


vysor的实现原理是什么？

Vysor 这是一款被大家称作神器的工具，在chrome安装一个插件无需root就能连接android，实现同步手机操作和投影显示。 android屏幕...显示全部

关注问题写回答添加评论分享邀请回答

4 个回答默认排序



stevensunzh
妈咪圈, mamiquan.cn

146 人赞同了该回答

跟据 @黑魔法师 的回复，搜索到了这篇文章：[vysor原理以及Android同屏方案](#)
原文内容如下：

vysor原理以及Android同屏方案

2016-07-02

vysor是一个免root实现电脑控制手机的chrome插件，目前也有几款类似的通过电脑控制手机的软件，不过都需要root权限，并且流畅度并不高。vysor没有多余的功能，流畅度也很高，刚接触到这款插件时我惊讶于它的流畅度以及免root，就一直对它的实现原理很感兴趣。这款插件我用了大半年，最近在升级后我发现它居然开始收费了，终生版需要39.99美元，不过经过简单的分析后我很轻松的破解了它的pro版，在分析的过程中发现它的原理并不复杂，所以就打算自己也实现一个类似的软件。

vysor原理以及Android同屏方案截屏常见的方案

在介绍vysor的原理前我先简单介绍一下目前公开的截屏方案。

- View.getDrawingCache()

这是最常见的应用内截屏方法，这个函数的原理就是通过view的Cache来获取一个bitmap对象，然后保存成图片文件，这种截屏方式非常的简单，但是局限性也很明显，首先它只能截取应用内部的界面，甚至连状态栏都不能截取到。其次是对某些view的兼容性也不好，比如webview内的内容也无法截取。

- 读取/dev/graphics/fb0

因为Android是基于linux内核，所以我们也能在android中找到framebuffer这个设备，我们可以通过读取/dev/graphics/fb0这个帧缓存文件中的数据来获取屏幕上的内容，但是这个文件是system权限的，所以只有通过root才能读取到其中的内容，并且直接通过framebuffer读取出来的画面还需要转换成rgb才能正常显示。下面是通过adb读取这个文件内容的效果。

- 反射调用SurfaceControl.screenshot()/Surface.screenshot()


SurfaceControl.screenshot()(低版本是Surface.screenshot())是系统内部提供的截屏函数，但是这个函数是@hide的，所以无法直接调用，需要反射调用。我尝试反射调用这个函数，但是函数返回的是null，后面发现SurfaceControl这个类也是隐藏的，所以从用户代码中无法获取这个类。也有一些方法能够调用到这个函数，比如重新编译一套sdk，或者在源码环境下编译apk，但是这种方案兼容性太差，只能在特定ROM下成功运行。

- screencap -p xxx.png/screenshot xxx.png

这两个是在shell下调用的命令，通过adb shell可以直接截图，但是在代码里调用则需要系统权限，所以无法调用。可以看到要实现类似vysor的同步操作，可以使用这两个命令来截取屏幕然后传到电脑显示，但是我自己实现后发现这种方式非常的卡，因为这两个命令不能压缩图片，所以导致获取和生成图片的时间非常长。

- MediaProjection,VirtualDisplay (>=5.0)

在5.0以后，google开放了截屏的接口，可以通过“虚拟屏幕”方式会弹出确认对话框，并且只在5.0上有效，所以我没有对这种



下载知乎客户端
与世界分享知识、经验和见解

相关问题

MVC 模式的原理，它在 Android 中是如何运用的？ 7 个回答


什么原理让谷歌浏览器如此流畅，但是占用了较多的内存？ 4 个回答

谷歌验证 (Google Authenticator) 的实现原理是什么？ 5 个回答

如何学好操作系统原理这门课？ 10 个回答


cs甩狙是什么原理？ 19 个回答

相关推荐




内向者优势演讲课
大卫祁
共 11 节课

▶ 试听



麦肯锡顾问教你金字塔原理
为轩 Welson
★★★★★ 311 人参与



深入分布式缓存：从原理到实践
521 人读过

阅读

刘看山 · 知乎指南 · 知乎协议 · 应用 · 工作

申请开通知乎机构号

侵权举报 · 网上有害信息举报专区

违法和不良信息举报：010-82716601

儿童色情信息举报专区

联系我们 © 2018 知乎

https://www.zhihu.com/question/46229570

146收起评论分享收藏感谢

1/9



可以看到，上述方案中并没有解决方案能够做到兼容性和效率都非常完美，但是我在接触到vysor后发现它不但画面清晰，流畅，而且不需要root。那么它是用了什么黑科技呢？下面我们反编译它的代码来研究一下它的实现机制。

vysor原理以及Android同屏方案vysor原理

反编译vysor的apk后可以发现它的代码并不多，通过分析后我发现它的核心代码在Main这个类中。

首先来看Main函数的main方法，这个方法比较长，这里直接贴出源码。

```
public static void main(String[] args) throws Exception {
    if (args.length > 0) {
        commandLinePassword = args[0];
        Log.i(LOGTAG, "Received command line password: " + commandLinePassword);
    }
    Looper.prepare();
    looper = Looper.myLooper();
    AsyncServer server = new AsyncServer();
    AsyncHttpServer httpServer = new AsyncHttpServer() {
        protected boolean onRequest(AsyncHttpRequest request, AsyncHttpSer
            Log.i(Main.LOGTAG, request.getHeaders().toString());
            return super.onRequest(request, response);
        }
    };
    String str = "getInstance";
    Object[] objArr = new Object[0];
    InputManager im = (InputManager) InputManager.class.getDeclaredMethod(r20, n
    str = "obtain";
    MotionEvent.class.getDeclaredMethod(r20, new Class[0]).setAccessible(true);
    str = "injectInputEvent";
    Method injectInputEventMethod = InputManager.class.getMethod(r20, new Class[
    KeyCharacterMap kcm = KeyCharacterMap.load(-1);
    Class cls = Class.forName("android.os.ServiceManager");
    Method getServiceMethod = cls.getDeclaredMethod("getService", new Class[]{St
    IClipboard clipboard = IClipboard.Stub.asInterface((IBinder) getServiceMetho
    clipboard.addPrimaryClipChangedListener(new AnonymousClass3(clipboard), null
    IPowerManager pm = IPowerManager.Stub.asInterface((IBinder) getServiceMethod
    IWindowManager wm = IWindowManager.Stub.asInterface((IBinder) getServiceMeth
    IRotationWatcher watcher = new Stub() {
        public void onRotationChanged(int rotation) throws RemoteException {
            if (Main.webSocket != null) {
                Point displaySize = SurfaceControlVirtualDisplayFactory.getCurre
                JSONObject json = new JSONObject();
                try {
                    json.put("type", "displaySize");
                    json.put("screenWidth", displaySize.x);
                    json.put("screenHeight", displaySize.y);
                    json.put("nav", Main.hasNavBar());
                    Main.webSocket.send(json.toString());
                } catch (JSONException e) {
                }
            }
        }
    };
    wm.watchRotation(watcher);
    httpServer.get("/screenshot.jpg", new AnonymousClass5(wm));
    httpServer.webSocket("/input", "mirror-protocol", new AnonymousClass6(watche
    httpServer.get("/h264", new AnonymousClass7(im, injectInputEventMethod, pm,
    Log.i(LOGTAG, "Server starting");
    AsyncServerSocket rawSocket = server.listen(null, 53517, new AnonymousClass8
    if (httpServer.listen(server, 53516) == null || rawSocket == null) {
        System.out.println("No server socket?");
        Log.e(LOGTAG, "No server socket?");
        throw new AssertionError("No server socket");
    }
}
```



```

System.out.println("Started");
Log.i(LOGTAG, "Waiting for exit");
Looper.loop();
Log.i(LOGTAG, "Looper done");
server.stop();
if (current != null) {
    current.stop();
    current = null;
}
Log.i(LOGTAG, "Done!");
System.exit(0);
}

```

这个软件koushikdutta是由开发的，这个团队以前发布过一个非常流行的开源网络库:async。在这个项目中也用到了这个开源库。main函数主要是新建了一个httpserver然后开放了几个接口，通过screenshot.jpg获取截图，通过socket input接口来发送点击信息，通过h264这个接口来获取实时的屏幕视频流。每一个接口都有对应的响应函数，这里我们主要研究截图，所以就看看screenshot这个接口。h264这个接口传输的是实时的视频流，所以就流畅性来说应该会更好，它也是通过virtualdisplay来实现的有兴趣的读者可以自行研究。

接下来我们来看screenshot对应的响应函数AnonymousClass5的实现代码。

```

* renamed from: com.koushikdutta.vysor.Main.5 */
static class AnonymousClass5 implements HttpServerRequestCallback {
    final /* synthetic */ IWindowManager val$wm;

    AnonymousClass5(IWindowManager iWindowManager) {
        this.val$wm = iWindowManager;
    }

    public void onRequest(AsyncHttpRequest request, AsyncHttpServerResponse response) {
        if (Main.checkPassword(request.getQuery().getString("password"))) {
            Log.i(Main.LOGTAG, "screenshot authentication success");
            try {
                Bitmap bitmap = EncoderFeeder.screenshot(this.val$wm);
                ByteArrayOutputStream bout = new ByteArrayOutputStream();
                bitmap.compress(CompressFormat.JPEG, 100, bout);
                bout.flush();
                response.send("image/jpeg", bout.toByteArray());
                return;
            } catch (Exception e) {
                response.code(500);
                response.send(e.toString());
                return;
            }
        }
        Log.i(Main.LOGTAG, "screenshot authentication failed");
        response.code(401);
        response.send("Not Authorized.");
    }
}

```

这个类传入了一个wm类，这个类是用来监听屏幕旋转的，这里不用管它。另外在vysor开始运行时，会随机生成一个验证码，只有验证通过才能进行连接，所以这里有一个验证的过程，这里也不管。可以看到这个类定义的响应函数的代码非常简单，就是通过EncoderFeeder.screenshot()函数来过去截图的bitmap，然后返回给请求端。那么EncoderFeeder.screenshot这个函数是怎样实现截图的呢？

```

c static Bitmap screenshot(IWindowManager wm) throws Exception {
    String surfaceClassName;
    Point size = SurfaceControlVirtualDisplayFactory.getCurrentDisplaySize(false);
    if (VERSION.SDK_INT <= 17) {
        surfaceClassName = "android.view.Surface";
    } else {

```

▲ 146

🗨 收起评论

★ 收藏

❤ 感谢



```

        surfaceClassName = "android.view.SurfaceControl";
    }
    Bitmap b = (Bitmap) Class.forName(surfaceClassName).getDeclaredMethod("screenshot")
    int rotation = wm.getRotation();
    if (rotation == 0) {
        return b;
    }
    Matrix m = new Matrix();
    if (rotation == 1) {
        m.postRotate(-90.0f);
    } else if (rotation == 2) {
        m.postRotate(-180.0f);
    } else if (rotation == 3) {
        m.postRotate(-270.0f);
    }
    return Bitmap.createBitmap(b, 0, 0, size.x, size.y, m, false);
}

```

这里的截图的核心代码也是反射调用Surface/SurfaceControl的screenshot方法。但是我们前面已经了解到，这个类只有在系统权限下才能获取到，那么vysor又是怎么调用到这个函数的呢？我们可以确认的是vysor不是通过重编译sdk和使用系统签名来完成的，因为那样只能对特定的rom适用。

当时看到这里的代码后我也非常困惑，vysor是怎么调用到这个类的。我注意到了vysor的核心代码不是在某个Activity或者Service中而是在一个Main类中，按照一般的逻辑来说，这种实时传屏应该是放在Service中不断截屏然后发给服务端，所以我决定再看下它的服务端的代码。

vysor的服务端是一个chrome插件，用javascript写成的，所以找到源码比java更加简单。虽然js经过混淆，但是很容易的可以通过一些工具来解密。然后就是分析它的代码了，终于被我找到了关键的代码。

```

function y(e, t, n) {
    m(e, "Connecting...");

    function o(o) {
        var i = Math.round(Math.random() * (1 << 30)).toString(16);
        var r = "echo -n " + i + " > /data/local/tmp/vysor.pwd ; chmod 600 /data
        Adb.shell({
            command: "ls -l /system/bin/app_process*",
            serialno: e
        }, function(s) {
            var c = "/system/bin/app_process";
            if (s && s.indexOf("app_process32") != -1) {
                c += "32"
            }
            Adb.sendClientCommand({
                command: 'shell:sh -c "CLASSPATH=' + o + " " + c + " /system/bin
                serialno: e
            }, function(o) {
                Adb.shell({
                    serialno: e,
                    command: 'sh -c "' + r + '"'
                }, function(e) {
                    Socket.eat(o);
                    n(t, i)
                })
            })
        })
    }
}

```

可以看到上面的代码是调用了adb shell命令来启动com.koushikdutta.vysor.Main类，并且上面获取了app_process这个程序。相信对android熟悉读者已经明白它的原理了。我简单解释一下。我们已经知道Surface/SurfaceControl这两个类是需要具有相应权限的程序才能调用到，用户进程无法获取到。adb shell可以调用screencap或者screenshot来截图



以也就是说adb shell是具有截屏权限的也就是能够调用到Surface/SurfaceControl。那么我们怎么通过adb shell来调用到这两个类呢，答案就是app_process。app_process可以直接运行一个普通的java类，详细的资料大家可以在网上找到。也就是说我们通过adb shell运行app_process，然后通过app_process来运行一个java类，在java类中就可以访问到Surface/SurfaceControl这两个类，是不是很巧妙？

理论有了，下面我们来通过代码验证。这里我们可以直接使用vysor的代码。因为是测试用所以我没有添加其他功能。

```
public class Main {

    static Looper looper;

    public static void main(String[] args) {

        AsyncHttpServer httpServer = new AsyncHttpServer() {
            protected boolean onRequest(AsyncHttpRequest request, AsyncHttpServletResponse response) {
                return super.onRequest(request, response);
            }
        };

        Looper.prepare();
        looper = Looper.myLooper();
        System.out.println("Andcast Main Entry!");
        AsyncServer server = new AsyncServer();
        httpServer.get("/screenshot.jpg", new AnonymousClass5());
        httpServer.listen(server, 53516);

        Looper.loop();

    }

    /* renamed from: com.koushikdutta.vysor.Main.5 */
    static class AnonymousClass5 implements HttpServerRequestCallback {

        public void onRequest(AsyncHttpRequest request, AsyncHttpServletResponse response) {
            try {
                Bitmap bitmap = ScreenShotFb.screenshot();
                ByteArrayOutputStream bout = new ByteArrayOutputStream();
                bitmap.compress(Bitmap.CompressFormat.JPEG, 100, bout);
                bout.flush();
                response.send("image/jpeg", bout.toByteArray());
                return;
            } catch (Exception e) {
                response.code(500);
                response.send(e.toString());
                return;
            }
        }
    }
}
```

编译成apk然后安装后，我们使用adb shell来运行这个类，主要方法如下，首先导出classpath，否则会提示找不到类。

```
export CLASSPATH=/data/app/com.zke1e.andcast-1/base.apk
```

然后调用app_process来启动这个类。

```
exec app_process /system/bin com.zke1e.andcast.Main '$@'
```

可以看到类已经成功运行了，正在监听请求。

▲ 146

🗨 收起评论

★ 收藏

♥ 感谢

编辑于 2016-09-28

⇒ 切换为时间排序

1 年前

1 年前

1 年前

1 年前

1 年前

1 年前

1 年前

1 年前

1 年前

👍 贊

♥ 感谢



知乎用户

1 年前

弱弱问下，到底pro版是怎么破解的哇

👍 赞



程凌杰

1 年前

很牛逼 已经实现了 膜拜大神

👍 赞



攻城狮

10 个月前

您好 您分享前请尊重原作者

原文地址[vysor原理以及Android同屏方案 | Zke1ev3n's Blog](#)

👍 1



stevensunzh (作者) 回复 攻城狮

10 个月前

您好，不好意思，我在回复前已经附上了原文网址，并做了说明，如果您觉得转内容过来不合适我把回复删除。

👍 赞 💬 查看对话



攻城狮 回复 stevensunzh (作者)

10 个月前

没事，附上地址没事的

👍 赞 💬 查看对话



Elio 回复 不说

7 个月前

自己手机用数据线连接电脑,隐私?

👍 赞 💬 查看对话



HalfmanG2

3 个月前

价值36美金的 Hello World! 学习了，收获很大，赞

👍 赞



知乎用户 回复 HalfmanG2

3 个月前

虽然本身并不复杂,但说是hw过分的

👍 赞 💬 查看对话



HalfmanG2 回复 知乎用户

3 个月前

核心代码不就是用app_process起了个 HelloWorld 嘛~~哈哈

👍 赞 💬 查看对话



默默无闻的菜鸟张

3 个月前

感谢分享

👍 赞



雄风屁驴子

20 天前

愿意和别人分享知识的人，是这个世界最需要的人

写下你的评论...



黑魔法师

Android程序员

8 人赞同了该回答

用系统API创建一个VirtualDisplay来实时获取屏幕录像,, 通过反射拿到SurfaceControl类,用MediaCodec createInputSurface()方法用于接收图像, 然后用h264编码 发到浏览器上。手机与浏览器用Websocket建立连接,

编辑于 2016-09-29

▲ 146

💬 收起评论

★ 收藏

♥ 感谢



12 条评论

切换为时间排序

孙圣翔

1 年前

那点击的原理呢？

赞

黑魔法师 (作者) 回复 孙圣翔

1 年前

反射。inputmanager里有一个injecttouchevent

赞 查看对话

孙圣翔 回复 黑魔法师 (作者)

1 年前

thanks a lot.

1 查看对话

亦可

1 年前

赞一个！

赞

Taozi

1 年前

我觉得vysor没有用到VirtualDisplay，因为这个api是5.0后开放的，并且投屏时需要用户确认。我觉得vysor是通过反射调用SurfaceControl的screenshot方法。想请教一下，该方法返回的是一个bitmap对象，具体怎样使用h264推流呢？

赞

黑魔法师 (作者) 回复 Taozi

1 年前

1. 对于5.0以上的机器是使用 virtualdisplay的。对于5.0以下的是使用 surfacecontrol里的 createdisplay等方法实现，只是需要反射和aidl。不是截屏。bitmap转h264不太了解，应该需要应用自己去引用一些库

赞 查看对话

Taozi 回复 黑魔法师 (作者)

1 年前

谢谢提醒还有createdisplay这么个方法，我看到这个方法返回的是一个IBinder，具体怎么使用呢？

赞 查看对话

黑魔法师 (作者) 回复 Taozi

1 年前

这个是aidl接口

赞 查看对话

黑魔法师 (作者) 回复 Taozi

1 年前

需要在系统源码里找相应的aidl文件复制到项目中，再具体的搜一下教程吧

赞 查看对话

陈端

1 年前

有没有针对vysor做类似的开源项目？

1

黑魔法师 (作者) 回复 陈端

1 年前

没有做，因为在github找到了一个remotedroid的项目，基本的实现跟vysor一样。可以看一下

1 查看对话

陈端 回复 黑魔法师 (作者)

好的谢谢

1

查看对话

写下你的评论...

 **yasy**

能否使用客户端以无线的方式去连接手机和截屏？

发布于 2018-01-01

0

添加评论

分享

收藏

感谢

 **宋先生**
工程师

vysor只能进行 屏幕截屏，远程控制没遇到比较细的原理讲解.

发布于 2017-04-01

0

收起评论

分享

收藏

感谢

1 条评论

切换为时间排序

 **Jrsen Zhu** 9 个月前

远程控制原理类似 Android shell中可以直接调用input命令去实现点击滑动等操作 所以我才
vysor应该实现了一个类似的命令然后通过chrome端用shell权限进行调用

赞

写下你的评论...

写回答