XLoader Android Spyware and Banking Trojan Distributed via DNS Spoofing

By By: Trend Micro Apr 20, 2018 Read time: 5 min (1359 words)

Published: 2018-04-20 · Archived: 2026-04-05 18:45:21 UTC

We have been detecting a new wave of network attacks since early March, which, for now, are targeting Japan, Korea, China, Taiwan, and Hong Kong. The attacks use Domain Name System (DNS) cache poisoning/DNS spoofing, possibly through infringement techniques such as brute-force or dictionary attacks, to distribute and install malicious Android apps. Trend Micro detects these as ANDROIDOS_XLOADER.HRX.

These malware pose as legitimate Facebook or Chrome applications. They are distributed from polluted DNS domains that send a notification to an unknowing victim's device. The malicious apps can steal personally identifiable and financial data and install additional apps. XLoader can also hijack the infected device (i.e., send SMSs) and sports self-protection/persistence mechanisms through device administrator privileges.

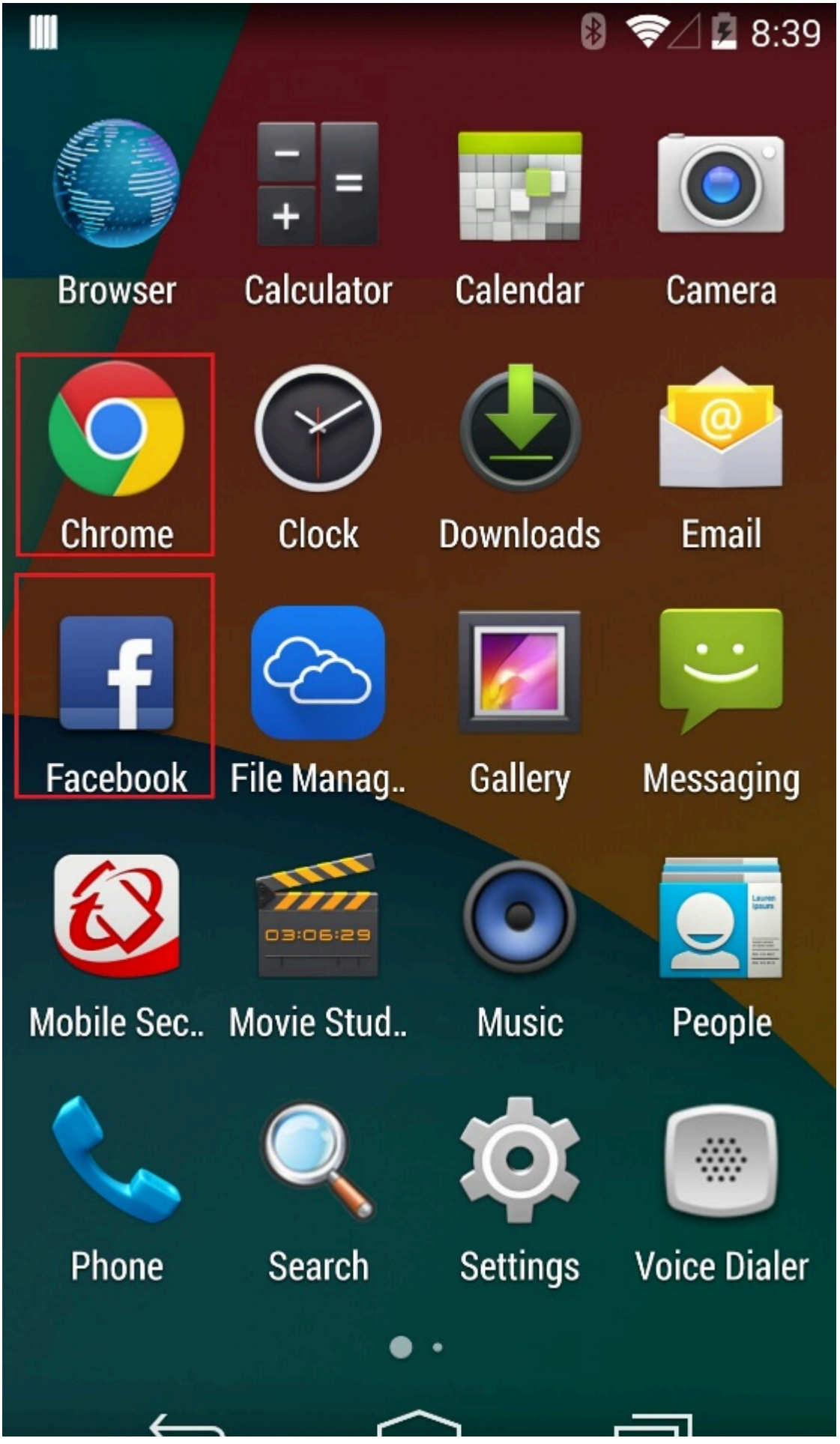
8:39
Browser
Calculator
Calendar
Camera
Chrome
Clock
Downloads
Email
Facebook
File Manag..
Gallery
Messaging
Mobile Sec..
Movie Stud..
Music
People
Phone
Search
Settings
Voice Dialer

Figure 1. Screenshot of the fake Facebook and Chrome apps (highlighted)

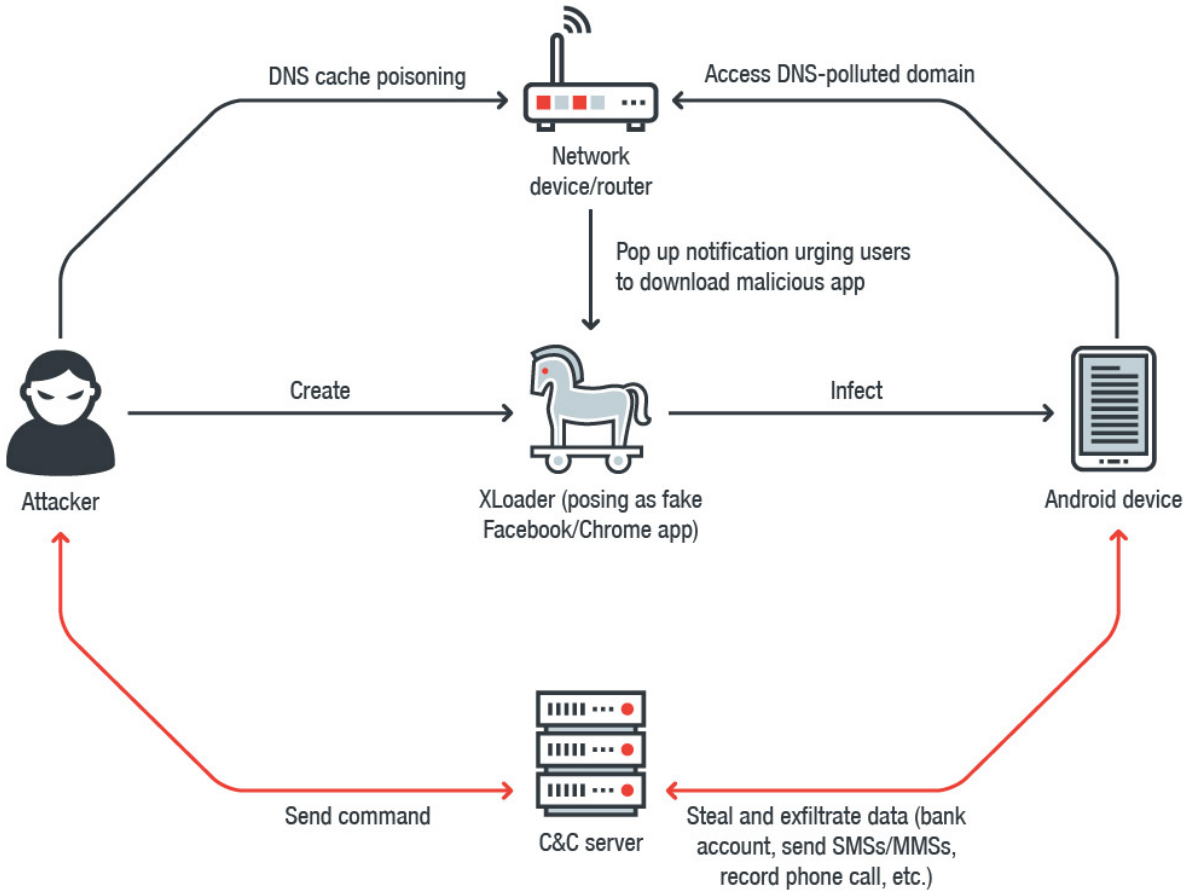

Figure 2. XLoader's infection chain

Infection Chain

As with our earlier reports in late March, the attack chain involves diverting internet traffic to attacker-specified domains by compromising and overwriting the router's DNS settings. A fake alert will notify and urge the user to access the malicious domain and download XLoader.

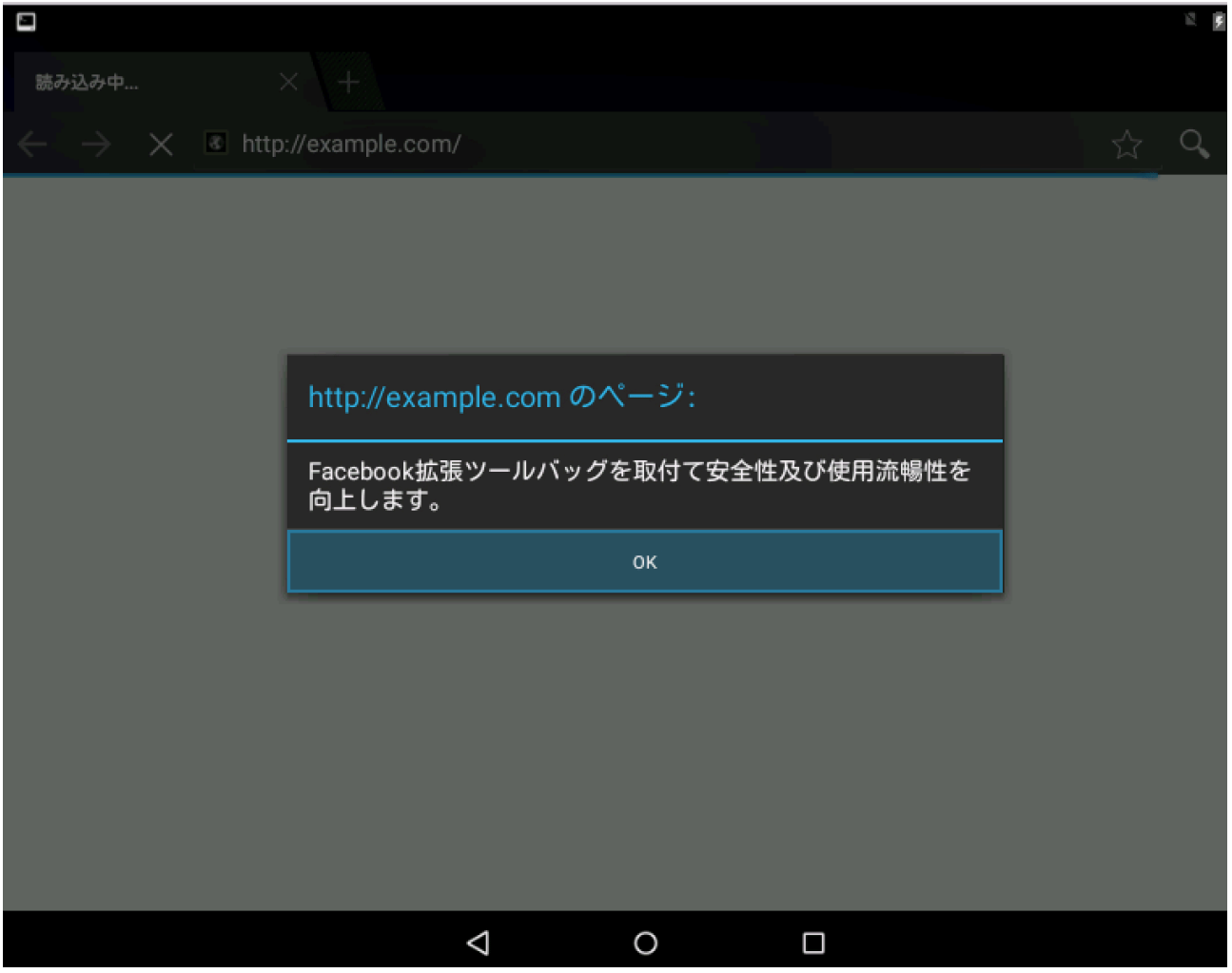

Figure 3. Screenshot of the fake notification on a spoofed/poisoned domain

Technical Analysis

XLoader first loads the encrypted payload from *Assets/db* as *test.dex* to drop the necessary modules then requests for device administrator privileges. Once granted permission, it hides its icon from the launcher application list then starts a service that it keeps running in the background. The background service uses the reflection technique (a feature that allows the inspection and modification of Java-based programs' internal properties) to invoke the method *com.Loader.start* in the payload.

```
public int onStartCommand(Intent arg9, int arg10, int arg11) {
    boolean v0_2;
    Object[] v5;
    Object v4;
    Method v3;
    if(!this.b) {
        this.b = true;
        Application v0 = this.getApplication();
        try {
            v3 = ((esMyApplication)v0).c.getMethod("start", Context.class, Boolean.TYPE);
            v4 = ((esMyApplication)v0).a;
            v5 = new Object[2];
            v5[0] = this;
            if(arg9 != null) {
                v0_2 = true;
            }
            else {
                goto label_29;
            }

            goto label_25;
        }
        catch(Exception v0_1) {
            goto label_32;
        }
```

Figure 4. Code snippet showing the main malicious module invoked via the reflection technique

Monitoring Broadcast Events

XLoader registers many broadcast receivers in the payload dynamically (to monitor broadcast events sent between system and applications). Registering broadcast receivers enable XLoader to trigger its malicious routines. Here is a list of broadcast actions:

- android.provider.Telephony.SMS_RECEIVED
- android.net.conn.CONNECTIVITY_CHANGE
- android.intent.action.BATTERY_CHANGED
- android.intent.action.USER_PRESENT
- android.intent.action.PHONE_STATE
- android.net.wifi.SCAN_RESULTS
- android.intent.action.PACKAGE_ADDED
- android.intent.action.PACKAGE_REMOVED
- android.intent.action.SCREEN_OFF
- android.intent.action.SCREEN_ON
- android.media.RINGER_MODE_CHANGED
- android.sms.msg.action.SMS_SEND
- android.sms.msg.action.SMS_DELIVERED

Creating a Web Server to Phish

XLoader creates a provisional web server to receive the broadcast events. It can also create a simple HTTP server on the infected device to deceive victims. It shows a web phishingnews- cybercrime-and-digital-threats page

whenever the affected device receives a broadcast event (i.e., if a new package is installed or if the device's screen is on) to steal personal data, such as those keyed in for banking apps. The phishing page is translated in Korean, Japanese, Chinese, and English, which are hardcoded in the payload. It will appear differently to users depending on the language set on the device.

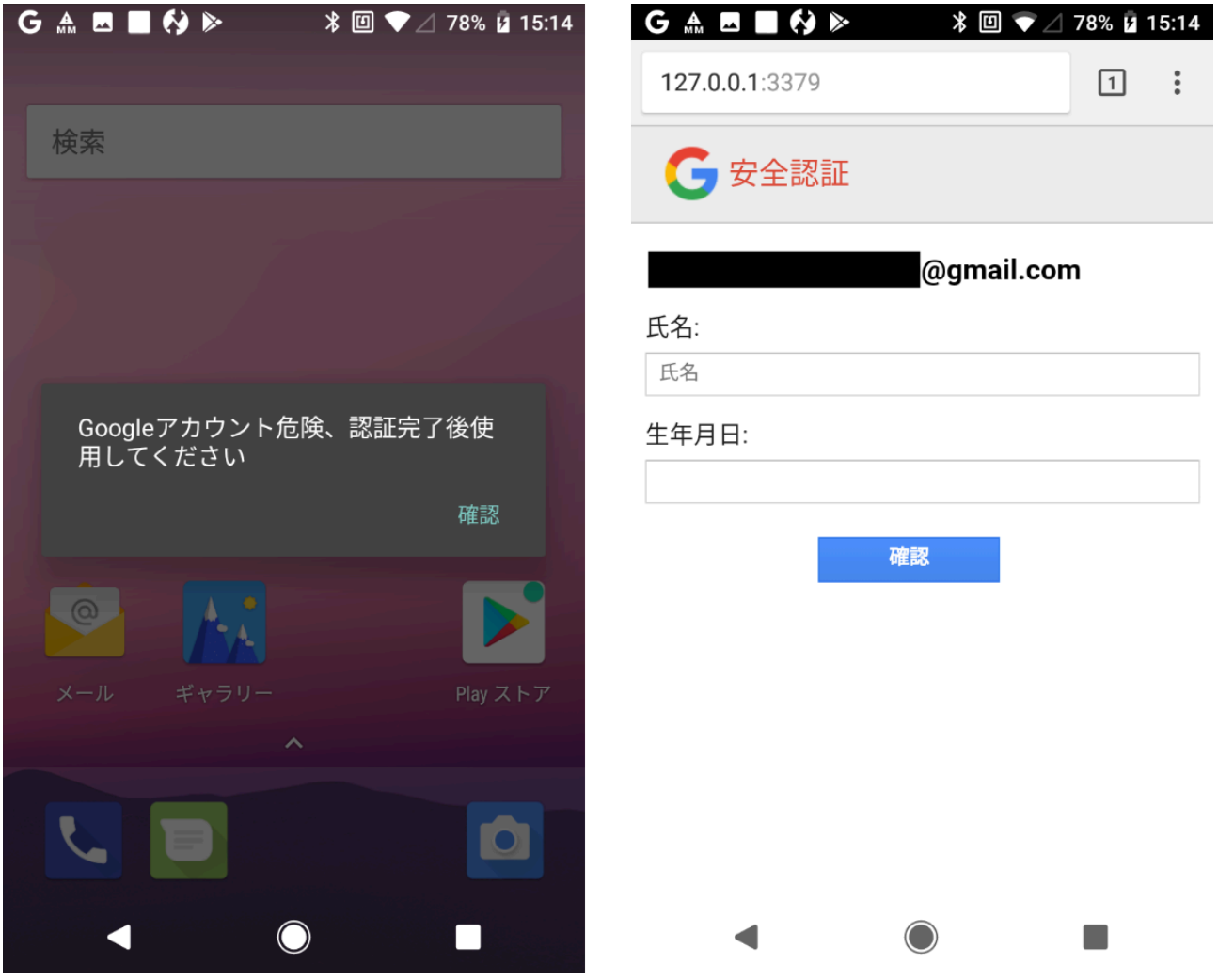

Figure 5. Screenshot of the phishing page (in Japanese)

XLoader as Spyware and Banking Trojan

XLoader can also collect information related to usage of apps installed in the device. Its data-stealing capabilities include collecting SMSs after receiving an SMS-related broadcast event and covertly recording phone calls. XLoader can also hijack accounts linked to financial or game-related apps installed on the affected device.

```
if(arg8 == v6) {
    this.a.setSpeekModle$loader_release(true);
    c.a.a.b.a.a().a(new Loader$b$c(this), 1000, TimeUnit.MILLISECONDS);
    if(!h.a(this.a(), arg9)) {
        return;
    }

    Log.d("WS", "OFFHOOK");
    this.b = new j(arg9);
    v0 = this.b;
    if(v0 == null) {
        h.a();
    }

    v0.a();
    return;
}
```

Figure 6. Code Snippet showing how XLoader records phone calls

XLoader can also start other attacker-specified packages. A possible attack scenario involves replacing legitimate apps with repackaged or malicious versions. By monitoring the package installation broadcast event, XLoader can start their packages. This enables it to launch malicious apps without the user's awareness and explicit consent.

We reverse engineered XLoader and found that it appears to target South Korea-based banks and game development companies. XLoader also prevents victims from accessing the device's settings or using a known antivirus (AV) app in the country.

XLoader can also load multiple malicious modules to receive and execute commands from its remote command-and-control (C&C) server, as shown below:

```
private final void d() {
    this.g.a("sendSms", new Loader$t(this));
    this.g.a("setWifi", new Loader$ad(this));
    this.g.a("gcont", new Loader$ae(this));
    this.g.a("lock", new Loader$af(this));
    this.g.a("bc", new Loader$ag(this));
    this.g.a("setForward", new Loader$ah(this));
    this.g.a("getForward", new Loader$ai(this));
    this.g.a("hasPkg", new Loader$aj(this));
    this.g.a("setRingerMode", new Loader$ak(this));
    this.g.a("setRecEnable", new Loader$u(this));
    this.g.a("reqState", new Loader$v(this));
    this.g.a("showHome", new Loader$w(this));
    this.g.a("getnpki", Loader$x.a);
    this.g.a("http", Loader$y.a);
    this.g.a("onRecordAction", new Loader$z(this));
    this.g.a("call", new Loader$aa(this));
    this.g.a("get_apps", new Loader$ab(this));
    this.g.a("show_fs_float_window", new Loader$ac(this));
}
```

Figure 7. Screenshot showing XLoader's malicious modules

Here's a list of the modules and their functions:

- *sendSms* — send SMS/MMS to a specified address
- *setWifi* — enable or disable Wi-Fi connection
- *gcont* — collect all the device's contacts
- *lock* — currently just an input lock status in the settings (*pref*) file, but may be used as a screenlocking ransomware
- *bc* — collect all contacts from the Android device and SIM card
- *setForward* — currently not implemented, but can be used to hijack the infected device
- *getForward* — currently not implemented, but can be used to hijack the infected device
- *hasPkg* — check the device whether a specified app is installed or not
- *setRingerMode* — set the device's ringer mode
- *setRecEnable* — set the device's ringer mode as silent
- *reqState* — get a detailed phone connection status, which includes activated network and Wi-Fi (with or without password)
- *showHome* — force the device's back to the home screen
- *getnpki*: get files/content from the folder named NPKI (contains certificates related to financial transactions)
- *http* — access a specified network using *HttpURLConnection*
- *onRecordAction* — simulate a number-dialed tone
- *call* — call a specified number
- *get_apps* — get all the apps installed on the device
- *show_fs_float_window* — show a full-screen window for phishing

Of note is XLoader's abuse of the WebSocket protocol (supported in many browsers and web applications) via *ws(WebSockets)* or *wss(WebSockets over SSL/TLS)* to communicate with its C&C servers. The URLs — abused as part of XLoader's C&C — are hidden in three webpages, and the C&C server that XLoader connects to differ per region.

The abuse of the WebSocket protocol provides XLoader with a persistent connection between clients and servers where data can be transported any time. XLoader abuses the MessagePack (a data interchange format) to package the stolen data and exfiltrate it via the WebSocket protocol for faster and more efficient transmission.

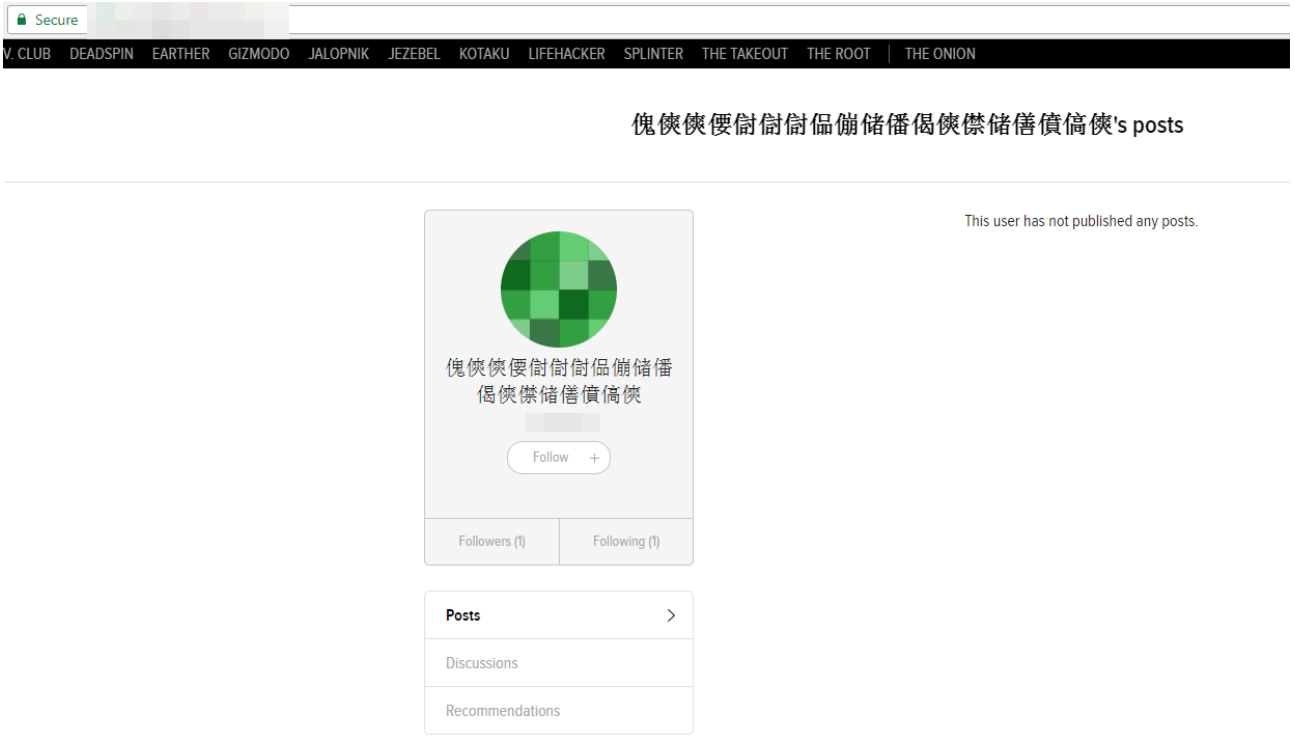

Figure 8. Screenshot showing one of the web pages with hidden C&C-related URL

```
String v2_2 = b.a(v1_1, v2_1.getString("addr_encoding", "utf-8"));
if(v2_2 == null) {
    goto label_45;
}

v1 = this.e;
if(v1 == null) {
    h.b("preferences");
}

String v3 = v1.getString("addr_pattern", "property=\"og:title\" content=\"([\\u4e00-\\u9fa5]+?)\"");
v1_1 = null;
Matcher v2_3 = Pattern.compile(v3).matcher(((CharSequence)v2_2));
if(v2_3.find()) {
    v1_1 = v2_3.group(1);
}
```

Figure 9. Code snippet showing how XLoader parses the C&C URL

Mitigations

XLoader will not download malicious apps if the Android device uses a mobile data connection. Nevertheless, users should practice proper security hygienenews article to mitigate threats that may take advantage of a home or business router's security gaps. Employ stronger credentials, for instance, to make them less susceptible to unauthorized access. Regularly update and patch the router's software and firmware to prevent exploits, and enable its built-in firewall.

For system administrators and information security professionals, configuring the router to be more resistant to attacks like DNS cache poisoningnews- cybercrime-and-digital-threats can help mitigate similar threats. Everyday users can do the same by checking the router's DNS settings if they've been modified. Even threats like DNS cache poisoning employ social engineering, so users should also be more prudentnews article against suspicious or unknown messages that have telltale signs of malware.

We have worked with Google and they ensure that Google Play Protect proactively catches apps of this nature. No instances of these apps were found in Google Play.

Trend Micro Solutions

Trend Micro™ Mobile Securityproducts blocks malicious apps that may exploit this vulnerability. End users and enterprises can also benefit from its multilayered security capabilities that secure the device's data and privacy, and safeguard them from ransomware, fraudulent websites, and identity theft.

For organizations, Trend Micro™ Mobile Security for Enterpriseproducts provides device, compliance and application management, data protection, and configuration provisioning. It also protects devices from attacks that leverage vulnerabilities, prevents unauthorized access to apps, and detects and blocks malware and access to fraudulent websites.

Trend Micro's Mobile App Reputation Service (MARS) covers Android threats using leading sandbox and machine learning technologies. It can protect users against malware, zero-day and known exploits, privacy leaks, and application vulnerability.

Indicators of Compromise

Hashes detected as ANDROIDOS_XLOADER.HRX (SHA-256):

Hash	Package	Label
0F49416B6BCB6E755D 999255FABB4C77C5EA 7DEDEB7E6CDB0925C 4F23C1FB00E	fddf.tre.hjgdsgkh	Chrome
958135E163E0518F24F BD1AF6EF18C30E30C1 A4DFB383FF47D111930 55D4CDCE	fghdf.rtghj.hjkh	Chrome
C65318AA58C9091B938 948B62C4B5D6E472376 97D8D2F96863F99EF17 7B6818D	ghd.et.hds	Chrome
62312475CF0EC1ED66F A29938C30D029BA2F02 BCD6B6ED5AC6C0E5DB E3626BF6	gfhd.rewq.cvxbdf	Chrome
17D1415176121AFF8C0020 C3A094B3D72F9802F5145 C80EBCA47DCFE10CC21F6	gfdg.qwe.gsdg	Facebook
1849E8DFD9D1C03DBE6C 1464F9B05492012A6C14A0 A5B63FEB938F1C8B70309B	jfgh.rtw.ghm	Facebook
AC0C7F59859B5DC3ACBC3 BACA6A6B0FD6ECD05375 D06995D7E28D3F6CB36322A	gwer.dfdf.cxv	Facebook
B623DA28673A1934BD61DE A94A88C37E5FBE9999ED3 D6BA311176D65F64C4A4D	ertt.fgh.nfg	Facebook

4232B36C2B306A47B6C67D 5D949349024F57CDBC4516 3A2CA7B7DEE304229C2B	gfdg.qwe.gsdg	Facebook
B8FB1857881F20E8E3223E 390E13E6DD97D47CABB81 870D51421C04631D63FC1	ertt.fgh.nfg	Facebook
AA183FDA57FDE0137AB931 F3729215956E6F9EE158D90 ED82151948F70DB841B	dfg67.as44f.cvx87df	Facebook
B125EA78FB390950893D14 6A51F513440314BE7648207 B59E5D0A1752740F273	jfgh.rtw.ghm	Facebook
6690FBA689E5AE957E0D01 565BA8F849E0F6AA214F2F 93535D1A7C9C75030BD3	ertt.fgh.nfg	Facebook
E690C05F2AB668A661CD21 9E324291819D5F5646775C2 A17F3B3A03E79332A04	tryrt.sdf.bfd	Facebook
4E32493E6C87B0E2EF3E6A E32F5C32D75AE36C92524A 185EABC88FEA3C7938C8	fddf.tre.hjgdsgkh	Facebook
82D7A496091BD8B0435359 BAFC9E7C923CF09BE58D3 ECC9C477E29E541811582	trghj.asdf.cvxebdf	Facebook
E1FB10B714420F23F1BB09B 6C4C55B674B6EFD93685EE 7D1D4574C7FA8B39A94	trghj.asdf.cvxebdf	Facebook
CC2617D7D904986B83BAF7 843DB6969151363000678E8 DA599EDBF6CF23CB827	jfgh.rtw.ghm	Facebook
3698DF22E8A4656FC53BD2 BDE2DA74DD9DA90083481 29347D5D3E6F976FABA6C	trghj.asdf.cvxebdf	Facebook
065E266016A15BB639C31D 49511DBCD0ADC83261D03 C6652DFBFCAB611B9DB53	trghj.asdf.cvxebdf	Facebook
6F20F227F79DEBFDAE322 33B59F4DC15C7FAF05036B 21E8CD46B24EBC52F0BF8	gfdg.qwe.gsdg	Facebook
A4031768A9F1AEB227389E DD99140303420F3A45F0C1 36D3863C703C685CDEF1	tryrt.sdf.bfd	Facebook
7E49B7C6ED359B4E910E8D 4D2C9436D99CDDEB7F9AF 2E2F1082D0CA45D469566	jfgh.rtw.ghm	Facebook

Source: https://www.trendmicro.com/en_us/research/18/d/xloader-android-spyware-and-banking-trojan-distributed-via-dns-spoofing.html