2018

Protecting apps against Jailbreaking and rooting

تأمين التطبيقات ضد الجيبلبريك والروت

Haboob Team

المحتوى

بسم الله الرحمن الرحيم

1 تعريف الجيلبريك والروت:

الجيلبريك والروت هي عملية كسر قيود حماية النظام من امكانية الوصول الى الملفات الداخلية عن طريق المستخدم العادي. او البرامج المثبته الآخرى. الهدف من ذلك هو منع المستخدمين من الوصول الى الجذر الخاص في النظام والتعديل عليه وتثبيت تطبيقات من خارج المتجر ومن مصادر غير معروفة. الطريقة تعتبر غير قانونية في بعض البلدان وبعض البلدان يعتبر نظامي إلا ان الشركات مثل أبل وقوقل يقومون بإلغاء الضمان في حالة تم عمل جيلبريك او روت.

2 تاريخ الجيلبريك وانواعه:

يعود تاريخ اول جيلبريك إلى عام 2007 عند اطلاق اول هاتف ايفون 3 في ذلك الوقت , قام باحثون بعمل كسر لحماية الجهاز وتثبيت برامج من خارج المتجر والتحكم بشكل كامل في النظام. يأتي الجيلبريك بثلاثة انواع: غير مقيد, شبه مقيد, وكسر حماية مقيد.

2.1 كسر الحماية الغير مقيد:

هي عملية كسر حماية للجهاز يسمح باعادة تشغيل الجهاز دون الحاجة إلى إعادة عمل خطوات الجيلبريك مرة أخرى.

2.2 كسر الحماية الشبه مقيد:

هي عملية مشابهه للعملية السابقة إلا ان المستخدم يحتاج إلى إعادة بعض الخطوات من نفس الهاتف دون الحاجة لمساعدة الكمبيوتر.

2.3 كسر الحماية المقيد:

هي عملية تحتاج إلى توصيل الهاتف إلى جهاز الكمبيوتر عند كل مرة يتم إغلاق الهاتف فيها . وهذا النوع من الجيلبريك مزعج وغير مناسب لذا يتجنب غالب محبي الجيلبريك من استعمال هذا النوع.

3 ساندبوكس

هي بيئة معزولة في نظام التشغيل هدفها أن تكون خاصة لكل تطبيق مثبت. بحيث كل تطبيق له بيئة منعزلة ومفصولة عن البيئات الخاصة بالتطبيقات الاخرى , كل بيئة تحتوي على ملفات أو قواعد بيانات خاصة لهذا التطبيق . مثلا التطبيق أ لايستطيع الوصول إلى بيئة التطبيق ب .

يقوم الجيلبريك في هذه الحالة بالغاء خاصية الـــ Sandbox فتكون جميع البيئات الخاصة لجميع التطبيقات مفتوحه ويمكن الوصـــول لها من التطبيقات الاخرى. لذا عند إختراق هاتف الجوال يكون فان البيانات تكون مكشوفه للهاكر ويمكن الوصول لقواعد البيانات تلك بكل سهولة.

في هذه الورقة البحثية ســيتم شـــرح كيفية عمل حماية لتطبيق معين ضـــد الجيلبريك, حيث يســـتحســـن في حال تطوير تطبيق حســـاس (كتطبيقات التعاملات المالية الخاصـــة بالبنوك . أو تطبيقات المراسلات الحساسة, أو تطبيقات حفظ معلومات مهمة مثل الارقام والصور وغيرها) فإن المطور الذكي يقوم بكتابة اكواد برمجية تقوم بعمل إيقاف لعملية التثبيت أو الخروج من البرنامج وحذف قواعد البيانات عند اكتشاف ان الجهاز قد حدث له عملية جيلبريك.

4 حماية تطبيق أي أو إس

الفكرة تكمن في كتابة كود يتم استدعائه عند في كل مرة يتم تشغيل التطبيق فيها. حيث يقوم بفحص مسارات السيديا`Applications/Cydia.app` وبعض المسارات الاخرى المشهوره , وفي حال كانت نتيجة الفحص ايجابية يقوم البرنامج بالخروج ولايستطيع المستخدم إستعماله. هذه الطريقة ليست آمنة 100% ولكنها ناجحه جدا في عملية حماية بيانات المستخدمين ضد الجيلبيريك (على الاقل لايتم حفظ بيانات داخل التطبيق)

ولشرح الطريقة بشكل تقني, يقوم المطور بداخل منصة تطوير التطبيق Xcode وبلغة Swift بكتابة إضافة الى كلاس UIDevice بداخله دالة يتم استدعائها عند كل مره يتم تشغيل التطبيق فيها.

```
1. extension UIDevice {
2.     func isJailBroken() -> Bool {
3.         var jailBroken = false
4.         let cydiaPath = "/Applications/Cydia.app"
5.         let aptPath = "/private/var/lib/apt/"
6.         if FileManager.default.fileExists(atPath: cydiaPath) {
7.             jailBroken = true
8.         }
9.         if FileManager.default.fileExists(atPath: aptPath) {
10.                jailBroken = true
11.        }
12.           return jailBroken
13.       }
14.   }
```

في الكود أعلاه تم كتابة الإضافة وبداخلها دالة للتحقق من النظام وتم تعريف متغيرات بداخلها مسار السيديا ومسار التطبيقات , وفي حالة تم اكتشاف مسار السيديا أو مسار التطبيقات تكون قيمة الدالة true.

```
15.   if UIDevice.current.isJailBroken() {
16.          // show a blank screen or some other view controller
17.          let jailbreakVC = JailBrokenViewController()
18.          self.navigationController?.present(jailbreakVC,
   animated: true, completion:nil)
19.       } else {
20.        // continue executing your next View controller
21.        let nextVC = NextViewController()
22.        self.navigationController?.present(nextVC, animated:
   true, completion:nil)
23.       }
```

في الكود اعلاه تم كتابة شرط أنه في حال كانت نتيجة إستدعاء الدالة السابقة true, فإن الجهاز يحتوي على جيلبريك ويتم اغلاق التطبيق والخروج فورا , يمكن للمطور إضافة أي اوامر اخرى مثل حذف قواعد البيانات, اشعار المستخدم برساله تنبيه, أو رساله صوتيه أو بحسب مايريده المطور.

5 حماية تطبيق اندرويد

نظريا لايوجد هناك أي اختلاف جوهري بين الطريقتين سوى في طريقة الكتابة ,وفحص المسارات. حيث في الطريقة الاولى تمت كتابة الكود بلغة الـ Swift بينما في نظام الاندرويد يتم كتابة الكود بلغة الجافا وبنفس الآلية مع اختلاف المسارات بالاضافة إلى اختبار صلاحية المستخدم ذو الصلاحيات العليا super user اذا كان يمكن الدخول به ام لا.

```
24.   public class RootUtil {
25.       public static boolean isDeviceRooted() {
26.           return checkRootMethod1() || checkRootMethod2() ||
   checkRootMethod3();
27.       }
28.
29.       private static boolean checkRootMethod1() {
30.           String buildTags = android.os.Build.TAGS;
31.           return buildTags != null &&
   buildTags.contains("test-keys");
32.       }
33.
34.       private static boolean checkRootMethod2() {
35.           String[] paths = {"/system/app/Superuser.apk",
   "/sbin/su", "/system/bin/su", "/system/xbin/su",
   "/data/local/xbin/su", "/data/local/bin/su",
   "/system/sd/xbin/su",
36.                   "/system/bin/failsafe/su", "/data/local/su",
   "/su/bin/su"};
37.           for (String path : paths) {
38.               if (new File(path).exists()) return true;
39.           }
40.           return false;
41.       }
42.
43.       private static boolean checkRootMethod3() {
44.           Process process = null;
45.           try {
46.               process = Runtime.getRuntime().exec(new
   String[]{"/system/xbin/which", "su"});
47.               BufferedReader in = new BufferedReader(new
   InputStreamReader(process.getInputStream()));
48.               if (in.readLine() != null) return true;
49.               return false;
50.           } catch (Throwable t) {
51.               return false;
52.           } finally {
53.               if (process != null) process.destroy();
```

```
54.           }
55.       }
56.   }
57.
```


```
58.   public class MainActivity extends AppCompatActivity {
59.
60.       @Override
61.       protected void onCreate(Bundle savedInstanceState) {
62.           super.onCreate(savedInstanceState);
63.           setContentView(R.layout.activity_main);
64.           this.setTitle("تطبيق");
65.           if (RootUtil.isDeviceRooted()) {
66.               AlertDialog.Builder builder = new
   AlertDialog.Builder(this);
67.               builder.setMessage("لا يمكن إستخدام هذا التطبيق في نظام بصلاحية
   الروت!!")
68.                       .setCancelable(false)
69.                       .setPositiveButton("الخروج وإلغاء التثبيت", new
   DialogInterface.OnClickListener() {
70.                           public void onClick(DialogInterface
   dialog, int id) {
71.                               //do things
```

```
72.                                 Uri packageUri =
   Uri.parse("package:com.example.developer.isrooted");
73.                                 Intent uninstallIntent =
74.                                         new
   Intent(Intent.ACTION_UNINSTALL_PACKAGE, packageUri);
75.                                 startActivity(uninstallIntent);
76.                                 System.exit(0);
77.                             }
78.                         });
79.                 AlertDialog alert = builder.create();
80.                 alert.show();
81.
82.             }
83.         }
84.    }
85.
```