Sandbox bypass through Google Admin WebView

13/08/2015

Software:	Google Admin (com.google.android.apps.enterprise.cpanel)
Affected Versions:	2014101605 and lower
CVE Reference:	N/A
Author:	Rob Miller, MWR Labs (http://labs.mwrinfosecurity.com/)
Severity:	Medium
Vendor:	Google
Vendor Response:	Issue to be resolved in updated apk

Description:

An issue was found when the Google Admin application received a URL via an IPC call from any other application on the same device. The Admin application would load this URL in a webview within its own activity. If an attacker used a `file://` URL to a file that they controlled, then it is possible to use symbolic links to bypass Same Origin Policy and retrieve data out of the Google Admin sandbox

Impact:

A malicious application on the same device as the Google Admin application is able to read data out of any file within the Google Admin sandbox, bypassing the Android Sandbox.

Cause:

The Google Admin application (`com.google.android.apps.enterprise.cpanel`), has an exported activity called `com.google.android.apps.enterprise.cpanel.activities.ResetPinActivity` that accepts an extra string called `setup_url`. This can be triggered by any application on the device creating a new intent with the `data-uri` set to `http://localhost/foo` and the `setup_url` string set to a file url that they can write to, such as `file://data/data/com.themalicious.app/worldreadablefile.html`

The ResetPinActivity will then load this in the WebView under the privileges of the Google Admin application.

The attacker adds HTML in to their world readable file, which includes an iframe that will load the world readable file again within the frame after a 1 second delay. The Google Admin application loads this file and renders it into its WebView.

Next the attacker deletes the world readable file and replaces it with a symbolic link of the same name that points to a file in the Google Admin sandbox.

After one second the iframe in the WebView will load the file, which will now point to one of its own files. Because the parent and child frames have the same URL, the Same Origin Policy allows the parent frame to query the contents of the child frame. This means that the HTML that the attacker controls can read from the files loaded into the iframe and extract their data.

Interim Workaround:

Devices with Google Admin installed should not install any untrusted third party applications.

Solution:

No updated version has been released as of the time of publication

Technical details

The Google Admin application has the following entry in its `AndroidManifest.xml`:

```
<activity android:configChanges="keyboard|keyboardHidden|orientation|screenSize"
android:name="com.google.android.apps.enterprise.cpanel.activities.ResetPinActivity">
            <intent-filter>
                <action android:name="android.intent.action.VIEW"/>
                <category android:name="android.intent.category.DEFAULT"/>
                <category android:name="android.intent.category.BROWSABLE"/>
                <data android:host="localhost" android:scheme="http"/>
            </intent-filter>
        </activity>
```

This means that other applications (and any web pages linking to `http://localhost/` URLs) are able to start the activity.

The `ResetPinActivity` starts by retrieving the `setup_url` extra,

```
  public void onCreate(Bundle arg3) {
        this.c = this.getIntent().getExtras().getString("setup_url");
```

This string is used later as a URL to load in a webview which has JavaScript enabled

```
        this.b.loadUrl(this.c);
```

A world readable file was created in `/data/data/com.malware/file.html`. Adb can be used to simulate the activity being queried from another app:

```
Adb shell am start -d http://localhost/foo -e setup_url
file:////data/data/com.malware/file.html
```

The file.html contained an iframe with `src=file:////data/data/com.malware/file.html` with JavaScript that would load this iframe after a one second delay and then copy the contents of the iframe and post back to a server. During this delay the file.html was deleted and replaced with a symbolic link to `/data/data/com.google.android.apps.enterprise.cpanel/shared_prefs/nP.xml`

After a one second delay, the iframe loaded with the nP.xml file and the parent frame read the XML data and posted it back to the attacker's server.

The solution to this issue depends on the intended use of this activity. If it is not required for other applications to start the `ResetPinActivity` then the `AndroidManifest.xml` file should be updated so that the activity has the "exported" flag set to false.

If the `ResetPinActivity` is intended to be started by other applications, then the URL provided should be subjected to filtering using a whitelist of acceptable URLs. Alternatively if only selected applications should start the activity, then filtering could be applied using Android IPC permissions.

Detailed Timeline

Date:	Summary:
17/03/2015	Issue disclosed to Google Security team
18/03/2015	Issue acknowledged by Google Security team
20/05/2015	MWR request update from Google Security team, Google Security team reply asking for 2 weeks to allow for update to be released
02/06/2015	MWR request update
18/06/2015	Google Security acknowledge they have exceeded their own 90 day deadline and request a delay on releasing details until July
05/08/2015	MWR announce to Google intention to disclose issue
13/08/2015	Advisory published