GDG Astrakhan

DevFest

Проблемы безопасности Android-приложений или как обезопасить ваше приложение

Сосновский Юрий

Telegram @JurySosnovsky vk.com/sosnoffsky

Модели угроз

Модели угроз

Локальная атака

1. Измененное приложение
2. Вредоносные приложение

Модели угроз

Удаленные атаки

1. С другого сервера
2. С другого клиента
3. В канале связи

Модели угроз

Доступ к устройству

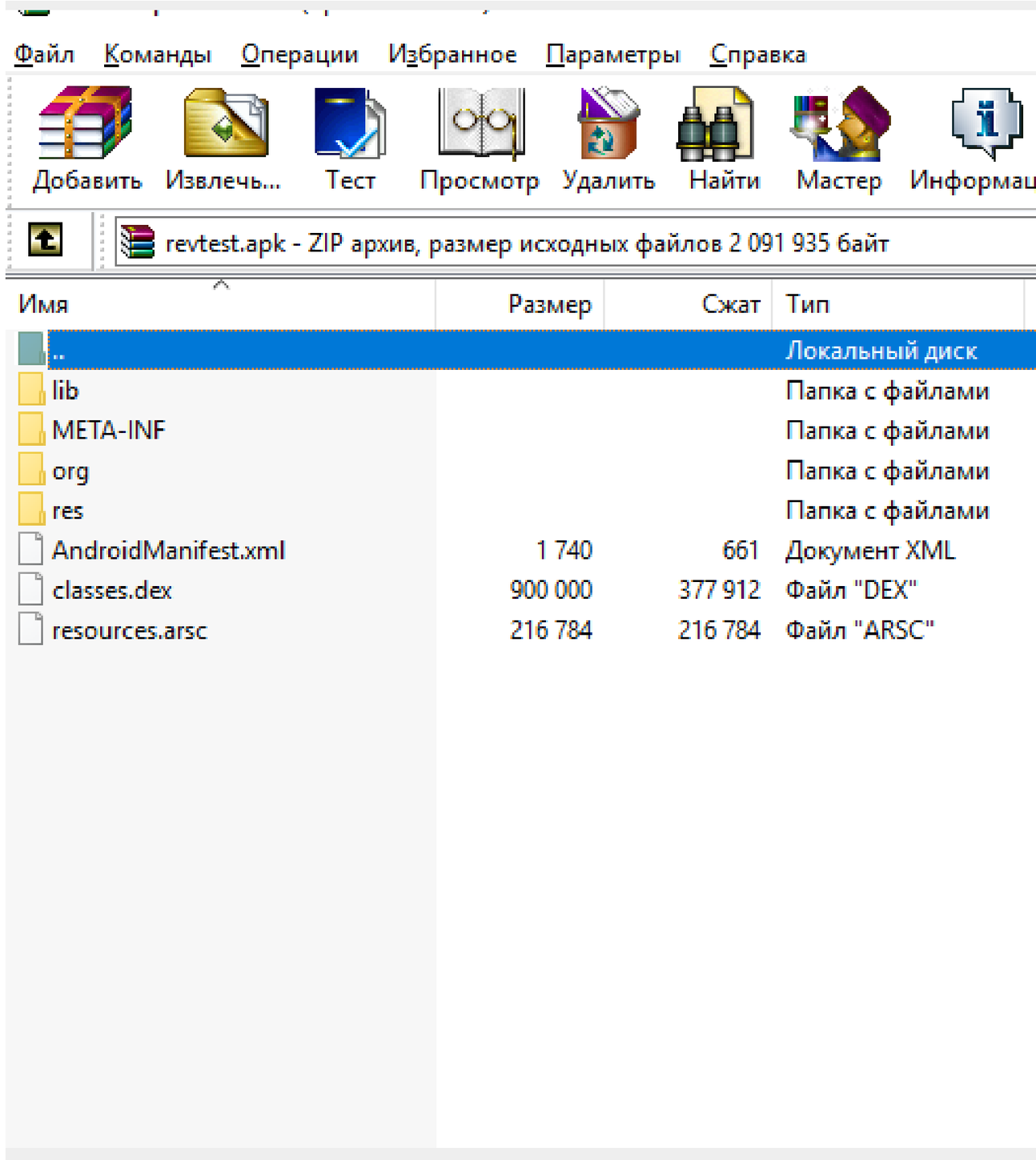
1. Кража или потеря
2. Кратковременный доступ

Локальные атаки

Локальные атаки

Устройство APK файла

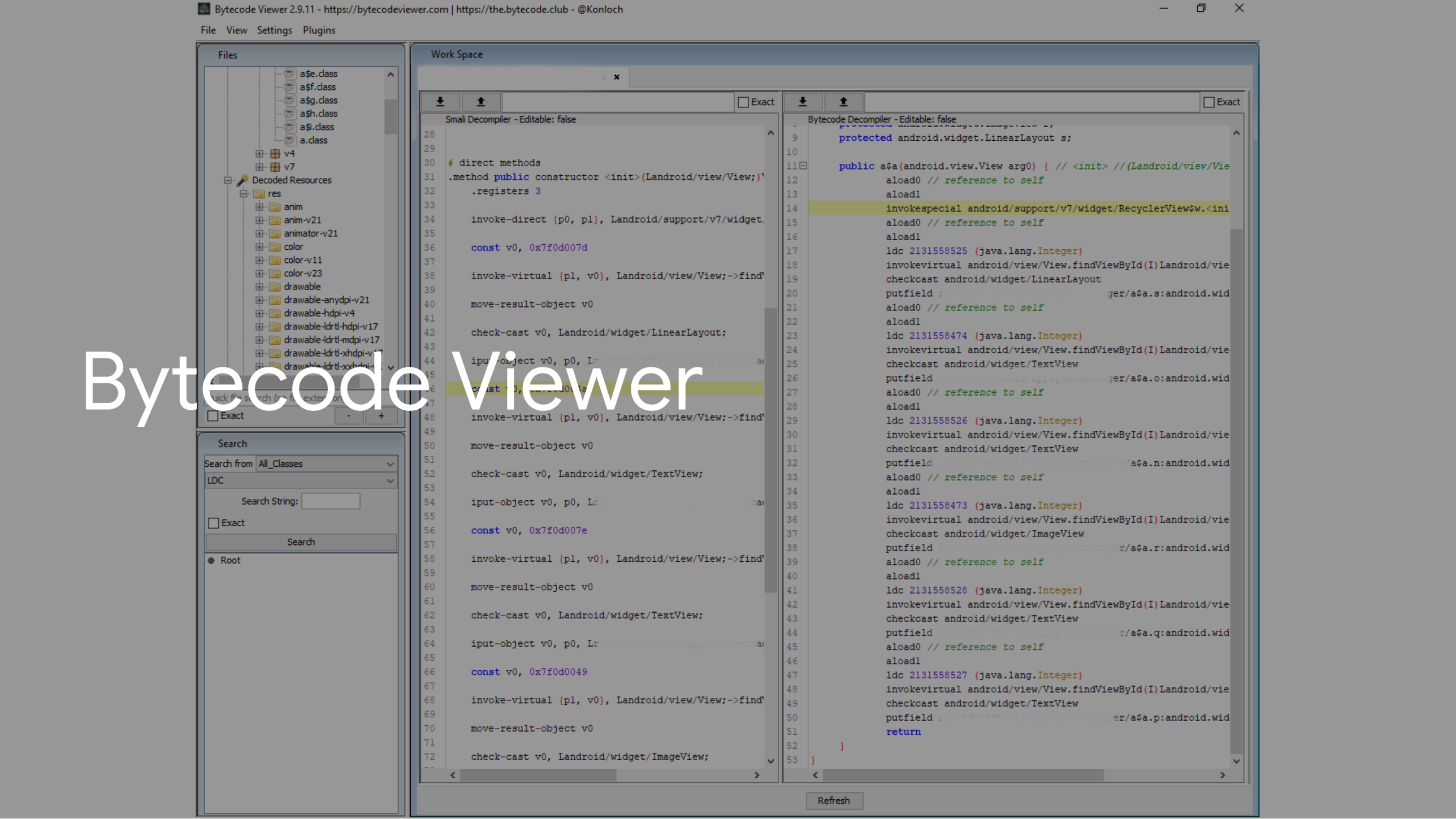
1. Обычный zip-архив
2. DEX – файлы с байткодом
3. ARSC – ресурсы
4. lib – системные библиотеки
5. Прочее (манифесты, конфигурации)



Локальные атаки

Декомпиляция APK

1. Получение APK (Google Play или с устройства)
2. Извлечение ресурсов
3. Извлечение байткода
4. ...
5. PROFIT!



Bytecode Viewer

Локальные атаки

Вектор атаки

1. Внедрение вредоносного кода
2. Фишинг, реклама, рескин

Локальные атаки

Способы защиты

Смириться, нет абсолютной защиты, можно усложнить процесс анализа приложение.

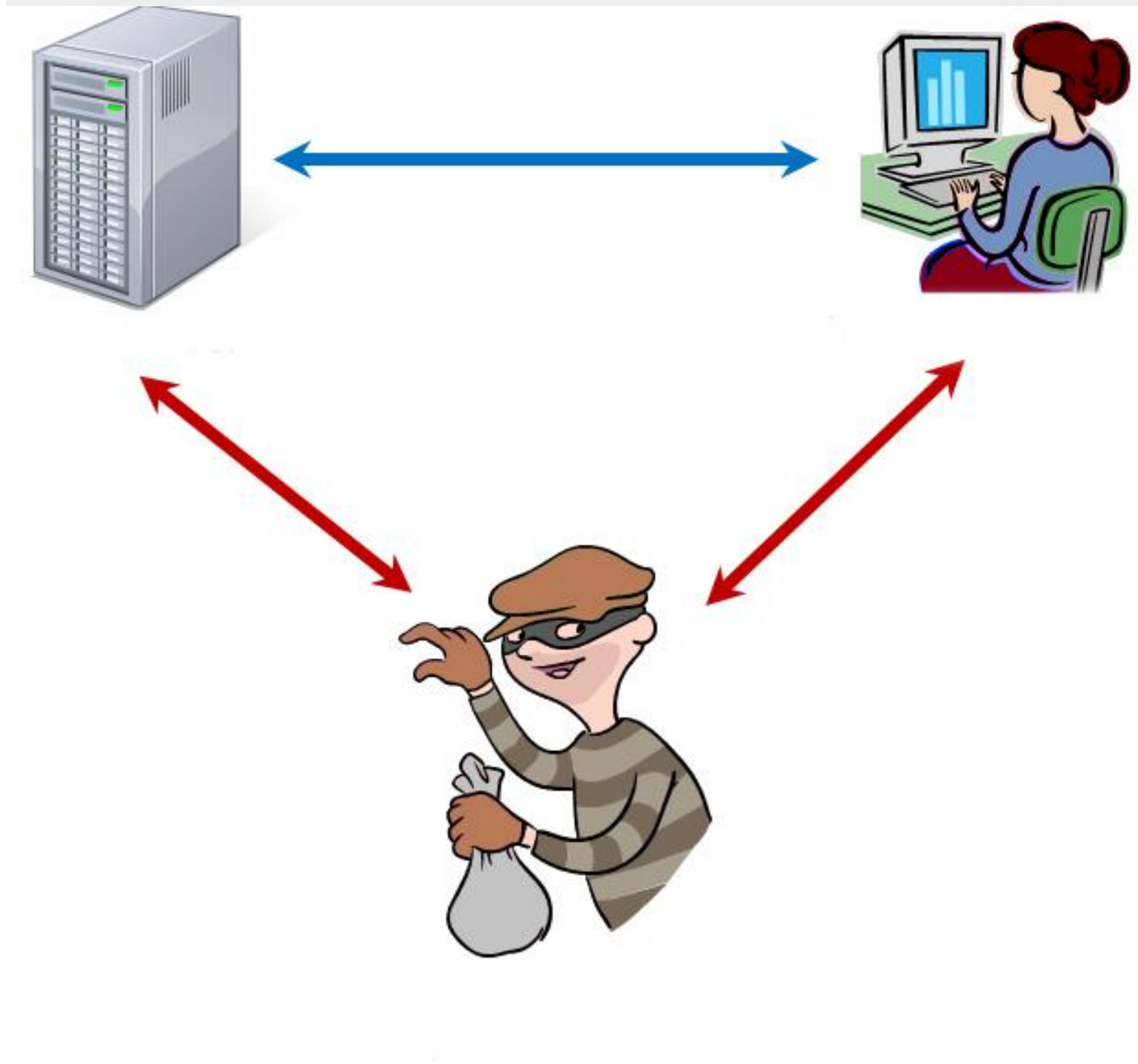
1. Обфусцирование кода (proguard)
2. Шифрование байткода
3. Утончение клиента и вынос важной логики на сервер
4. Реализация логики в нативных библиотеках
5. Контроль и защита «чувствительных» данных

Удаленные атаки

Удаленные атаки

Основные виды атак

1. Man in the middle
2. SSL unpinning



Удаленные атаки

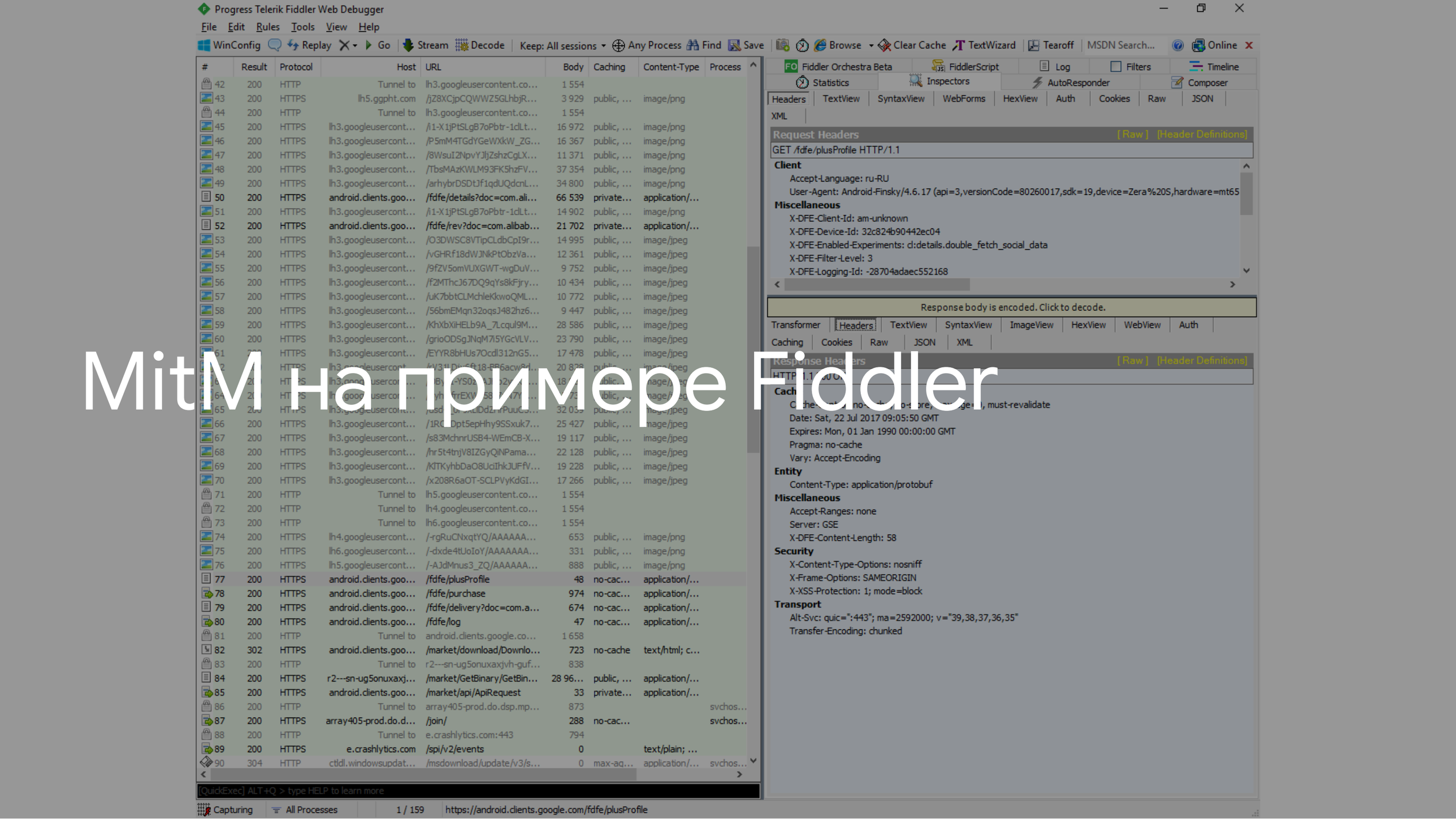
Вектор атак

Атака MitM

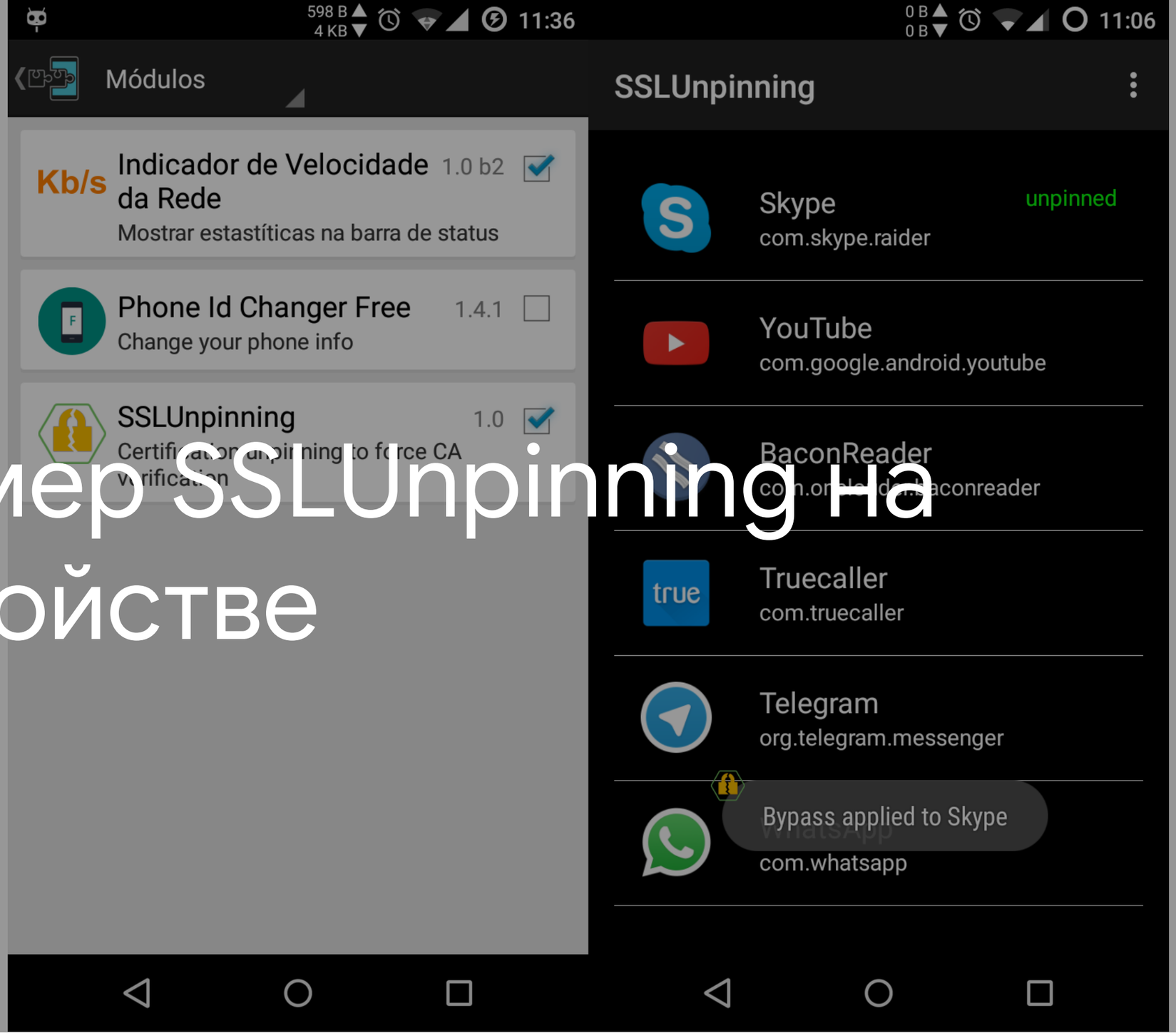
1. Внедрение доверенного сертификата
2. Перехват или изменение трафика

Атака ssl unpinning

1. Переопределение методов java для работы с сетью и сертификатами
2. Перехват трафика



MitM на примере Fiddler



Пример SSLUnpinning на устройстве

Удаленные атаки

Способы защиты

1. SSL pinning
2. Проверка цепочки сертификата
3. SSL unpinning работает на рутованных устройствах – проверка на root
4. Дополнительное шифрование
5. Усложнение протокола обмена

Доступ к устройству

При данных видах атаки, злоумышленник заполучает устройство целиком. Если вышеописанные методы атак безуспешны применяется анализ на наличие уязвимостей в программном коде или операционной системы Android.

Эксплуатация уязвимостей

Ключи авторизации в коде

Примеры уязвимости

1. Логины пароли в коде или манифесте
2. Ключи шифрования в открытом виде
3. Слабые ключи шифрования

```
private void InitializeSQLCipher() {
        SQLiteDatabase.loadLibs(this);
        File databaseFile = getDatabasePath("demo.db");
        databaseFile.mkdirs();
        databaseFile.delete();
        SQLiteDatabase database = SQLiteDatabase.openOrCreateDatabase(databaseFile, "test123", null);
        database.execSQL("create table t1(a, b)");
        database.execSQL("insert into t1(a, b) values(?, ?)", new Object[]{"one for the money",
                                                                            "two for the show"});
    }
```

```
    <!--
        <meta-data>
            <user>admin</user>
            <password>p@$sWOrd</password>
        </meta-data>
    -->

    <application
        android:allowBackup="true"
        android:debuggable="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Security App"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme"
```

Ключи авторизации в коде

Защита

1. Не хранить важные данные в коде или настройках
2. Генерировать ключи шифрования на основе логина/пароля пользователя на лету

Атака на WebView

Вектор атаки

1. Неверный ssl-pinning или его отсутствие
2. Эксплуатация кеша WebView
3. Доступ к файлам через WebView

```
// Защита от кеширования

mWebView.getSettings().setAppCacheEnabled(false);

mWebView.getSettings().setCacheMode(WebSettings.LOAD_NO_CACHE);

// Пример доступа к файлам через WebView

webView.getSettings().setJavaScriptEnabled(true);

webView.getSettings().setAllowFileAccessFromFile(true);

WebView.loadUrl(url);

// url = “file:///data/data/bankclient_android/shared_prefs/prefs.xml”
```

Атака на WebView

Защита

1. Использование ssl-pinnig для webview
2. Запрещать кеширование данных
3. Запрет на загрузку файлов

Атака через allowBackup и debuggable

Вектор атаки

1. Позволяет сделать дамп всей информации приложении
2. Позволяет выполнить отладку приложения

```
// Пример запуска полного дампа устройства
$ adb backup –all
// Отладка приложения
$ adb forward tcp:54321 jdwp:543
$ jdb –attach localhost:54321
```

Атака через отправку сообщений

Вектор атаки

1. Экспорт компонентов
2. Отправка широковещательных сообщений
3. Отсутствие проверки источника сообщения
4. Отсутствие фильтрации во входящем intent
5. Вложенные intent

```
<receiver android:name=".SecurityReceiver"
            android:enabled="true"
            android:exported="true">
            <intent-filter>
                <data
                    android:scheme="https"
                    android:host="securehost.com"
                    android:pathPattern="/auth/.*/"
                    />
                <category android:name="android.intent.category.DEFAULT"/>
            </intent-filter>
        </receiver>
```

Пример перенаправления Intent

Атака через отправку сообщений

Защита

1. Правильно экспортировать компоненты
2. Проверка источника сообщения
3. Фильтрация входных сообщений

Статистика

Соотношение кода ~82% - библиотеки ~18% собственный код
Вторая по популярности библиотека – facebook
Класс «a» - 257 место по использованию.
Класс «o» - 81 по использованию
99% приложений используют доступ в сети Интернет

GDG Astrakhan

DevFest

Спасибо!

Сосновский Юрий

Telegram @JurySosnovsky, vk.com/sosnoffsky

GDG Astrakhan

DevFest

Вопросы

Вопросы можно направить на qa@sosnoffsky.com