Security Analysis of Anti-Theft Solutions by Android Mobile Anti-Virus Apps

Laurent Simon
lmrs2@cam.ac.uk
https://www.cl.cam.ac.uk/~lmrs2/
Talk outline

• Background
• Mobile Anti Virus (MAV) sample
• Lock
• Wipe
Background

- Phone theft is a growing problem
  - 2013:
    - 3.1M devices stolen in the USA
    - 120,000 in London
  - 50% of users don't lock their phone
Anti-Theft Solutions

• Wide offering – enterprise and consumer-grade
  => This talk: *consumer grade* only

• Top 10 Mobile Anti Virus apps (MAV), downloaded from Google Play hundreds of millions of times (top 2 between 100M and 500M)

• Anti-theft enable *remote wipe* and *remote lock* with an app on phone + remote trigger via
  • web page
  • SMS
Partition storing user data

- **Data partition** mounted on `/data`
  - Sensitive info, ext4 (eMMC), yaffs2 ("raw flash")

- **Internal (primary) "SD card"**: mounted on `/sdcard`
  - Music, pictures, FAT, emulated (FUSE)

- **External SD card**: removable
  - Same as internal one, FAT
  - Secondary SD card, or primary if no internal one
Admin API

- Provides admin features, i.e. sensitive functions
- Access to various "policies": e.g. force-lock, wipe-data, reset-password
- Like traditional Android permissions, each policy declared in Android manifest file
- Like traditional Android permissions, policies not accepted at installation but manually enabled/disabled in the phone Settings
Admin API (Cont'ed)

Device administrators

- avast! Anti-Theft
- McAfee Security
- Sample Device Admin

McAfee Security

Activating this administrator will allow the application McAfee Security to perform the following operations:

- **Erase all data**
  Erase the phone’s data without warning, by performing a factory data reset

- **Change the screen-unlock password**
  Change the screen-unlock password

- **Set password rules**
  Control the length and the characters allowed in screen-unlock passwords

- **Monitor screen-unlock attempts**

Activate | Cancel
Admin API (Cont'ed)

• If user does not grant admin access, app can still run ... without admin privileges

• To uninstall/remove admin app, admin privileges must be disabled first

• Restrictions imposed: cannot read other apps' data or read/write chip at block level
Admin API (Cont'ed)

• Focus of this talk: force-lock and wipe-data policies

• `wipeData(int flag)`:
  • Triggers the built-in Factory Reset
  • Flag indicates:
    – Wipe only data partition
    – Wipe data partition AND primary SD card

• `LockNow()`: lock the screen with default Android PIN

• No admin granted: ad-hoc solutions
Modes

- Normal mode: Android
- Safe mode
- Recovery/Bootloader mode
Talk outline

- Background
- Mobile Anti Virus (MAV) sample
- Lock
- Wipe
Apps studied

• 10 most downloaded Mobile Anti Virus (MAV) apps on Google Play
  • AVG, Lookout, Avast, Dr.web, Norton, McAfee, Kaspersky, TrustGo, TrendMicro, Avira
• Top 2 downloaded 100M-500M
• Following top 4 10M-50M
Talk outline

- Background
- Mobile Anti Virus (MAV) sample
- Lock
- Wipe
Removal of MAVs & API Misuse

- **Scenario: admin + non-locked:**
- **7/10** MAVs do not prevent disabling admin privileges
- McAfee and Avast prompt user with PIN when trying to disable admin
Removal of MAVs & API Misuse

public class McAfeeReceiver extends DeviceAdminReceiver {

    public void onDisabled(Context paramContext, Intent paramIntent) {
        [...] // removed
        displayLockScreen();
    }
}

- Android doc: "called prior to the administrator being disabled"
- BUT called after on Gingerbread (GB, v2.3.x)
- OnDisabledRequested() called prior on GB, ICS, JB
Other API Misuses

- **Scenario: admin + locked**: proper lock implementation requires:
  - Force-lock policy declared in manifest file by MAV
  - Manual granting of admin by users
  - Proper use of API by MAV, e.g. `lockNow()`
- **4/10 MAVs do not use `lockNow()` even when granted admin privileges**
  - Bypass thru Safe mode
Rate Limiting

- Scenario: admin + locked + use lockNow()
- Overlay of custom lock screen on top of default Android PIN screen
Rate Limiting

- **5/10** MAVs do not enforce rate limiting in their screen => brute-force PIN feasible

- For a 4-digit PIN and 5sec/PIN attempt, about 7hrs on average for randomly selected PINs

- <5mn for 60 most common PINs ~ 30%
- <40mn for 400 most common PINs ~ 50%
Rate Limiting

- **Scenario**: admin + locked + use `lockNow()` + rate limiting
- Some devices have no rate limiting (e.g. Samsung Galaxy S Plus)
- Reboot into Safe mode where user-installed apps do not run automatically
- Counter storing glitches: e.g. for Lookout, removing battery resets the state
Network-level attacks: GSM

- Avast (100M-500M download) sends temp PIN in clear
- Similar issue for Dr.Web with commands sent via SMS
Network-level attacks: TLS

- Impersonate as cloud server to send an unlock command
- One app did not validate the CN of certs
Vendor customisations

- Charging mode gives shell: e.g. LG L7 running JB (v4.1.2)

- Unprotected Recovery/Booloader: flash arbitrary binaries to access data regardless of Android lock. Most Samsung/LG phones in our sample.
Talk outline

- Background
- Mobile Anti Virus (MAV) sample
- Lock
- Wipe
Wipe implementations

- Data partition: 10/10 use admin API to wipe it
  - If no admin privileges, just use phone APIs (contact, SMS, etc)
- Primary SD: 5/10 MAVs use admin API to wipe it
  - Other MAVs unlink and/or overwrite files and/or format partition
- Secondary SD: 10/10 MAVs use ad-hoc solutions (unlink, overwrite files, format partition). *Android has no API to wipe it.*
Lookout implementation

- Overwrites files and unlinks them
- Dev assume file update occurs "in-place"
- On Galaxy S Plus, FAT-formatted primary SD: >90% data recoverable
Avast implementation

- "Thorough wipe" option:
  - Unlinks all files from external storage
  - Creates a 1MB file and overwrites it 1000 times with zeros
- Dev assume file update does NOT occurs "in-place", so 1GB (1000x1MB) unallocated space is overwritten
- Partitions formatted with ext4 update "in-place", 99% of data is recoverable
Conclusion

- Lock implementations can be circumvented because of misuse of APIs, vendor customisations, restrictions imposed by Android

- Wipe implementations are not better than the built-in (possibly flawed) Factory Reset

- Vendor solutions only have the potential to increase reliability
Thanks!

Laurent Simon

lmrs2@cam.ac.uk

https://www.cl.cam.ac.uk/~lmrs2/