XCTF_MOBILE12_你是谁

原创

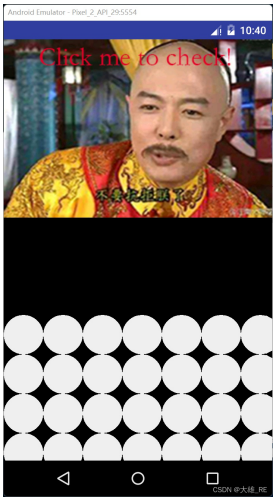
大雄_RE 于 2022-03-01 16:38:48 发布 3631 收藏
分类专栏： CTF 文章标签： android 逆向 ctf
版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。
本文链接：https://blog.csdn.net/shadow20080578/article/details/123201805
版权

CTF 专栏收录该内容
17 篇文章 1 订阅
订阅专栏

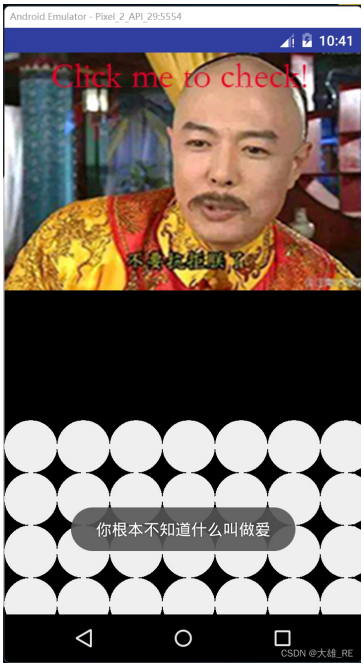
初见

附件为一个apk。在模拟器中安装打开先看到一个骚年：

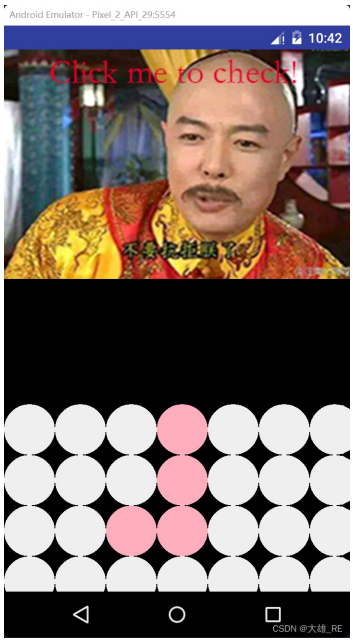
等一会这个画面自动跳过，然后看到一个熟悉的面孔：



随便点击一下上面的图片，提示“你根本不懂什么叫做爱”，并且能听到语音“你是个好人，但是我们不适合。”：



发现下面的白色圆圈可以点击：



暂时没有其它信息，下面先静态分析。

静态分析

使用jadx打开apk，先看MainActivity类。

通过jadx的注释信息，可以看到这里面重载了com.iflytek.cloud.SynthesizerListener类的许多函数：

```
@Override // com.iflytek.cloud.SynthesizerListener
public void onCompleted(SpeechError error) {
}

@Override // com.iflytek.cloud.SynthesizerListener
public void onBufferProgress(int percent, int beginPos, int endPos, String info) {
}
```

百度一下com.iflytek，得知这是科大讯飞的语音库，可以识别语音为文字，也可以将文字转换为语音，这么说之前听到的语音“你是个好人，但是我们不适合。”应该就是这个库生成的。

在一系列对科大讯飞语音库中函数的重载后，是onCreate，onDestory，setParam，getsna四个函数。

在onCreate里对科大讯飞语音库进行了初始化，并创建了语音识别器：

```
this.mIat = SpeechRecognizer.createRecognizer(this, this.mInitListener);
```

语音识别器是用来将语音转化为文字的。

又创建了一个语音合成器：

```
this.mTts = SpeechSynthesizer.createSynthesizer(this, null);
```

语音合成器是用来将文字转换为语音的。

关于科大讯飞的语音库，大家可以查阅这篇文章。

如此看来，这道题可能要靠说话来做。

上面重载的语音库的诸多函数大多函数体都为空，有一个除外：

```
@Override // com.iflytek.cloud.RecognizerListener
        public void onResult(RecognizerResult results, boolean isLast) {
            Log.d(MainActivity.this.TAG, results.getResultString());
            try {
                JSONObject res = new JSONObject(results.getResultString()).getJSONArray("ws").getJSONObject
                MainActivity.this.ss = res.getString("w");
            } catch (Exception e) {
                Log.d(MainActivity.this.TAG, "catch Excepetion");
            }
            if (MainActivity.this.ss.equals("你好")) {
                MainActivity.this.getsna();
            }
            Log.d(MainActivity.this.TAG, MainActivity.this.ss);
}
```

这个onResult就是语音识别器将语音识别为文字后的响应函数，识别的文字结果在参数result中。

识别结果是JSON格式的，这里用JSON类进行解析。

这个函数核心就是判断，是否说了“你好”，如果说了，就调用getsna打印“haha”：

```
    public void getsna() {
        Toast.makeText(this, "haha", 0).show();
    }
```

但这是app的原layout，也就是第一个骚年画面的行为，我们知道，在app运行一段时间后，界面会从骚年图像变成“皇上”图像，这对应了MainActivity.onCreate里的这条语句：

```
setContentView(new background(this));
```

接下来我们就看看background这个类的代码。

background类

background类的构造函数中也创建了语音识别器和语音合成器：

```
this.mIat = SpeechRecognizer.createRecognizer(getContext(), this.mInitListener);
this.mTts = SpeechSynthesizer.createSynthesizer(getContext(), null);
```

并且在构造函数中将自身设置为屏幕触摸的响应类：

```
setOnTouchListener(this);
```

从构造函数中可以猜测，语音识别和屏幕触摸为两个重点。

通过上面对科大讯飞语音库的简单了解我们知道，将我们通过麦克风说的语音转化为文字后，调用的回调函数为onResult。另外屏幕触摸的响应函数为onTouch。

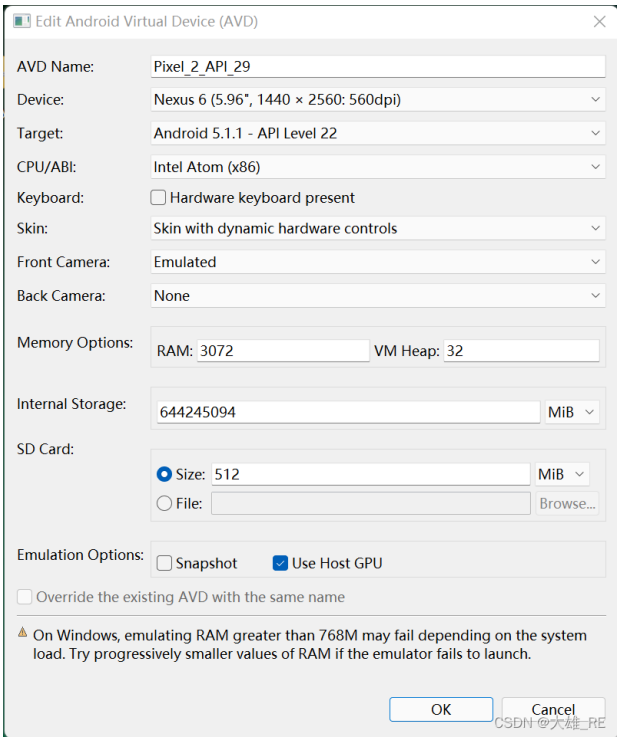
下面我们重点分析这两个函数。

onTouch

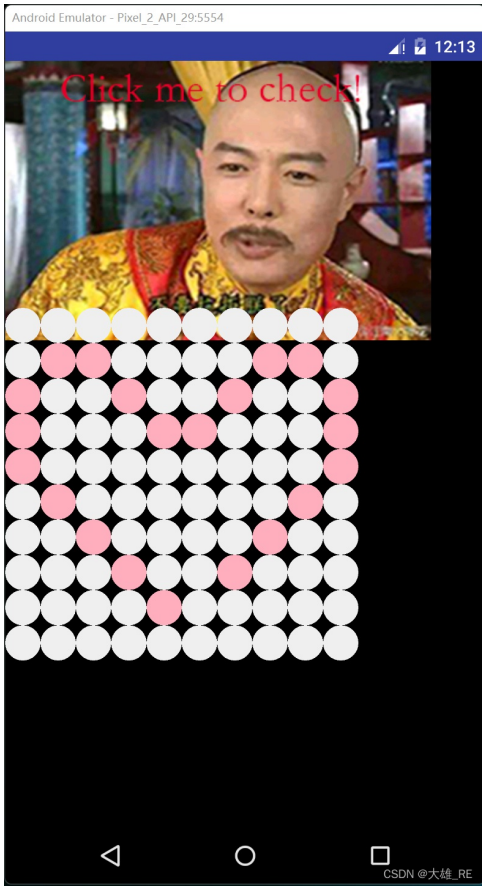
经过一些精简，onTouch函数代码为：

```
    public boolean onTouch(View arg0, MotionEvent e) {
        if  (e.getY() < 815.0f) //触摸位置在上面照片部分
        {
                if (!check()) {
                    this.mTts.startSpeaking("你是个好人，但是我们不适合。 ", this.mSynListener);
                    Toast.makeText(getContext(), "你根本不知道什么叫做爱", 0).show();
                } else {
                    setParam();
                    Log.d(this.TAG, "startListening ret:" + this.mIat.startListening(this.recognizerListene
                    Toast.makeText(getContext(), "通过爱的验证", 0).show();
                }
        } else {
                int y = (int) ((e.getY() / 106.0f) - 7.0f);
                int x = (int) (e.getX() / 106.0f);
                getcircle(x, y).setStatus(getcircle(x, y).getStatus() ^ 1);
                redraw();  //修改按的圆形的颜色
                if (check()) {
                    Toast.makeText(getContext(), "Right design", 0).show();
                }
        }
        return true;
    }
```

到这里我发现从代码看，app界面上的圆圈矩阵应该是10*10的，我的模拟器显示不全，怀疑是因为分辨率问题，故调大模拟器的分辨率：



重新打开app，就能看到完整的圆圈矩阵了：



onTouch函数的逻辑是：

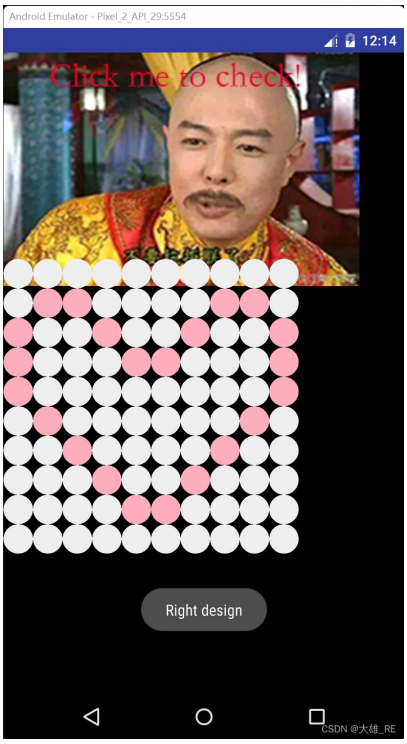
如果点击位置在上面的图像，调用check函数检查是否通过。

如果点击下面的矩阵，则修改对应圆形的颜色，并修改矩阵对应的数组的值（0对应白色，1对应红色）。

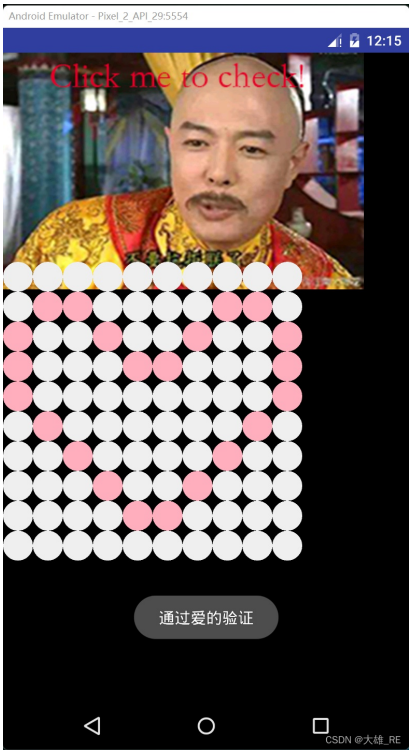
而check函数逻辑也很简单，就是检查这个矩阵数组中部分成员是否为1：

```
public boolean check() {
    return this.matrix[1][1].getStatus() == 1 && this.matrix[1][2].getStatus() == 1 && this.matrix[1][7
}
```

按照check函数，将部分圆点为红色，得到一个爱心（好骚的题）：



拼出爱心后，再点击上面的图片，提示“通过爱的验证”：



onTouch这个验证就通过了，没发现flag。

再看看onResult函数。

onResult

onResult将语音转化为文字后，交给setsna进行处理：

```
    public void onResult(RecognizerResult results, boolean isLast) {
        Log.d(background.this.TAG, results.getResultString());
        try {
            JSONObject res = new JSONObject(results.getResultString()).getJSONArray("ws").getJSONObject
            background.this.ss = res.getString("w");
        } catch (Exception e) {
            Log.d(background.this.TAG, "catch Excepetion");
        }
        background.this.getsna(background.this.ss);
        Log.d(background.this.TAG, background.this.ss);
    }
```

setsna检查转换为的文字是否为中午“傻我是逼”：

```
public void getsna(String flag) {
    if (flag.length() == 4) {
        int[] as = new int[flag.length()];
        for (int i = 0; i < flag.length(); i++) {
            as[i] = flag.charAt(i) & 65535;
        }
        for (int j = 0; j < 4; j++) {
            for (int k = j + 1; k < 4; k++) {
                if (as[j] > as[k]) {
                    int temp = as[j];
                    as[j] = as[k];
                    as[k] = temp;
                }
            }
        }
        if (as[0] == 20667 && as[1] == 25105 && as[2] == 26159 && as[3] == 36924) {
            Toast.makeText(getContext(), "You get the sorted flag: 20667 25105 26159 36924", 0).show();
        } else {
            Toast.makeText(getContext(), "wrong input", 0).show();
        }
    }
}
```

“傻我是逼”就是20667/25105/26159/36924这几个数字对应的中文。

如果语音输入为“傻我是逼”的话就打印："You get the sorted flag：20667 25105 26159 36924"

大致意思是，你得到了打乱顺序的flag。

打乱顺序是“傻我是逼”，原顺序估计就是“我是傻逼”。

最终得到flag为：flag{25105 26159 20667 36924}

欢迎关注我的微博：大雄_RE。专注软件逆向，分享最新的好文章、好工具，追踪行业大佬的研究成果。