软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, [doi: 10.13328/j.cnki.jos.000000]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于分布式信息流控制的无障碍服务安全加固*

李晓娟, 陈海波

(上海交通大学 并行与分布式系统研究所,上海 上海 200240)

通讯作者: 陈海波, E-mail: haibochen@sjtu.edu.cn

摘 要: 随着安卓系统的广泛使用，系统提供的功能也越来越多，其中一个重要特性是 1.6 版本中引入并在 4.0 及以上版本中优化的无障碍辅助性服务。通过无障碍辅助功能，应用不仅可以获得输入框输入文本等窗口元素信息，还可以与应用窗口自动地进行双向交互（如获得按钮信息点击按钮）。然而，这些特性一旦被滥用将会给用户带来巨大的安全威胁。本文对安卓系统中的无障碍辅助性服务进行深入研究，分析其可能被滥用的途径，并找出安全缺陷及产生原因。然后提出了基于分布式信息流的控制机制标记并跟踪无障碍辅助性服务和无障碍事件以进行安全加固。我们实现了一个名为的 Tassel 安全系统以防止无障碍辅助性服务滥用。经过测试，该系统可以在不影响系统其他功能正常使用的前提下，保证服务的使用安全，且系统整体的性能影响很小。

关键词: 安卓系统; 无障碍辅助性服务; 系统漏洞与安全; 分布式信息流控制
中图法分类号: TP311

中文引用格式: 李晓娟,陈海波.基于分布式信息流控制的无障碍服务安全加固.软件学报. http://www.jos.org.cn/1000-9825/0000.htm

英文引用格式: Xiaojuan Li, Haibo Chen. The Security Reinforcement of the Accessibility Service Based on DIFC. Ruan Jian Xue Bao/Journal of Software, 2016 (in Chinese). http://www.jos.org.cn/1000-9825/0000.htm

The Security Reinforcement of the Accessibility Service Based on DIFC

XIAOJUAN Li, HAIBO Chen

(Institute of Parallel and Distributed Systems, Shanghai Jiao Tong University, Shanghai 200240, China)

Abstract: With the ubiquity of Android system, the Android offers more and more functions, one important feature is the Accessibility service, which is introduced in Android 1.6 and optimized in Android 4.0 and above. With the Accessibility service, applications can retrieve active window information, such as the text contents users input, and can communicate with applications' window automatically, such as getting the button content and then clicking it and etc. However, these advanced features once are abused, will bring mobile users considerable safety threats. Our work does research into this system service, analyzing its possible ways being abused and its cause from system API design. Then we propose and implement a new system called Tassel, based on the decentralized information flow control mechanism, to make this service API usage securer. In our system, we tag, track and control the accessibility service handling as well as the accessibility event. In our evaluation, we proof that our system can prevent accessibility service from being abused, meanwhile, it does not impact the system normal functions as well as the performance.

Key words: Android system; accessibility service; system vulnerability and security; decentralized information flow control

收稿时间: 0000-00-00; 修改时间: 0000-00-00; 采用时间: 0000-00-00; jos 在线出版时间: 0000-00-00
CNKI 在线出版时间: 0000-00-00

安卓系统的普及使其提供更加丰富的功能以满足越来越多的用户需求。无障碍辅助性服务(Accessibility Service) [1,2] 在安卓版本 1.6 中提出并在安卓 4.0 版本及以后进行了优化。基于该特性，第三方应用可以获得活动窗口信息，例如获得输入框用户输入内容；通过该服务操作窗口元素，比如模拟点击按钮等。另外，应用开发者也可注册自己的无障碍辅助性服务，为用户提供更多便利，如为视觉障碍用户提供语音提示等。例如 Zhong 等人 [4] 使用无障碍辅助性服务开发了 Touch Guard 来增强安卓智能机触摸区域的管理并提供了更多的补充特性，来解决残障人士操作设备时人手抖动造成的点击错误。

由于安卓系统中该服务 API 设计的一些缺陷， 恶意开发者滥用该 API 可以执行恶意行为。例如，静默安装恶意软件或后门、窃取用户密码、劫持攻击登陆界面以及绕过安卓安全保护等[3,8,9,11,14,15]。2014 年的 A11y[5]工作对 Window、Linux、iOS6 和 Android4.4 四大系统中的辅助性服务进行研究。该工作发现这些系统中的辅助性服务都存在一定的漏洞，可被用来绕过系统中固有的安全机制，进行信息窃取等。但该工作仅针对系统中自带的辅助性服务进行研究，不包括 4.0 版本以上的可被开发者操作的辅助性服务。Kraunelis J[6] 等对安卓系统中无障碍辅助性服务框架进行了学习研究，发掘其潜在的安全隐患。

然而，针对安卓系统中无障碍辅助性服务的滥用问题，目前仍没有明确的解决方案。当前，仅有一些粗粒度的使用建议。例如使用安卓系统时，当有些软件请求打开无障碍辅助性服务时，用户需要多注意这些服务的使用用途； 建议用户尽量不要在第三方市场下载安装软件等。然而，这些建议并不能有效地为用户提供安全保障，并且用户需要在功能性与安全性之间进行取舍。

本工作主要针对该安全问题，提出了基于分布式信息流控制机制（DIFC）的服务加固策略并实现了安全系统。我们通过分析安卓系统中无障碍辅助性服务的调用及处理无障碍事件的过程，分别在服务注册及无障碍事件产生时添加标记，然后在调用服务处理无障碍事件时进行标记检查，若匹配控制规则，则处理返回；反之，不予处理。经过测试，本系统中无障碍辅助性服务正常工作，且由控制策略带来的性能开销很小，相比原系统处理延迟不超过 20ms。

本文贡献点是：

(1) 全面地分析研究安卓系统中优化的无障碍辅助性服务的安全性；

(2) 基于分布式信息流控制机制来设计安全的系统无障碍辅助性服务；

(3) 通过实验证明了本文提出的系统方法可以有效地解决无障碍辅助性服务 API 被滥用的情况。

本文第 1 节介绍安卓系统中的无障碍辅助性服务及其安全性问题.第 2 节介绍基于分布式信息流控制机制的无障碍辅助性服务加固系统的设计与实现. 第 3 节是对本工作实现系统的测试.第 4 节介绍相关工作. 最后总结全文,并对未来的工作进行展望。

1 安卓无障碍辅助性服务及其安全性问题

无障碍辅助是安卓应用的一个重要部分。据调查，世界上大约 15%[7]的人口或多或少的有一些残障。对于这些用户来说，灵活地使用智能手机存在一定的问题。而无障碍辅助性服务正是为了给这类用户提供交流、学习、生活上的便利。考虑到这些用户在视力、听力及肢体等方面的限制，其阅读、听声及操作应用等会存在一定困难，安卓系统开发者设计无障碍辅助性服务来放大字体、将文字转化为语音及进行自动点击等，从而很好地帮助用户便捷地使用安卓智能设备。根据官方开发文档[10]，该服务是在不修改系统代码的基础上设计的一个系统服务。通过增加类似 contentdescription 的属性，开发者可以通过该服务获得活动视图的内容，并进行自动点击，语音或触摸的提示等等。

1.1 无障碍辅助性服务的开发使用

无障碍辅助性服务是一个系统服务，生命周期不由应用本身管理，而是由系统和用户的显式操作控制。安卓系统提供了该服务的开发 API，应用开发者在编写应用时可以直接调用该 API 配置自己的无障碍辅助性服务。使用步骤如下：

1) 在应用的配置文件中声明该服务权限：android.permission.BIND_ACCESSIBILITY_SERVICE。这

样保证了系统对该服务的绑定，并进一步进行管理调用，处理事件；

2) 为应用配置文件中的无障碍辅助性服务设置 action:android.accessibilityservice.AccessibilityService 过滤器，保证当无障碍事件发生时，该服务被触发；

3) 在应用中给该服务进行配置，编写对应的 XML 文件，声明该服务接收的事件类型，反馈类型，监听包名等内容。

若开发者调用该服务实现一些功能时，优先考虑是否方便残障人士操作设备，例如为方便视弱者操作设备，将文字信息转换为语音，则是对无障碍辅助性服务的合理利用行为。除此，目前一些应用开发者利用该无障碍辅助性服务获得视图内容及自动点击的特性进行免 root 智能安装，这一功能可为任何用户提供便利。一般的，设备安装应用时需要用户点击“下载”，“安装”等按钮，并赋予一些权限以保证应用正常运行。而在 Root 后的设备中，应用商店类应用可以通过执行一些需要高权限的命令，例如“pm install” 来自动安装应用。但随着系统补丁的增加，应用通过系统漏洞获得 Root 权限越来越难。无障碍辅助性服务使得开发者可以利用其特性实现自动安装，捕获系统中视图变化事件，触发无障碍辅助性服务获得视图中请求安装的按钮，执行模拟点击，完成安装过程。该过程不需要用户参与交互。除了执行自动安装，该服务还可以用来执行自动更新。无障碍辅助性服务的自动安装功能省去了用户一步步点击的繁琐操作，为用户带来了方便。例如，豌豆荚应用，该应用进行智能安装的代码片段如下表 1.

Table 1 Code snippet for using accessibility service to install automatically

表 1 利用无障碍辅助性服务执行自动安装片段

```
1  private static boolean b(Context arg3) {
2      try {
3          v1_1 = Settings$Secure.getInt(arg3.getContentResolver(), "accessibility_enabled"); //判断服务开启
4      }catch(Settings$SettingNotFoundException v1) {
5          v1_1 = 0;}
6  public static void  a(ApiContext arg7) {
7      HashMap v2 = new HashMap();
8      ((Map)v2).put("deviceModel", AccessibilityManager.a(Build. MODEL));
9      ((Map)v2).put("deviceBrand", AccessibilityManager.a(Build. BRAND));
10     d v0 = new d("http://api.wandoujia.com/v3/autoinstall", (Map)v2), arg7, new c(), new
11             com.wandoujia.accessibility.d(),  new e());
12      v0.a("Accept"," application/json");
13      arg7.getRequestQueue().a((Request)v0);
14      …
15      AccessibilityNodeInfo v0 = arg4.getSource();
16      if(v0 != null && v0.getPackageName() != null) {
17          v0 = arg5.getRootInActiveWindow(); //获得视图
18          if(v0 != null && !TextUtils.isEmpty(a.b(v0))) {
19              Iterator v1 = v0.
20 findAccessibilityNodeInfosByViewId("com.xiaomi.gamecenter:id/replace_btn_install_oldpkg").iterator();
21              while(v1.hasNext()) {
22                  a.a(v1.next(), TaskEvent$Action.AUTO_INSTALL_CLICK); //若视图包含安装按钮则自动点击
23  }}}
```

应用先判断无障碍辅助性服务是否开启，若没有则会引导用户开启。然后发出一个自动安装的请求，该请求被包装为一个无障碍事件并由视图发出，事件被系统中符合类型的无障碍辅助性服务处理。无障碍辅助性服务被触发并获取视图中的按钮信息，若按钮是执行安装相关的则模拟点击允许请求最终完成自动安装。这一过程除了引导用户开启无障碍辅助性服务外再无其他与用户的交互，减少了用户的频繁点击操作，提升了用户的用机体验。

1.2 无障碍辅助性服务的滥用

除了一些对无障碍辅助性服务的合理利用，近年来，应用开发者对该服务的滥用案例越来越多。根据

[3]，去年发现的利用无障碍辅助性服务的样本数量相比前几年呈现快速增长状态。无障碍辅助性服务的特殊性可以被用来完成一些需要高权限才能执行的操作，对安卓设备使用者造成巨大的安全威胁。该服务被滥用来进行自动抢红包[16]，静默安装恶意软件[17,18]，窃取用户隐私数据[3]，防止软件被卸载[3]等等。下面介绍这些滥用案例及操作过程。

a. 抢红包。首先，开发者配置应用中的无障碍辅助性服务使其对指定应用包例如微信进行监听，并设置该服务监听的事件类型为窗口变化事件和通知栏改变事件。在微信应用运行过程中，当应用界面发生变化或通知栏变化通知抢红包时，视图发出无障碍事件，这类事件被该无障碍辅助性服务截取。无障碍辅助性服务从窗口视图中遍历窗口树节点找出“拆红包”或“领取红包”按钮（不同的微信版本按钮的 ID 不一样），然后执行模拟点击，允许事件请求，实现抢红包。该服务继续进行监听，自动处理下一个事件。这是无障碍辅助性服务被利用的灰色地带，已经背离了安卓官方设计该无障碍辅助性服务的初衷。实际上，自动抢红包功能带有外挂特性，会造成某种程度的不公平。

b. 静默安装恶意软件。最开始开发者利用无障碍辅助性服务可以获得窗口信息元素的特性，获取视图中“安装”按钮信息并进行模拟点击完成安装请求，其初衷是为了减少用户的点击操作，节省用户时间。但由于该过程不需要其他额外的权限，恶意开发者利用这一特性获取安装应用的相关按钮，执行模拟点击以通过任意安装请求。这样使得在用户不知情的情况下，恶意软件开发者安装用户并不想要的应用甚至恶意应用。由于该过程中没有安装进度提示及与用户的交互操作，所以用户并不能发现该恶意安装过程并及时阻止。恶意应用一旦安装在用户设备上，将会给用户隐私安全及财产安全造成威胁。

c. 窃取用户隐私数据。恶意应用开发者利用无障碍辅助性服务可以获得当前活动窗口的内容的特性窃取用户隐私数据。当一个用户打开银行客户端，应用会显示一个登陆界面。若一个无障碍辅助性服务是对系统中的视图变化事件进行监听时，该无障碍辅助性服务被触发。若恶意开发者的实现逻辑是捕获视图中的文本信息，则当用户输入用户名和密码时，应用通过该服务获得文本框内容然后通过网络传输出去，或者直接利用“文本转语音”服务将文本内容转换为语音朗读出来，这样造成用户隐私信息的泄露，给用户带来巨大损失。

d. 软件拒卸载。恶意软件开发者可以利用无障碍辅助性服务进行软件防卸载。恶意开发者注册一个无障碍辅助性服务，该服务监听系统中的窗口变化事件。当用户想卸载一个应用时弹出卸载界面，这一界面变化会触发处于监听状态的无障碍辅助性服务并进行响应处理，在用户点击“卸载”按钮前，自动跳转到系统设置界面或者系统主界面，从而阻止用户正常卸载应用。

1.3 无障碍辅助性服务安全性分析

无障碍辅助性服务给用户提供便利的同时，确实存在着一定的威胁。该服务之所以被滥用，是因为系统在设计该服务时存在一定的缺陷。下面针对该服务的安全性进行分析。

通过阅读资料以及安卓源码，我们总结了以下无障碍辅助性服务被滥用的原因：

1) API 自身设计的缺陷，服务权限的粗粒度管理。安卓系统给开发者提供了一些服务接口，一旦用户开启了该服务。该服务的配置文件开始发挥作用，直接赋予该服务获取活动窗口视图的权限。这样服务在运行时，即使在有运行时权限检查机制的 6.0 及以上的系统版本，这些权限也无法被修改。另外，该服务在开启后还被自动授予访问语音引擎的权限。当设置了服务处理事件的返回类型为语音时，无障碍辅助性服务就可以将视图的文本信息转换为语音输出。
2) 安卓系统的安全机制并没有随着无障碍辅助性服务的优化而增强。在安卓系统中无障碍辅助性服务相关代码中，我们发现无障碍辅助性服务的管理服务在决定是否分发事件时只是判断了事件类型，看其是否是无障碍事件，而没有检查事件是否与无障碍辅助性服务属于一个应用或者该事件是否来自恶意应用。这样会导致任何满足指定类型的事件可以被无障碍辅助性服务处理并获得处理结果。
3) 安卓系统开放的 API 说明文档和源代码实现不匹配。这使得开发者在调用该 API 实现功能的时候

并不能完全按照源代码中的安全策略进行，最终导致该服务被滥用。我们发现在无障碍辅助性服务的开发文档中，分发无障碍事件时判断的是应用包名。而在源代码中则判断事件类型。这样，攻击者可以直接在应用代码中设置监听某个包名，当应用包产生事件时，系统只判断事件类型进行无障碍辅助性服务处理，最终恶意应用对监听的应用进行了操作并获得返回结果。

4) 开发者的恶意使用。无障碍辅助性服务设计的初衷是帮助残障人士更好地操作设备。但是恶意开发者会忽视该正当目标，转而利用其提供的一些特性去执行恶意操作。例如静默安装、窃取活动视图界面内容等。

由于越来越多的应用开发者开始利用无障碍辅助性服务实现各种便捷功能，一旦该服务被滥用，用户隐私信息受到威胁；同时当前的很多用户使用移动应用管理财产，运行在安卓系统上的这些移动应用的安全得不到保证，用户的财产安全同样地受到威胁。

2 无障碍辅助性服务的安全加固

本节主要介绍基于分布式控制流机制的无障碍辅助性服务加固系统设计与实现。

2.1 系统设计

2.1.1 威胁模型

在介绍本系统的设计之前，我们先给出威胁模型。在本安全加固系统设计中：

1）无障碍辅助性服务仅针对单视图下产生的事件进行处理，不考虑分屏，多屏共享情况。

我们知道安卓 7.0 版本及以上设计了分屏多任务功能。该特性允许系统中多个活动共用一个屏幕。屏幕共享的情况下，事件的模拟处理会相对复杂。且通过阅读 7.0 版本的安卓系统代码，我们发现开发者还没处理好分屏情况下的一些无障碍辅助性服务事务。例如无障碍辅助性服务对共享屏幕下多个区域事件的模拟处理。

2）应用程序不能对系统造成破坏。

我们假设应用不能对系统造成破坏。这样可以保证系统功能正常，攻击者无法修改我们系统中新增添的 Tassel 服务。由于我们系统中新增的 Tassel 服务主要负责无障碍辅助性服务及事件的标记、检查与处理，该服务不被修改确保本工作设计系统的有效性。

3）不考虑应用间的攻击：一个应用不可攻击破坏其他应用的活动，服务属性等。

这样保证了应用注册的无障碍辅助性服务不会被其他应用修改配置来监听不当事件执行恶意操作。

4）非 root。

移动设备是非 root 的；应用自身无法获得 root 权限。

2.1.2 系统设计中的挑战

DIFC 策略大多应用在 Linux 系统上，很少有工作实现在安卓系统上。本文设计的系统目标是确保安卓中无障碍系统服务的安全使用。考虑到安卓系统的一些特性，我们需要解决的挑战有以下几点：

1）安卓系统中的无障碍辅助性服务是一个系统服务。在默认情况下该服务是全局的，即当一个无障碍事件产生时，系统先寻找可以处理该类事件的第一个注册的无障碍辅助性服务，并交由其处理。这样会使得待处理的事件和处理事件的服务可能来自于不同的应用，进而导致应用间信息泄漏。在我们设计的系统中需要阻止这样的操作，确保只有同一个应用或关系相近的应用才可互相调用无障碍辅助性服务处理无障碍事件；

2）安卓系统是一个多任务系统。多个应用的活动或服务共同完成一个任务。这就使得某个应用注册的无障碍辅助性服务可以被多个应用共用，且视图中的无障碍事件可能被多个应用的服务进行处理并获得返回结果。为了保证安全，这种情况下的共用应该受到限制，否则一些处理事件的返回值会导致信息泄漏。本文设计的系统需要控制哪些任务中的无障碍辅助性服务及事件可以被共享并返回处理结果。

3）新版本的安卓系统为应用开发者提供了可调用的无障碍辅助性服务 API，且该服务可被动态配置。

这意味着在应用运行过程中，无障碍辅助性服务的相关信息会发生变化。在我们的系统中如何对这一变化进行处理是一个挑战。

以上三个挑战最终可以总结归为两个主要问题：应用内部无障碍辅助性服务对无障碍事件的处理过程和应用之间无障碍辅助性服务对事件的处理过程。

2.1.3 解决挑战的方法

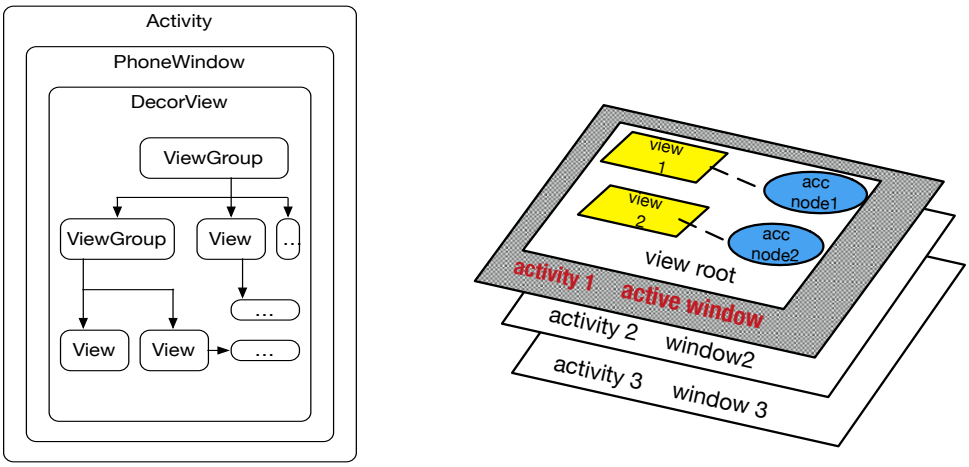
我们的系统为了解决以上主要问题采用如下方法：

1） 针对应用自身的无障碍事件的处理过程：我们的系统借鉴分布式控制流机制，对应用在系统中注册的无障碍辅助性服务及活动视图中产生的无障碍事件进行标记。然后制定安全策略控制事件的流向，以确保本应用产生的无障碍事件只能被本应用或有关联关系的应用的无障碍辅助性服务处理，本应用处理无障碍事件的结果不会被其他应用获得，且本应用的无障碍辅助性服务不能处理其他无关应用的无障碍事件；

2） 针对应用间的无障碍事件的处理过程：在一些情况下，应用之间是有“合作”关系的。即多个不同的应用分别提供活动、服务等组件完成一个任务。在这种场景下，活动视图中的无障碍事件也会被“合作”方的无障碍辅助性服务应用处理。为了解决应用间的无障碍辅助性服务及事件共享问题，同时确保应用中无障碍辅助性服务的正常功能，我们的系统为应用添加“亲密性”属性。该属性决定一个应用的无障碍辅助性服务是否有权处理来自另一个应用的无障碍事件以及一个应用是否可以使用另一个应用的无障碍辅助性服务处理本应用的事件。

2.1.4 系统设计

为了更好的阐述本系统的设计，先来介绍下与无障碍辅助性服务相关的视图部分。我们知道无障碍辅助性服务是作用于应用视图的，为了很好地说明该服务的工作原理，这里介绍下安卓系统中应用活动、窗口和视图的关系。通过查找资料[13] 和阅读源代码，我们得到了它们之间的关系，如图 1 所示.



(a) Activity 与 View 、ViewGroup　　(b) Activity 与 Window、AccessibilityNode

Fig.1 The relationships among Activity, Window, View and Accessibility elements

图 1 活动、窗口视图及无障碍辅助性服务元素的关系

图 1(a)展示了应用活动与视图，视图组之间的关系。当用户点击应用的某个按钮时，一个活动被启动。同时系统会创建一个 PhoneWindow 窗口对象，该对象接着创建一个 DecorView 视图。由于一个应用活动可能包含多个不同的视图，系统在 DecorView 对象中声明 ViewGroup 视图组来管理应用运行过程中产生的多个视图。图 1(b)表示了活动窗口与视图及无障碍辅助性服务节点的关系。当前处于前端的窗口来自于是活动 1。该活动在运行期间生成两个视图，这两个视图受根对象的管理。每个视图对应一个 AccessibilityNode 对

象。在无障碍辅助性服务运行期间，通过服务获得的窗口元素及进行模拟点击等操作的对象实质是一个 View 对象。当活动窗口发生变化时，系统将这一变化转换为一个事件。若活动窗口设置“importantForAccessibility”属性为真，代表当前视图产生的事件允许被无障碍辅助性服务处理。于是该事件被分发给无障碍辅助性服务，最终由系统的无障碍辅助性服务管理器根据该事件的类型选择恰当的无障碍辅助性服务进行处理，并返回处理结果。最后系统根据这些返回值向用户反馈。

另外，我们知道正常情况系统安装一个应用时，系统的包管理服务起主要作用。包管理服务对应用压缩包进行解析，获得其中的四大组件包括服务组件及声明的权限信息。应用在运行时向用户询问是否开启无障碍辅助性服务。若用户开启，则系统根据应用中的无障碍辅助性服务配置文件设置该服务的一些属性：能否获得窗口内容，监听哪个应用包以及处理返回的结果类型。

以上是一般情况下该服务的有关工作流程。为了使应用开发者安全地使用该服务，我们对无障碍辅助性服务使用过程中涉及的系统服务及组件分别进行加固。根据之前对无障碍辅助性服务的安全性分析，该无障碍辅助性服务 API 被滥用的根本原因是系统对服务权限的粗粒度管理。一种直接的加固方法是当该服务开启时，禁止相关权限。但这样会使功能受到限制，与该服务的设计初衷相违背。另一种是用户完全禁止无障碍辅助性服务。这样的话，系统提供的便利用户的辅助性服务将会失效，服务失去意义。

既然不能全面禁止，我们需要灵活地进行权限管理。为了达到此目标，我们需要对这一过程中产生的事件及调用来处理事件的服务进行追踪并检查。由此，我们引入分布式控制流机制进行保护。设计的加固系统概要如下图 2 所示。

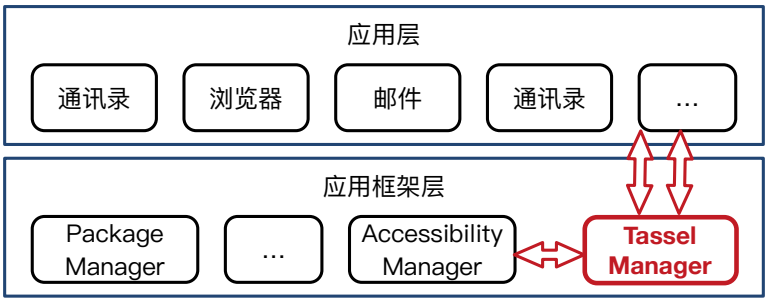


Fig.2 The design of Tassel system enhancing Android Accessibility service

图 2 Tassel 系统概要

我们的系统添加了 TasselManager 系统服务，该服务控制管理系统中的无障碍辅助性服务。在我们的系统中，任何一个应用可以注册无障碍辅助性服务，但一个应用的无障碍辅助性服务只能处理该应用或与之有关联关系的应用产生的无障碍事件。这类事件的分发及处理受到 TasselManager 服务的策略约束。一个应用 A 无法通过无障碍辅助性服务获得另一个应用 B 的事件处理结果及活动视图信息；同样的，应用 B 也无法调用应用 A 的无障碍辅助性服务处理事件。

在我们的 Tassel 系统中，策略管理主要包括以下两个部分：

a. 引用监视器 (Reference Monitor)。为了保证以上操作，我们的系统对应用注册的无障碍辅助性服务及在运行过程中产生的无障碍事件分别设置标记并进行信息流跟踪。本系统考虑到无障碍辅助性服务和事件的属性差异，对它们设计不同的标记：对于无障碍辅助性服务，本系统采用 UID 和包名组合。在安卓系统中，不同的应用可能设置”sharedUserId”属性导致共享 UID，因此我们为标记添加了包名信息；对于无障碍事件，本系统采用 UID，包名和 PID 的组合。为了匹配处理事件与无障碍辅助性服务，本系统没有直接使用 PID 标记。考虑到有的应用可能会伪造应用采用同样的包名进行“占坑”，故在包名的基础上增加了 UID 值。而应用间的“sharedUserId”属性又会导致 UID 相同，所以对事件的标记保留了 PID。另外，考虑到 6.0 版本的安卓系统中运行时权限机制，我们的系统将

一些在无障碍辅助性服务中自动授予的权限加入到应用的动态权限列表。这样，当应用通过该服务获得某些资源时需要动态申请权限，使得对资源使用在用户可控范围内。

b. 应用“亲密性” (Application Affinity)检查。应用运行过程中，来自不同应用的组件会共同完成一件任务。考虑到这种情况，本系统为了保证这些应用在能够正常完成任务的基础上，防止无障碍辅助性服务被滥用，引入了应用间的“亲密性”属性。具有该属性的两个及以上的应用可以互相调用无障碍辅助性服务对应用中产生的无障碍事件进行处理。在应用安装时，系统解析应用包提取其中的服务并写入一个全局变量中。在该解析过程中，Tassel 服务对应用要注册的无障碍辅助性服务进行标记：获取应用包名，”sharedUserId” 属性及活动组件中的“taskaffinity”属性。这些属性将共同决定应用之间的“亲密性”。若两个应用 A, B 是“亲密”的，则应用 A 产生的无障碍事件可以被应用 B 注册的无障碍辅助性服务处理；反之，同理。若应用间无“亲密”关系，则在事件处理时即使用户选择了优先注册的无障碍辅助性服务，该服务也无法处理当前活动窗口产生的无障碍事件并获得处理结果。

除了判断应用间的“亲密性”， 我们还分析应用注册的无障碍辅助性服务的属性：若一个应用的无障碍辅助性服务设置监听的包名为空，代表着该服务可以对系统中任何应用产生的指定类型的无障碍事件进行处理。为了防止滥用，我们的系统规定：当无障碍辅助性服务监听包为空时，该服务只能处理本应用及有“亲密性”关系的应用产生的无障碍事件。

在以上策略的控制下，系统在分发无障碍事件时，会首先判断产生事件的应用包名和 UserId；在将事件传递给无障碍辅助性服务进行处理时，则对该事件的标记和即将被调用来处理该事件的无障碍辅助性服务的标记进行检查，并做以下判断：事件与服务是否属于同一个应用包；产生事件的应用与处理该事件的无障碍辅助性服务所属的应用是否具有“亲密“关系。当事件与服务满足以上任意一个条件时，该无障碍事件可以被服务处理，并返回结果。

图 3 中展示了本系统中两个应用运行时的对象标记及信息流情况。在一个应用 APP1 被安装时，系统的包管理服务解析该应用中的无障碍辅助性服务，并根据应用包名及分配的 UID 信息对应用中的无障碍辅助性服务设置标记 src1{t_1, t_2}。当 APP1 的一个活动产生一个无障碍事件时，系统对该活动中的事件进行标记 Activity1{t_1, t_2, t_3}。当第二个应用被安装时系统会对其进行同样的处理，但标记为 src2{t_4}。通过对应用的属性进行解析，这两个应用并不具有“亲密”关系。在应用运行过程中，Activity1 产生的事件被标记为 src1 的无障碍辅助性服务处理，而不能被带有 src2 标记的无障碍辅助性服务处理。而 APP1 的无障碍辅助性服务在执行操作期间只能调用带有 service1 标记的服务，且调用该服务的处理结果只能返回给 APP1 而不是 APP2。由于标记不匹配，服务 service1 不能被 APP2 中的无障碍辅助性服务启动，并获得服务处理结果。

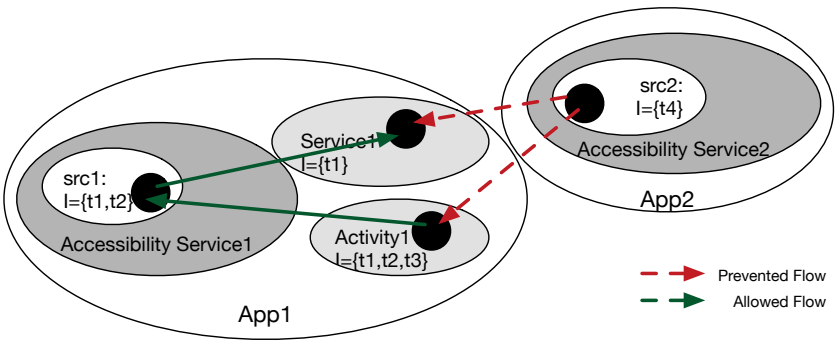


Fig.3 The information flow control of Accessibility service in Tassel system

图 3 Tassel 系统中无障碍辅助性服务信息流控制图

2.2 系统实现

我们的系统实现在安卓 6.0 版本上。该系统版本具有完整的无障碍辅助性服务功能模块及动态权限管理特性，且该版本下系统没有多分屏共享功能。

根据前面的介绍，我们知道在应用安装时需要系统中的包管理服务(PMS)；在应用运行启动一个活动时最终会创建一个 View 对象。该对象会产生各种事件并执行处理逻辑。若应用当前视图带有“importForAccessibility” 属性，则产生的事件优先交由系统中的无障碍辅助性服务处理。系统中可能注册了多个无障碍辅助性服务，在决定当前的无障碍事件该由哪个无障碍辅助性服务处理时，系统调用 AccessibilityManagerService 服务，该服务通过判断事件类型，从注册的服务列表中选择可以处理当前事件类型的无障碍辅助性服务。若系统中存在多个可以处理当前事件的无障碍辅助性服务，系统会展示给用户可用的服务列表，让用户选择。最后将处理结果返回给调用方。我们为系统添加了 TasselManagerService 服务。该服务是一个 JAVA 层服务，在系统启动后作为主要服务被启动。在该服务中我们实现了为服务及应用创建标签(包括“亲密性”属性)、获取标签等功能。这些方法可以被其他服务调用。在以上调用过程的基础上，为了使无障碍辅助性服务被安全地使用，我们的系统主要从以下几个方面进行加固。如下图 4。

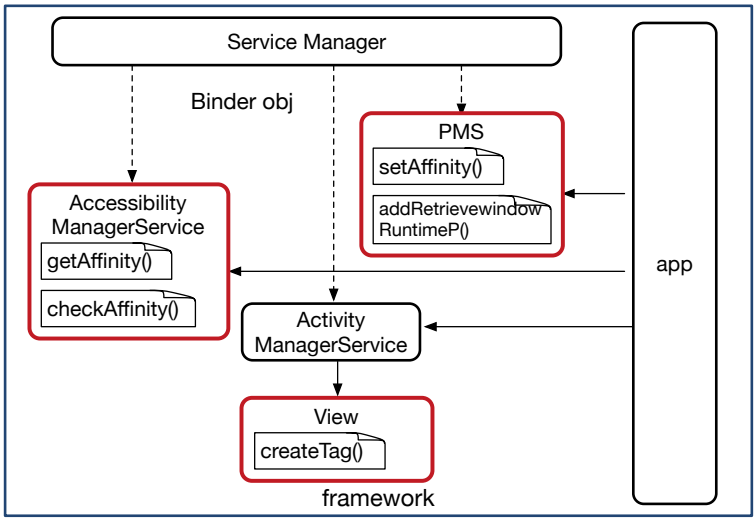


Fig.4 The main modules for enhancing Accessibility service in Tassel system

图 4 Tassel 系统实现中的主要模块

1） PackageManagerService 服务。在应用安装时，该服务负责解析应用压缩包。在我们的系统设计中，我们在应用安装解析过程中获得应用的 “sharedUserId” 属性，并获得活动组件中的 “taskaffinity” 值。这些属性用来标记应用服务、事件及应用的“亲密性”。应用自身的无障碍事件的处理过程取决于标记。应用间的无障碍事件处理则由“亲密性”决定。

2）View 视图。一个应用启动活动视图，生成视图对象是通过 View 类完成的。在该类中系统完成了对事件的初始化及分发操作。在我们的系统中，我们设计在初始化无障碍事件时，利用提取的一些属性信息及包名和 UID 值，生成标签并对无障碍事件进行标记。

3）AccessibilityManagerService 服务。该无障碍辅助性服务管理服务负责调用哪个无障碍辅助性服务处理哪些事件。在我们的系统中，为了保证事件得到正当处理，无障碍辅助性服务不被滥用，我们在启动无障碍辅助性服务处理事件时，首先获得服务的“亲密性”属性并进行检查。该步骤的检查包含两个方面：检查无障碍辅助性服务与待处理的事件得标签是否匹配或具有亲密性；检查无障碍辅助性服务与将

要调用的其他服务的“亲密性”来决定是否有权访问其他服务的处理结果。若满足，则继续处理，反之拒绝处理。

除此之外，为了保证一些隐私数据不被无障碍辅助性服务处理，我们在实现时还进行如下操作：

1）当一个应用活动启动创建一个视图，而该视图中含有“密码”等隐私信息时，我们设置该视图“importantForAccessibility” 属性为 “False”。这样，无障碍辅助性服务就不可以对该视图进行操作，无法通过获得视图内容窃取隐私数据。

2）当一个应用调用无障碍辅助性服务处理事件时，我们首先判断该应用是否在黑名单列表中。目前应用商店的一些应用已经被报道利用无障碍辅助性服务执行恶意操作，我们将这些应用加入黑名单。若一个应用在黑名单列表或与黑名单列表中的应用有关联关系，则系统拒绝使用无障碍辅助性服务处理事件。

在系统实现过程中，我们还利用安卓系统中接口定义语言 AIDL 的特性，设置接口参数的 ”in/out” 属性来控制信息的流动方向。这样可以保证一些无障碍辅助性服务处理结果不流向非法应用。

3 无障碍辅助性服务安全加固系统的测试

本节对本文实现的系统进行测试评估。

3.1 实验环境

我们的安全加固系统是在安卓 6.0.1 AOSP_HammerHead 基础上进行修改实现的，内核版本是 3.4.0-g67595d3。测试使用的设备为 Google Nexus5，处理器是 Qualcomm Snapdragon 800 2.27 GHz。

3.2 实验结果及分析

我们主要从系统的可用性，安全性及性能方面对本工作进行评估。

3.2.1 可用性

本文设计的 Tassel 系统主要是在安卓系统基础上添加了一个 Tassel 系统服务来管理无障碍辅助性服务及处理无障碍事件。本节我们主要说明在我们的系统中：安卓整体依然正常运行；系统中的无障碍辅助性服务仍可正常工作，并发挥应有的作用。同时该服务处理事件的操作受系统制定策略的约束，例如当活动视图中包含私密信息时该无障碍辅助性服务的一些特性失效，不能获取视图中的一些信息，并且不能对产生的无障碍事件进行处理。

编译修改后的系统并将生成的镜像文件刷入手机设备，我们发现设备系统正常开机运行。分析系统运行日志，本系统中的 Tassel 服务正常开启运行， 如下表 2。

Table 2 Piece of the log from the Tassel System

表 2 系统运行日志片段

系统运行日志片段
…
SystemServiceManager: Starting com.android.server.tassel.TasselManagerService
TasselManagerService: initialized.
Tassel: start the tasselmanagerservice.
…

系统中无障碍辅助性服务运行正常。从下图 5（a）可以看到，“No services installed” 表示系统应用没注册无障碍辅助性服务，也无第三方应用注册无障碍辅助性服务。编写一个样本应用安装在设备中，该应用注册一个无障碍辅助性服务，如下图 5（b），可以看到正常注册，并正常获取所需权限。在该应用安装注册无障碍辅助性服务过程中，我们系统中的 Tassel 服务被包管理服务调用，并根据当前应用的运行环境初始化，获得应用的一些配置信息对注册的无障碍辅助性服务进行标记：

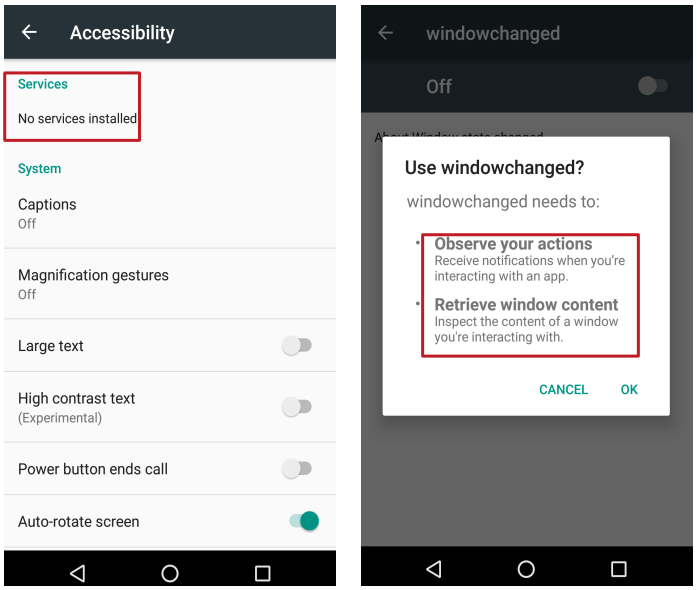
taTag(for service): com.example.accessibility_apk1000

当该无障碍辅助性服务开启后，系统中一些窗口界面发生变化时，视图发出无障碍事件，我们的 Tassel 服务在事件初始化时进行标记。由于该简单操作场景下界面变化来源有两种：系统 UI 变化和应用自身窗口变化，下面是对这两个事件的标记：

taTag(for event1): com.android.systemui1001612020

taTag(for event2): com.example.accessibility_apk100029158

在对这两个事件分发前，Tassel 服务判断事件 1 不是本应用发出的，所以不予分发处理，而是处理匹配的事件 2。



（a）系统原始状态　　（b）应用注册无障碍辅助性服务

Fig.5 Normal functioning of Accessibility service in Tassel system

图 5 Tassel 系统中无障碍辅助性服务正常运行

3.2.2　安全性

我们的系统主要是为了解决无障碍辅助性服务 API 被滥用的问题设计的。在本系统中，我们规定一个应用注册的无障碍辅助性服务只能对本应用或者有“亲密性”关系的应用产生的无障碍事件处理。一个应用的无障碍事件只能被本应用或者有关联关系的应用注册的无障碍辅助性服务消费处理。

测试时，我们编写并安装两个应用 com.example.accessibility_apk.apk 和 com.example.accessibilityDemo.apk。这两个应用没有被设置“sharedUserId”属性，它们分别注册自己的无障碍辅助性服务并被标记：

taTag(for service1): com.example.accessibility_apk1000

taTag(for service2): com.example.accessibilityDemo1002

在应用运行期间，我们监视系统的运行情况，从运行日志中分析这两个服务对运行过程中产生的各种无障碍事件的处理，处理情况如下表 3。通过标签匹配，我们发现这些服务都只对各自的无障碍事件进行处理，而不会去处理其他应用产生的无障碍事件。下一步修改应用并设置“sharedUserId”属性，再次运行。通过分析系统运行日志，我们发现这两个应用产生的无障碍事件可以被互相处理，但依然不能处理系统中其他应用产生的无障碍事件，如上表 3。我们修改应用使其中一个应用的活动界面包含“password”或者“密码”字样。当应用启动该活动，视图界面变化触发该应用的无障碍辅助性服务试图获得视图内容，系统阻止该无障碍辅助

性服务的操作，记录如下表 4。

Table 3 Piece of the log from the Tassel System

表 3 系统运行日志片段

系统运行日志片段
…
//开启两个应用的无障碍辅助性服务
create tag for accessibility event. TMS
get the packagename of the event: com.android.settings
get the affinity for event:
Tassel: Mismatch, cannot dispatch the event to the service…
//设置“sharedUserId”属性后
//只开启 com.example.accessibility_apk 应用的无障碍辅助性服务
get the packagename of the event:com..examplei.accessibilityDemo
get the affinity of the event: com.example.accessibility_apk
Tassel: Match! the event can be dispatched to the service
get the packagename of the event: com.android.launcher3
Tassel: Mismatch, cannot dispatch the event to the service…
…

Table 4 Piece of the log from the Tassel System

表 4 系统运行日志片段

系统运行日志片段
…
get the packagename of the event: com.example.accessibility_apk
Tassel: the accessibility service is blocked because this view may have something private
…

我们还将被暴露滥用无障碍辅助性服务 API 的应用安装在该系统中运行进行测试。例如游戏类执行静默安装的应用，工具类 WIFI 增强应用等。实验过程中，我们发现一些没有注册无障碍辅助性服务的应用不能调用其他应用的无障碍辅助性服务获得视图内容并执行自动点击而完成静默安装；还有一些应用注册了无障碍辅助性服务，由于系统中设置的约束策略，一些自动点击的行为无法执行，静默安装操作无法完成。

3.2.3 性能

在保证安全性的同时，本系统标记及检测策略的引入不应该导致整体性能下降。由此，我们检测本系统中 Tassel 服务运行过程中的耗时。我们设计以下：

a. 在我们的系统中安装应用时，Tassel 服务获得包管理服务解析的一些应用属性，进行服务标记并设置“亲密性”属性。这里我们获得启动 Tassel 服务消耗的时间以及获得标记进行标记消耗的时间，并与源系统测试对比。

b. 在我们的系统中，我们设计在视图生成、初始化无障碍事件并分发处理该事件时进行标记检查。为了测评该过程的耗时，我们记录在 Tassel 系统及源系统中生成一个无障碍事件的时间和该事件被分发处理的时间：从产生事件到事件被处理得到处理结果的时间差。

通过测试，我们获得在第一部分中，消耗总时间平均为 1ms；在第二部分中，消耗总时间平均为 14ms。相对安卓原系统来说，本系统中新策略的引入对性能的影响很小，事件处理延迟不超过 20ms。

4 相关工作

4.1 无障碍辅助性服务问题研究

早前有一些学者针对无障碍辅助性服务进行研究。文献[5]中对安卓 2.3 版本的无障碍辅助性服务进行研究。该工作中研究的无障碍辅助性服务是一个系统内部的服务，该系统版本下的无障碍辅助性服务不允许开发者操作调用。此工作从各种系统包括 Ubuntu, Window, Android, iOS 中的无障碍辅助性服务入手，主要从 I/O 的角度来考虑攻击，进行分析。CVE-2014-4368[19] 利用了 iOS 中的无障碍辅助性服务进行攻击，攻击者通过 AssistiveTouch 事件进行锁屏干扰，该漏洞已经被处理。

而本文对无障碍辅助性服务的研究是针对高版本系统下优化过的无障碍辅助性服务进行的。安卓 4.0 及以上版本的无障碍辅助性服务相对 2.3 系统增添了一些特性，并且提供了供应用开发者调用的接口。目前针对 4.0 以上版本的无障碍辅助性服务的全面研究及有效防范策略还没有。本文主要针对这类系统无障碍辅助性服务进行研究，并设计实现安全加固方案。

4.2 信息流保护机制

目前的计算机系统安全策略有很多，例如访问控制策略[20]。这种策略往往考虑的是对象的即时访问特性，而且不考虑相关信息流传播路径。这种模式对于现在的很多场景是不适用的。于是研究者提出了信息流控制机制[21,22]。信息流控制机制是一种用来增强系统安全性的技术，在很多安全类研究中都用到过[23,24,25]。该机制通过使用定义和一些增强手段可以保证数据的私密性，使其在系统中被正确传递，并防止数据被不正当暴露。在信息流系统中，主体和客体被预定义的安全规则标记。安全策略规定了数据流向，该流向遵循有向有限格。该机制有从语言层面来进行标记跟踪的：这种模型往往不能很好的对隐式数据流进行处理；也有从系统层面解决的：这种模型则不能很好地对系统中的应用数据标签进行追踪处理。且该机制暂无广泛地实际应用。

由于传统的安卓权限管理框架不足够来保护用户的数据，从而导致用户的一些隐私数据很容易遭到泄露。信息流跟踪技术可以解决安卓自身权限系统管理不够的问题。该技术通过一些限定条件来控制系统中的信息流，从而保证一些隐私数据在没授权的情况下不被泄漏出去。然而，该技术仅针对一些已知的对象数据例如手机设备的 IMEI 进行保护。但在安卓多任务系统中，存在着很多无法预知的数据，例如各个应用中的操作数据，传统的信息流跟踪技术将不足够解决多任务应用中的数据泄漏问题。于是研究者提出了分布式的信息流跟踪技术（DIFC）[26]。该技术是信息流跟踪技术的一个拓展，最早由 Andrew C. Myers 和 Barbara Liskov 两位研究者提出。该机制可以让每个应用针对自己的上下文场景做特有的安全标记。这些标记可以用来防止特定应用的数据泄漏、防止不安全代码运行在安全系统中带来的安全隐患等。之后，很多系统都采用此技术实现安全策略，例如银行、医务系统等[27]。而针对安卓系统的分布式信息流跟踪机制却很少，目前的系统有 Adwait Nadkarni 和 William Enck 提出的 Aquifer 系统[28], Jia 等提出的 DIFC 系统[29]，Xu 和 Witchel 提出的 Maxoid 系统[30] 和 Adwait Nadkarni 等提出的 Weir 系统[31]。第一个系统通过追踪应用中的数据流，用来保护已知数据的意外泄漏，但该系统禁止了多任务机制，且为了防止标记过多，不对后台组件例如服务、内容提供者等对象进行标记；Jia 等提出的使用 DIFC 的安全系统中同样地，通过在一个应用中限制新方法的调用来禁止多任务机制；在 Maxoid 系统中，为了避免标记爆炸，使用了多实例化方法来分离不同的标记数据，但这个机制仅用在磁盘上的数据，没有考虑内存中的数据。考虑到应用运行时内存中数据的共享，该系统的处理还是不够的。Weir 是 Adwait Nadkarni 等最近提出的一个安卓上实用的分布式信息流跟踪技术的增强系统。该系统解决了前面工作中限定多任务、禁止后台组件标记、标记磁盘存储数据的问题，并进一步考虑到安卓系统是网络事件驱动的环境，需要对标记的数据进行“解码”传输，制定了“解码”机制。但该系统依然存在着一些问题，例如 Overlay 文件系统中存储的多实例会带来一定的内存、电池等的消耗。

本文将根据我们的问题场景：安卓系统中无障碍辅助性服务的滥用，基于 DIFC 策略：从设计标记、进行标记、分发及处理解码标记等方面对无障碍辅助性服务进行管理，实现出安全的无障碍辅助性服务安卓系统。

5 总结与展望

本文对安卓系统中无障碍辅助性服务进行研究，深入分析了该服务的安全性问题及产生原因，并针对此问题，引入分布式信息流控制机制，设计实现了一个安全的无障碍辅助性服务安卓系统。本工作将系统部署在真机上进行测试，测试结果表明本文设计的系统可以在不影响整体性能的基础上，保证无障碍辅助性服务使用的安全性，防止该 API 被滥用。

随着移动设备的广泛使用，安卓系统也被越来越多的用户使用。而安卓系统及安卓应用安全是目前亟待解决的问题。本文设计的系统对安卓系统安全问题有指导意义。对于未来的工作：为了提供给设备使用者更加灵活的选择，我们可以在系统中增添一个本地服务，该服务与 JAVA 层服务共同作用，将设置无障碍辅助性服务标记的选择权交给用户。若用户开启标记设置选项，则系统的无障碍辅助性服务策略起作用，反之意味着用户不需要这层保护。

References:

[1] Android 开发无障碍指南. http://informationaccessibilityassociation.github.io/androidAccessibility/services.htm

[2] AccessibilityService. https://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html

[3] Android Accessibility 安全性研究报告. http://blogs.360.cn/360mobile/2016/09/07/research_of_accessibility/

[4] Yu Zhong, Astrid Weber, Casey Burkhardt, Phil Weaver, and Jeffrey P. Bigham. 2015. Enhancing Android accessibility for users with hand tremor by reducing fine pointing and steady tapping. In *Proceedings of the 12th Web for All Conference* (W4A '15). ACM, New York, NY, USA, , Article 29 , 10 pages. DOI: http://dx.doi.org/10.1145/2745555.2747277.

[5] Y. Jang, C. Song, S. P. Chung, T. Wang, and W. Lee. A11y Attacks: Exploiting Accessibility in Operating Systems. In ACM Conference on Computer and Communications Security (CCS), November 2014.

[6] Kraunelis J., Chen Y., Ling Z., Fu X., Zhao W. (2014) On Malware Leveraging the Android Accessibility Framework. In: Stojmenovic I., Cheng Z., Guo S. (eds) Mobile and Ubiquitous Systems: Computing, Networking, and Services. MobiQuitous 2013. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 131. Springer, Cham

[7] Learn about global development. http://www.worldbank.org/en/topic/disability/overview#1

[8] Yair Amit. (03 May, 2016) Android Clickjacking – Android Malware Evolution. https://www.skycure.com/blog/accessibility-clickjacking/

[9] Yair Amit. (17 May, 2016) 95.4 Percent of All Android Devices Are Susceptible to Accessibility Clickjacking Exploits. https://www.skycure.com/blog/95-4-android-devices-susceptible-accessibility-clickjacking-exploits/

[10] Accessibility. https://developer.android.com/guide/topics/ui/accessibility/index.html

[11] 逆巴. (12 18，2015) 滥用 Accessibility Service 自动安装应用. http://ju.outofmemory.cn/entry/227941

[12] 小七在简书. （08 30，2015）Android 事件传递机制. http://www.jianshu.com/p/cf22ea3b09a5

[13] ztelur （04 17, 2016）Android 视图架构详解. http://blog.csdn.net/u012422440/article/details/51173387

[14] Vedprakash Rout. (Aug 22, 2016) Security issues with Android Accessibility. https://android.jlelse.eu/android-accessibility-75fdc5810025

[15] Dinesh Venkatesan. (04 May, 2016) Malware May abuse Android's Accessibility Service to bypass security enhancements. https://www.symantec.com/connect/blogs/malware-may-abuse-android-s-accessibility-service-bypass-security-enhancements

[16] Kompasim. (04 10, 2017) Android 实现自动抢红包. https://github.com/kompasim/android-wechat-tool/blob/master/README.md

[17] Guolin. (12 14, 2015) Android 静默安装实现方案. http://blog.csdn.net/guolin_blog/article/details/47803149

[18] 董. (02 28, 2017). 安卓新木马. http://www.shujubo.com/news/15/2280.html

[19] CVE-2014-4368, http://www.nsfocus.net/vulndb/27932

[20] Quiet Heart. (03 21, 2017) Linux 访问控制权限基本原理. http://www.jianshu.com/p/56d5c68b5363

[21] D. E. Denning. A lattice model of secure information flow. *CACM*, 19(5):236–243, May 1976.

[22] 吴泽智,陈性元,杨智,杜学绘.信息流控制研究进展.软件学报,2017,28(1):135−159. http://www.jos.org.cn/1000- 9825/5131.htm

[23] Krohn M, Yip A, Brodsky M, Kaashoek MF, Kohler E, Morris R. Information flow control for standard OS abstractions. In: Proc. of the ACM SIGOPS Operating Systems Review. New York: ACM Press, 2007. 321−334. [doi: 10.1145/1294261.1294293]

[24] Efstathopoulos P, Krohn M, VanDeBogart S, Frey C, Ziegler D, Kohler E, Morris R. Labels and event processes in the Asbestos operating system. In: Proc. of the SOSP. Brighton: ACM Press, 2005. 17−30. [doi: 10.1145/1095810.1095813]

[25] Zeldovich N, Boyd-Wickizer S, Mazieres D. Securing distributed systems with information flow control. In: Proc. of the NSDI. San Francisco: USENIX, 2008. 293−308.

[26] Myers A C, Liskov B. A decentralized model for information flow control[M]. ACM, 1997.

[27] 许帅;分布式系统中的信息流控制模型的研究[D];上海交通大学;2011 年.

[28] NADKARNI, A., AND ENCK, W. Preventing Accidental Data Disclosure in Modern Operating Systems. In *Proceedings of the ACM Conference on Computer and Communications Secu- rity (CCS)* (2013).

[29] JIA, L., ALJURAIDAN, J., FRAGKAKI, E., BAUER, L., STROUCKEN, M., FUKUSHIMA, K., KIYOMOTO, S., AND MIYAKE, Y. Run-Time Enforcement of Information-Flow Prop- erties on Android (Extended Abstract). In *Proceedings of the European Symposium on Research in Computer Security (ES- ORICS)* (2013).

[30] XU, Y., AND WITCHEL, E. Maxoid: transparently confining mobile applications with custom views of state. In *Proceedings of the Tenth European Conference on Computer Systems* (2015), ACM.

[31] Nadkarni A, Andow B, Enck W, et al. Practical DIFC enforcement on Android[C]//25th USENIX Security Symposium (USENIX Security 16). USENIX Association, 2016: 1119-1136.