SUBEX

Pegasus Spyware Threat Report

Date: 05-08-2021

K. Narahari

Overview:

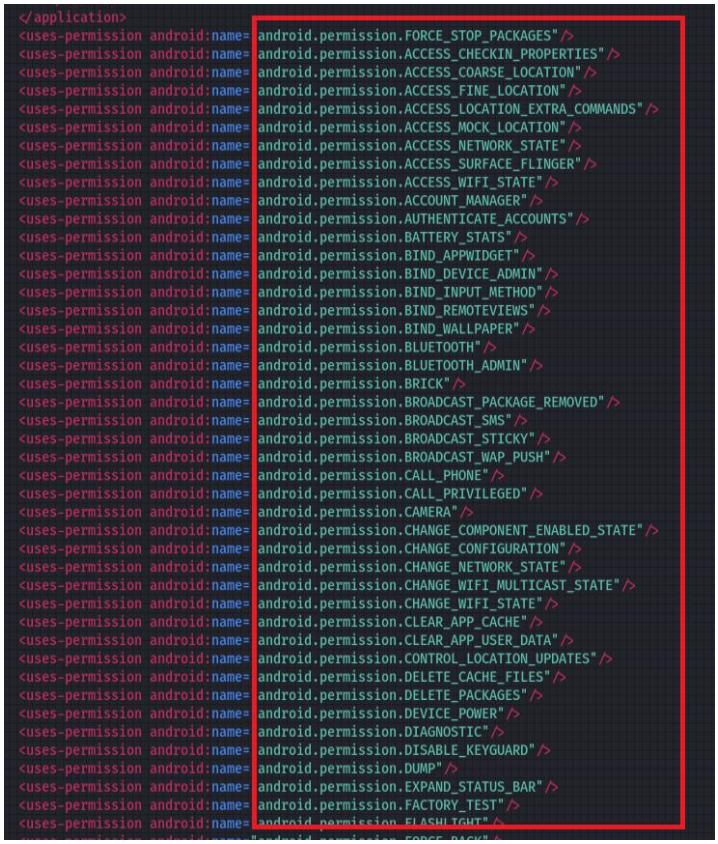
In our Honeypot, we found malicious Pegasus spyware. It can infect the victim's phone through different approaches. It may involve malware through SMS or iMessage that provides a link to a website. If clicked, this link delivers malicious software that compromises the device.

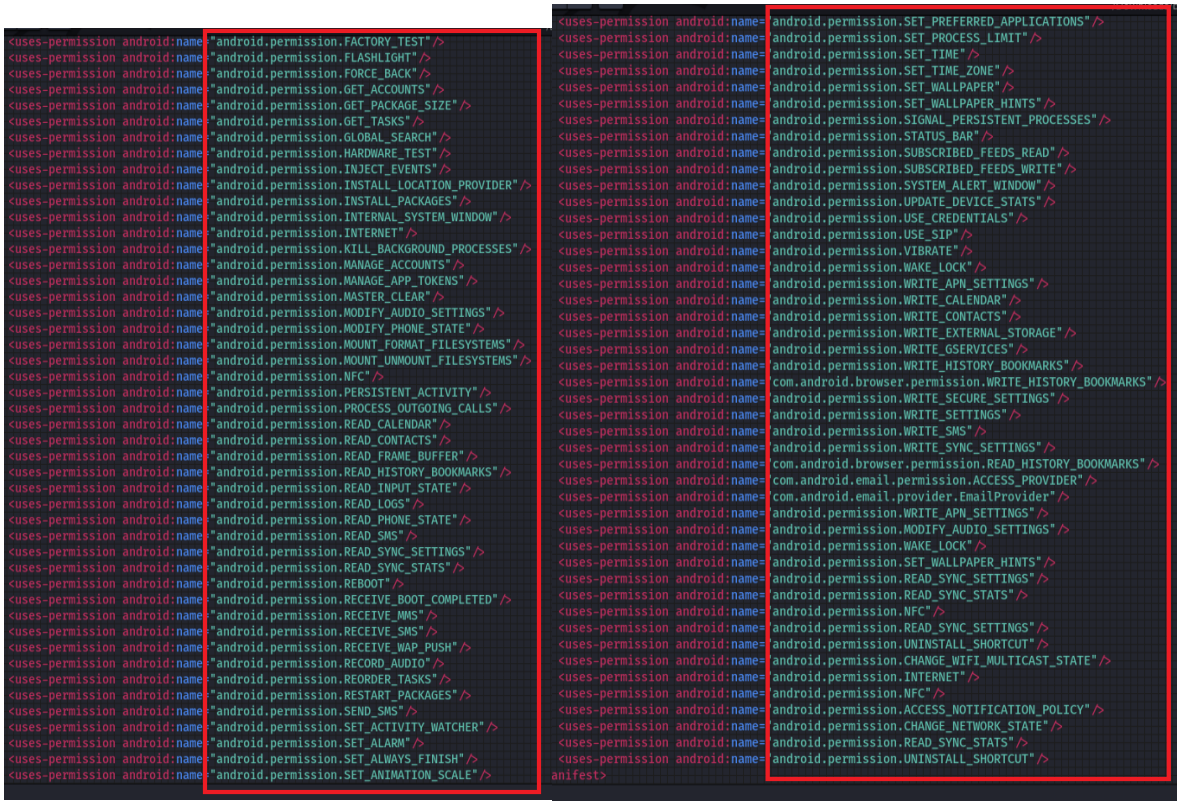
The aim is to take full control of the mobile device's operating system, either by Rooting (on Android devices) or Jailbreaking (on Apple devices). Usually, Rooting on Android device is done by the user to install applications and games from non-supported app stores, or re-enable a functionality that has been disabled by the manufacturer.

File Hash: **8d4b77fa3546149f25bd17357d41fbf0**

Technical Analysis:

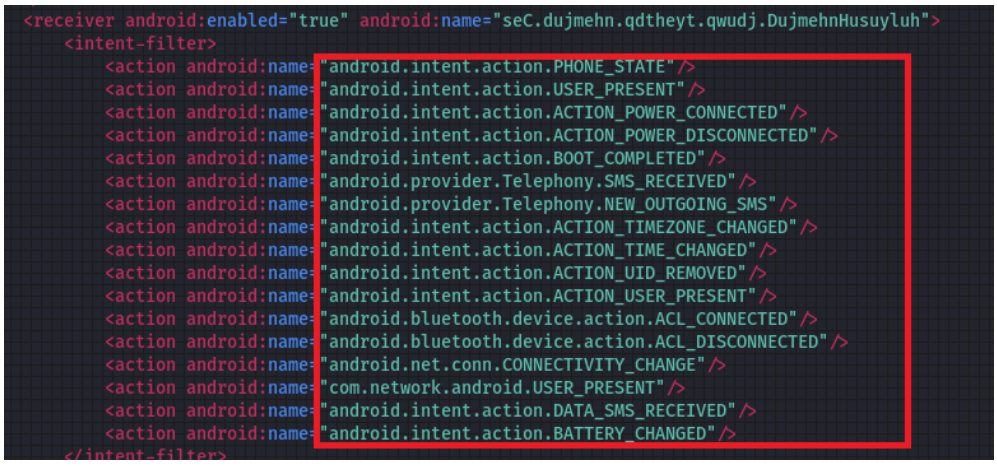
We downloaded the malware sample and reverse-engineered the apk, then performed static analysis and went through the manifest.xml, which is the primary file to begin the analysis.





From the analysis of the manifest file, we can observe that there are plenty of permissions given to the application. It has complete control and access to packages (apps installed on the device), location-related, network-related (Wi-Fi, cellular details), account manager, battery-related, device admin, root control of the device, Bluetooth, broadcast-related, calls, SMS, camera, power, keyguard, flashlight, accounts and passwords, hardware-related, logs, complete settings control, microphone, audio, video, alarm, wallpaper and screen lock, calendar, contacts, and read and write control on memory and storage.

The permissions used by this application seem to be very dangerous and they are used to perform malicious operations. This type of malware is very advanced and can be seen in rare scenarios. Malware has complete root access, control to the device, and commands are executed through remote C2 Server.



From the above screenshot, we can see that this application is using broadcast receivers to pass the data from the application to other malicious application which is installed on the device. In general, broadcast receivers are used to pass the data from one application to another app in the device. For example, OTP is sent from messaging app to other applications like uber, Facebook, etc., this happens through the usage of the broadcast receiver.

Now, in this case, attacker uses receivers to pass phone state, power details, SMS-related, contacts, user details, connectivity details, OTP, login credentials, credit card details, and battery information. This application uses many receivers and the receiver displayed above is one of many receivers used by this malicious application.

```
public static void lPqvQrwhSinkoiCF(Camera camera) {
    Camera.class.getMethod(lPqvQrwhSinkoiCF5538(), new Class[0]).invoke(camera, new Object[0]);
}

Camera camera = null;
for (int i = 0; i < hZXbSmTxBxgXEvMn(); i++) {
    Camera.CameraInfo cameraInfo = new Camera.CameraInfo();
    QmFhcKglgmnECoRM(i, cameraInfo);
    if (cameraInfo.facing == 1) {
        camera = plWybDpztDxxNFNZ(i);
    }
}
return camera;
```

Pegasus Spyware is opening the camera of the victim device. It can get camera access of both the front camera as well as the back camera. It can open and monitor the camera without victim's knowledge that the camera is running. It can take pictures, record videos, and send them to the attacker's server.

```
case 4096:
    return "ACTION_SCROLL_FORWARD";
case 8192:
    return "ACTION_SCROLL_BACKWARD";
case 16384:
    return "ACTION_COPY";
case 32768:
    return "ACTION_PASTE";
case 65536:
    return "ACTION_CUT";
case 131072:
    return "ACTION_SET_SELECTION";
default:
    return "ACTION_UNKNOWN";
```

Using a switch case to transfer the data from the device storage to the attacker's machine. These are some of the actions which can be triggered by the attacker to steal the data from the victim's device. Some of those actions include selecting the file, cut, copy, paste, transmit, and deleting the file, these operations can be performed by the attacker.

```
public void run() {
    int[] grantResults = new int[permissions.length];
    Context context = fragment.getActivity();
    if (context != null) {
        PackageManager packageManager = context.getPackageManager();
        String packageName = context.getPackageName();
        int permissionCount = permissions.length;
        for (int i = 0; i < permissionCount; i++) {
            grantResults[i] = packageManager.checkPermission(permissions[i], packageName);
        }

public String toString() {
    StringBuilder sb = new StringBuilder("NotifyTask[");
    sb.append("packageName:").append(this.packageName);
    sb.append(", id:").append(this.id);
    sb.append(", tag:").append(this.tag);
    sb.append("]");
    return sb.toString();
}
```

Package manager is used to retrieve the complete information of the list of packages, and applications installed on the android device. The attacker is using this to get the information related to the applications which are installed on the victim device. With the declaration of excessive permissions, attacker can manipulate the applications installed on the device i.e... they can delete, add, or modify the existing applications, or even install new malicious applications.

```
public void connect(InetAddress address, int port) throws IOException {
    throw new UnsupportedOperationException();
}

/* access modifiers changed from: protected */
@Override // java.net.SocketImpl
public void create(boolean isStreaming) throws IOException {
    throw new UnsupportedOperationException();
}

/* access modifiers changed from: protected */
@Override // java.net.SocketImpl
public InputStream getInputStream() throws IOException {
    throw new UnsupportedOperationException();
}

/* access modifiers changed from: protected */
@Override // java.net.SocketImpl
public OutputStream getOutputStream() throws IOException {
    throw new UnsupportedOperationException();
}

/* access modifiers changed from: protected */
@Override // java.net.SocketImpl
public void listen(int backlog) throws IOException {
    throw new UnsupportedOperationException();
}
```

The Malicious application uses Sockets to establish communication with attacker C2 Server. The Malware is connected to the attacker server and commands are executed by the attacker in the victim device. Input stream, Output streams are used by the attacker to maintain a flow of commands and control of the communication establishment. All the stolen data is

transmitted after the connection is established and the attacker has total control of creating, maintaining, and closing the connection.

IOCS:

http://qualityfeeling.net
http://bahrainsms.co
http://humandiven.com
http://top10leadsgen.com
http://agilityprocessing.net
http://bytlo.com
http://clubmovistar.com
http://damanhealth.online
http://icrcworld.com
http://smsmensaje.mx
http://redstartnews.net
http://jeeyarworld.com
http://securedlogin.org
http://tahmilmilafate.com
http://jeeyarworld.com
http://shtraf.info
http://accountnotify.com
http://alive2plunge.com
http://bestsalesaroundme.com
http://cdnwa.com

PROCESSES:

bluetoothfs
CommsCenterRootHelper
Dhcp4d
Fdlibframed
MobileSMSd
launchrexd
netservcomf
otpgrefd

MITRE Techniques:

Broadcast receivers (T1402)
Access Contact List (T1432)
Masquerade as Legitimate Application (T1444)
Access Sensitive Data in Device Logs (T1413)

Execution (TA0041)

Subex Secure Protection

Subex Secure detects the android sample as “SS_Gen_PegasusSpyware_A”.

Our Honeypot Network

This report has been prepared from the threat intelligence gathered by our honeypot network. This honeypot network is today operational in 62 cities across the world. These cities have at least one of the following attributes:

- Are landing centers for submarine cables
- Are internet traffic hotspots
- House multiple IoT projects with a high number of connected endpoints
- House multiple connected critical infrastructure projects
- Have academic and research centers focusing on IoT
- Have the potential to host multiple IoT projects across domains in the future.

Over 3.5 million attacks a day is being registered across this network of individual honeypots. These attacks are studied, analysed, categorized, and marked according to a threat rank index, a priority assessment framework that we have developed within Subex. The honeypot network includes over 4000 physical and virtual devices covering over 400 device architectures and varied connectivity mediums globally. These devices are grouped based on the sectors they belong to for purposes of understanding sectoral attacks. Thus, a layered flow of threat intelligence is made possible.