
ISC
2015
中国互联网安全大会
360互联网安全中心
ISC
2015
数据驱动安全
2015 中国互联网安全大会
China Internet Security Conference
Android平台
磁盘数据安全
现状分析
束骏亮
@上海交通大学 LoCCS GoSSIP

关于我

- GoSSIP成员
  - 研究方向：系统安全、软件安全、Android安全
- 0ops队员
  - 主攻：Android相关、steganography、forensics、misc

http://weibo.com/u/3659841663

ANDROID平台磁盘数据安全现状分析

一块容易被忽视的传统战场

- 问题存在吗？
- 有多严重？

ANDROID平台磁盘数据安全现状分析

- 超过90%的Android设备正面临磁盘数据泄漏的风险
- 不仅仅局限于隐私数据，同时还包括了大量和App认证相关的数据

ANDROID平台磁盘数据安全现状分析

- 磁盘数据最主要的风险来自于擦除操作时产生的数据残留
- 数据残留的出现由底层介质和文件系统特性共同决定
- Android系统中并没有易于使用和推广的缓解机制

大纲

- Android数据存储
- Android中的数据擦除操作
- 如何去攻击残留的数据
- 实验
- 案例分析
- 我们能做什么

数据存储模型

操作系统
(Android)

文件系统
(yaffs2 & ext4)

硬件虚拟层
(FTL)

存储介质
(Flash memory)

Android数据存储——存储介质

Flash Memory(闪存)

- 非易失性、可反复擦写的存储介质，是目前移动设备中常用的存储介质。

Android数据存储——存储介质

闪存的特性:

- 可以进行字节级别的读写操作，但是只能进行块(block)级的改写/擦除。
- 每一个物理块具有10^4~10^5擦除的次数限制。

Android数据存储——FTL

Flash Translation Layer(FTL):

- 直接操作底层的存储介质，将磁盘物理地址映射为逻辑上的块地址，将Flash Memory封装成一个块设备供上层操作。

Android数据存储——FTL

FTL同样会有一些特性:

- 更新一段数据时，会将新的数据写入一个新的未使用的块，之后修改地址之间的映射关系，将原数据所属的块标记为未使用，原有的数据仍旧保留在磁盘上。
- 通过不定期的垃圾回收(GC)操作来统一对未使用的块进行数据擦除。
- 为了延长寿命，垃圾回收时优先选择擦除次数较少的块。

Android数据存储——文件系统

从YAFFS2到ext4:

- YAFFS2(Yet Another Flash File System)
- Ext4(The fourth extended file system)
- 都是日志型文件系统(Log-structured File System)

Android数据存储——文件系统

日志型文件系统的特性:

- 所有对文件的写入操作都会在日志分区中添加一份对应的记录
- 基于底层的特性，对文件的改写并不会擦除原文件的内容，而是新建一个文件然后改变文件节点内的地址指针，原文件的信息依旧保留在日志中，原文件的内容依旧保留在磁盘上。

Android数据存储——总结

- 多个层次的特性合力导致了文件擦除是一个极其不安全的操作。
- 设想中被擦除的数据将会残留在磁盘上。
- 残留时间和数据量与磁盘大小呈正相关，和设备使用频率呈负相关。

Android数据存储——总结

磁盘容量	数据残留时间(小时)	
	中位数	>95%
200MB	41.5±2.6	46.2±0.5
1GB	163.1±7.1	169.7±7.8
2GB	349.4±11.2	370.3±5.9

Android数据存储——总结

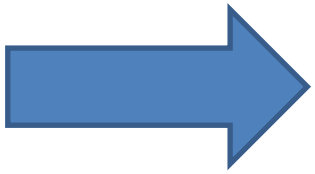
- 在这种情况下，上层操作系统必须进行针对性的安全设计，以此保证磁盘数据的安全。

Android中的数据擦除操作

- 通过系统API进行文件擦除
- 应用程序卸载
- Recovery模式恢复出厂设置
- Fastboot模式解锁、刷机

Android中的数据擦除操作

-系统API



Android中的数据擦除操作

-系统API

- File.delete()

/libcore/luni/src/main/java/java/io/File.java/delete()
-->/libcore/luni/src/main/java/libcore/io/Posix.java/remove()
-->/libcore/…/libcore_io_Posix.cpp/Posix_remove()
-->remove()

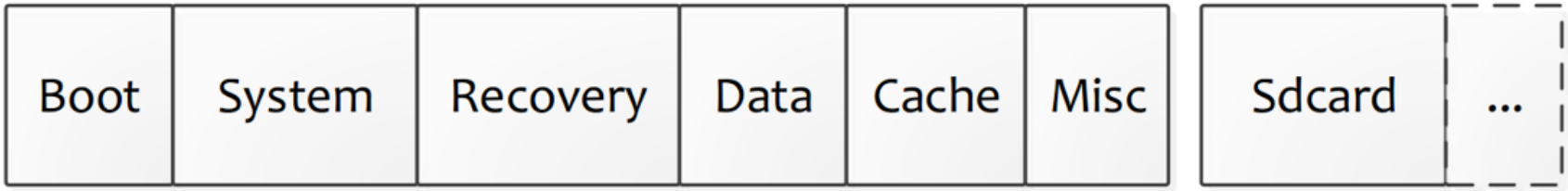
- 最终调用Linux syscall：*unlink(2)* 和*rmdir(2)*
- 不安全

Android中的数据擦除操作

-应用程序卸载

Android中的数据擦除操作

-应用程序卸载



Android中的数据擦除操作

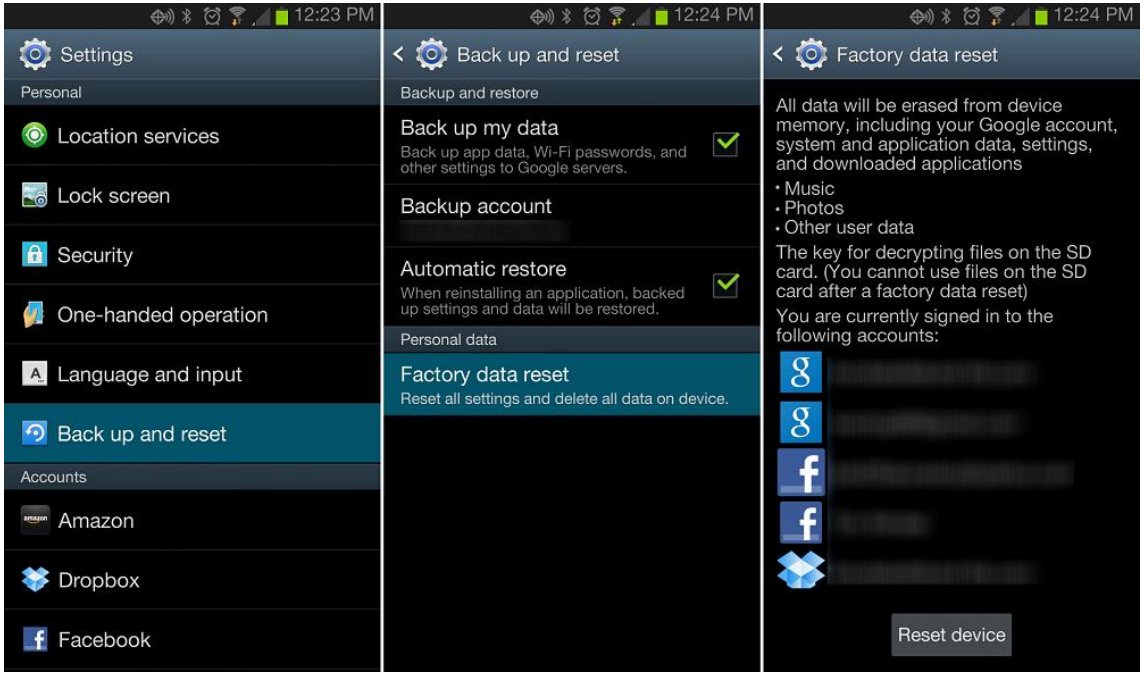
-应用程序卸载

deletePackage()
->deletePackageAsUser()
->deletePackageX()
->deletePackageLI()
->removeDataDirsLI()
->remove()
->uninstall()
->_delete_dir_contents()

- 最终会调用Linux syscall: unlinkat(2)
- 不安全

Android中的数据擦除操作

-恢复出厂设置



Android中的数据擦除操作

-恢复出厂设置

- Android4.0之前的恢复出厂设置功能均不安全
- Android4.0之后，Google官方Recovery会使用make_ext4fs来对文件系统进行重建，我们的实验证明该方案能够正确擦除userdata分区下的数据。
- 安全

Android中的数据擦除操作

-恢复出厂设置

- 部分第三方、定制Recovery在进行恢复出厂设置操作时，为了避免将SD卡格式化(针对那些SD卡和userdata属于同一分区的设备)，会使用rm –rf命令来擦除userdata分区内的数据。
- 最为流行的CWM Recovery和TWRP均存在该问题，影响上千万设备。
- 不安全

Android中的数据擦除操作

-fastboot擦除数据

Android中的数据擦除操作

-fastboot擦除数据

- Fastboot模式提供针对userdata分区的擦除功能，同样使用make_ext4fs对文件系统进行重建。
- 安全
- 但是，对于普通用户来说，操作过于复杂。

Android中的数据擦除操作

-补救措施

- 全磁盘加密(FDE)
- FDE主密钥来自于锁屏密码，导致Android 5.0之前的FDE有被暴力破解的风险。
- Android 5.0之后引入了安全硬件提升安全性，但过高的成本以及Android版本的碎片化导致普及存在难度，目前仅有不到7%的设备更新了Android 5.0。

Android中的数据擦除操作

-总结

- Android系统中绝大部分的数据擦除操作是不安全的
- 仅有的能够成功擦除数据的方式对一般用户来说太不友好
- 现有的补救措施仅能够解决极小部分的问题。

如何攻击残留的数据

- 和设备的物理接触
- 取得磁盘的镜像

如何攻击残留的数据

-常规磁盘取证

- 磁盘未遭破坏，log信息完整
- *Extundelete*、*recuva*、*ufs* 等
- 所有类型的文件

如何攻击残留的数据

-file carving

- 磁盘遭到破坏，镜像不完整
- 以文件结构为特征，以块(block)为单位进行搜索
- *binwalk*、*foremost* 等
- 针对特定的文件类型进行定制、优化(SQLite)

如何攻击残留的数据

-cell carving

- 磁盘被严重破坏，块(block)也不完整
- 以文件内部结构为特征进行深层次的数据挖掘
- 比如针对SQLite数据库内的每一条数据记录进行carving

实验

- 存在性实验
- 模拟攻击
- 案例分析

实验

- API文件删除

数据集大小	数据残留率
85.64MB	100%
150.63MB	100%
400.30MB	100%

实验

- 应用程序卸载

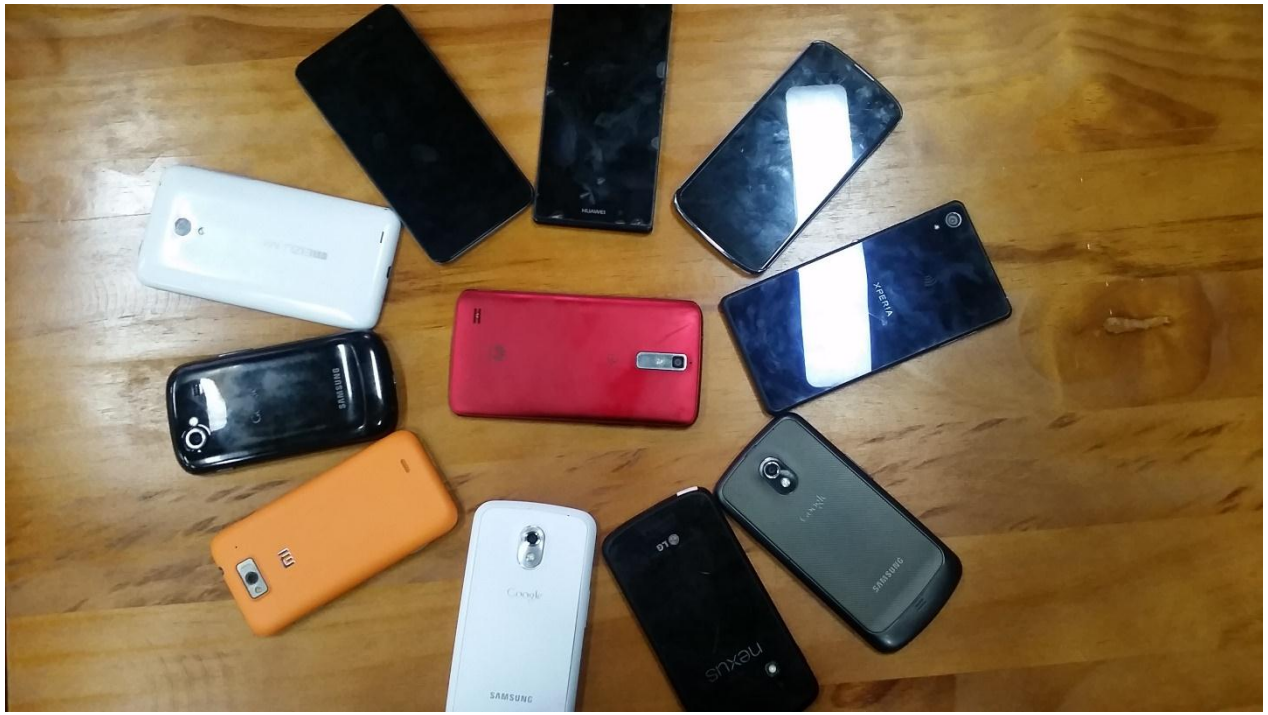
文件数/残留	微信	QQ	微博	Facebook	Snapchat
实验一	86/100	184/190	64/79	65/70	23/25
实验二	79/79	160/160	73/78	66/70	13/15
实验三	80/80	149/156	40/69	66/70	15/15

实验

恢复出厂设置

设备(系统)	Recovery版本	分区大小/块数	一次数据残留率
Samsung Galaxy S3 (Android 4.2.2)	TWRP 2.7.0.0	11.5GB / 3022848	98.03%
	CWM 6.0.4.6		84.86%
Sony Lt28H (Android 4.4.2)	TWRP 2.6.3.0	2.00GB / 524288	0.48%
	CWM 6.0.3.0		0.00%
ASUS Nexus7 (Android 4.4.2)	TWRP 2.6.3.1	3.84GB / 1007616	89.06%
	CWM 6.0.4.3		93.26%
LG Nexus4 (Android 4.4.2)	TWRP 2.6.3.3	13.1GB / 3449600	95.41%
	CWM 6.0.4.7		94.21%

实验



- 15台二手Android手机和平板，不同品牌，不同型号，不同来源

实验

- 对所有设备镜像进行取证分析，并比对原始镜像
- 平均恢复出数据约100MB/GB
- 平均恢复出数据文件约350个/GB
- 数据种类包括：图片、视频、数据库、apk、dex、文本等
- 很多有意思的东西

实验

- 对所有设备镜像进行取证分析，并比对原始镜像
- 平均恢复出数据约100MB/GB
- 平均恢复出数据文件约350个/GB
- 数据种类包括：图片、视频、数据库、apk、dex、文本等
- 很多有意思的东西

实验

- 案例一，某炒股软件加密数据和dex

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>↓
<map>↓
    <boolean name="save_pwd" value="true" />↓
    <string name="cookie">sessionid=mfyp6fy[redacted]cs9f7ff</string>↓
    <string name="user_name">ffz[redacted]</string>↓
    <string name="user_pwd">[redacted]aa84ec3d0cd2c731a</string>↓
    <int name="login_source" value="0" />↓
</map>↓
←
```

实验

- 案例二，某金融软件登录token

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="str">4830845,mzdj[redacted]mNfqt27iTQZKZKh0cio*,0</string>
</map>
```

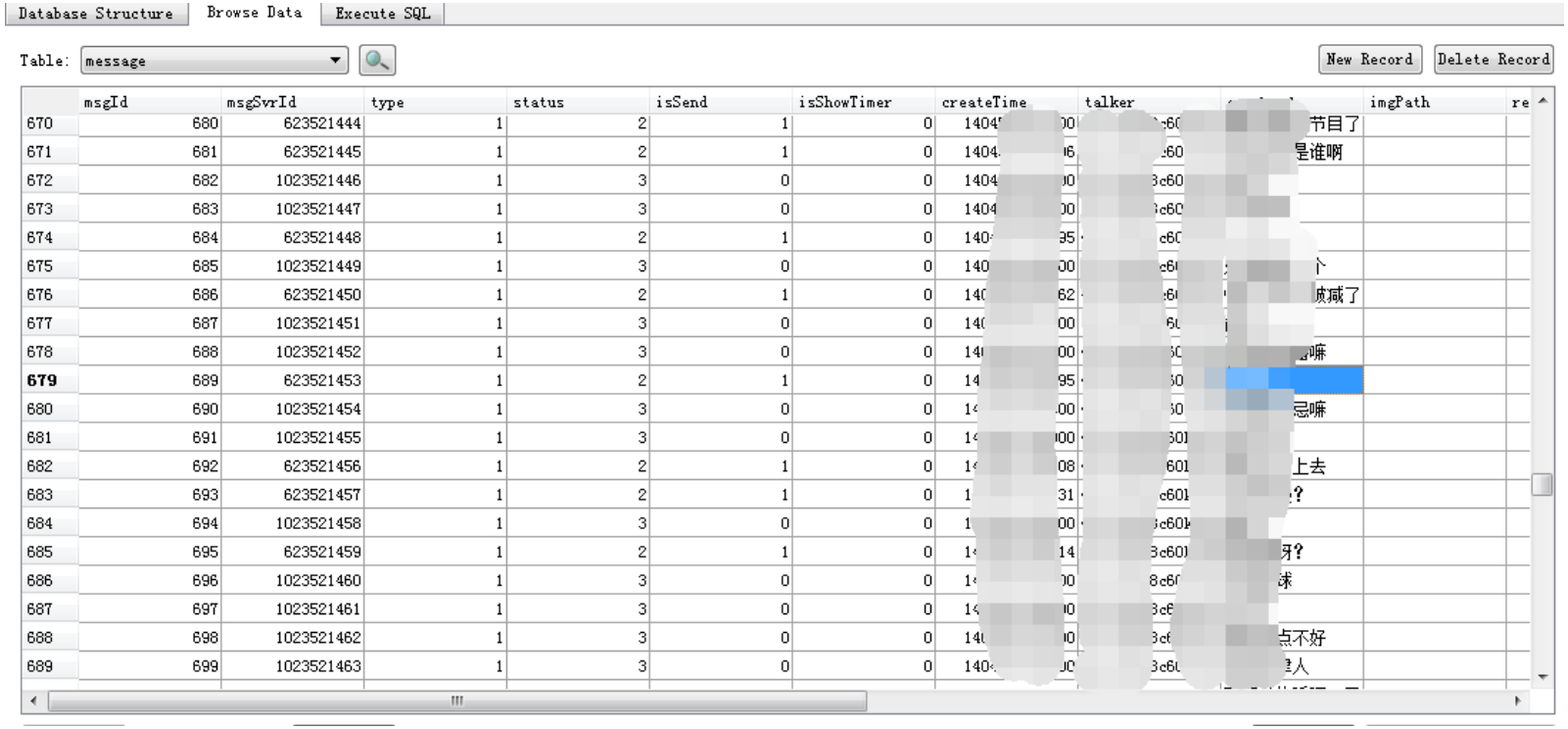
实验

- 案例三，某金融软件明文数据

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="actionImageId">1</string>
    <string name="accessToken">[redacted]e97911f6948990e</string>
    <string name="actionTurnValueId">http:/[redacted]tml?clientLogin=true&amp
    <string name="mac">10:68:3f:ff:ae:52</string>
    <boolean name="isAutoLogin" value="false" />
    <string name="password">[redacted]6</string>
    <string name="balance">0.00</string>
    <string name="bankNo"></string>
    <null name="exchangeCash" />
    <string name="username">[redacted]721</string>
    <boolean name="mainStartGuid" value="true" />
    <string name="hasPayPwd">0</string>
    <boolean name="auto_login" value="true" />
    <string name="name"></string>
    <boolean name="veryPayPassword" value="false" />
    <string name="certid"></string>
    <string name="bankUserName"></string>
    <boolean name="isShortcut" value="true" />
    <string name="autologintoken">[redacted]3-09347866a3e5</string>
    <string name="nickname"></string>
    <string name="imei">[redacted]0802947</string>
    <string name="bankName"></string>
    <boolean name="isFirst" value="true" />
    <string name="userno">[redacted]0530438</string>
    <string name="goldBalance">0.0</string>
    <null name="point" />
    <string name="deviceToken">[redacted]v36VVw9GTauktwKWtEvXr6-u8</string>
    <string name="noDrawBalance">0.00</string>
    <string name="drawBalance">0.00</string>
    <string name="safeQuestion"></string>
    <string name="mobileid"></string>
    <string name="channel">980</string>
</map>
```

实验

- 案例四，某IM软件加密数据库



	msgId	msgSvrId	type	status	isSend	isShowTimer	createTime	talker	imgPath	re
670	680	623521444	1	2	1	0	1404[illegible]	[illegible]	节目了	
671	681	623521445	1	2	1	0	1404[illegible]	[illegible]	是谁啊	
672	682	1023521446	1	3	0	0	1404[illegible]	[illegible]		
673	683	1023521447	1	3	0	0	1404[illegible]	[illegible]		
674	684	623521448	1	2	1	0	140[illegible]	[illegible]		
675	685	1023521449	1	3	0	0	140[illegible]	[illegible]	个	
676	686	623521450	1	2	1	0	140[illegible]	[illegible]	减了	
677	687	1023521451	1	3	0	0	140[illegible]	[illegible]		
678	688	1023521452	1	3	0	0	140[illegible]	[illegible]	嘛	
679	689	623521453	1	2	1	0	14[illegible]	[illegible]		
680	690	1023521454	1	3	0	0	1[illegible]	[illegible]	忌嘛	
681	691	1023521455	1	3	0	0	1[illegible]	[illegible]		
682	692	623521456	1	2	1	0	1[illegible]	[illegible]	上去	
683	693	623521457	1	2	1	0	1[illegible]	[illegible]	?	
684	694	1023521458	1	3	0	0	1[illegible]	[illegible]		
685	695	623521459	1	2	1	0	1[illegible]	[illegible]	牙?	
686	696	1023521460	1	3	0	0	1[illegible]	[illegible]	球	
687	697	1023521461	1	3	0	0	1[illegible]	[illegible]		
688	698	1023521462	1	3	0	0	140[illegible]	[illegible]	点不好	
689	699	1023521463	1	3	0	0	140[illegible]	[illegible]	人	

我们能做什么

- 设备/ROM制造商：
  - 确保Recovery中恢复出厂设置功能的正确实现
  - 在内核中添加对底层安全擦除API的支持(BLKSECDISCARD)

我们能做什么

- App开发者：
  - 避免在磁盘上存储重要数据
  - 无法避免时，使用正确的密钥交换协议和加密算法来处理重要数据

我们能做什么

- 用户：
  - 在丢弃/赠送/出售手机前，确保使用正确的Recovery进行恢复出厂设置。
  - 有能力的用户请使用fastboot模式对磁盘进行清理。

我们能做什么

- 安全研究人员：
  - 提出易用、易普及的解决方案
  - 现有的解决方案均存在问题

ISC 2015 中国互联网安全大会

360互联网安全中心

thx

Q&A