In total we found 52 entities that Zou Tao is either the legal representative or a director for. Zou Tao is the primary connecting figure that would make all these companies to be probably considered as related entities.

¹⁴ <https://www.beijing2022.cn/a/20200630/011345.htm>

¹⁵ <https://trumpwhitehouse.archives.gov/presidential-actions/executive-order-addressing-threat-posed-applications-software-developed-controlled-chinese-companies/>

¹⁶ https://www.baike.com/wikiid/4264469297351337403?from=wiki_content&prd=innerlink&view_id=50lg7tk4bm4000



邹涛 Zou Tao legal Representative for:

- Chengdu Xishanju Interactive Entertainment Technology Co., Ltd. 成都西山居互动娱乐科技有限公司
- Beijing Kingsoft Digital Entertainment Technology Co., Ltd. 北京金山数字娱乐科技有限公司
- Beijing Kingsoft Office Software Co., Ltd. 北京金山办公软件股份有限公司

and company director for Cheetah which is run by 傅盛 Fu Sheng

- Beijing Cheetah Mobile Technology Co., Ltd. 北京猎豹移动科技有限公司
- Beijing Kingsoft Security Software Co., Ltd. 北京金山安全软件有限公司

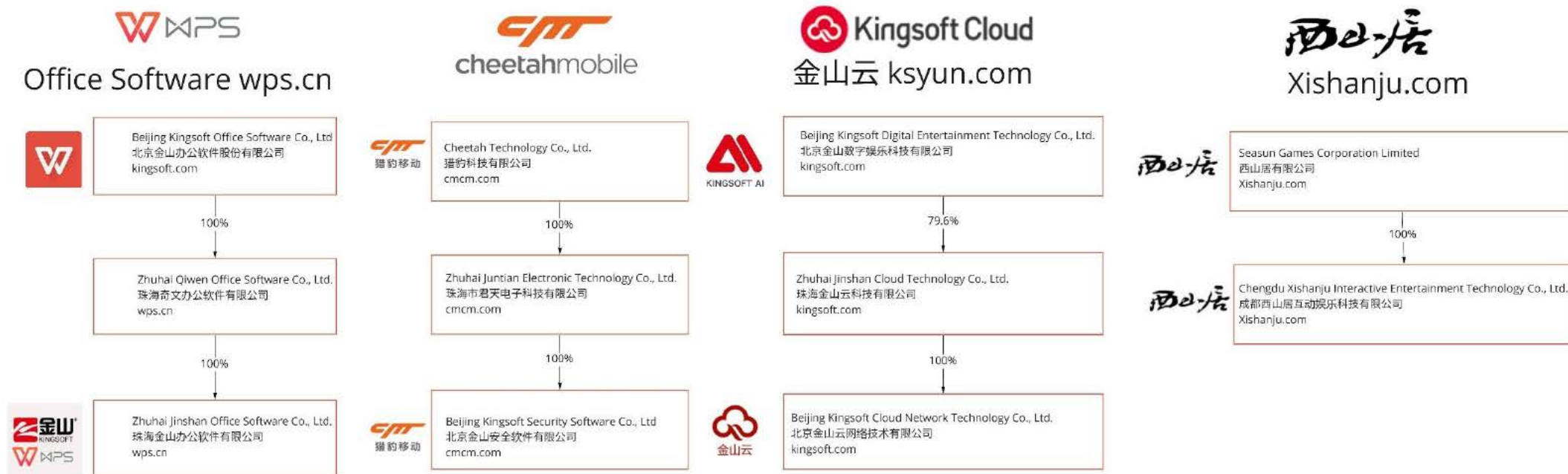


Figure 12: Kingsoft company structure.

Kingsoft Anti-virus

For thoroughness the antivirus software was copyrighted 1998-2010 Kingsoft Corporation and signed by the Zhuhai Kingsoft Office Software Co. Ltd. For Kingsoft antivirus creator the names Kingsoft and Jinshan are interchangeable as seen on Figure 12. The software has multiple http connections to kingsoftsecurity.com and info.duba.net both domains associated with Beijing Kingsoft Security Software Co., Ltd.¹⁷ When the software installer is downloaded the file has multiple different names as seen on VirusTotal. The antivirus installer is named “kingsoft-free-antivirus-*.*.*.exe”. These names are seen in Figure 13 below.¹⁸ On analysis there are a total of 2,809 files that come compiled inside of the installer.

Names ⓘ

kingsoft-free-antivirus-2010.11.06.318.exe
klivesetup
klivesetup.exe
5152a08d9bfd304cb6b4da95d3fe83c4b61f24cb181b9584a715a7db07a48fe100dec736729f1289679a7799c0b291b076fff74df82b715b51e85dd2893b309
kingsoft-free-antivirus.exe
file-4779217_exe
file-1533763_exe
output.2427236.txt
2427236

Figure 13: Kingsoft installer file names

Importantly, according to VirusTotal, the installer is detected by multiple Sigma rules as being CoViper malware, autorun keys modifications, code integrity check failure, port sweeping, as well as being detected by a Yara rule as having a Ursnif3 payload embedded inside of one of the packed files. It is also detected by two antivirus software as a “Trojan” and “Malware”.¹⁹ This is seen in Figure 14. VirusTotal is a community driven security platform that allows security professionals and companies to input flags for malicious files, IP addresses and websites. As it is a community driven platform, we cannot assess the veracity of these malicious flags for the installer by the members. However, as named and referenced researchers of the VirusTotal community have placed flags and their names to these citations, we would flag that multiple professionals have flagged the file as having malicious behaviours or properties and that this is definitely an indicator of probable risk on use.

¹⁷ <https://www.virustotal.com/gui/file/6080728583494c7d09ff91ede4cb5b3dcc0c56b82e043646ee6c25fc08cfc2a8/behavior>
<https://www.virustotal.com/gui/file/6080728583494c7d09ff91ede4cb5b3dcc0c56b82e043646ee6c25fc08cfc2a8/behavior>

¹⁸ <https://www.virustotal.com/gui/file/6080728583494c7d09ff91ede4cb5b3dcc0c56b82e043646ee6c25fc08cfc2a8/details>

¹⁹ <https://www.virustotal.com/gui/file/6080728583494c7d09ff91ede4cb5b3dcc0c56b82e043646ee6c25fc08cfc2a8>

The screenshot shows the VirusTotal analysis page for a file named `klivesetup.exe` (SHA256: 6080728583494c7d09ff91ede4cb5b3dccc56b82e043646ee6c25fc08cfc2a8). The file is 18.23 MB and was uploaded on 2021-09-30 19:35:30 UTC. It has a Community Score of 2/53 and is flagged as malicious by 2 security vendors. The file is categorized as EXE. The analysis shows several detection rules, including Crowdsourced YARA Rules, Crowdsourced Sigma Rules, and Crowdsourced IDS Rules. The Sigma Rules section shows matches for rules like `Ursnif3`, `GoViper Malware`, `Autorun Keys Modification`, and `Failed Code Integrity Checks`. The IDS Rules section shows a match for the rule `(port_scan) TCP filtered portsweep`.

2 security vendors flagged this file as malicious

6080728583494c7d09ff91ede4cb5b3dccc56b82e043646ee6c25fc08cfc2a8

klivesetup.exe

18.23 MB Size

2021-09-30 19:35:30 UTC

3 months ago

EXE

calls-wm detect-debug-environment direct-cpu-clock-access invalid-signature long-sleeps nls overlay peeka revoked-cert runtime-modules signed software-collection

Community Score

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY 2

Crowdsourced YARA Rules

Matches rule `Ursnif3` by kevoreilly from ruleset Ursnif3 at <https://github.com/kevoreilly/CAPEv2>

↳ `Ursnif Payload`

Crowdsourced Sigma Rules

CRITICAL 2 HIGH 0 MEDIUM 11 LOW 1

2 matches for rule `GoViper Malware` by Ariel Millhuel from SOC Prime Threat Detection Marketplace

↳ `GoViper is a Wiper that appears during the COVID-19 situation`

11 matches for rule `Autorun Keys Modification` by Victor Sergeev, Danil Yugoslavskiy, GL... from Sigma Integrated Rule Set (GitHub)

↳ `Detects modification of autostart extensibility point (ASEP) in registry.`

1 match for rule `Failed Code Integrity Checks` by Thomas Patzke from Sigma Integrated Rule Set (GitHub)

↳ `Code integrity failures may indicate tampered executables.`

Crowdsourced IDS Rules

HIGH 0 MEDIUM 1 LOW 0 INFO 0

Matches rule `(port_scan) TCP filtered portsweep` from Snort registered user ruleset

↳ `attempted-recon`

Figure 14: Kingsoft installer detection from virustotal.

As the installer runs, we found a file that we saw risk with. This was `kcookie.bin.exe`. This file was 245 KB in size and comes with the installer. See Figure 15.

Name	Date modified	Type	Size
bcinstall32.bin.zip	1/4/2022 9:50 AM	ZIP File	78 KB
driver32.bin.zip	1/4/2022 9:50 AM	ZIP File	76 KB
kavlog2.bin.zip	1/4/2022 9:49 AM	ZIP File	251 KB
kavsetup.bin.zip	1/4/2022 9:47 AM	ZIP File	18,178 KB
kcookie.bin.zip	1/4/2022 9:46 AM	ZIP File	245 KB
kdevmgr.bin.zip	1/4/2022 9:49 AM	ZIP File	40 KB
kingsoft-free-antivirus-2010.11.06.318....	1/4/2022 9:38 AM	BIN File	18,670 KB
kislive.bin.zip	1/4/2022 9:44 AM	ZIP File	324 KB
kismain.bin.zip	1/4/2022 9:51 AM	ZIP File	156 KB
kisuisp.bin.zip	1/4/2022 9:45 AM	ZIP File	221 KB

Figure 15: Kingsoft installer files

As we ran the executable file through an online sandbox (any.app.run) it flagged as a 75 per cent risk, as it appears to access personal information and takes credentials and cookies from the internet browser that the user is running. There is also a command line argument in the source code that states “getandsendcookies”. See Figure 16 for a detailed view of the file from the sandbox.

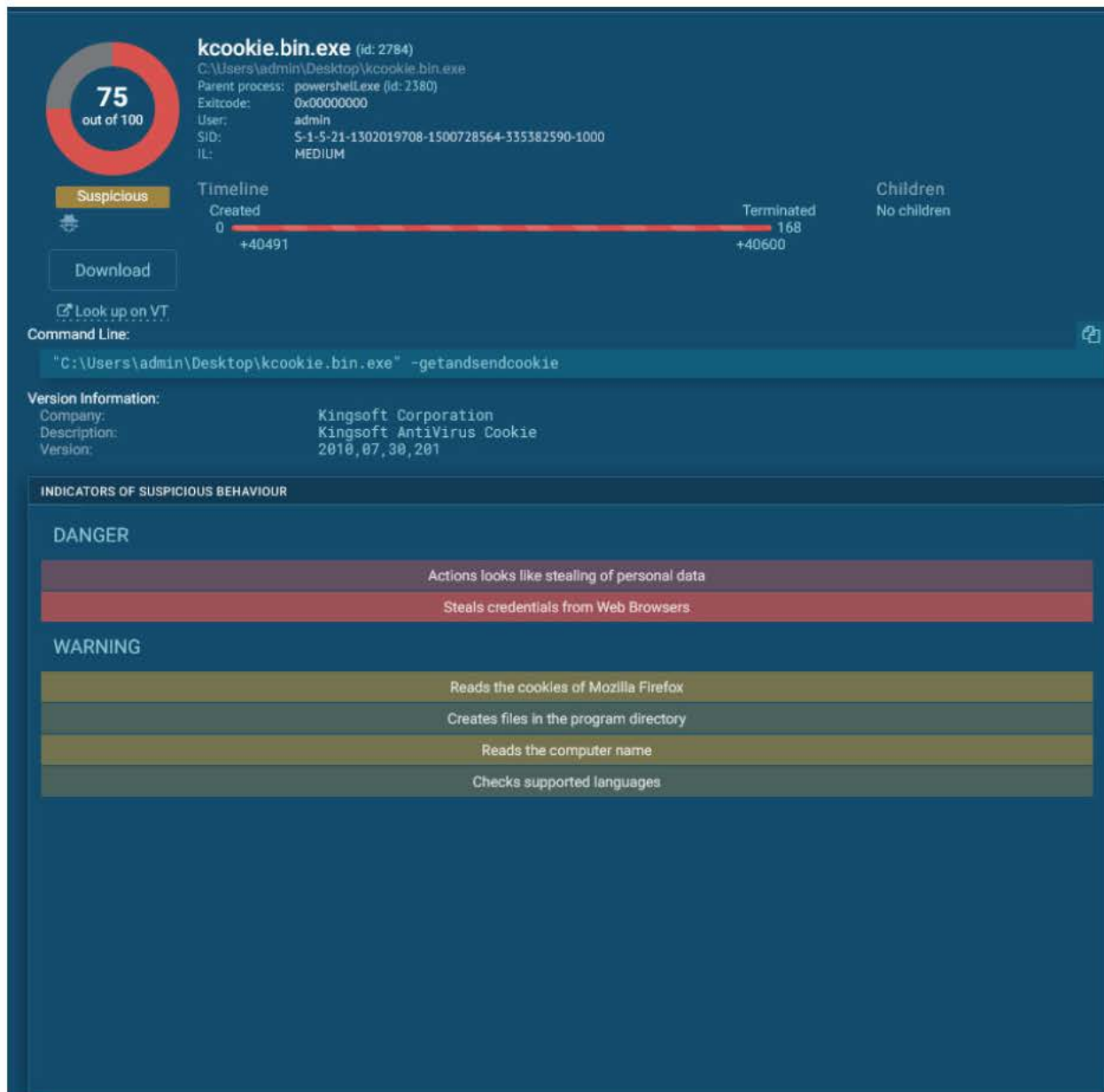


Figure 16: kcookie.exe taken from the Kingsoft AV installer detected on the sandbox.

As the installation continued it appeared that “kcookie.exe” is responsible for accessing the internet browsers information, as you can see in the following image “kcookie.exe” takes cookies out of the Firefox browsers SQLite file. See Figure 17 for this command.

```

...xt:004028c9 6a 19      PUSH     0x19
...xt:004028cb 69 c0 45      PUSH     u_Mozilla\Firefox\Profile_
47 00
...xt:004028d0 8d 4c 24 18    LEA      EAX,[EIP + 0x10]
...xt:004028d4 m5 b7 ee      CALL     KTE::CString::CStringT<wchar_t,
ff ff
...xt:004028d9 8b 74 24 10    MOV     EAX,dword ptr [ESP + 0x10]
...xt:004028dd 56          PUSH     ESI
...xt:004028de ff 15 10      CALL     dword ptr [->SHIMAP1.DLL::Pa
e2 46 00
...xt:004028e4 95 c0      TEST     EAX,EAX
...xt:004028e6 75 17      JNZ     LAB_004028ff
...xt:004028e8 56          PUSH     ESI
...xt:004028e9 68 f8 45      PUSH     s_Get_firefox_cookie_path_fa
47 00
...xt:004028ee e8 7d 12      CALL     FUN_00403b70
00 00
...xt:004028f3 83 c4 08      AND     EAX,0x8
...xt:004028f6 89 7e 24 2c    MOV     dword ptr [ESI + 0x2c],EDI
...xt:004028fa 9d 46 f0      LEA      EAX,[ESI + -0x10]
...xt:004028fd eb 9d      JMP     LAB_0040289c

LAB_004028ff
...xt:004028ff e8 60 33      CALL     FUN_00405e64
00 00
...xt:00402904 33 c9      XOR     EAX,EAX
...xt:00402906 85 c0      TEST     EAX,EAX
...xt:00402908 0f 95 c1      SETNB    CL
...xt:0040290b 85 c9      TEST     EAX,EAX
...xt:0040290d 74 04      JNZ     LAB_00400919
...xt:0040290f 68 05 40      PUSH     0x80004005
00 80
...xt:00402914 e8 57 f0      CALL     FUN_00401970
.. ..

```

```

35 FUN_00402cd0(0xstack24,1,"Software\Microsoft\Windows\CurrentVersion\Explorer\Shell
Guides\");
40 ;
41 uStack4_0_1_ = 0;
42 iVar6 = FUN_00402010(0xstack22);
43 uStack4_0_1_ = 1;
44 piVar4 = (int *) (0xstack24 + -4);
45 LOCK();
46 iVar5 = *piVar4;
47 *piVar4 = *piVar4 + -1;
48 if (iVar5 == 1 || iVar5 + -1 < 0) {
49     (**(code **)) (**(int **)) (0xstack24 + -0x10 + 4)) ((int **) (0xstack24 + -0x10));
50 }
51 uStack4 = (uint) uStack4_0_1_ << 0;
52 piVar4 = (int *) (0xstack20 + -2);
53 LOCK();
54 iVar5 = *piVar4;
55 *piVar4 = *piVar4 + -1;
56 if (iVar5 == 1 || iVar5 + -1 < 0) {
57     (**(code **)) (**(int **)) (0xstack20 + -8 + 4)) (0xstack20 + -8);
58 }
59 if (iVar6 == 0) {
60     FUN_00403b70("Get Firefox cookie path fail reason is: read regedit fail!\n");
61     ppiVar7 = (int **) (0xstack32 + -8);
62 }
63 else {
64     Append((CString::CStringT<wchar_t,0> *) (0xstack32, (wchar_t
*) L"\\Mozilla\\Firefox\\Profile",0x10);
65     pWVar2 = pWStack32;
66     iVar8 = pathFileExists(pWStack32);
67     if (iVar8 == 0) {
68         FUN_00403b70("Get Firefox cookie path fail the reason is : %s folder didnot exist!\n");
69         ppiVar7 = (int **) (0xstack32 + -8);
70     }

```

Figure 17: kcookie.exe source code.

WPS Office

The WPS Office software was copyrighted 2021 Kingsoft Corporation which is run by Beijing Kingsoft Office Software Co., Ltd as seen in figure 12. The WPS Office software comes in four major packages which are very similar and is compatible with the Microsoft Office Software. The software is marketed as being free to download. The packages are titles Writer, Presentation, Spreadsheet and PDF files. The packages are usable across all major platforms including Windows, Mac, Android, IOS and Linux. They also have a cloud platform and templates store.²⁰ For the purpose of this analysis we will only refer to information pertaining to the android version of the software package.

In our analysis of the android version of the software we found several items that in our opinion can be used for data gathering purposes. The software gathers nearly all information about the current system it is on such as: GPS location, MAC address, installed applications, the phone number of the device, operating system information and taking screenshots (it is worth mentioning that these screenshots may be used to provided previews for files). It also takes information on device details and network and sim information. As seen in figures 18 to 22 below. We did note that a lot of the features including access to clipboard probably enhance the compatibility and user experience of the software. We also noted that the software has multiple http connections to upload its user information back to iciba.com which links back to kingsoft.com and that this information probably comes under Chinese Government jurisdiction. This upload function can be seen in figures 23 and 24.

From a risk perspective we would state that gathering network connection information like sim data and location data in unnecessary collection for the function of office software. Complete access to clipboard, camera access and device details would be needed for functional use but the broad scope of the data collection shown is an example of the surveillance culture which imposed by the Chinese Government national security legislation. From a risk perspective users should be aware that this data is able to be processed and sent back via network connections to Kingsoft.com which would place the data under Chinese Government jurisdiction.

²⁰ <https://www.wps.com/>

```
public class DeviceInfo implements Parcelable {
    public static final Parcelable.Creator<DeviceInfo> CREATOR = new a();
    @SerializedName("identify_info")
    @Expose
    public IdentifyInfo B;
    @SerializedName("client_info")
    @Expose
    public ClientInfo I;
    @SerializedName("os_info")
    @Expose
    public OsInfo S;
    @SerializedName("net_info")
    @Expose
    public NetInfo T;
    @SerializedName("additional_info")
    @Expose
    public AdditionalInfo U;
    @SerializedName("ext")
    @Expose
    public String V;
    @SerializedName("status")
    @Expose
    public int W;
    @SerializedName("register_time")
    @Expose
    public long X;
}
```

Figure 18: WPS Office Android software gathering information in preparation of sending out to other networks

```
public class IdentifyInfo implements Parcelable {
    public static final Parcelable.Creator<IdentifyInfo> CREATOR = new a();
    @SerializedName(MopubLocalExtra.APP_ID)
    @Expose
    public String B;
    @SerializedName("user_id")
    @Expose
    public String I;
    public String S;
    @SerializedName("device_id")
    @Expose
    public String T;
    @SerializedName("device_name")
    @Expose
    public String U;

    /* loaded from: classes.dex */
    public static class a implements Parcelable.Creator<IdentifyInfo> {
        /* renamed from: a */
        public IdentifyInfo createFromParcel(Parcel parcel) {
            return new IdentifyInfo(parcel);
        }

        /* renamed from: b */
        public IdentifyInfo[] newArray(int i) {
            return new IdentifyInfo[i];
        }
    }

    public IdentifyInfo() {
        this.B = "wps-office";
    }
}
```

Figure 19: WPS Office Android software obtaining user ID, Device ID and Device name.


```

public enum b {
    MEMORY,
    DISK
}

boolean a(a aVar, int i, OutputStream outputStream);

boolean b();

boolean c();

void d(b bvar);

hq1 e(int i, int i2);

boolean f();

int g();

int getHeight();

int getWidth();

void recycle();
}

```

Figure 20: WPS Office Android software gathering device height and width dimensions.

```

private ClientMetadata(@NonNull Context context) {
    ApplicationInfo applicationInfo;
    Preconditions.checkNotNull(context);
    Context applicationContext = context.getApplicationContext();
    this.q = applicationContext;
    this.r = (ConnectivityManager) applicationContext.getSystemService("connectivity");
    this.n = a(applicationContext);
    PackageManager packageManager = applicationContext.getPackageManager();
    String packageName = applicationContext.getPackageName();
    this.o = packageName;
    try {
        applicationInfo = packageManager.getApplicationInfo(packageName, 0);
    } catch (PackageManager.NameNotFoundException unused) {
        applicationInfo = null;
    }
    if (applicationInfo != null) {
        this.p = (String) packageManager.getApplicationLabel(applicationInfo);
    }
    TelephonyManager telephonyManager = (TelephonyManager) this.q.getSystemService(writer.g.bfE);
    if (telephonyManager != null) {
        this.a = telephonyManager.getNetworkOperator();
        this.b = telephonyManager.getNetworkOperator();
        if (telephonyManager.getPhoneType() == 2 && telephonyManager.getSimState() == 5) {
            this.a = telephonyManager.getSimOperator();
            this.c = telephonyManager.getSimOperator();
        }
        if (MoPub.canCollectPersonalInformation()) {
            this.d = telephonyManager.getNetworkCountryIso();
            this.e = telephonyManager.getSimCountryIso();
        } else {
            this.d = "";
            this.e = "";
        }
        try {
            this.f = telephonyManager.getNetworkOperatorName();
            if (telephonyManager.getSimState() == 5) {
                this.g = telephonyManager.getSimOperatorName();
            }
        } catch (SecurityException unused2) {
            this.f = null;
            this.g = null;
        }
    }
    this.h = new MoPubIdentifier(this.q);
}

```

Figure 21: WPS Office Android software gathering device sim information

```

public enum b {
    NETWORK("network"),
    GPS("gps");

    public final String B;

    b(String str) {
        this.B = str;
    }

    public final boolean b(Context context) {
        int i = a.ordinal();
        if (i == 1) {
            return n5d.a(context, "android.permission.ACCESS_FINE_LOCATION");
        }
        if (i != 2) {
            return false;
        }
        return n5d.a(context, "android.permission.ACCESS_FINE_LOCATION");
    }

    @Override // java.lang.Enum, java.lang.Object
    public String toString() {
        return this.B;
    }
}

public static Location a(Context context) {
    return c(b(context, b.GPS), b(context, b.NETWORK));
}

public static Location b(Context context, b bVar) {
    if ((VersionManager.isProVersion() && VersionManager.N()) || !bVar.b(context)) {
        return null;
    }
    try {
        return ((LocationManager) context.getSystemService("location")).getLastKnownLocation(bVar.toString());
    } catch (IllegalArgumentException unused) {
        dii.a("LocationService", "Failed to retrieve location: device has no " + bVar.toString() + " location provider.");
        return null;
    } catch (NullPointerException unused2) {
        dii.a("LocationService", "Failed to retrieve location: device has no " + bVar.toString() + " location provider.");
        return null;
    } catch (SecurityException unused3) {
        dii.a("LocationService", "Failed to retrieve location from " + bVar.toString() + " provider: access appears to be disabled.");
        return null;
    }
}

```

Figure 22: WPS Office Android software gathering location data


```
private HttpPost d(String str) {  
    try {  
        String encode = URLEncoder.encode(str, "UTF-8");  
        StringBuffer stringBuffer = new StringBuffer();  
        stringBuffer.append("http://dict-mobile.iciba.com/interface/index.php");  
        stringBuffer.append("?c=word");  
        stringBuffer.append("&list=");  
        stringBuffer.append("1");  
        stringBuffer.append("&client=");  
        stringBuffer.append(1);  
        String valueOf = String.valueOf(System.currentTimeMillis() / 1000);  
        stringBuffer.append("&timestamp=");  
        stringBuffer.append(valueOf);  
        stringBuffer.append("&sign=");  
        stringBuffer.append(f("word#ICIBA!(*&R$@#LOVE#1" + valueOf).substring(5, 21));  
        stringBuffer.append("&uuid=");  
        stringBuffer.append(b(this.e));  
        stringBuffer.append("&sv=");  
        stringBuffer.append("android" + Build.VERSION.RELEASE);  
        stringBuffer.append("&v=");  
        stringBuffer.append("2.0.4");  
        stringBuffer.append("&uid=");  
        stringBuffer.append("&tc=");  
        stringBuffer.append(o.f);  
        HttpPost httpPost = new HttpPost(stringBuffer.toString());  
        ArrayList arrayList = new ArrayList();  
        arrayList.add(new BasicNameValuePair("word", encode));  
        try {  
            httpPost.setEntity(new UrlEncodedFormEntity(arrayList, "UTF-8"));  
        } catch (UnsupportedEncodingException e) {  
            e.printStackTrace();  
        }  
        return httpPost;  
    } catch (Exception unused) {  
        return null;  
    }  
}
```

Figure 23: WPS Office Android software uploading information to iciba.com

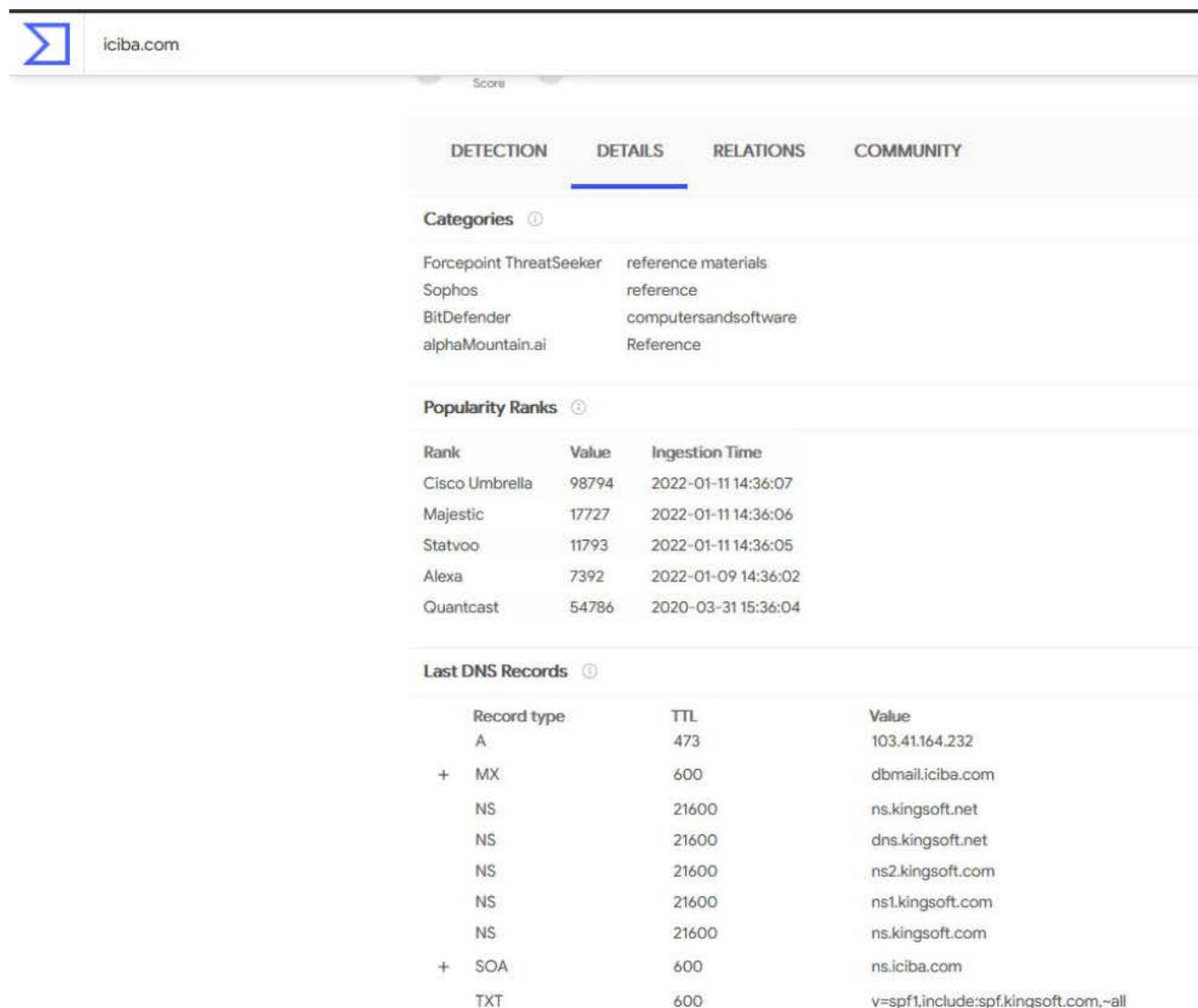


Figure 24: WPS Office Android software uploading information to iciba.com which links back to kingsoft.com data from virustotal.com

Key Terms used in report

Anti-Virtual Machine (anti-VM) and Anti-Debug	Anti-virtual machine (Anti-VM) and Anti-debugging techniques thwart attempts at malware analysis. With these techniques, the malware attempts to detect whether it is being run inside a virtual machine. If a virtual machine is detected, it can act differently or simply not run. ²¹
Application Programming Interface (API)	API is a software intermediary that allows two applications to talk to each other. Each time you use an app like Facebook, send an instant message, or check the weather on your phone, you're using an API. ²²
IMEI	The International Mobile Equipment Identity (IMEI) is a number, usually unique to identify mobile phones, as well as some satellite phones. ²³
IMSI	An International Mobile Subscriber Identity (IMSI) is a 15-digit number for every user in a Global System for Mobile communication (GSM). The IMSI is used by Mobile Network Operators (MNOs) and is an important part of the Subscriber Identity Module (SIM) profile. ²⁴
MAC Addresses	Every device connected on a network has a Media Access Control (MAC) address, that uniquely identifies it. The MAC address is a 12-digit hexadecimal number that is most often displayed with a colon or hyphen separating every two digits (an octet), making it easier to read. ²⁵
Malware	Malware (malicious software) is a file or code, typically delivered over a network, that infects, explores, steals or conducts virtually any behavior an attacker wants.
SIM	A SIM card, or subscriber identity module, is a small card in your cellphone that connects you to the network. A SIM card contains a phone number, and lets you make phone calls, send text messages, and more.
Trojan	A type of malware that downloads onto a computer disguised as a legitimate program.
Virtual Private Network (VPN)	A virtual private network, or VPN, is an encrypted connection over the Internet from a device to a network. The encrypted connection helps ensure that sensitive data is safely transmitted. ²⁶

²¹ <https://www.oreilly.com/library/view/practical-malware-analysis/9781593272906/ch18.html>

²² <https://www.mulesoft.com/resources/api/what-is-an-api>

²³ https://en.wikipedia.org/wiki/International_Mobile_Equipment_Identity#cite_note-3gppspec-1

²⁴ <https://www.simoniot.com/what-is-an-imsi/>

²⁵ <https://slts.osu.edu/articles/whats-a-mac-address-and-how-do-i-find-it/>

²⁶ https://www.cisco.com/c/en_a/products/security/vpn-endpoint-security-clients/what-is-vpn.html

Internet 2.0

RELENTLESS SECURITY

AUSTRALIA

L1, 18 National Circuit, Barton
ACT, 2600

ABN: 17 632 726 946

UNITED STATES

Suite 100 211 N Union St
Alexandria, 22314

EIN: 86-1567068

E: contact@internet2-0.com
AUS: 1300 583 007
INTL: +611300583007

internet2.0

MILITARY-GRADE

CYBER PROTECTION

WeChat Analysis

Authors

David Robinson

Robert Potter

Thomas Perkins

Luke McWilliams



Table of Contents

Executive Summary	2
Introduction	4
Part 1 – Technical Analysis of WeChat.....	7
Part 2 – Chinese Propaganda Department Contracts.....	13
Survey.....	17

Executive Summary

This report is a technical analysis of the source code of WeChat versions 8.0.15s as well as an analysis of publicly available Chinese government procurement documents. It has been prepared by Internet 2.0 for policy makers and legislators to make evidence-based decisions. WeChat is significant because it is a digital communications gateway to China as it has a monopoly on digital communication in Mandarin. Nearly all citizens of the Peoples Republic of China (PRC) use WeChat in their daily lives to communicate by private message, to make payments for services, and as an application to connect through social media. WeChat has an estimated monthly user base of more than 1 billion people.¹ While most 'Weixin' (the domestic version of WeChat in China) users reside in China, WeChat also has an active user base globally including Australia (0.6 million users)², the UK (1.3 million users)³ and the United States (1.5 million users)⁴.

This report has been broken into two primary parts. Part One is a technical analysis of WeChat and Part Two is an analysis of WeChat procurement contracts with Chinese Communist Party (CCP) propaganda departments. We have based our analysis primarily on WeChat's written submission on 30 September 2020 to the Australian Parliamentary Select Committee on Foreign Interference through Social Media (the Select Committee). The context of this report is in the statements made by WeChat to this Select Committee as well as the deteriorated legal autonomy of Hong Kong over the past two years. This erosion of legal autonomy is important because WeChat has structured its operational architecture, terms of service and company structure on the consensus that Hong Kong is a more compatible legal jurisdiction to the international community. In our opinion the creation and enforcement of the Hong Kong National Security Legislation has changed the nature of this compatibility and we expect the international community will be asking more questions of WeChat's handling of their citizens' data.

Our summary findings in Part One are:

- Tencent Holdings (owners of WeChat) refers to WeChat and Weixin as sister applications. In our opinion the technical architecture is more like Weixin as the parent and WeChat as the child. This is based on URL⁵ architecture and management as well as probable focus due to the higher Weixin user count. Further, Weixin has all available functions and the app is governed by the CCP whereas WeChat has limited functionality and has to manage the myriad of international jurisdictions, while allowing communication with Weixin users in mainland China.

¹ <https://www.cnbc.com/2019/02/04/what-is-wechat-china-biggest-messaging-app.html>

² <https://www.zdnet.com/article/wechat-sets-the-record-straight-for-its-690000-aussie-users/>

³ <https://financesonline.com/wechat-statistics/>

⁴ <https://99firms.com/blog/wechat-statistics/#gref>

⁵ URL is short for Uniform Resource Locator is the address of a given unique resource on the Web.

- WeChat has a sophisticated technical feature that prohibits SSL⁶ pinning of the software to enable analysis under sandbox conditions. WeChat also uses AES ECB7 for encryption. This encryption method is used for all log files while stored on the device and while the logs are posted back to their central logging server. In our opinion it can be interpreted as a deliberate measure to avoid analysts from having any insight into the log management of WeChat user's data.
- WeChat states that all its servers are kept outside of mainland China. This is critical because our analysis uncovered that while all chat and audio/video calls are probably managed by servers internationally, all user data that WeChat logs and posts to its logging server about its users goes directly to Hong Kong. We argue it is reasonable to consider that under the Hong Kong National Security Legislation there is little difference between Hong Kong resident servers and those on mainland China.
- We ascertained that WeChat users can interact directly with servers on mainland China and we consider that it is becoming quite difficult to manage the competing jurisdictional priorities of a software platform that connects mainland China under the CCP and international data privacy laws such as General Data Protection Regulation in the European Union and the California Consumer Privacy Act in California.
- During our analysis we found no evidence that contradicts the claim that chats are not stored outside of the user's device. There is no logging attempt on chats we could find. The only security flag we saw was that WeChat has the potential to access all the data in the user's clipboard (see Figure 6). This is a risk to flag as users that have a password manager rely on the clipboard to copy and paste their passwords.
- WeChat Pay has functionality in multiple global currencies but it is tied to the US Dollar as the default base currency and posts all its user logs to a Hong Kong server.

Our summary findings in Part Two are:

- WeChat's submission to the Select Committee on combating disinformation states it prohibits accounts that spread content that breaches any applicable laws or regulations, or content which may constitute a genuine risk of harm or direct threat to public safety. WeChat also prohibits paid promotional content regarding:
 - a candidate for an election;
 - a political party or any elected or appointed government official appealing for votes for an election;
 - appeals for financial support for political purposes; and
 - a law, regulation or judicial outcome.

In our opinion these policies were contradictory to democratic free speech and must be difficult to implement. We see the structural pressure the CCP has over WeChat by barring democratic speech.

⁶ SSL is short for Secure Socket Layer, and an SSL certificate will give you a way of encrypting information while it travels online.

⁷ AES (Advanced Encryption Standard) is one of the most used algorithms for block encryption. ECB (Electronic Code Book) mode is one of five modes of AES.

- We also found these policies do not align with WeChat's own practices on mainland China. This means that democratic advertising is banned but authoritarian speech is not. This gives the application an explicit authoritarian bias. We found 10 contracts between 2016 and 2019 from Chinese Communist Party Propaganda Departments to conduct influence or propaganda actions over Tencent platforms on behalf of CCP governing departments. The sum of these contracts is 2,327,000 Yuan (see Figure 9). These contracts were awarded to subsidiaries in China owned by either Tencent Holdings or companies which Ma Huateng (马化腾), Chairman and CEO of Tencent, has a controlling stake in.
- We consider that these policies favour the CCP to the disadvantage of all other democratic governments and in nature are contradictory to their written testimony to the Select Committee. This policy bias privileges authoritarian voices and renders democratic speech as second class and prohibited. To see Tencent defend this as a feature in their submission was noteworthy.
- Internet 2.0 conducted an industry survey in response to our initial reaction to these findings. The question was asked with no context nor mention of WeChat. Respondents could infer any platform or data was behind this question. The question was "Should the data of electoral/political communications be housed within the sovereignty of the country conducting its own electoral process?". Out of the respondents 97 per cent agreed with the above statement with an affirmative yes. All respondents did so under their own name.

Introduction

WeChat is the dominant digital platform for communications in Mandarin and digital gateway to China. Nearly all PRC citizens, and Mandarin speakers wishing to connect digitally with China, use WeChat in their daily lives to communicate by private message, to make payments for services and as an application to connect through social media. WeChat is the dominant communication platform for communicating in Mandarin. This was best described by Li Yuan writing for the New York Times:

*"There's no company in the world like Tencent. It's a true monopoly on many levels. It wields the kind of influence in China that Facebook, Amazon, Apple and Google can only aspire to."*⁸

Because of this monopoly most Chinese based globally also use it as a preferred social media and communication platform. Anyone wishing to communicate using Mandarin on social media, but are overseas, are shaped into using WeChat through this monopoly. This is true even if both audiences are based outside of mainland China.

We have analysed the application for several reasons, firstly with the explosion in dis- and misinformation social media applications themselves play a large role in policing and supporting what our society holds in consensus to be true. As the dominant Mandarin messaging and social media platform globally, WeChat is governed predominantly under the CCP. This is through their Hong Kong stock market listing as well as the fact that most of their users are geographically based

⁸ <https://www.nytimes.com/2021/06/02/technology/china-tencent-monopoly.html>

in mainland China. We consider that WeChat is under structural pressure to support the CCP's rule of law. We believe this structural market pressure may come at a disadvantage to all other governments' rule of law.

Secondly, in our opinion, WeChat has structured its international operations and terms of services in the context of Hong Kong's legal autonomy as a more compatible legal jurisdiction for international governments. Hong Kong's legal autonomy has been eroded over the last two years by the creation and enforcement of Hong Kong's National Security Legislation. In our opinion, we now struggle to see any difference between the legal rights of user's data on servers based in Hong Kong over mainland China. Given Hong Kong's changing legal compatibility, international policy makers and legislators will be asking more questions of WeChat's handling of their citizens' data.

It is immediately noticeable in WeChat's submission to the Select Committee that it is operating under the culture that Hong Kong still affords legal autonomy from mainland China through its stated operational policies. In our opinion the perceptions on this question of legal autonomy are changing. This changing perception has informed our analysis and findings.

Lastly, the Prime Minister of Australia has had difficulty himself with his WeChat account after he was deregistered as its authorized user effectively removing or blocking his access to the platform.⁹ Prime Minister Morrison uses WeChat to reach out to his 75,000 followers in Australia through Mandarin. We note other prominent Australian politicians, including Leader of the Australian Labor Party, Anthony Albanese, use WeChat to also communicate to Australian residents through Mandarin. Key concerns raised by the media and politicians include the extent to which platforms such as WeChat, with structural pressure under the CCP, could control and censor accounts and whether this amounts to foreign interference by the CCP. According to recent media reports, the Prime Minister's Office is now in direct discussions with WeChat to resolve the matter.¹⁰

As Tencent is the parent company of WeChat we found it necessary to record the complex company structure that has been built by its founder, CEO and Chairman Ma Huateng. This structure records Tencent as the parent of WeChat, which is registered in the Cayman Islands and listed on the Hong Kong Stock Exchange as well as several of its sister companies registered on mainland China that were recorded within Chinese government propaganda procurement records (see Figure 1 below).

⁹ <https://www.afr.com/politics/federal/wechat-s-hijacking-of-pm-could-lead-to-banning-of-chinese-app-20220124-p59qqx>

¹⁰ <https://www.canberratimes.com.au/story/7600856/wechat-parent-tencent-reaches-out-to-pm/>

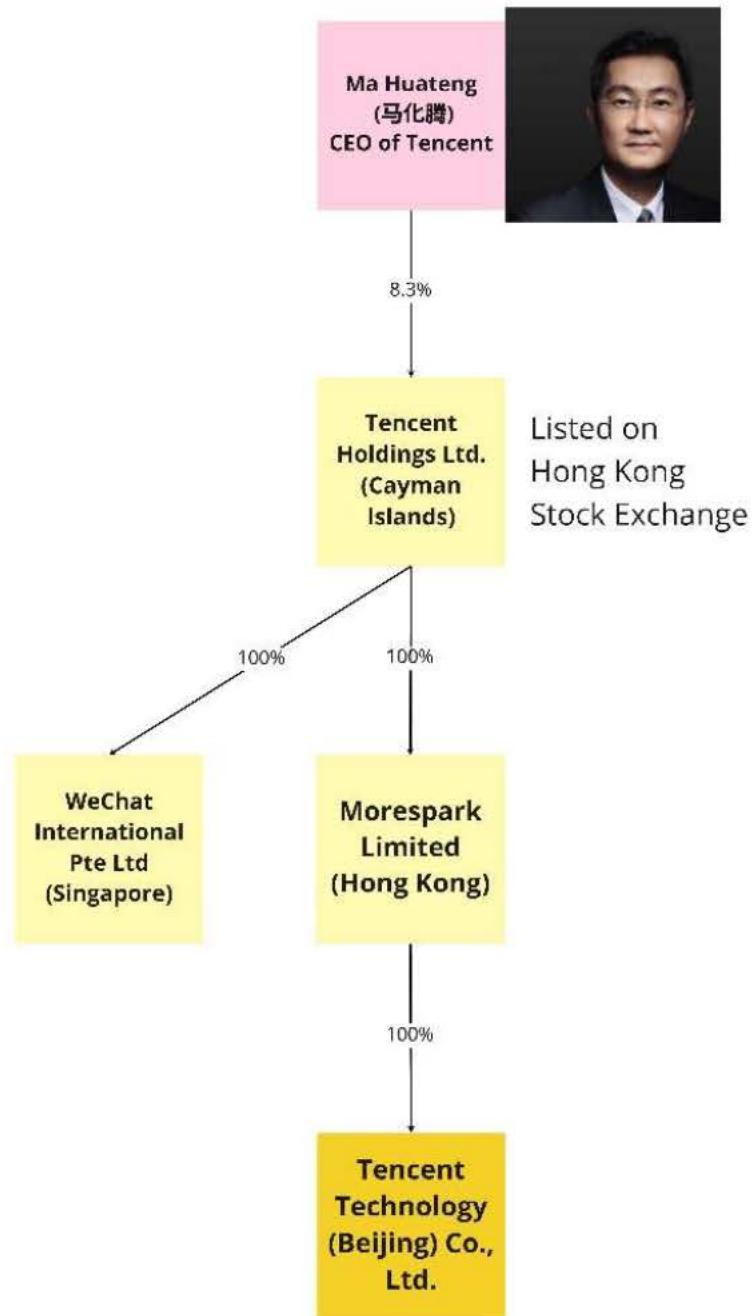


Figure 1: Ownership structure of Tencent and WeChat.

Part 1 – Technical Analysis of WeChat

There are two versions of the WeChat application. Weixin and WeChat. Tencent refers to the duopoly of WeChat and Weixin as “interoperable sister” applications. In this technical analysis we only conducted analysis of the WeChat version 8.0.15 available globally but not in China. We note that Tencent states:

“While each is based on a different server architecture and subject to different laws, WeChat users can chat and share with Weixin users...this was a conscious decision designed to serve different users while ensuring compliance with applicable laws across different jurisdictions.”¹¹

WeChat advises that users outside of China are not governed by Chinese law, as opposed to Weixin users, and that all servers for overseas users are located outside of mainland China. Users who register with a Chinese mobile phone number will be made a Weixin user, while users who register with a non-Chinese mobile number will be made a WeChat user.

Further, WeChat does allow users to access certain Weixin functions through the WeChat application.¹² WeChat advised in its submission to the Select Committee that when it allows some Weixin features through WeChat it advises the user they are coming under Weixin terms of service. This is not the case for core functions including messaging, audio/video calls and moments.

We determined that Weixin is probably hierarchically higher than WeChat in its architecture management. A better term would be parent and daughter than sister apps. WeChat uses Weixin URLs for its support and agreement functions as below. There are a total of 1207 references to Weixin URLs in the WeChat source code and Tencent’s QQ domains are the higher hierarchical logging server:

- (new Image).src="https://support.weixin.qq.com/cgi-bin/mmsupportmeshnologicsvr-bin/cube?biz=3512&label=wx110.support.frozen"
- <url><![CDATA[https://.weixin.qq.com/agreement?lang=en&cc=US&s=privacy&scene=reg&v=1&needopenplatform=0]]></url>
- <url><![CDATA[https://res.wx.qq.com/t/wx_fed/weixin_portal/res/static/js/agreement_a5f6151.js]]></url>

From a software development and data management perspective we assess that this is a complex architecture management problem. The questions that immediately come up are: If a Weixin user and WeChat user are communicating or a WeChat user is accessing Weixin functions where is the data stored? How does Tencent manage the competing legal priorities of the CCP that governs Weixin users and international residents that come under laws such as the GDPR in Europe or the CCPA in California? At some point there will be compromise on the storage of user’s data as it travels through competing jurisdictions. We noted this is a complexity for any software company operating

¹¹ WeChat Submission to Australian Select Committee on Foreign Interference through Social Media on 30 September 2020.

¹² <https://www.zdnet.com/article/wechat-sets-the-record-straight-for-its-690000-aussie-users/>

with interoperability between China and internationally. This is also true for Tesla as Keith Zhai writing in the Wall Street Journal noted:

“Some Chinese state-owned companies, along with military staff, have been restricted from using Tesla Inc.’s vehicles over Beijing’s concerns that data the cars gather could be a source of national-security leaks. Tesla’s chief executive, Elon Musk, said the company would never provide the U.S. government with data collected by its vehicles in China or other countries.”¹³

Anti-sandbox feature

The Internet 2.0 team have analysed many mobile applications in our careers. We firstly must compliment the engineers that worked on WeChat as it had some unique and sophisticated technical features that made these findings initially difficult to come to. Their ability to avoid SSL pinning during static sandbox analysis as well as the encryption using AES ECB enabled the application to encrypt and obfuscate how WeChat managed and pushed its user logs. From a professional standpoint it would be unfair not to call out and compliment this skilled engineering. In our opinion, it was a deliberate measure to avoid analysts from having any insight into the user log management of WeChat.

WeChat User Data and Logs

WeChat logs data with the intent to gain marketing data and improve on their user experience. It is possible to log the user’s interactions with the application and the device characteristics the application is installed upon. This type of data is not chat specific but interaction specific such as when you press on the keypad, to when you click a link or watch a video. The data that is collected includes information concerning the user’s current network, device information, GPS information, cell phone ID (most likely the advertiser ID), and build version (or android API version) as per Figure 2 below. The data is logged into two places within the application while on the mobile, these locations are:

- \$ANDROID_HOME/files/tbslog/tbslog.txt or
- \$ANDROID_HOME
/files/Tencent/tbs_live_log/\$APP_ID/com.tencent.mm_\$STRING_\$STRING.livelog

¹³ <https://www.wsj.com/articles/chinas-state-run-firms-limit-use-of-tencents-messaging-app-11637837474>

```

125 private static String hpe() {
126     NetworkInfo activeNetworkInfo;
127     long currentTimeMillis = System.currentTimeMillis();
128     HashMap hashMap = new HashMap();
129     if (!shouldReportCellInfo()) {
130         return "";
131     }
132     hashMap.put("is_ci_permitted", AppEventsConstants.EVENT_PARAM_VALUE_NO);
133     hashMap.put("net_type", NetStatusUtil.getFormattedNetType(MMAApplicationContext.getContext()));
134     Log.e("MicroMsg.GpsReportHelper", "RecordCostTime: readCellInfo cost 01- %d ms", Long.valueOf(System.currentTimeMillis() - currentTimeMillis));
135     long currentTimeMillis2 = System.currentTimeMillis();
136     if (hpf()) {
137         hashMap.put("is_ci_permitted", "1");
138         hashMap.put("uid", UUID.randomUUID().toString());
139         hashMap.put("sample_time", new StringBuilder().append(System.currentTimeMillis() / 1000).toString());
140         hashMap.put("phone_brand", Build.BRAND);
141         hashMap.put("phone_model", Build.MODEL);
142         try {
143             if (MMAApplicationContext.getContext().checkCallingOrSelfPermission("android.permission.ACCESS_NETWORK_STATE") == 0 && (activeNetworkInfo = ((Connect
144                 hashMap.put("net_subtype", new StringBuilder().append(activeNetworkInfo.getSubtype()).toString());
145         }
146     } catch (Exception unused) {
147     }
148     Log.e("MicroMsg.GpsReportHelper", "RecordCostTime: readCellInfo cost 02- %d ms", Long.valueOf(System.currentTimeMillis() - currentTimeMillis2));
149     long currentTimeMillis3 = System.currentTimeMillis();
150     List<A> h5 = hG(MMAApplicationContext.getContext());
151     Log.e("MicroMsg.GpsReportHelper", "RecordCostTime: readCellInfo cost 03- %d ms", Long.valueOf(System.currentTimeMillis() - currentTimeMillis3));

```

Figure 2: Snapshot of a data logging record.

From here the data logs are sent using a POST request as seen in Figure 3 and located in the following locations in Hong Kong as per Figure 4 below.

- log.tbs.qq.com (129.226.107.80)
- qprostat.imtt.qq.com (101.32.212.183)

```

<time>Mon Jan 31 11:57:05 CST 2022</time>
<url><![CDATA[http://log.tbs.qq.com/ajax?c=pu&v=2&k=a82de80a8229dead90ff1ebc716109aacc86e8738e00e9d3a
<host ip="129.226.107.80">log.tbs.qq.com</host>
<port>80</port>
<protocol>http</protocol>
<method><![CDATA[POST]]></method>
<path><![CDATA[/ajax?c=pu&v=2&k=a82de80a8229dead90ff1ebc716109aacc86e8738e00e9d3a973447476a1df616e4a3
<extension>null</extension>
<request base64="false"><![CDATA[POST /ajax?c=pu&v=2&k=a82de80a8229dead90ff1ebc716109aacc86e8738e00e9d3a
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 536
User-Agent: Dalvik/2.1.0 (Linux; U; Android 10; SM-G960U Build/QP1A.190711.020)
Host: log.tbs.qq.com
Accept-Encoding: gzip, deflate

..:Y
6..F.A\S...x...z...o07 ..&u..mX...yK...%.....j..b.....h...==ja.....h.mvE..c
.g...H.U..a..v.IU9ux...3...p...s.L.z.g...wW..8xe.1... ..y...N-$...%
(...V.v)...*-u\XQ.z.[]).e#.....3.?..WV;V">.....w@T.....2.p.....>2)#.....1.....h\C.t.N56.....[.
+...HY.....f\B.*.../...c..j...
$.8...o~Y.W {....(1....ss.F..3.W.....B9 ..H..{.,L.e].QI.1 .....E5..x8..sz...|...Y.....\
.*4LDM..^..4..(....C.F...S+.f.))]></request>
<status>200</status>
<responselength>273</responselength>
<mimetype></mimetype>
<response base64="false"><![CDATA[HTTP/1.1 200 OK
Date: Mon, 31 Jan 2022 17:57:05 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 32
Connection: close
Set-Cookie: tgw_l7_route=0679de2af948455342ed9a473e4314d; Expires=Mon, 31-Jan-2022 18:27:05 GMT; Path=/

```

Figure 3: WeChat user logs POST command.



Elapsed Time	10 seconds	IP Address	101.32.212.183 Find Sites IP Whois
Blacklist Status	POSSIBLY SAFE 0/115	Reverse DNS	Unknown
IP Address	129.226.107.80 Find Sites IP Whois	ASN	AS132203
Reverse DNS	Unknown	ASN Owner	Tencent Building, Kejizhongyi Avenue
ASN	AS132203	ISP	Tencent cloud computing
ASN Owner	Tencent Building, Kejizhongyi Avenue	Continent	Asia
ISP	Tencent cloud computing	Country Code	 (HK) Hong Kong
Continent	Asia	Latitude / Longitude	22.2908 / 114.1501 Google Map
Country Code	 (HK) Hong Kong	City	Central
Latitude / Longitude	22.2908 / 114.1501 Google Map	Region	Central and Western District
City	Central		
Region	Central and Western District		

Figure 4: Central Logging post locations for WeChat user logs.

The logging data is also encrypted using AES ECB. This encryption method is used for all log files while stored on the device and while the logs are posted back to their central logging server in Hong Kong. This cipher mode allows data to look very similar to one another after being encrypted. One of the characteristics of this encryption method is that it is difficult to use data pattern recognition to be able to spot encrypted data. In our opinion, it is possible to encrypt multiple different types of data without changing the pattern, providing a unique way of preventing detection on what data is being encrypted and uploaded.

It is important to note that while Tencent and WeChat manage multiple servers around the globe, these internationally based servers exist to ensure chats and audio/video calls work with good user experience. During the analysis the user's log data is posted directly to only Hong Kong servers. This is the boundary of the insight we had into WeChat's management of user data. It is reasonable to take Tencent at their word that the data is managed in Hong Kong. It is also reasonable to consider that as the data is posted directly to Hong Kong Tencent would be legally required to provide user data on request of the CCP under the new National Security Legislation.

We must note this record was taken at the time of analysis which was the first week of February 2022. WeChat can easily change their logging processes with a simple update and the IP address and logging records we have outlined are only accurate as at the time of analysis.

WeChat states in its submission to the Select Committee that "servers are all located outside of mainland China".¹⁴ During the analysis we were able to determine there were multiple IP addresses within the application that touched mainland China. We did note that these Aliyun servers were also being used by third parties. In our opinion Weixin functions like a marketplace that are enabled for WeChat users and we consider if data was to be logged as Tencent suggests WeChat would advise

¹⁴ WeChat Submission to Australian Select Committee on Foreign Interference through Social Media on 30 September 2020.

the user they are coming under Weixin terms of service. CHINANET is China's national internet backbone and it was impossible to determine what this function was. IP addresses we found were in mainland China are at Figure 5.

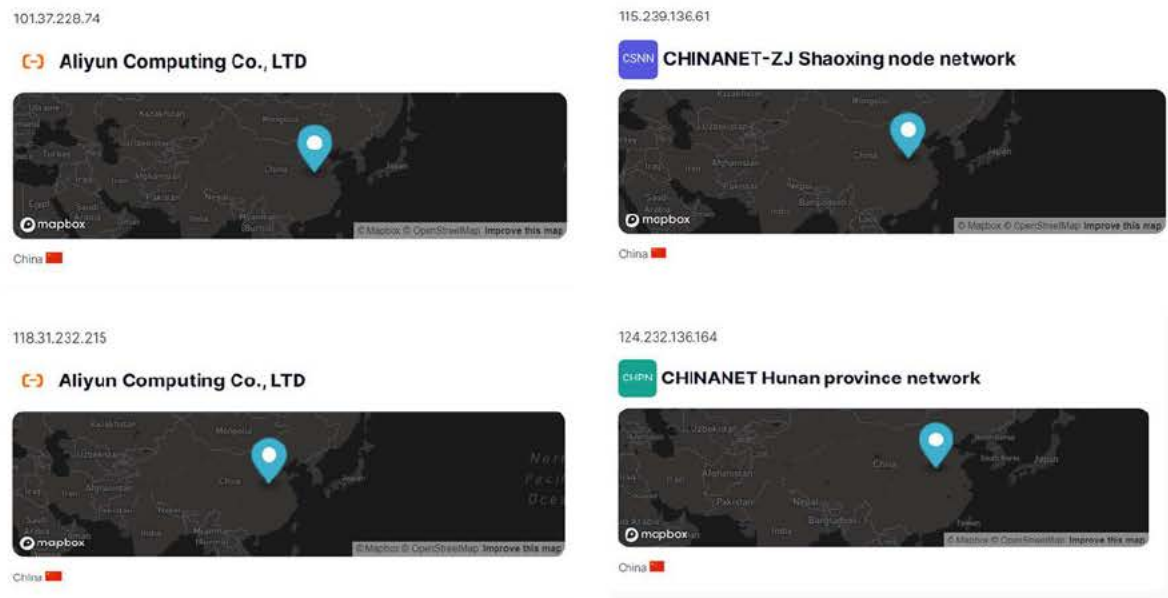


Figure 5: Mainland China IP locations that touch WeChat users.

WeChat states in their submission that:

*"WeChat complies with applicable privacy laws and transparently discloses its collection and processing of users' data in its Privacy Policy (which is GDPR compliant). WeChat also adheres to industry best practices such as data minimisation and the concepts of privacy by default and privacy by design (e.g. chats are not monitored and are stored on a user's device rather than on our servers)."*¹⁵

During our analysis we found no evidence that contradicts the claim that chats are not stored outside of the user's device. There is no logging attempt on chats as we could find. The only security flag we saw was that WeChat has the potential to access all the data in the user's clipboard as per Figure 6 below. This is a risk to flag as users that have a password manager rely on the clipboard to copy and paste their passwords.

¹⁵ WeChat Submission to Australian Select Committee on Foreign Interference through Social Media on 30 September 2020.


```

@Override // android.view.View.OnLongClickListener
public final boolean onLongClick(View view) {
    com.tencent.mm.hellhoundlib.b.b bVar = new com.tencent.mm.hellhoundlib.b.b();
    bVar.br(view);
    com.tencent.mm.hellhoundlib.a.a.c("com.tencent/mm/plugin/finder/profile/uic/FinderProfileHeaderUIC$handleFinderLiveNoticeInfo", this, "Label", this, "Ank.fCb");
    ClipData newPlainText = ClipData.newPlainText("Label", this, "Ank.fCb");
    Object getSystemService = MMApplicationContext.getContext().getSystemService("clipboard");
    Objects.requireNonNull(systemService, "null cannot be cast to non-null type android.content.ClipboardManager");
    ((ClipboardManager) getSystemService).setPrimaryClip(newPlainText);
    com.tencent.mm.ui.base.w.showToast(j, this, getActivity(), "已复制noticeId");
    com.tencent.mm.hellhoundlib.a.a.a(true, this, "com.tencent/mm/plugin/finder/profile/uic/FinderProfileHeaderUIC$handleFinderLiveNoticeInfo");
    return true;
}

```

Figure 6: WeChat's potential access to data in user's clipboard.

WeChat Payment function

WeChat has a payment functionality called WeChat Pay where users can send currency exchange between each other. WeChat maintains a digital wallet record of these payments and manages the currency against their digital account. Payments are available in the following currencies according to the WeChat Pay guide:

*"WeChat Pay support major currencies including but not limited to GBP, HKD, USD, JPY, CAD, AUD, EUR, NZD, KRW settlement. WeChat Pay will have settlement with vendors according to the price in local currency. For unsupported currencies, trade can be made through settlement on US dollar."*¹⁶

After analysing the WeChat Pay function we noted that during payments, the data is logged in Hong Kong at IP address 203.205.239.155 as per Figure 7. Part of this function is seen at Figure 8.


IP Address	203.205.239.155 Find Sites IP Whois
Reverse DNS	Unknown
ASN	AS132203
ASN Owner	Tencent Building, Kejizhongyi Avenue
ISP	Tencent cloud computing
Continent	Asia
Country Code	 (HK) Hong Kong

Figure 7: Location of WeChat Pay logging server in Hong Kong.

¹⁶

https://pay.weixin.qq.com/wechatpay_guide/help_faq.shtml#:~:text=WeChat%20Pay%20support%20major%20currencies,through%20settlement%20on%20US%20dollar.

```
PS C:\Users\rando> ping api.unipay.qq.com

Pinging apiunipay.ms.tencent-cloud.com [203.205.239.155] with 32 bytes of data:
Reply from 203.205.239.155: bytes=32 time=195ms TTL=44
Reply from 203.205.239.155: bytes=32 time=200ms TTL=44

Ping statistics for 203.205.239.155:
    Packets: Sent = 2, Received = 2, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 195ms, Maximum = 200ms, Average = 197ms
Control-C
PS C:\Users\rando>
```

Figure 8: WeChat Pay Logging server.

Part 2 – Chinese Propaganda Department Contracts

WeChat policies towards democratic speech

In its submission to the Select Committee WeChat states the following on combating disinformation and its policies on political communication:

“WeChat’s policy is to ensure that content and behaviour on its application is authentic and to remove false news, disinformation, misinformation, false advertising and security breaches. It does this by setting and enforcing acceptable use standards [...] WeChat prohibits [...] accounts that coordinate, spread, distribute, or participate in inauthentic behaviour. This includes in relation to false news, disinformation, or misinformation in relation to a topic or individual [...] content which breaches any applicable laws or regulations [...] content which may constitute a genuine risk of harm or direct threat to public safety [...]

WeChat prohibits paid promotional content regarding: a candidate for an election; a political party; or any elected or appointed government official appealing for votes for an election; appeals for financial support for political purposes; and a law, regulation, or judicial outcome, including changes to any such matter. WeChat enforces this restriction through its usual advertising content review process before the advertisement is accepted and by the user report function which appears in the application to report undisclosed, miscategorized advertisements and any other inappropriate, offensive or inauthentic material.”¹⁷

¹⁷ WeChat Submission to Australian Select Committee on Foreign Interference through Social Media on 30 September 2020.

In our opinion the interpretation of this statement is crucial. One could argue that any democratically elected politician at all times was either appealing for votes or making content regarding a legislative outcome. This, in essence, is what politicians do. We also found this policy contradictory in its implementation as most Australian politicians have been making this type of content and distributing it on WeChat. Lastly, we struggled to see how WeChat could enforce these policies while also adhering to their other policy of not breaching any applicable laws and regulations. In the United States for example freedom of speech is a constitutional right, WeChat's policies possibly run-in direct contradiction to these constitutional rights. Basically, we struggled to comprehend how WeChat could regulate content and combat misinformation while also regulating content which breached any applicable laws but then bar all political content. It is in this muddled policy we see the structural pressure the CCP has over WeChat.

Propaganda Department Contracts

During our analysis of Chinese government procurement records there were at least 10 contracts between 2016 and 2019 from CCP propaganda departments to conduct influence or propaganda actions over Tencent platforms. The sum of these contracts was 2,327,000 Yuan. These contracts were awarded to subsidiaries in China owned by either Tencent Holdings (owners of WeChat) or companies which Ma Huateng (马化腾), Chairman and CEO of Tencent, has a controlling stake in. These companies are listed in Figures 9 and 10. In our opinion the award of these contracts from CCP propaganda departments is contradictory in nature to the policies it provided in their submission to the Select Committee.

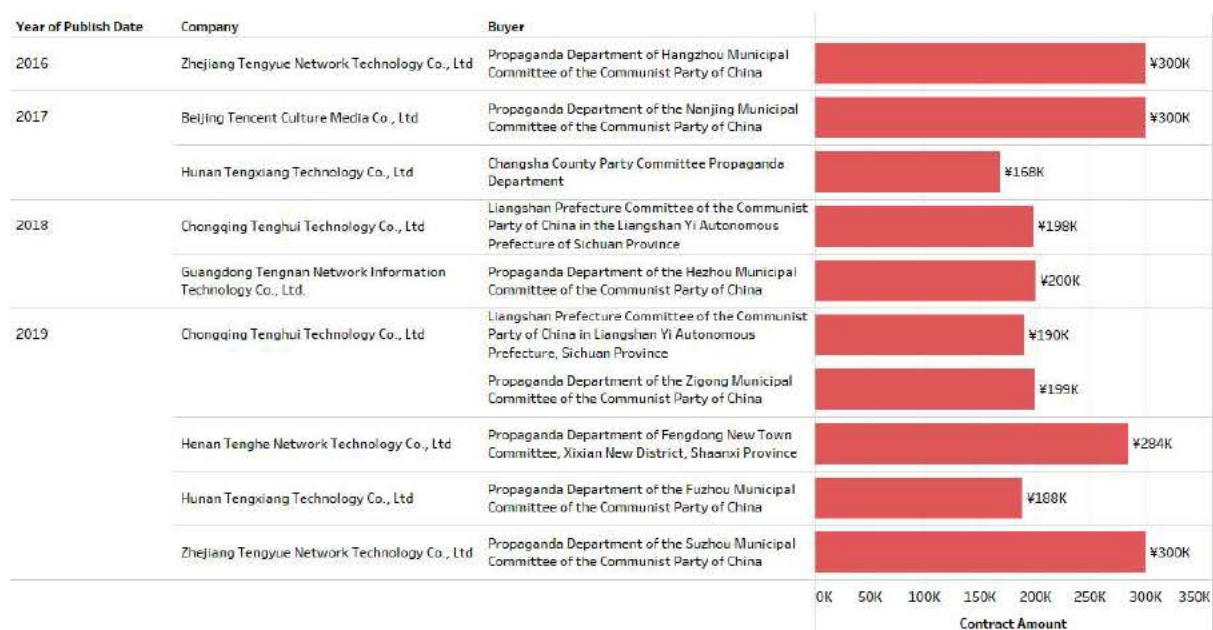


Figure 9: Procurment Contracts between Tencent subsidiaries and CCP propaganda departments.

We have provided below a detailed description of the 10 contracts between 2016 and 2019 from CCP propaganda departments to conduct influence or propaganda actions over Tencent platforms.

Zhejiang Tengyue Network Technology Co., Ltd. (浙江腾越网络科技有限公司), which is 51 per cent owned by Shenzhen Tencent Computer System Co., Ltd. (深圳市腾讯计算机系统有限公司), successfully won two propaganda tenders totalling 600,000RMB (\$94,324 USD) in 2016 and 2019.

1. On the 2 August 2016, Zhejiang Tengyue Network Technology Co., Ltd. (浙江腾越网络科技有限公司) was announced as the successful supplier for a tender titled “Hangzhou released and the city government’s Weibo and WeChat construction funds - Hangzhou released, Tencent Weibo, WeChat publicity and promotion project list.....”. The buyer was the Propaganda Department of Hangzhou Municipal Committee of the Communist Party of China for 300,000RMB (\$47,162 USD).¹⁸
2. On 21 November 2019, Zhejiang Tengyue Network Technology Co., Ltd. (浙江腾越网络科技有限公司) was announced as the successful supplier for a tender titled “Announcement on the transaction of the Propaganda Department of the Suzhou Municipal Committee of the Communist Party of China on the publicity and promotion of the content of the WeChat public account ‘Suzhou Release’”. The buyer was the Propaganda Department of the Suzhou Municipal Committee of the Communist Party of China for 300,000RMB (\$47,162 USD).¹⁹

Hunan Tengxiang Technology Co., Ltd. (湖南腾湘科技有限公司), which is 51 per cent owned by Shenzhen Tencent Computer System Co., Ltd. (深圳市腾讯计算机系统有限公司), successfully won two propaganda tenders totalling 356,000RMB (\$55,966USD) in 2017 and 2019.

1. On 24 March 2017, Hunan Tengxiang Technology Co., Ltd. (湖南腾湘科技有限公司) was announced as the successful supplier for a tender titled “Changsha County released the announcement of the single-source procurement transaction results of the WeChat public account platform construction project”, The buyer was the Changsha County Party Committee Propaganda Department for 168,000RMB (\$26,410 USD).²⁰
2. On 9 October 2019, Hunan Tengxiang Technology Co., Ltd. (湖南腾湘科技有限公司) was announced as the successful supplier for a tender titled “Announcement on the results of the event to praise the new China and talk about the new Fuzhou (Package 1)”. The buyer was the Propaganda Department of the Fuzhou Municipal Committee of the Communist Party of China for 188,000RMB (\$29,555 USD).²¹

Henan Tenghe Network Technology Co., Ltd. (河南腾河网络科技有限公司), which is 51 per cent owned by Shenzhen Tencent Computer System Co., Ltd. (深圳市腾讯计算机系统有限公司), successfully won a propaganda tender on 18 October 2019 for 284,000RMB (\$44,647 USD). The tender was titled “Media Publicity and Promotion Project of the Propaganda Department of

¹⁸ <https://archive.ph/2lqzC>

¹⁹ <https://web.archive.org/web/20220201215855/https://www.bidcenter.com.cn/newscontent-82377800-4.html>

²⁰ <https://archive.ph/itosu>

²¹ https://web.archive.org/web/20220201214839/http://www.ccgp.gov.cn/cggg/dfgg/cjgg/201910/t20191009_13060106.htm

Fengdong New Town Committee, Xixian New District, Shaanxi Province - Tencent (Secondary Purchase) Single Source Transaction Announcement". The buyer was the Propaganda Department of Fengdong New Town Committee, Xixian New District, Shaanxi Province.²²

Guangdong Tengnan Network Information Technology Co., Ltd. (广东腾南网络信息科技有限公司), which is 51 per cent owned by Shenzhen Tencent Computer System Co., Ltd. (深圳市腾讯计算机系统有限公司), successfully won a propaganda tender on 7 November 2018 for 200,000 RMB (\$31,441 USD). The tender was titled "Hezhou Public Resources Trading Centre About the Propaganda Department of the Communist Party of China Hezhou Municipal Committee Tencent Daguang Network 2018 Guangxi (Hezhou) Elite" and the buyer was the Propaganda Department of the Hezhou Municipal Committee of the Communist Party of China.²³

Chongqing Tenghui Technology Co., Ltd. (重庆腾汇科技有限公司) is 51 per cent owned by Shenzhen Century Kaixuan Technology Co., Ltd. (深圳市世纪凯旋科技有限公司), making it the major shareholder. The major shareholder of Shenzhen Century Kaixuan Technology Co., Ltd. (深圳市世纪凯旋科技有限公司) is Pony Ma (马化腾) who owns 54.29 per cent of the company.

Chongqing Tenghui Technology Co., Ltd. (重庆腾汇科技有限公司) successfully won 3 propaganda tenders between 2018 and 2019 totalling 587,000RMB (\$92,281 USD)

1. On the 30 November 2018, **Chongqing Tenghui Technology Co., Ltd. (重庆腾汇科技有限公司)** was announced as the successful supplier for a tender titled "Announcement on single-source procurement of the annual new media publicity cooperation project of the Liangshan Prefecture Committee of the Communist Party of China in the Liangshan Yi Autonomous Prefecture of Sichuan Province". The buyer was the External Propaganda Office of the CPC Liangshan Prefecture Committee for 198,000RMB (\$31,127 USD).²⁴
2. On the 24 October 2019, **Chongqing Tenghui Technology Co., Ltd. (重庆腾汇科技有限公司)** was announced as the successful supplier for a tender titled "Announcement on the single-source transaction of the 2019 Tencent Dacheng.com publicity and promotion service procurement project by the Propaganda Department of the Zigong Municipal Committee of the Communist Party of China in Zigong City, Sichuan Province". The tender involved targeting more than 100 million people in Zigong City to "promote strong external public opinion support". The buyer was the Propaganda Department of Zigong Municipal Committee of the Communist Party of China for 199,000RMB (\$31,274 USD).²⁵

²² <https://archive.ph/8wssk>

²³ <https://archive.ph/qYzc4>

²⁴ <https://archive.ph/YqFKI>

²⁵ https://web.archive.org/web/20220129134434/http://www.ccgp-sichuan.gov.cn/view/staticpags/shiji_cjgg/2c9240ea6dfa5dfe016dfcf7729505e1.html

3. On the 20 December 2019, Chongqing Tenghui Technology Co., Ltd. (重庆腾汇科技有限公司) was announced as the successful supplier for a tender titled “Announcement on the single-source transaction of the annual new media publicity cooperation project of the Liangshan Prefecture Committee of the Communist Party of China in Liangshan Yi Autonomous Prefecture, Sichuan Province”. The buyer was the Propaganda Department of the CPC Liangshan Prefecture Committee for 190,000RMB (\$29,869 USD).²⁶

Beijing Tencent Culture Media Co., Ltd. (北京腾讯文化传媒有限公司) successfully won a propaganda tender on 27 November 2017, titled “[NJZC-2017R064] 2017 World Intelligent Manufacturing Conference Tencent Publicity Service Procurement Announcement”. The buyer was the Propaganda Department of the Nanjing Municipal Committee of the Communist Party of China for 300,000RMB (\$47,162 USD).²⁷

Beijing Tencent Culture Media Co., Ltd. (北京腾讯文化传媒有限公司) is 100 per cent owned by Tencent Technology (Beijing) Co., Ltd. (腾讯科技(北京)有限公司), which in turn is 100 per cent owned by Morespark Limited (添曜有限公司). SEC filings show that Morespark Limited is a British Virgin Islands company and a direct wholly-owned subsidiary of Tencent (“Morespark”). The authorised representative is listed as Ma Huateng, (马化腾), Chairman and CEO of Tencent.²⁸

We note the legal representative of Beijing Tencent Culture Media Co., Ltd. (北京腾讯文化传媒有限公司) is Luan Na (栾娜). An article posted on the Tencent website in 2020 describes Luan Na (栾娜) (aka Helen Luan) as Vice President of Tencent.²⁹ She joined Tencent in 2008 and is currently the head of Tencent's advertising accounts. The legal representative for Tencent Technology (Beijing) Co., Ltd. (腾讯科技(北京)有限公司) is Xi Dan (奚丹) who is the Senior Vice President of Tencent, having joined Tencent in 2002.³⁰

Survey

Internet 2.0 conducted an industry survey in response to our initial reaction to these findings. The question was asked with no context nor mention of WeChat. Respondents could infer any platform or data was behind this question. The question was “Should the data of electoral/political communications be housed within the sovereignty of the country conducting its own electoral process?”. Out of the respondents, an overwhelming 97 per cent agreed with an affirmative yes. All respondents did so under their own name.

²⁶ <https://archive.ph/AkKae>

²⁷ <https://archive.is/N2A2c>

²⁸

https://web.archive.org/web/20220202213624/https://www.sec.gov/Archives/edgar/data/1293451/000095014216003982/eh1600731_13d-bit.htm.

²⁹ <https://web.archive.org/web/20210815184019/https://www.tencent.com/zh-cn/articles/2201039.html>

³⁰ <https://web.archive.org/web/20211117150220/https://www.crunchbase.com/person/xi-dan>

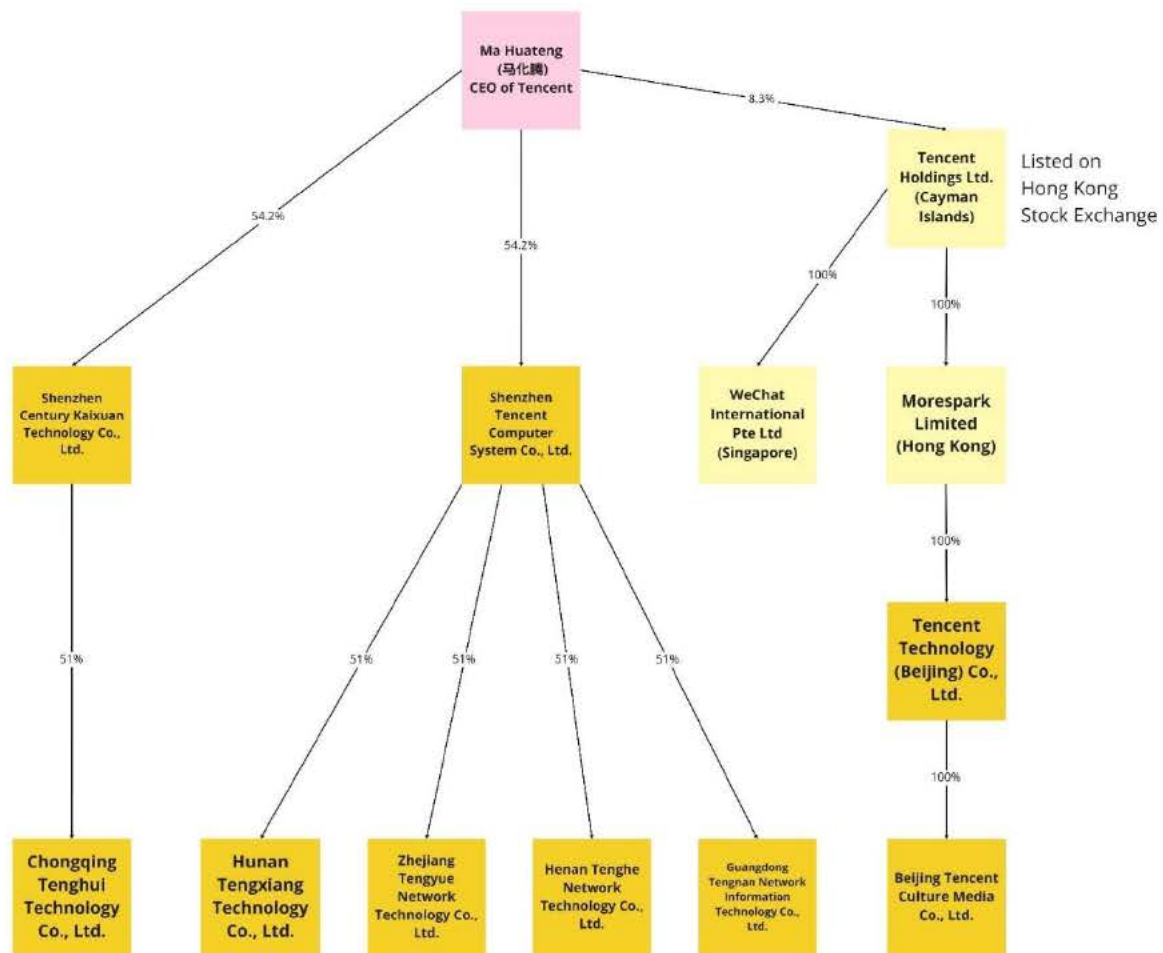


Figure 10: Tencent and Ma Huateng's company structure.

internet2.0

MILITARY-GRADE

CYBER PROTECTION

Australia

Level

18 National

Barton ACT 2600

ABN: 17 632 726

United States

Suite 100

211 N Union

Alexandria

EIN: 86-1567068

contact@internet2-0.com