2018 年全国大学生信息安全竞赛

作品报告

作品名称	基于蜜罐机制的 Android 主动防御系统
电子邮箱	610572182@qq.com
提交日期	2018.6.5

填写说明

1. 所有参赛项目必须为一个基本完整的设计。作品报告书旨在能够清晰准确地阐述（或图示）该参赛队的参赛项目（或方案）。

2. 作品报告采用A4纸撰写。除标题外，所有内容必需为宋体、小四号字、1.5倍行距。

3. 作品报告中各项目说明文字部分仅供参考，作品报告书撰写完毕后，请删除所有说明文字。（本页不删除）

4. 作品报告模板里已经列的内容仅供参考，作者可以在此基础上增加内容或对文档结构进行微调。

5. 为保证网评的公平、公正，作品报告中应避免出现作者所在学校、院系和指导教师等泄露身份的信息。

摘要

随着移动智能时代的到来，和智能手机相关的犯罪行为和用户私密信息泄露问题凸显。其中，大多数案例均与用户手机失窃或丢失相关。当前，绝大多数手机在丢失后对用户信息的保护机制是不健全的，仅存在密码开锁这一单一的被动防御方式，该方式存在诸多缺点，如密码被偷窥、弱口令、无法防止暴力刷机等问题。

本作品基于当前占据中国市场份额最大的Android手机平台，利用Android系统的多用户机制，创新性地融合了在信息安全领域当前广泛研究的蜜罐技术，通过看似不设密码的手机解锁手势，引诱偷窃者进入蜜罐系统。在隐藏并保护真正用户系统的同时，监控偷窃者的位置、图像和行为信息，降低手机被暴力刷机的风险，提高手机的找回概率，帮助警方进行犯罪取证。

本作品使得Android手机的安全性获得了很大的提升，具有极大的使用性和应用前景。本作品所具体的创新之处如下：

（1）蜜罐技术+智能手机

本作品将蜜罐技术创新性地应用到了智能手机领域，利用Android系统的多用户机制实现蜜罐系统，在保护真实系统的同时，为违法使用者呈现出一个看似不设防的虚假系统。

（2）信息保护+犯罪取证

盗窃者进入蜜罐系统后，真实系统将被隐藏，用户信息将被备份。同时，蜜罐系统会通过各种手段记录并上传盗窃者行为及手机所处的环境信息，增大手机找回成功率。在确定无法找回手机时，用户还可以远程控制手机，抹去所有信息。

（3）主动的防御理念

传统的防御机制在手机丢失后，需要被动地接受机主的指令，才能使得防御机制生效，存在诸多问题。本作品则通过判断手机是否进入蜜罐系统从而主动完成整

个信息保护流程，防御机制更加有效。

（4）高兼容度+高效率

由于本作品基于Android原生系统开发，理论上可以适配市面上所有Android机型，同时作品本身所占资源极少，不会对用户正常使用手机产生影响。

我们希望通过本作品，对用户手机进行切实保护，降低手机犯罪成功率，保护用户私密数据，同时，唤醒国内群众的信息安全及隐私保护意识。

关键词： 蜜罐系统 Android多用户 信息保护 数据备份 犯罪取证

目 录

第一章 作品概述

1.1 研究背景

随着科技的发展和社会的进步，手机已经成为全世界普及率最高的电子产品，而随着手机的更新换代，其作用逐渐从单一的通话功能向全方面发展。如今人们可以方便快捷地使用手机进行社交娱乐，移动支付等操作。

承载了多功能的同时，手机逐渐成为了一个储存着大量个人信息与隐私的数据库，随着个人信息的价值的提升，信息泄露的可能性也越来越大。

如图 1-1 为国内手机网站中关村在线（ZOL）的调查结果。

图 1-1 中关村在线调查结果

从图中我们可以看出，超过六成的用户表示曾经丢失过手机。手机的丢失带来的不仅仅是直接的经济损失，更严重的是因处理不当，保护不周而导致的个人隐私的泄露。

因此，我们希望设计一款产品，在真正保护失窃手机用户的个人信息的同时，记录偷窃者的犯罪路径以及犯罪行为，并在作品普及的过程中，逐渐唤醒国内用户对于个人信息的保护意识。

在计算机领域中，如果想要捕获以及分析攻击者的行为，最常见的做法是采用蜜罐技术。蜜罐技术本质上是一种对攻击方进行欺骗的技术，通过布置一些作为诱饵的网络服务或者信息，诱使攻击方对它们实施攻击，从而可以对攻击行为进行捕获和分析。由此我们希望把蜜罐技术应用于智能手机领域，引诱偷盗者进入虚假的手机系统，在保护隐私信息的同时，记录其在虚假手机系统中的行为，为手机用户提供多方面的信息保护。

1.2 相关工作

1.2.1 蜜罐系统产品的发展现状

目前国外的学术界对蜜罐技术研究最多的是蜜网项目(Honey net Project)组织，它是一个非官方、非营利的由安全专家组成的组织，专门致力于了解黑客团体使用的工具、策略和动机，并形成了一系列的理论基础。除此之外其它研究机构和公司也已开始广泛研究蜜罐及商业化生产，出现了 KFSensor 及 Specter 等商业蜜罐产品。

1.2.2 Android 手机系统的保护

Android 是以 Linux 为核心的手机操作平台，作为一款开放式的操作系统，随着 Android 的快速发展，它越来越多地受到开发者的欢迎，其系统的开放性使其所占的全球市场份额高达 85%。在现实生活中，用户在 Android 手机被盗时往往束手无策。Android 手机生产厂家对于其手机的保护，往往局限于密码解锁保护，手势解锁保护以及人脸识别解锁保护，而这些保护措施在面对刷机时毫无办法，偷窃者会把盗取的手机进行备份后，将该备份刷到另一台备用手机上，从而在备用手机上提取所需要的信息。市面上，部分品牌的 Android 手机会在手势认证失败多次后调用摄像头进行拍照，尽管这是一个不错的尝试，但是拍摄的照片仍存放在本机，一旦手机丢失就毫无作用。此外，手机丢失者往往对偷窃者的信

息一无所知，这使得 Android 手机一旦丢失很难找回。综上所述，如今失窃手机的信息保护和找回方面还存在着很大问题。

1.2.3 多用户的相关研究

Android中的多用户与Windows的多用户类似，可以支持多个用户使用系统。不同用户的设置各不相同，并且不同用户安装的应用及应用数据也不相同。但是系统中和硬件相关的设置则是共用的。目前大多数用户们使用多用户功能只是为了防止某个自己认识的人窥探信息，或者防止孩童无意间抹去手机信息，并不具备手机丢失时的信息保护功能。

因此为了保护用户个人信息，我们将多用户功能与蜜罐技术相结合，引诱偷窃者进入新的用户即蜜罐系统，收集其犯罪证据的同时，有效地保护了用户的个人信息与隐私。

1.3 研究意义

在当今的信息社会中，手机在人们生活中占据了越来越高的地位，随着智能手机日新月异的发展，其能提供的服务也越来越多样化，伴随而来的是越来越多的信息交流，小到手机应用的下载注册，大到股票基金等移动支付，可以说手机就是个人信息的集中营。与此同时，手机丢失事件层出不穷，带来的个人信息的丢失影响更为严重。而手机厂商们对于手机中个人信息的保护还存在着巨大的漏洞。因此，保护机主的个人隐私是一个十分紧迫的命题。

1.3.1 保护手机个人信息与隐私

在手机丢失的场景中，由于作为隐私信息载体的手机落入不法分子手中，且目前的技术破坏手段难以防范，使得隐私保护陷入被动。本作品基于蜜罐概念，通过诱导不法分子获取事先伪造的虚假信息，误以为无需使用更复杂的技术破坏手段，

使得真正的隐私信息得到保护；本作品在降低不法分子获取机主隐私的可能性的同时，结合备份以及远程指令功能，实现隐私信息的找回以及危险信息的销毁。

1.3.2 为手机找回提供依据

本作品利用蜜罐系统的“犯罪取证”功能，不仅可以将手机位置信息上传至服务器，还能够记录偷窃者的行为以及个人信息(例如拍照)，从而帮助用户增加追回手机的可能性，尽力弥补因手机丢失造成的损失。

1.3.3 补救 Android 手机固有的安全隐患

Android手机由于其固有的开放性，导致个人信息泄露的风险较大。同时对于没有备份习惯的用户，其手机一旦丢失，个人信息将无法找回，同时面临着信息被盗取的风险。而本作品的信息保护机制可以极大程度地避免机主信息的丢失与泄露，其具体表现为：本作品的自动备份的功能使得机主可以随时获得其手机中的个人信息，此外手机中蜜罐系统能够保护信息不被不法操作者盗取，从而消除了Android手机在信息保护方面的安全隐患。

1.3.4 为手机个人信息保护提供了新的研究方向

本作品把“蜜罐”、“手机”、“多用户”跨界结合，为智能手机上个人信息保护的研究提供了新的思路与方向。蜜罐系统是防御方为了改变攻防博弈不对称局面而引入的一种主动防御技术，它可以诱骗攻击者对其进行非法使用，从而对攻击者的行为进行捕获和分析，这一技术如今广泛应用在计算机领域，并成为互联网安全威胁监测与分析的一种主要技术手段。手机丢失的用户正是处于被动的一方，本作品提供的方案，可以在很大程度上扭转用户的被动局面：引诱盗窃者进入基于Android多用户模式设计的蜜罐系统，主动备份手机个人信息，记录盗窃者个人信息；在保证信息不丢失的基础上，为找回手机提供更大的可能。

1.4 作品简介

基于蜜罐技术与Android系统的多用户模式，本作品设计了一个能够主动防御的信息保护系统。用户手机丢失后，可以自动化完成一整套信息保护流程。其中包括：备份用户手机的个人信息，上传至服务器供用户下载还原；引诱偷窃者进入蜜罐系统并且识别敏感操作，捕捉其个人信息，上传至服务器供用户取证。这一保护流程很大程度上弥补了Android手机在隐私保护这一方面的不足，挽回了用户的损失，其操作流程如下：

（1）用户在日常使用中可以自定义两种手势解锁方法。一种手势是对应进入机主系统，一般较为复杂，偷窃者无法暴力破解；另一种是蜜罐系统的手势，较为简单，并且以文字说明的方式显示在手机锁屏界面上，如‘滑动解锁’，诱使偷窃者使用简单的手势进入蜜罐系统。

（2）偷窃者进入蜜罐系统后，机主系统的隐私被隔离，个人信息将得到充分保护，同时，一部分隐私信息会立刻备份并加密传输到监控服务器供用户下载还原。

（3）偷窃者使用蜜罐系统的过程中，一旦被识别到敏感操作，前置摄像头会开启并取证，同时手机定位将自动打开，手机位置信息将上传。蜜罐系统与正常系统无异，提供了虚假的应用程序比如qq、微信、支付宝等，同时还伪造了的短信，联系人等关键信息，这就使得偷窃者误以为自己正在使用用户的手机。本作品通过这种方式隐藏了真正的系统，达到了保护真正系统中的个人信息的目的。

(4)为了更加保险地保护个人信息，用户可以通过服务器远程向手机下达命令，将手机中个人信息彻底抹去，彻底防止隐私的泄露。

在整个防御流程中，本作品不仅可以成功保护用户手机中的个人信息，还可以通过犯罪取证功能，为后续手机的找回创造条件，而远程控制功能也解决了用户的后顾之忧，杜绝了隐私的泄露。此外，本作品基于原生Android系统开发，理论上可以适配市面上的所有机型，同时其内存占比很小，不会对用户正常使用产生影响，应用前景广阔。

1.5 特色创新

1.5.1 跨界结合——创新性

蜜罐技术是在计算机领域实现的一种主动防御安全技术，后来发展而成的蜜罐系统更是改变了网络攻防博弈的不对称形势。

在智能手机领域，手机丢失常常让用户陷入极其被动的处境，用户对此束手无策，Android手机用户因为其自身手机开放性的特点，更是面临着信息丢失以及隐私泄露等巨大的风险。

为了改变这一不平衡的局面，我们创新性地结合Android手机的多用户模式，将蜜罐技术运用在智能手机领域。本作品设计的蜜罐系统，在保护个人信息完整的同时，会记录其犯罪过程并取证，而这些信息所具有的抗抵赖性，会在追回手机这一过程中发挥巨大的作用。这在很大程度上扭转了用户失窃时被动的局面。

1.5.2 防御理念——主动性

传统的防御机制中，保护手机的操作是需要用户主动发出指令才会执行的，而从用户丢失手机到发出指令这段时间，恰恰是信息保护的黄金时间。本作品弥补了这一缺陷，通过判断手机是否进入蜜罐系统，从而主动完成整个信息保护流程，使得保护在第一时间生效。即使机主的远程指令下达不及时，甚至未下达指令，蜜罐防御机制也能达到很好的防御效果。

1.5.3 系统伪造——隐蔽性

本作品将蜜罐技术与Android系统的多用户机制结合，设计的蜜罐系统与机主系统无差别，内容完善，具有极高的隐蔽性。同时，作品的诱导界面具有很好的欺骗性，使得偷窃者更容易直接进入蜜罐系统。在偷窃者误以为进入了机主系统，并希望继续窥探用户隐私信息时，为机主个人信息的备份传输以及偷窃者个人信息

的记录提供更多的时间。

1.6 小结

本章介绍了当前时代个人信息保护越来越受到重视这一大环境，更进一步引出了Android手机面临的信息保护方面的安全隐患；同时本章分析了当前有关蜜罐技术以及多用户等相关工作的研究进展，进一步引出这一将蜜罐技术与手机多用户结合的作品。紧接着本章介绍了作品的工作流程、作品突出的“保护+取证”的特色，应用前景的广阔，更进一步突出了该作品在如今这个时代的重要性与必要性。

第二章 作品设计与实现

本章主要阐述系统的设计以及具体功能的实现方法。由于技术破坏获取隐私信息的方法难以防御，本系统将以“降低”不法分子“获取”隐私信息的成本的方式，诱导不法分子更长时间地处于预设的安全机制之下，避免隐私信息遭到技术性破坏而泄露。本作品大胆地取消了显示身份认证的安全机制（如明文密码），而采用隐式手势认证，结合 Android 的多用户机制实现了一个完成度较高的蜜罐系统，实现功能主要有锁屏诱导界面部分、手势识别和系统切换部分以及手机找回和信息保护部分。在设计部分，将以流程图为主，说明整个逻辑流程。在实现部分，则会根据不同的功能模块给出对应的实现方法和技术分析。

2.1 整体结构

在不采用显示身份认证的前提之下，锁屏将是不法分子接触到手机的第一环。因此锁屏功能将作为安全机制的逻辑第一层。同时在传统的安全机制中，锁屏与安全验证互相独立，并未参与到安全机制当中，因此锁屏也十分适合作为第一层的诱导 UI。

从锁屏开始，本作品的安全机制生效的流程如图 2-1 所示。

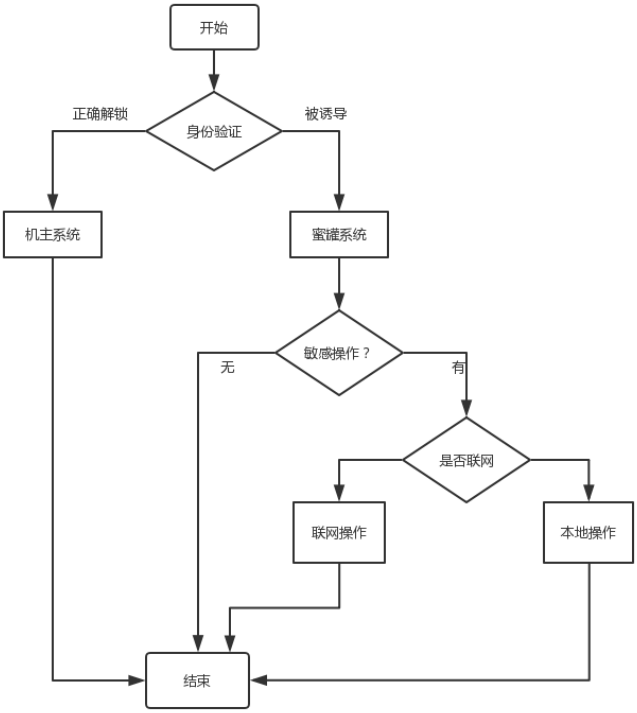

图 2- 1 安全机制生效流程

将逻辑流程涉及到的步骤按照功能点分类，可以将本作品分为诱导界面，系统切换和隐私隔离，信息搜集和隐私保护三个功能模块。在模块设计中，将给出这三个模块的更细化的逻辑流程。

逻辑流程如图 2-2 所示。

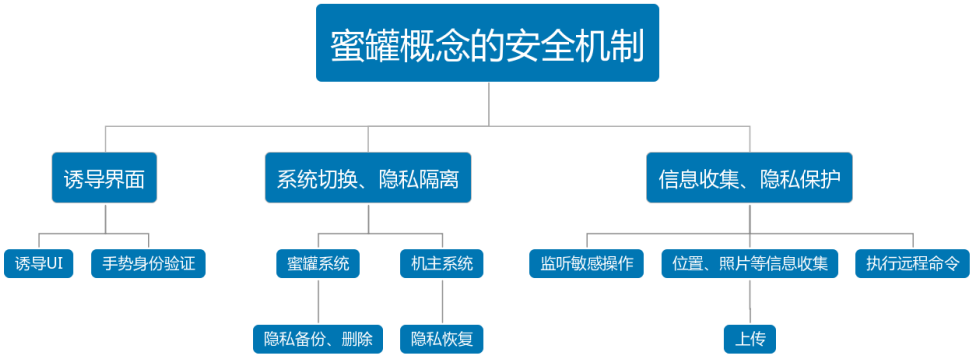

图 2- 2 作品逻辑流程

2.2 模块设计

2.2.1 诱导界面

诱导界面（即锁屏界面）是我们的安全机制生效的第一环，其中诱导界面的诱导性、简易性以及机主手势的隐蔽性、复杂性、识别准确性都有较高的要求。

诱导界面的流程如图 2-3 所示。

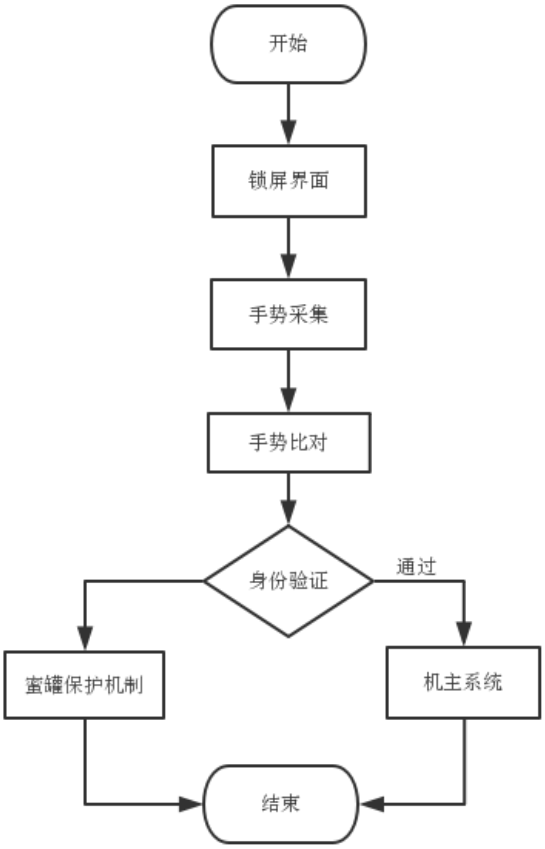

图 2- 3 诱导界面流程

2.2.2 系统切换、隐私隔离

本模块以手势（身份）识别为起点，具体流程图 2-4 所示：

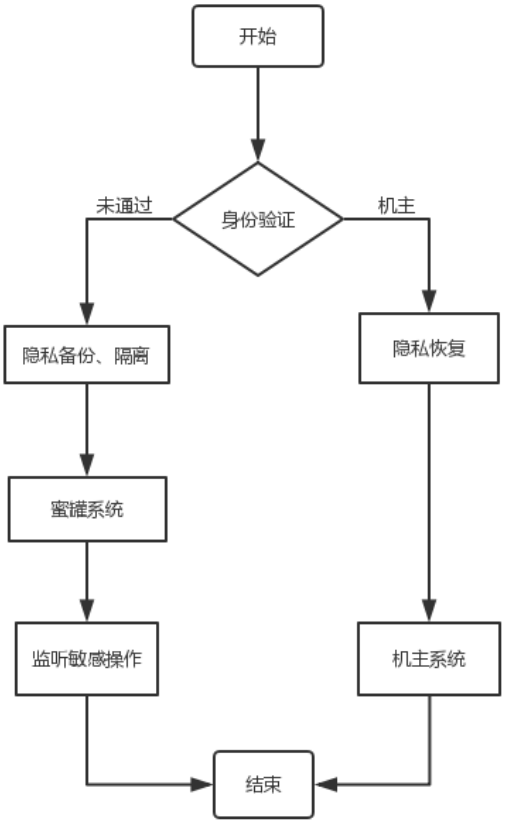

图 2- 4 系统切换、隐私隔离流程

2.2.3 信息搜集、隐私保护

本模块生效的流程如图 2-5 所示：

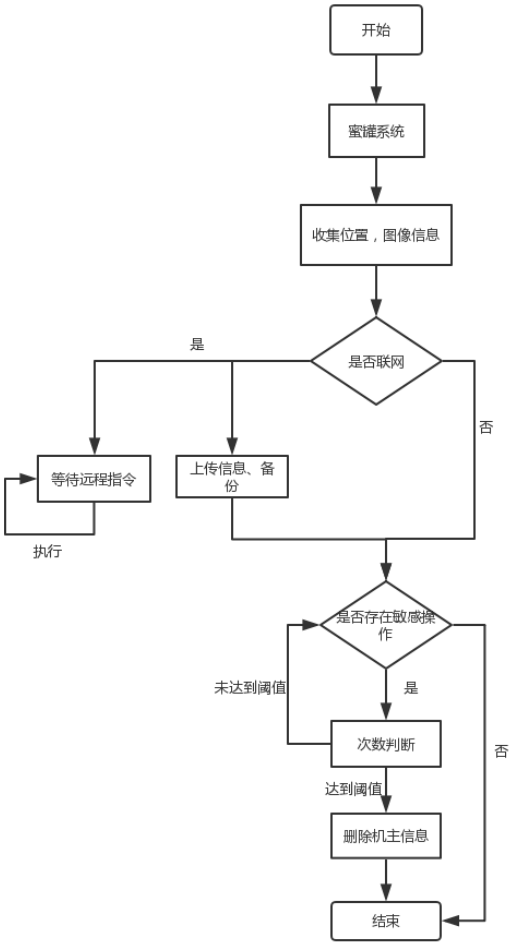

图 2- 5 信息收集、隐私保护流程

图 5

2.3 模块实现

2.3.1 诱导界面的实现

（一）前台 UI

（1）相关按键操作的屏蔽

通过实现屏蔽 Home 键、返回键，防止通知栏下拉等功能，保证诱导界面与系统的锁屏功能相同。具体实现方法采用了全屏悬浮窗的实现方法：

```
fun setParams(){
    params.type = WindowManager.LayoutParams.TYPE_SYSTEM_ALERT;
    //params.type = WindowManager.LayoutParams.TYPE_SYSTEM_OVERLAY;
    params.flags = WindowManager.LayoutParams.FLAG_FULLSCREEN or
            WindowManager.LayoutParams.FLAG_LAYOUT_IN_SCREEN or
            WindowManager.LayoutParams.FLAG_LAYOUT_NO_LIMITS or
            WindowManager.LayoutParams.FLAG_LAYOUT_INSET_DECOR or
            WindowManager.LayoutParams.FLAG_DISMISS_KEYGUARD or
            WindowManager.LayoutParams.FLAG_SHOW_WHEN_LOCKED or


            0x40000000;

    params.width = WindowManager.LayoutParams.MATCH_PARENT;
    params.height = WindowManager.LayoutParams.MATCH_PARENT;
    params.gravity = Gravity.TOP;
}

mView = LayoutInflater.from(context).inflate(R.layout.lock_view,null);
mWindowManager.addView(Singleton.mView, Singleton.params);
```

（2）滑动效果的实现

对于用户的滑动手势，需要有相关的滑动动画相应，以达到与系统锁屏功能相同的目标。相关动画实现：

```
mView.setOnTouchListener(object : View.OnTouchListener {
    //保存悬浮框最后位置的变量
    internal var lastX: Int = 0
    internal var lastY: Int = 0
    internal var paramX: Int = 0
    internal var paramY: Int = 0
    internal var downX: Int = 0
    internal var downY: Int = 0
    override fun onTouch(v: View, event: MotionEvent): Boolean {
        when (event.action) {
            MotionEvent.ACTION_DOWN -> {... }
            MotionEvent.ACTION_MOVE -> {... }

            MotionEvent.ACTION_UP -> {... }
```

```
            }
            return true
        }
})
```

（3）获取用户手势

由于不采用显式身份验证，我们采用隐式的手势验证来代替显式身份验证。获取手势的方法为：

```
gestureOverlayView.addOnGesturePerformedListener(object :
GestureOverlayView.OnGesturePerformedListener {


    override fun onGesturePerformed(p0: GestureOverlayView?, gesture: Gesture?) {

    }
})
```

（二）后台操作

（1）监听息屏操作

诱导界面（即锁屏界面）必定出现在亮屏之后，因此我们采取监听息屏操作来加载诱导界面的方式，实现锁屏的唤醒。具体方法为：

```
intentFilter.addAction(Intent.ACTION_SCREEN_OFF);
intentFilter.addAction(Intent.ACTION_USER_BACKGROUND);
intentFilter.addAction(Intent.ACTION_USER_FOREGROUND);
mReceiver = MyBroadcastReceiver();
registerReceiver(mReceiver,intentFilter);
```

（2）防杀保护

将诱导界面（实质为一个悬浮窗）设置为最无法跳出的最前台 UI，使得该界面无法通过内存清理等操作被杀掉

```
params.type = WindowManager.LayoutParams.TYPE_SYSTEM_ALERT;
```

（3）后台常驻

由于在 Android 机制下，进程常驻无法在用户完全不察觉的方式下实现，因此我们改用进程拉活的思路，即继承 NotificationListenerService 和 AccessibilityService，并配置相关权限文件，使得我们的进程与系统进程绑定，当进程结束后，由系统拉活。相关代码：

```
class MyAccessibility: AccessibilityService() {...}
class ImmortalService:NotificationListenerService() {...}
```

（4）开启自启

为了防止通过重启手机的方式绕过诱导界面，需要添加开机自启功能。相关实现方法为：

设置 BroadcastReceiver，对

actionandroid：name=″android. intent. action. BOOT_COMPLETED″监听。

2. 3. 2 系统切换

（1）身份验证

不采用显式身份验证，通过获取用户手势与预设手势的比较来进行隐式身份验证。相关实现方法：

```
gestureOverlayView.addOnGesturePerformedListener(object :
GestureOverlayView.OnGesturePerformedListener {
    override fun onGesturePerformed(p0: GestureOverlayView?, gesture: Gesture?) {
        val predictions = gestureLib.recognize(gesture)
        for (pred in predictions) {
            if (pred.score > -0.1) {

                Log.i("ges","与手势【" + pred.name + "】相似度为" + pred.score);
            }
            if (pred.score > 8) {
                gestureOverlayView.visibility = View.INVISIBLE;
                switchUser(0);
                Log.i("ges","与手势【" + pred.name + "】相似度为" + pred.score);
            }
```

```
        }
    }
})
```

（2）蜜罐系统切换

由于在 Android 源码中，进行多用户切换的方法，ActivityManager 中的 switchUser()方法被隐藏，并且会进行严格的 System 权限（不同于 root 权限）检测，因此不能简单地通过反射机制调用该方法完成用户切换功能。在此情况下，我们采用 ADB 命令的方式实现用户切换。相关实现方法为：

```
public fun switchUser(userID:Int){
    if (myID != userID) {
        lockScreen();
        Log.i("myID2",""+ myID)
        var process = Runtime.getRuntime().exec("su");
        var os = DataOutputStream(process.outputStream);
        os.writeBytes("am switch-user " + userID + " \n");
        os.flush();
        os.writeBytes("exit\n");
        os.flush();
        process.waitFor();

    }
    removeScreenLock();
}
```

2. 3. 3 信息搜集及隐私保护

（一）信息搜集

（1）搜集 Sim 卡序列号信息、服务商信息：

```
@SuppressLint("MissingPermission")
fun getIccid(): String {
    var iccid = "N/A"
    iccid = myTelephonyManager.getSimSerialNumber()
    return iccid
```

```
fun getProvidersName(): String {
    var providersName = "N/A"
    var NetworkOperator = myTelephonyManager.getNetworkOperator()
    //IMSI 号前面 3 位 460 是国家，紧接着后面 2 位 00 02 是中国移动，01 是中国联通，03 是中国电信。
    //            Flog.d(TAG,"NetworkOperator=" + NetworkOperator);
    if (NetworkOperator.equals("46000") || NetworkOperator.equals("46002")) {
        providersName = "中国移动"//中国移动
    } else if (NetworkOperator.equals("46001")) {
        providersName = "中国联通"//中国联通
    } else if (NetworkOperator.equals("46003")) {
        providersName = "中国电信"//中国电信
    }
    return providersName
}
```

（2）搜集位置信息：

```
@SuppressLint("MissingPermission")
fun getLocation(): String{
    Log.i("test","hereLoc")
    val location =
locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
    Log.i("test","hereLoc")
    Log.i("test","hereLoc")
    return "经度:"+location.longitude+";纬度:"+location.latitude+";海拔:"+location.altitude+"\n" +
getStreet(location);
}

    fun getStreet(location: Location): String {
        Log.i("test","hereStreet")
        val geocoder = Geocoder(context)
        val stringBuilder = StringBuilder();
        //根据经纬度获取地理位置信息
        var addresses:List<Address> = geocoder.getFromLocation(location.getLatitude(),
location.getLongitude(), 1);
        Log.i("test","hereStreet")
        //根据地址获取地理位置信息
        //var addresses = geocoder.getFromLocationName(adressStr, 1);
```

```
        if (addresses.size > 0) {
            var address = addresses.get(0);
            for (i in addresses.indices) {
                stringBuilder.append(address.getAddressLine(i)).append("\n");
            }
            stringBuilder.append(address.getCountryName()).append("_");//国家
            stringBuilder.append(address.getAdminArea()).append("_");//省份

            stringBuilder.append(address.getLocality()).append("_");//市

            stringBuilder.append(address.getSubLocality()).append("_");//乡洲区
/*              stringBuilder.append(address.getThoroughfare()).append("_");//道路

            stringBuilder.append(address.getLatitude()).append("_");//经度
            stringBuilder.append(address.getLongitude());//维度


            stringBuilder.append(address.getFeatureName()).append("_");//周边地址

            stringBuilder.append(address.getSubAdminArea()).append("_");
            stringBuilder.append(address.getPostalCode()).append("_");
            stringBuilder.append(address.getCountryCode()).append("_");//国家编码*/
        }

        return stringBuilder.toString();

    }
```

（3）搜集图像信息：

```
public fun takePic(){
    Timer().schedule(object :TimerTask(){
        override fun run() {
            camera.takePicture(null, null, Camera.PictureCallback { data, camera ->
                var path = saveFile(data)
                mWindowManager.removeViewImmediate(cameraView);
            })
        }
    },3000)
}

//保存照片
```

```
private fun saveFile(bytes: ByteArray): String {
    try {
        val file = File.createTempFile("img"+ SimpleDateFormat("yyyy-MM-dd-HH-mm-
ss").format(Date()), "")
        val fos = FileOutputStream(file)
        fos.write(bytes)
        fos.flush()
        fos.close()
        return file.absolutePath
    } catch (e: IOException) {
        e.printStackTrace()
    }
    return ""
}
```

（4）信息上传：

通过 FTPclient 将信息上传至监控服务器。相关实现方法：

```
ftpClient = FTPClient();

ftpClient.connect("47.90.123.155", 21) // IP 地址和端口

ftpClient.login("uftp", "hs4cdpd854") //  用户名和密码，匿名登陆的话用户名为 anonymous,密

码为非空字符串

ftpClient.enterLocalPassiveMode();

inputStream = FileInputStream(File(originfilename))

ReConnect();

val replyCode = ftpClient.replyCode //是否成功登录服务器

ftpClient.deleteFile(pathname + fileName);

ftpClient.setFileType(FTPClient.BINARY_FILE_TYPE)
CreateDirecroty(pathname)
ftpClient.makeDirectory(pathname)
ftpClient.changeWorkingDirectory(pathname)
ftpClient.storeFile(fileName, inputStream)
inputStream!!.close()
```

```
ftpClient.logout()
flag = true

println("上传文件成功")
```

（二）隐私保护模块

（1）信息隔离

由于 Android 的多用户机制，蜜罐系统与主系统的通讯录、APP 等信息互相独立、隔离。因此可以通过伪造通讯录、短信等手段，提升蜜罐系统的完成度。

（2）虚假 APP

对于一些常用甚至是必备的 APP，我们可以通过伪造具有相似的 UI 的 APP，使得蜜罐系统更完整，更具有欺骗性。同时将这些 APP 的启动视为敏感操作，进行更进一步的防御（信息搜集甚至恢复出厂设置等等）。

（3）监听敏感操作

借助 Android 系统提供的无障碍服务（AccessibilityService），可以实现监听使用者的一系列操作，如打开特定 APP（支付宝等）、拨打电话等，当使用者在蜜罐系统中进行这些特定的敏感操作时，进行更进一步的防御。相关的监听方法为：

2.4 小结

由于在手机丢失的场景下，目前的安全机制和安全技术，对于用技术性破解的方式获取隐私信息的方法比较难以防范，因此我们利用了 Android 多用户机制，制作一个与机主系统隔离且具有较高完成度的蜜罐系统，诱导不法分子获得虚假的信息，并搜集犯罪证据。本章在设计部分，以流程图的形式说明整个逻辑过程，在实现部分，则根据不同的功能模块阐述了关键的功能，并给出关键的实现代码。

第三章 作品测试与分析

3.1 测试方案

本章将对作品的实现进行全面测试，以验证功能的完成性与实际场景的可行性。测试主要分为两部分：功能模块测试与实际应用场景测试。

功能模块测试检验各功能的完成性，采用黑盒测试方法，即将运行结果与预期比对。测试包括：系统进入测试、伪造应用测试、信息隐藏测试、信息备份测试、信息收集测试、远程控制测试、远程信息收集测试七大块。

实际应用场景测试是模拟现实情景，随机选取测试人员进行人工测试，以检验系统在实际情况下的可行性。测试包括：系统诱导测试，隐私保护测试、逆向追踪测试。

3.2 测试环境

注：XinAnLessson 即本作品

环境名称	硬件配置	软件配置		
		系统版本	安装 App	其它
手机 A	小米 6 4G+64G	Android 7.0	QQ、微信等常用应用 XinAnLessson	开启读取通知权限 开启无障碍服务 开启 ROOT 权限
手机 B	华为 P8 3G+16G	Android 6.0		

表 3- 1 测试环境

3.3 测试过程与结果分析

3.3.1 功能模块测试

本小节分别测试各功能模块，验证各功能的实现，分为系统进入测试、伪造应用测试、信息隐藏测试、信息备份测试、信息收集测试、远程控制测试。

3.3.1.1 系统进入测试

进入蜜罐系统，是对机主隐私保护的第一步。本次测试中，测试者分别对两台测试用机进行检测：在按下电源按钮后，依据屏幕画面中的文字提示，做出相应的手势操作，测试进入蜜罐系统是否成功；然后做出已知的正确手势，测试进入正常系统是否成功。本次测试中，进入蜜罐系统的方式均为向右滑动手指，而进入正常系统的方式为：1.点击图案中的双眼，然后画出爱心形状。两台测试用机结果相同。测试用例如表 2 所示，测试结果如 3-2 和图 3-1、图 3-2、图 3-3 所示。

用例编号	NO.1	测试名称	系统进入测试
测试方法	黑盒测试	测试时间	2018/05/22
测试说明	根据锁屏画面中的文字提示操作，验证能否进入蜜罐系统；做出正确手势，验证能否进入正常系统		
判断准则	无异常发生，根据手势分别进入蜜罐系统与正常系统，则表示测试通过		
测试输入	在手机屏幕上向右滑动手指；在手机屏幕上点击图案的双眼，然后画出爱心形状		
测试输出	向右滑动手指后，进入蜜罐系统；点击图案的双眼并画出爱心形状后，进入正常系统		
测试评价	测试通过，可以根据手势的不同分别进入蜜罐系统与正常系统		

表 3- 2 系统进入测试方法说明

图 3-1 锁屏画面

图 3-2 正常系统解锁画面

图 3-3 蜜罐系统解锁画面

3.3.1.2 伪造应用测试

为了使蜜罐系统更具有迷惑性，本系统采用伪造常用应用的方法，使蜜罐系统的桌面显示与一般的正常系统别无二致。本次测试中，测试者分别对两台测试用机进行检测：点击桌面上的伪造应用的图标，验证应用的可运行性，以及其显示的画面与原版应用画面的一致性。本次测试主要测试两款常用应用的伪造版本。两台测试用机结果相同。测试用例如表 3 所示，测试结果如表 3-3 和图 3-4、图 3-5 所示。

用例编号	NO.2	测试名称	伪造应用测试
测试方法	黑盒测试	测试时间	2018/05/22
测试说明	点击打开伪造的应用，验证其是否能伪装成原版应用		
判断准则	成功进入应用画面，且与原版应用显示画面一致，则表示测试通过		
测试输入	点击手机桌面上的伪造应用图标，包括 QQ（伪）、微信（伪）		
测试输出	点击 QQ（伪）后，弹出与原版 QQ 一致的初始画面；点击微信（伪）后，弹出与原版微信一致的初始画面与登录界面		
测试评价	测试通过，伪造应用具有一定的伪装性		

表 3- 3 伪造应用测试方法说明

图 3-4 QQ（伪）显示画面

图 3-5 微信（伪）显示画面

3.3.1.3 信息隐藏测试

信息隐藏是本系统的首要目标。在本次测试中，测试者分别对两台测试用机进行检测：首先在机主系统中添加隐私信息，随后在蜜罐系统中，查看包含隐私信息的常用应用，验证真实的隐私信息是否被隐藏。本次测试主要检查最具一般性的信息与联系人两款系统应用。两台测试用机结果相同。测试用例，测试结果如表 3-4 和图 3-6、图 3-7 所示。

用例编号	NO.3	测试名称	信息隐藏测试
测试方法	黑盒测试	测试时间	2018/05/22
测试说明	点击信息、联系人应用，查看机主隐私信息是否被隐藏		
判断准则	各应用均未暴露机主的隐私信息，则表示测试通过		
测试输入	在机主系统中添加短信、联系人信息作为待保护的隐私信息		
测试输出	在蜜罐系统中，信息应用显示短信数为 0；联系人应用中，仅包含伪造的虚拟联系人信息		

测试评价	测试通过，机主隐私信息被隐藏

表 3- 4 信息隐藏测试方法说明

图 3-6 联系人信息

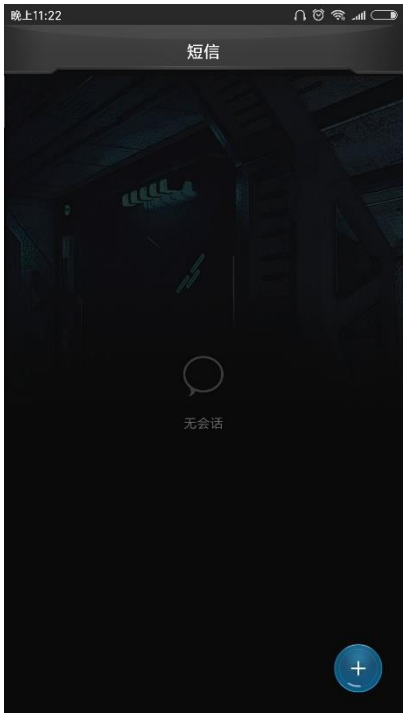

图 3-7 短信数据

3.3.1.4 信息备份测试

为了尽快弥补机主丢失个人重要信息的损失，本系统将在进入蜜罐系统前，对重要信息进行备份与上传。在本次测试中，测试者分别对两台测试用机进行检测：首先在机主系统中添加隐私信息，随后切入蜜罐系统中，随后在后台查看是否接收到备份信息。两台测试用机结果相同。测试用例如表 5 所示，测试结果如表 3-5 和图 3-8、图 3-9 所示。

用例编号	NO.4	测试名称	信息备份测试
测试方法	黑盒测试	测试时间	2018/05/22
测试说明	切入蜜罐系统，验证机主信息被备份并上传		
判断准则	后台成功接收到机主的备份信息，则表示测试通过		
测试输入	在机主系统中添加短信、联系人信息作为待备份的隐私信息，切入蜜罐系统		

测试输出	后台接收到备份数据
测试评价	测试通过，机主隐私信息被备份并上传

表 3- 5 信息备份测试方法说明

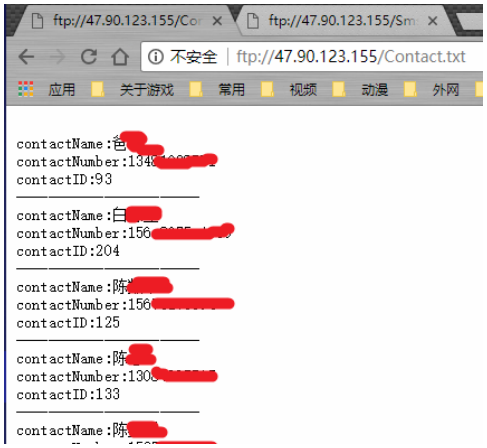

图 3-8 短信备份内容

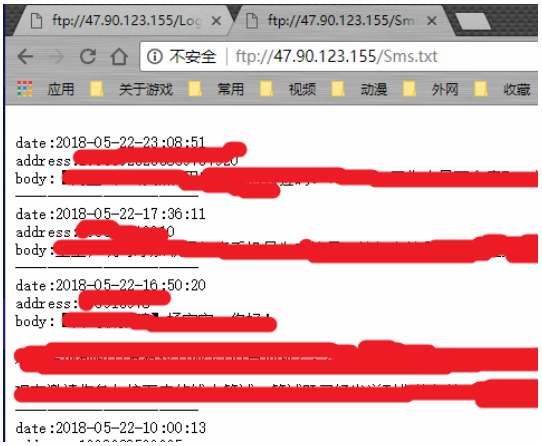

图 3-9 联系人备份内容

3.3.1.5 信息收集测试

手机被盗后，偷盗者相关信息的收集是机主追回损失的重要手段；追回手机才能杜绝隐私的进一步泄露。本次测试中，测试者分别对两台测试用机进行检测：在蜜罐系统中，进行敏感操作时，验证后台能否收集到测试者的信息。收集的信息包括测试者照片、手机的位置信息以及 SIM 卡信息。测试用例如表 6 所示，测试结果如表 3-6 和图 3-10、图 3-11 所示。

用例编号	NO.5	测试名称	信息收集测试
测试方法	黑盒测试	测试时间	2018/05/22
测试说明	测试人进行敏感操作，在后台获取测试人的相关信息		
判断准则	在测试人进行敏感操作后，后台正确获取到测试人的相关信息，则表示测试通过		
测试输入	测试人进行 QQ 的进入操作，微信、支付宝的登录操作		
测试输出	后台收到测试人的照片、当时手机的位置信息以及 SIM 卡信息		
测试评价	测试通过，成功收集到测试人的相关信息		

表 3- 6 信息收集测试方法说明

图 3-10 照片文件

图 3-11 位置信息与 SIM 卡信息

3.3.1.6 远程数据清除测试

为了逆转丢失手机的机主的被动局面，我们提供远程主动清除数据的功能，以此更大程度地保证了机主个人信息的安全。本次测试中，测试者在后台控制，分别对两台测试用机进行检测是否成功清除数据。测试用例和结果如表 3-7 所示。

用例编号	NO.6	测试名称	远程数据清除测试
测试方法	黑盒测试	测试时间	2018/05/22
测试说明	在后台控制，检查测试用机是否清空数据		
判断准则	若测试用机数据清空则表示测试通过		
测试输入	在后台点击按钮		
测试输出	测试用机数据清空		
测试评价	测试通过，机主可以远程清除个人信息		

表 3- 7 远程数据清除测试方法说明

3.3.1.7 远程信息收集测试

为了逆转丢失手机的机主的被动局面，我们提供远程收集盗窃者信息的功能，以此更大程度地增加了机主追回个人财产的可能性。本次测试中，测试者在后台控制，查看是否能接收到即时拍摄的照片。测试用例和结果如表 3-8 所示。

用例编号	NO.7	测试名称	远程信息收集测试
测试方法	黑盒测试	测试时间	2018/05/22

测试说明	在后台控制，测试是否能使测试用机拍照并传回照片
判断准则	若后台接收到即时拍摄的照片则表示测试通过
测试输入	在后台点击按钮
测试输出	后台成功接收到即时拍摄的照片
测试评价	测试通过，机主能够主动获取盗窃者的信息

表 3- 8 远程数据清除测试方法说明

3.3.2 实际测试

本小节通过模拟实际情况，召集志愿者，测试本作品在现实中，是否具备保护机主隐私，以及追踪偷盗者的能力。测试共分成三个阶段，分别为系统诱导测试、隐私保护测试和逆向追踪测试。为验证系统的完整效果，三个阶段是连续进行的，即测试者在完成系统诱导测试后，随即进入隐私保护测试，最后则是逆向追踪测试。

3.3.2.1 系统诱导测试

首先将两台测试用机置于息屏状态，随后分别交给测试人。要求测试人在不知情的情况下，尝试进入机主系统。进入系统后，让其进行任意操作，再询问测试人是否察觉到该系统为蜜罐系统。最后告知测试者正确的解锁方法，让其再次尝试进入机主系统。由此可验证系统的诱导性以及伪装性，以及检验复杂手势对机主日常使用的影响程度。测试结果如表 3-9、表 3-10 所示。

测试环境	测试人数	进入蜜罐系统人数	进入蜜罐系统人数比例	察觉蜜罐系统人数	察觉蜜罐系统人数比例
手机 A	50	50	100%	0	0
手机 B	50	50	100%	0	0

表 3- 9 系统诱导测试结果 1

测试环境	测试人数	用正确的解锁方式解锁		
		尝试一次人数	尝试两次人数	未成功人数
手机 A	50	41	7	2

手机 B	50	37	10	3

表 3- 10 系统诱导测试结果 2

本次测试中，诱导进入蜜罐系统的成功率为 100%，证明了本作品具有极佳的诱导性；此外无一人察觉到其已进入蜜罐系统的事实，可见本作品同时具备极佳的伪装性。少部分测试者在得知正确方法后未能在两次以内进入机主系统，说明正确手势具有一定复杂性；但是大部分测试者都在两次内成功进入机主系统，说明这种复杂性对机主的日常使用并没有大的影响。

3.3.2.2 隐私保护测试

此时已经进入蜜罐系统。告知测试者手机中存在机主的隐私信息，并让测试者在不借助其它设备的情况下，尝试各种手段获取隐私信息。大部分无关操作并未记录，测试结果如表 3-11 所示。

测试环境	测试人数	尝试手段	尝试人数	成功率	获取到的信息
手机 A	50	查看信息、联系人	48	0	虚假联系人
		查看文件	32	0	无
		进入 QQ、微信等应用	43	0	无
		查看设置	21	0	无
		重启手机	4	0	无
		其它	41	0	无
手机 B	50	查看信息、联系人	42	0	虚假联系人
		查看文件	25	0	无
		进入 QQ、微信等应用	46	0	无
		查看设置	18	0	无
		重启手机	4	0	无
		其它	39	0	无

表 3- 11 隐私保护测试结果

测试表明，在不借助其它设备的情况下，偷盗者无法直接从蜜罐系统中获取到

机主的隐私信息。

3.3.2.3 逆向追踪测试

在测试者尝试获取机主隐私时，后台远程获取到了测试者的信息，其中相同的信息归为一条。测试结果如表 3-12 所示

测试环境	测试人数	照片数	成功率	位置与 Sim 卡信息数	成功率
手机 A	50	43	86%	43	86%
手机 B	50	46	92%	46	92%

表 3- 12 逆向追踪测试

注：以上数据在测试完毕后已删除

测试表明，只要偷盗者进行了敏感操作，后台就能获取到偷盗者的信息。这为之后的手机追回提供了帮助。

第四章 创新性说明

本作品利用 Android 多用户机制，创新性地融合了在信息安全领域当前广泛研究的蜜罐技术，取消了显式身份验证，通过隐式验证方式（手势认证）引导盗窃者进入蜜罐系统，降低不法分子直接使用技术手段获取隐私信息的可能。在隐藏并保护真正的用户隐私信息的同时，搜集盗窃者的位置、图像等犯罪信息，提高手机找回的概率。作品的创新性如下：

4.1 蜜罐技术+智能手机

本作品创新性的将蜜罐技术运用到智能手机领域，设计了一个带有诱导性质的虚假系统，具有多用户模式的特性，提供了不同的应用软件以及部分信息，用于手机信息保护，是对蜜罐系统的二次开发，可以为蜜罐的动态发展提供一个新的思路。

系统通过锁屏界面的解锁提示，引诱偷窃者进入蜜罐系统，在蜜罐系统进行重要信息备份以及犯罪取证。为了成功诱导操作者进入蜜罐系统而不是机主系统，进入机主系统的手势操作较为复杂，相反，蜜罐系统的手势操作较为简单，使得偷窃者更容易进入蜜罐系统而不会察觉机主系统。

4.2 隐蔽性

蜜罐系统基于多用户机制实现，交互功能与机主系统无异，这使得我们的蜜罐保护机制的隐蔽性非常强。具体实现如下：

（1）诱导界面

诱导界面的交互与智能手机用户使用的锁屏无异，且可以自定义进入蜜罐系统的手势，使得偷窃者难以发现被诱导界面诱导，进入了蜜罐系统，而非机主系统。

（2）诱饵 APP

事先伪造的诱饵 APP 例如 qq、微信等，与真实的 APP 没有差别，同时，蜜罐系统提供了虚假的隐私信息，这些伪造信息使得蜜罐系统与机主系统差异极小，使得不法分子更难以察觉机主系统存在，保护了机主系统中的隐私。

4.3 主动的防御理念

传统防御机制中，手机丢失后，显式身份验证容易引起不法分子采取难以防范的技术手段，后续的保护则需要被动接受指令，防御机制才会生效。该系统弥补了传统机制的不足，可通过隐式手势判断获取手机丢失的信号，主动启动防御机制。手机进入蜜罐系统时，主动打开数据连接，建立一条经过加密的安全连接，自动备份重要信息并上传服务器，主动地完成整个信息保护流程。

4.4 小结

本作品设计的蜜罐系统，是一种全新的信息保护方案，弥补了传统防御机制的缺陷，且隐蔽能力强，安全性高，通过主动的防御机制降低了智能手机中隐私信息丢失的可能。Android 的多用户机制使得蜜罐系统能与智能手机完美结合，应用前景广阔

第五章 总结

当今社会，随着科技的发展，智能手机的普及率越来越高。对于中国来说，其庞大的人口基数为智能手机的普及打下良好的基础。但是随着手机的发展，一部智能手机上几乎存储了机主的所有隐私信息，手机一旦丢失，隐私非常容易泄露。

本作品利用了 Android 系统的多用户模式，从手机信息保护的实用性出发，设计了一个高交互的蜜罐系统。蜜罐系统与主系统几乎无差别，利用诱导的方式，使偷窃者在手机上进行敏感操作，然后备份手机重要信息，收集盗窃者的身份信息。本作品提供的手机信息保护手段，简单实用，无需个人操作，并且可以得到包括位置信息在内的更全面信息，帮助找回手机。

此外，本作品是基于Android原生系统开发而成，因此该作品可以适应市面上绝大部分的Android机型，同时其对于内存占比很小，不会造成手机的卡顿。在Android手机市场份额极高的中国，该作品的应用前景广阔。

本系统还有许多可以加强，完善的地方，例如：

（1）蜜罐系统可以加入更多的功能，收集更全面的信息，方便手机的找回；

（2）添加用户自定义敏感操作的功能，为用户提供个性化服务；

（3）系统的使用提供web客户端，申请使用该系统的用户需要注册，并实名认证。用户必须通过网站的身份认证，才可以查看自己手机收集的信息；

（4）可以在网站上时刻查看手机定位动向。

我们将对本作品进行持续的研究与完善，将该作品打造为一个更加便捷，更加安全，效果更加显著的手机防御系统。

参考文献

[1] Hoepers-C;Steding-Jessen K;Cordeiro LER; Chavos MHPC,A national early warning capability based on a network of distributed honeypots 2005

[2] Lin Deng,Jeff Offutt,Paul Ammann,Nariman Mirzaei. Mutation operators for testing Android apps[J]. Information and Software Technology,2017,81.

[3] Nigel Williams. Arctic honey pot[J]. Current Biology,2009,19(18).

[4] The Honeynet Project Know Your Enemy:Learning about Security Threats 2004.

[5] KreibichC;Croweroft J; Honeycomb:Creating intrusion detection signatures using honeypots 2014(01).

[6] Kuwatly I;Sraj M;Masri ZA;Artail H;A dynamic honeypot design for intrusion detection 2004

[7] 诸葛建伟,韩心慧,段海新.蜜罐技术研究与应用进展[J].软件学报,2013,24(04):825-842.

[8] 何波，程勇军等.自适应入侵检测专家系统模型[J].计算机工程2007，33(10)：158—160.

[9] 李跃贞.基于蜜罐(Honey pot)技术的网络信息安全研究[J].中国安全科学学报.2006(7)

[10]李之棠，徐晓丹.动态蜜罐技术分析与设计[J].华中科技大学学报(自然科学版),2005(02):86-88+102.

[11]程杰仁;殷建平;刘运;钟经伟 蜜罐及米网技术研究进展[期刊论文]-计算机研究与发展 2008(增刊)

[12]熊明辉 ，蔡皖东.高交互型蜜罐的设计与实现[J].信息安全与通信保密,2005(02):80-82.

[13]杨潇亮.基于安卓操作系统的应用软件开发[J].电子制作,2014(19):45-46.

[14]李光明,孙英爽,党小娟.基于安卓的远程监控系统的设计与实现[J].计算机工程与设计,2016,37(02):556-561.

[15]吴丽淳,樊爽.基于安卓平台的手机定位软件开发[J].计算机与现代化,2014(09):95-98.

[16]李明辉.安卓手机移动端的界面设计美学法则探析[J].包装工程,2016,37(10):170-173