آسیب پذیری اندروید
StrandHogg

شناسه سند:	A-Android-StrandHogg
طبقه بندی سند:	**عادی**
شماره نگارش:	R1.0
تاریخ آخرین ویرایش:	۹۸/۱۰/۱۷
تعداد صفحات:	۸

محمدرضا تیموری
Senior Security Specialist

تهران دانشگاه تربیت مدرس دانشکده برق و کامپیوتر آزمایشگاه تست نفوذ شبکه
www.moreti.ir

فهرست مطالب

۱. مقدمه

اطلاع از آ سیب پذیری‌های روز در دنیای فناوری اطلاعات، در دفاع از امنیت سازمان‌ها و اشـخاص بسـیار مفید بوده و باعث می‌گردد که کارشـناسـان امنیت و به طور کلی همه کسانی که در زمینه IT مشغول فعالیت هستند از این آسیب پذیری‌ها آگاه شوند و بتوانند رویکرد مناسبی را در برخورد با این نوع از تهدیدات اتخاذ نمایند.

از این رو moreti.ir به شرح بعضی از آسیب‌پذیری‌های مهم در حوزه شبکه، سیستم‌عامل، وب اپلیکیشن و موبایل اپلیکیشن می پردازد. در این گزارش به بررسی آسیب پذیری جدید StrandHogg که حتی در اندروید ۱۰ (آخرین نسـخه اندروید در حال حاضـر) نیز قابلیت اکسـپلویت کردن را دارا می‌باشـد، پرداخته‌ایم. این گزارش برای مطالعه کارشـناسـان متخصـص در حوزه امنیت اطلاعات تهیه گردیده اسـت و دانشنامه ای در خصوص اصطلاحات به کار برده شده در این سند وجود ندارد و فرض نویسنده بر این بوده که خواننده با ابزارها و اصطحلاحات به کار برده شده آشنایی لازم و کافی را دارد.

۱.۱. معرفی

آسیب پذیری مورد بحث، در هفته اول ماه دسامبر توسط Promon منتشر گردیده است. اما سوالی که در ابتدا باید به آن پاسخ داده شود تا ایده کلی درباره این آسیب پذیری به دست آید این است که، با استفاده از این آسیب پذیری، نفوذگر چه اقداماتی را می تواند انجام دهد ؟ چه ضـربه هایی را می تواند به امنیت دسـتگاه اندرویدی وارد کرد ؟ چه اطلاعاتی را بدون اطلاع کاربر می تواند سرقت نمود ؟ به صورت خلاصه در پایین مهمترین تهدیدات به وجود آمده با استفاده از این آسیب‌پذیری را مشاهده می‌نماییم.

نفوذگر می‌تواند

- با استفاده از میکروفون دستگاه، به کاربر گوش دهد.
- از طریق دوربین، عکس بگیرد.
- پیام‌های متنی (SMS) را بخواند و ارسال نماید.
- تماس تلفنی ایجاد و یا ضبط نماید.

- اطلاعات ورود به حساب را فیشینگ کند یا حساب‌های کاربری شبکه اجتماعی را سرقت کند.
- به تمام عکس‌های شخصی و یا فایل‌های دیگر در دستگاه دسترسی داشته باشد.
- موقعیت مکانی یا GPS را ردیابی کند.
- به لیست دفترچه تلفن دسترسی داشته باشد.
- به لاگ‌های دستگاه دسترسی داشته باشد.

نکته مهم در خصوص این آسیب‌پذیری این است که تمام مواردی که در بالا ذکر شد، در آخرین ورژن اندروید در زمان نوشتن این مقاله (ژانویه ۲۰۲۰) قابل اکسپلویت و بهره‌برداری می‌باشد.

۲. StrandHogg چیست ؟ چطور می‌توان آن را اکسپلویت کرد ؟

برای پاسخ به این سوال نیاز به این است که مقداری در جزییات سیستم اندروید وارد شویم.

هر اپلیکیشن اندرویدی به وسیله چندین activity ساخته شده است. با توجه به layman، در هر زمانی که شما بر روی آیکون مربوط به یک اپلیکیشن خاص کلیک می نمایید، صفحه ای که به شما نشان داده می شود، به صورت معمول یک activity را به شما نشان می‌دهد. براساس کاربری یک اپلیکیشن، امکان این وجود دارد که activity های بیشتری وجود دارد، اما برای سادگی بیشتر اینطور فرض می‌نماییم که هر UI جدید، یک activity جدید می‌باشد. (ذکر این نکته لازم است که سیستم عامل اندروید از مکانیزم‌های دیگر مانند Fragments نیز پشتیبانی می‌نماید). هر Activity شامل مجموعه‌ای از Attributeهای از پیش تعریف شده می‌باشد که در یک فایل XML به نام AndroidManifest تعریف شده است. در این جا نمونه ای از این فایل را مشاهده می‌نمایید:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.android.systemsupport">

    <uses-permission android:name="android.permission.READ_SMS" />
    <uses-permission android:name="android.permission.CAMERA" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="systemsupport"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".AnotherAffinityActivity"
            android:label="AnotherAffinityActivity"
            android:theme="@style/AppTheme.NoActionBar"
            android:taskAffinity="com.abc.xyz"
            ></activity>
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".ui.login.LoginActivity"
            android:allowTaskReparenting="true"
            android:label="Sign in"
            android:taskAffinity="com.abc.xyz" />
    </application>

</manifest>
```

همانطور که مشـــاهده می‌نمایید هر تگ <Activity> چند Attribute یا ویژگی مانند name، label و یا غیره را دارا می‌باشـد. چیزی که در این جا مورد بحث ما اسـت، taskAffinity می‌باشـد. حالا این تگ چطور مورد اسـتفاده قرار می‌گیرد ؟ قبل از دانسـتن این موضـوع، نیاز به این اسـت که یک سـطح دیگر در معماری taskهای اندروید عمیق تر شویم.

یک task مجموعه ای از activityهایی اســت که کاربران برای انجام یک job مشــخص با آن در تعامل هســتند. activityها در یک stack (back stack) سـامان‌دهی می‌گردند، به صـورتی که هر activity باز می‌باشـد. به عنوان مثال یک اپلیکیشـن ایمیل، به صـورت معمول یک activity دارد که در آن لیسـتی از ایمیل‌ها را به نمایش می‌گذارد. زمانی که کاربر یک ایمیل را انتخاب می‌کند، یک activity جدید باز می گردد تا محتوای آن ایمیل را نمایش دهد.

این activity جدید در back stack اضافه می‌گردد. حال، زمانی که کاربر دکمه back را کلیک نماید، activity جدید به پایان می‌رسد و از stack بیرون می‌رود (pop off می‌شود).

پس به صورت معمول، همه activityهای موجود در یک اپلیکیشن با یک task در تعامل هستند. همانطور که آنها باز می‌گردند، آنها در یک back-stack مربوط به یک task اصطلاحا push می‌شوند. هر زمانی که شما یک اپلیکیشن را به وسیله کلیک بر روی آیکون آن و یا نوتیفیکیشن مربوطه و یا دکمه back کلیک نمایید، این stack مربوط به task معرفی می‌شود و بر این اساس، نمایه صحیح در سیستم پیش‌زمینه Android ارائه شده است.

به صورت پیش‌فرض همه activityها در یک اپلیکیشن یک پیوستگی را دارا می‌باشند که این نزدیکی دلالت می‌کند بر این موضوع که همه به یکدیگر تعلق دارند و در یک stack بارگذاری می‌گردند. با این حال، شما می‌توانید ویژگی taskAffinity را تنظیم نمایید تا آن‌ها را به صورت‌های مختلف گروه‌بندی نمایید و حتی activityهای تعریف شده مختلف در برنامه‌های مختلف را در یک stack بارگذاری نمایید. با استفاده از taskAffinity می‌توان پیوستگی activityهای مربوط به یک اپلیکیشن را تغییر داد. در نتیجه برای اپلیکیشن ماشین حساب، برای یکی از activityها می‌توان taskAffinty را به جای این که به ماشین حساب پیوستگی داشته باشد به اپلیکیشن Facebook تعریف نمود. در این صورت این activity می‌تواند در stack مربوط به اپلیکیشن Facebook بارگذاری شود. این مورد دقیقا همان نقطه آسیب‌پذیر است.

نفوذگر یک برنامه را با activityهای مشخص طراحی می‌نماید که یکی از آنها ویژگی taskAffinity با برنامه Facebook را دارا می‌باشد. حال اگر اپلیکیشن مخرب اجرا گردد، activity دارای ویژگی taskAffinity برای اپلیکیشن Facbook، یک stack برای اپلیکیشن Facebook را شروع می‌کند. حال اگر شما بر روی آیکون اپلیکیشن Facebook کلیک نمایید، سیستم عامل اندروید این stack را چک می‌نماید و activityهای مربوط به اپلیکیشن مخرب را در عوض اپلکیشن Facebook نمایش می‌دهد. ترکیب taskAffinity با فلگ های دیگر مانند

FLAG_ACTIVITY_CLEAR_TASK یـــــا FLAG_ACTIVITY_NEW_TASK و FLAG_ACTIVITY_MULTIPLE_TASK می‌تواند به مهاجم دسترسی‌های بیشتری را نیز هدیه بدهد.

اکسپلویتی که یک نفوذگر بر اساس این رفتار می‌تواند طراحی نماید، بر اساس ابتکار و خلاقیت مهاجم می‌تواند محدود گردد.

۳. آیا واقعا این آسیب‌پذیری کار می‌کند ؟

بله ! تست های انجام شده نشان می‌دهد که با ساخت یک برنامه مخرب که با یک برنامه کاربردی دیگر پیوستگی یا Affinity دارد، حتی در دستگاه Pixel امکان اکسپلویت وجود دارد. با این حال برای اپلیکیشن‌های سیستمی و یا برنامه های مربوط به شرکت Google به این نتیجه رسیده ایم که رفتار مشاهده شده قابل اعتماد نیست. اما در مقابل اپلیکیشن‌هایی مانند Telegram و Instagram آسیب پذیری هنوز وجود دارد.

۴. چرا این ویژگی کماکان در اندروید وجود دارد ؟

همانطور که سیستم عامل اندروید برای پشتیبانی از multi-tasking طراحی شده است، این ویژگی به صورت مشخص مورد اســـتفاده قرار می‌گیرد. این ویژگی به کاربران تجربه یکپارچه و interaction با برنامه های دیگر را می‌دهد. این یک حدس است اما برای رهایی از این آسیب‌پذیری ممکن است تغییراتی در سطح سیستم عامل صورت گیرد.

۵. چگونه از ما حمایت کنید ؟

با اشتراک گذاشتن این فایل می‌توانید ما را در امر انتشار گزارشات جدید آسیب پذیری‌ها حمایت نمایید.

باتشکر