Breaking Business As Usual

Attacking Android Enterprise Solutions

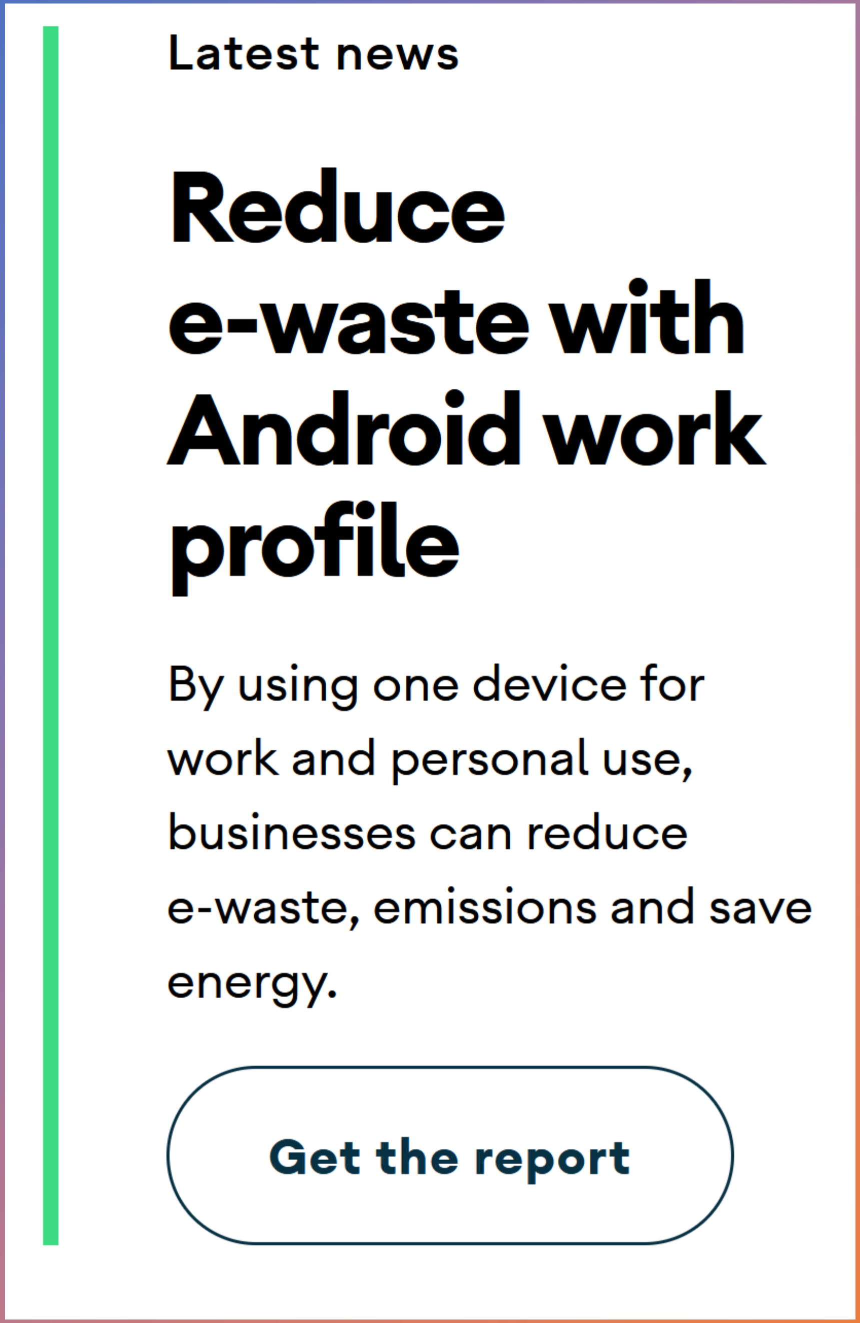
Aug 09, 2023

Priyank Nigam

Microsoft Red Team

बैकग्राउंड

- Red Team @Microsoft
- Previous
  - Offensive Security Consulting
  - App Dev (C++)
  - Speaker at BSidesLV, THOTCon, BSidesSEA, DenverISSA etc. conferences
- Research Interests
  - AppSec (Web/mobile) + Network Sec
  - IoT
  - MS Azure
- Blue team @Home
  - My toddler in YOLO'ing, RED team mode



Latest news

Reduce e-waste with Android work profile

By using one device for work and personal use, businesses can reduce e-waste, emissions and save energy.

Get the report

What is Android Enterprise?

Company-owned device (*corp-liable device)*

- Device Admin - A device owned and fully managed by an employee's organization.
- Can be set up exclusively for work use (fully managed), or to allow both work and personal use (fully managed with a work profile).

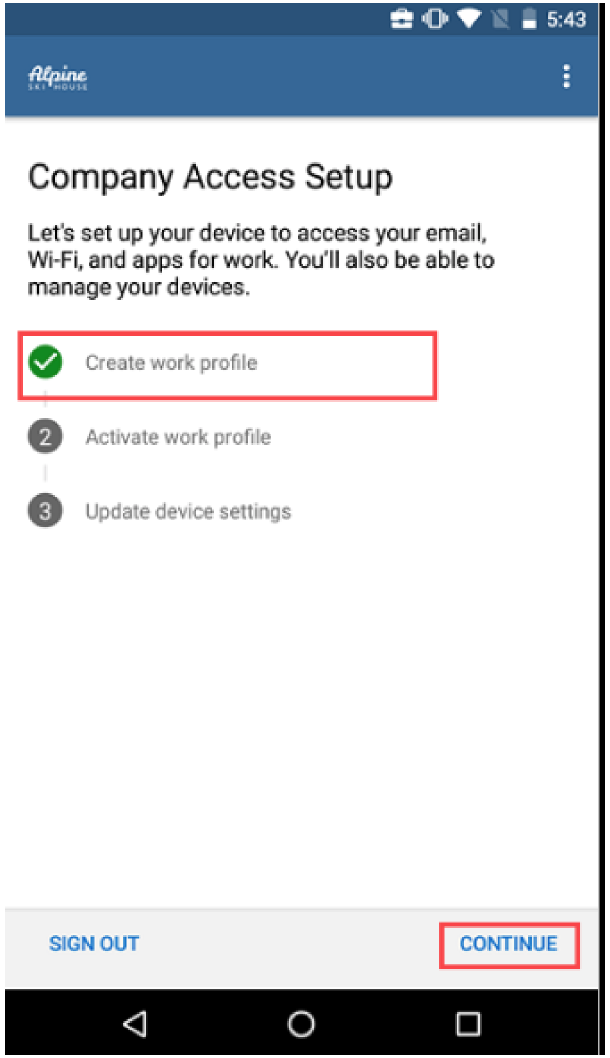
Personally Owned Devices/bring your own device (BYOD)

- Work profiles, which separate work apps from personal apps, are the recommended deployment method for BYOD devices

The trend is towards the later.

Some countries still require the device to operate in device admin mode.

Personally Owned Work Profile (WP)



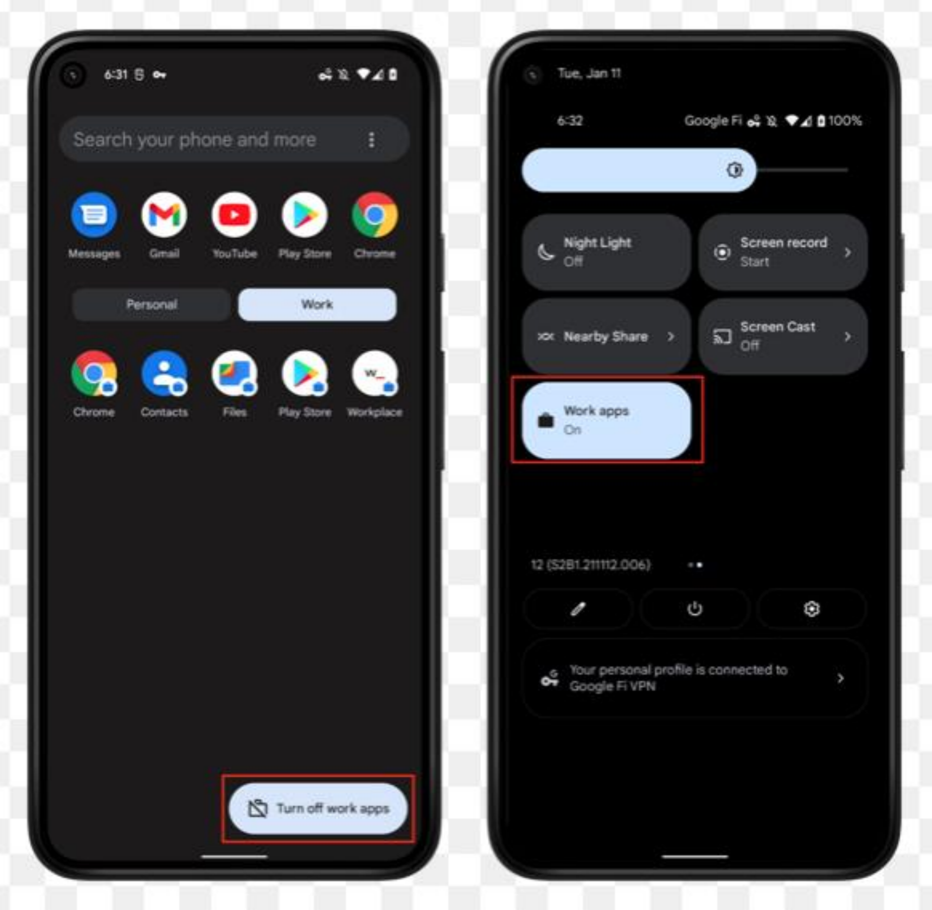
Company Access Setup

Let's set up your device to access your email, Wi-Fi, and apps for work. You'll also be able to manage your devices.

✓ Create work profile

2 Activate work profile

3 Update device settings



Night Light Off

Screen record Start

Nearby Share

Screen Cast Off

Work apps On

12 (S2B1.211112.006)

Your personal profile is connected to Google Fi VPN

EMM - Device Policy Controller

- An enterprise mobility management (EMM) device policy controller application allows IT administrators to separately manage access to corporate apps and data on supported Android 5.0+ devices.
- The device policy controller app
  - Communicates with the EMM software to apply profile and device restrictions and settings.
  - Implements managed configurations and verifies device compliance with the EMM's policies.

Threat Model

Filesystem (Work Apps within personal profile)

- Add personal accounts within work apps?
- Logs
- IPC (Content Providers/Activity)

Network Comms (Shared Cert Store)

Connected Work Apps

EMM App Security

Work Apps Security

Rootkits

Install Work apps in Personal Profile?

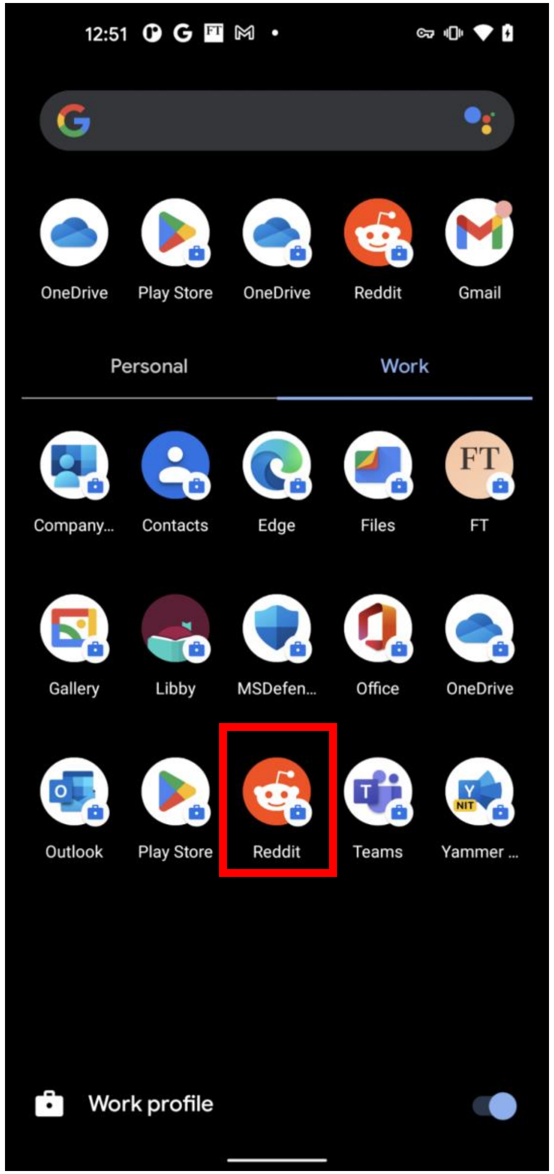
- Use regular Play store or sideload the work app.
- Some apps allow to be installed as a personal profile
- Difficult to run compliance checks on a personal app. Only server-side conditional access can prevent this. (Eg – Authentication at Azure AD)

Install personal apps in work profile?

```
redfin:/sdcard # pm install --user 11
/data/local/tmp//Reddit_v2022.38.0_apkpure.com.apk
Success
```

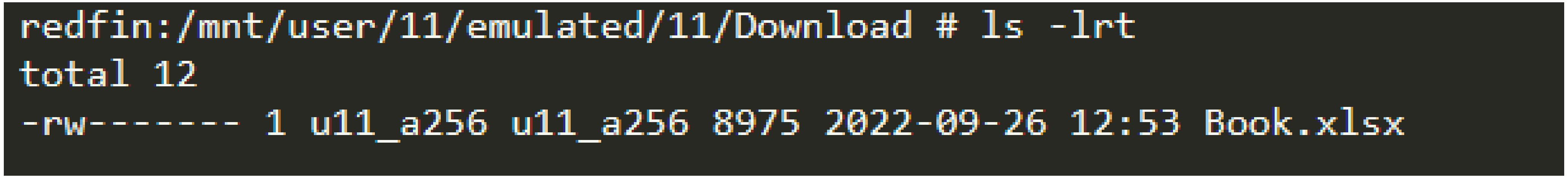
- Interact with work apps via IPCs.
- No root access required, adb would install the app for all users.

- **Trust boundary is broken!**



IPC (Content Providers/Activity)

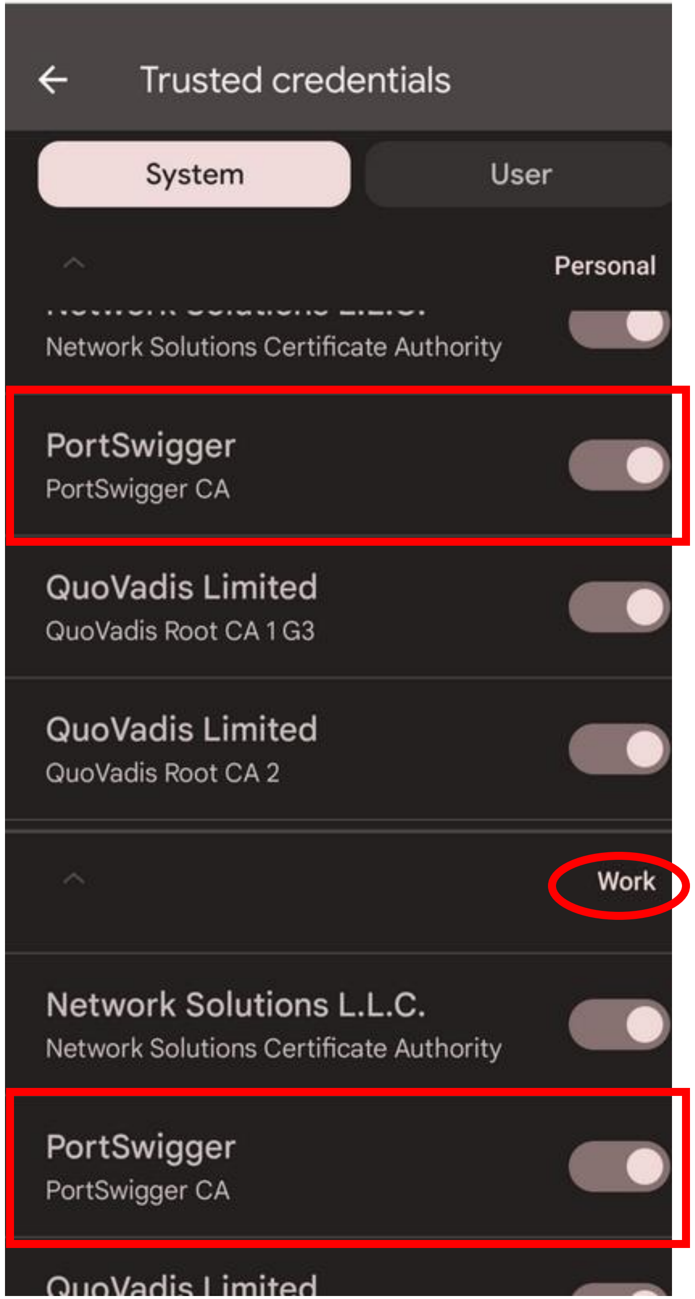
- By default, most Intents do not cross from one profile to the other.
- File URI that is valid on one profile is not valid on the other. (Since the storage areas are separate)
- No longer valid if non-work apps are not detected via the EMM application.
- Eg- A non-approved file manager has access to the Documents/Downloads directory for the work profile apps.



```
redfin:/mnt/user/11/emulated/11/Download # ls -lrt
total 12
-rw------- 1 u11_a256 u11_a256 8975 2022-09-26 12:53 Book.xlsx
```

Network Comms (Shared Cert Store)

- No Separate cert store for work profile!
- Once a system cert is installed at `/system/etc/security/cacerts/*`
- Unrestricted interception of HTTP(s) network calls for the work apps via network proxy.
  - Malware on the device
  - The end user



Trusted credentials

System | User

Personal

Network Solutions Certificate Authority

PortSwigger
PortSwigger CA

QuoVadis Limited
QuoVadis Root CA 1 G3

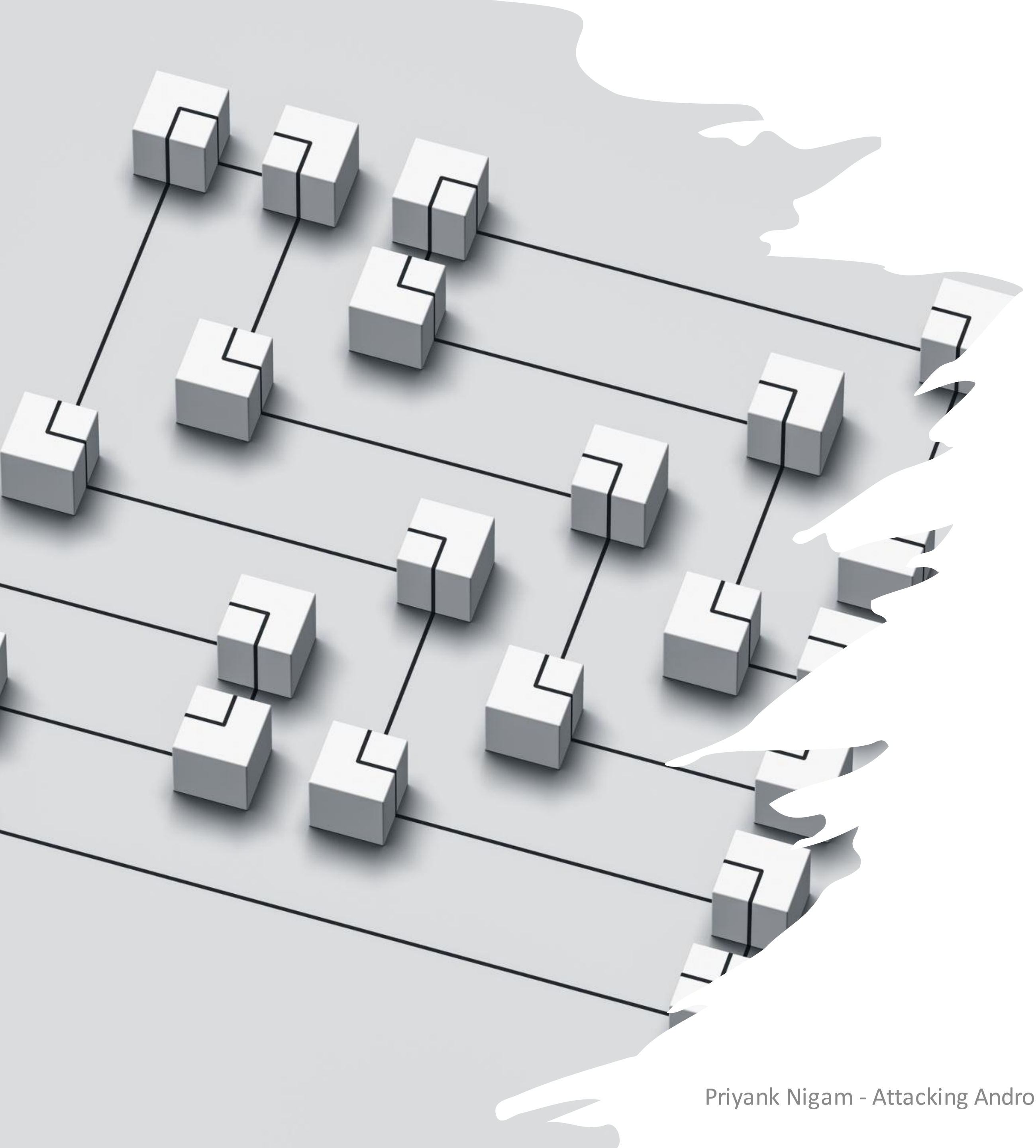
QuoVadis Limited
QuoVadis Root CA 2

Work

Network Solutions L.L.C.
Network Solutions Certificate Authority

PortSwigger
PortSwigger CA

QuoVadis Limited



Device Logs

- Shared streams of logs from both personal & work apps.
- Outside the control of EMM provider
- Work Apps logs sensitive data -> Exposes debug info and/or user’s PII.
- Inherited from Linux – By design
  - Except, android apps are much chattier within logs.

Connected Work Apps

If your organization allows it, you can enable some of these apps to share data and connect with themselves across your work and personal profiles.

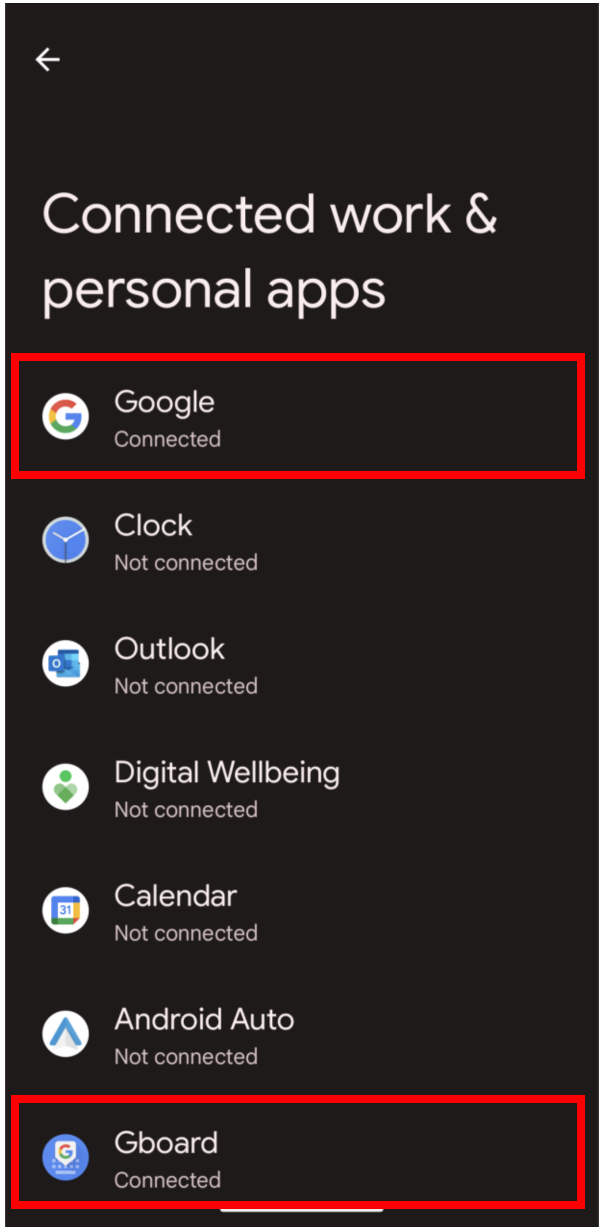
Eg - by connecting your calendar app you could view your work and personal events together.

Settings > Apps > Special app access > Connected work & personal apps.

(Enabled connected apps may share your personal data with your employer or other work apps)

They run under separate UIDs -

```
redfin:/data/data/com.google.android.googlequicksearchbox/shared_prefs # ps -A | grep search
u0_a190        7350    855 15824908 247772 do_epoll_wait      0 S
com.google.android.googlequicksearchbox:interactor
u11_a190      30059    855 15209848 292760 do_epoll_wait      0 S
com.google.android.googlequicksearchbox:search
u11_a190      30281    855 14823284 186408 do_epoll_wait      0 S
com.google.android.googlequicksearchbox:interactor
u0_a190       30621    855 32622652 369008 do_epoll_wait      0 S
com.google.android.googlequicksearchbox:search
```



Connected work & personal apps

Google
Connected

Clock
Not connected

Outlook
Not connected

Digital Wellbeing
Not connected

Calendar
Not connected

Android Auto
Not connected

Gboard
Connected

EMM App Security

Device Admin vs Profile Admin – Super ROOT !

- A Profile admin enforces work profile policies which cannot be bypassed even via root access.

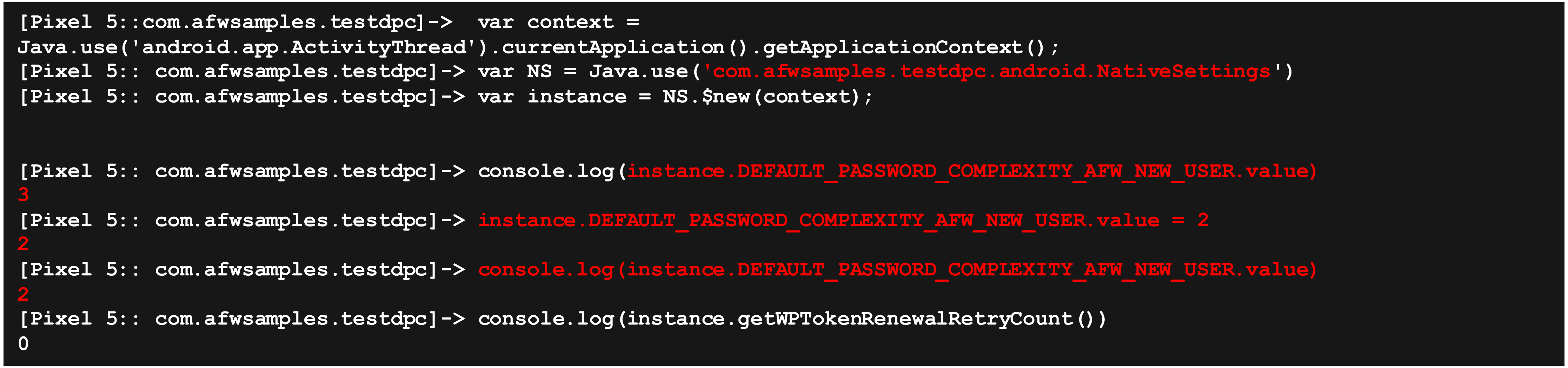
```
255|redfin:/ # dpm list-owners
2 owners:
User 10: admin=com.afwsamples.testdpc/.DeviceAdminReceiver,ProfileOwner
User 10: admin=com.afwsamples.testdpc/.DeviceAdminReceiver,ManagedProfileOwner(parentUserId=0)
```

Doesn't actually disable the lock screen!

```
redfin:/# locksettings clear --old 0000 --user 10
New credential doesn't satisfy admin policies: Weak credential type
redfin:/# locksettings set-disabled --old 0000 --user 10 true
Lock screen disabled set to true
```

Runtime Application Self-Protection (RASP)

- **Really** Strong Controls needed.
- Hijack flow using dynamic instrumentation such as Frida.
- Pretty much bypass ALL Enforced policies



```
[Pixel 5::com.afwsamples.testdpc]->  var context =
Java.use('android.app.ActivityThread').currentApplication().getApplicationContext();
[Pixel 5:: com.afwsamples.testdpc]-> var NS = Java.use('com.afwsamples.testdpc.android.NativeSettings')
[Pixel 5:: com.afwsamples.testdpc]-> var instance = NS.$new(context);


[Pixel 5:: com.afwsamples.testdpc]-> console.log(instance.DEFAULT_PASSWORD_COMPLEXITY_AFW_NEW_USER.value)
3
[Pixel 5:: com.afwsamples.testdpc]-> instance.DEFAULT_PASSWORD_COMPLEXITY_AFW_NEW_USER.value = 2
2
[Pixel 5:: com.afwsamples.testdpc]-> console.log(instance.DEFAULT_PASSWORD_COMPLEXITY_AFW_NEW_USER.value)
2
[Pixel 5:: com.afwsamples.testdpc]-> console.log(instance.getWPTokenRenewalRetryCount())
0
```

Other EMM "Features"

- Send logs for troubleshooting via email-> Breaks out of Work profile confinement.
- SSO Authentication BEFORE device enrollment?
- Chicken and egg problem
- Work profile is configured before device configuration check is performed. Potential abuse even if work apps are not setup. Rootkits can exfil data before the device policies go into affect.

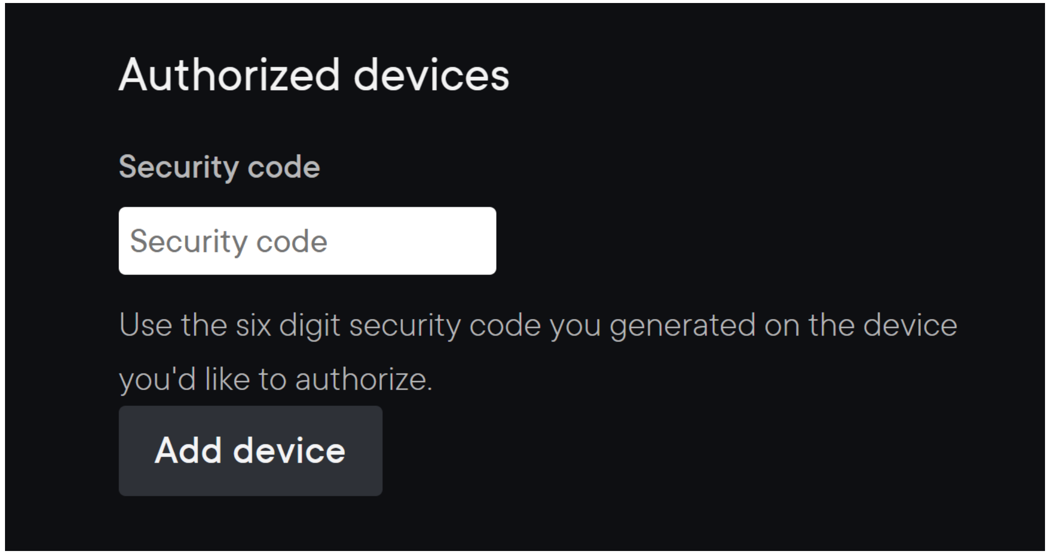
- Check for device integrity first!

General Application Security Considerations for work apps

- Add personal accounts on the work app (Eg- Outlook?)
- General OWASP Security Guidelines –
  - Sensitive data storage on device/memory (Eg- Auth tokens)
  - Network communications
  - Platform-specific security issues
  - Runtime Application Self-protection (RASP)

Work Apps - Alternate forms of authentication

- If work apps provide other ways of authentication (Eg- Linked devices), the Managed device requirement can be bypassed.
- Eg – [REDACTED] (Linked Devices)
- Bypasses MDM Restrictions



Authorized devices

Security code

Security code

Use the six digit security code you generated on the device you'd like to authorize.

Rootkits (or users with root access)

Can almost always evade detection.

Can they access work data?

Work profile unlocked – Yes

Work profile locked – It depends

File-based Encryption(FBE) Crash Course

Device Encrypted (DE) storage is accessible once the device boots, as well as after the user unlocks the device.

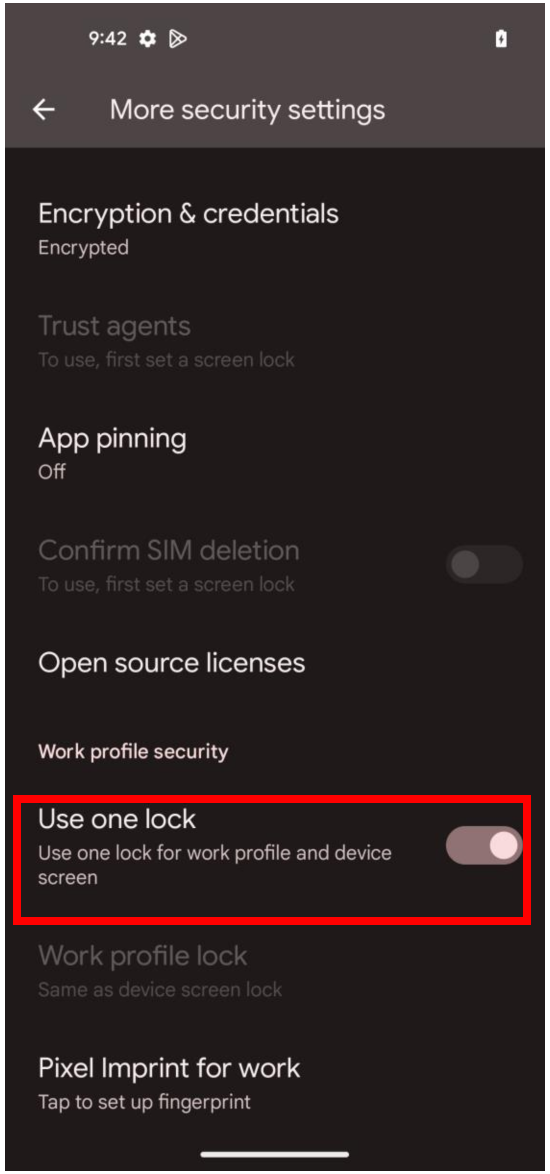
Credential Encrypted (CE) storage is only available after the user enters their credentials and unlocks the device

A Work profile runs under a separate user (typically user id 10)

One lock for both profiles is enabled by default

- When the user unlocks the device, the work profile is unlocked as well

```
Installed fscrypt key with ref b97bdcc9717116964cd59ab93cb13e49 to
/data
vold     : Added fscrypt-provisioning key for
b97bdcc9717116964cd59ab93cb13e49 to session keyring
vold     : Installed ce key for user 10
```



More security settings

Encryption & credentials
Encrypted

Trust agents
To use, first set a screen lock

App pinning
Off

Confirm SIM deletion
To use, first set a screen lock

Open source licenses

Work profile security

Use one lock
Use one lock for work profile and device screen

Work profile lock
Same as device screen lock

Pixel Imprint for work
Tap to set up fingerprint

Pause Work profiles

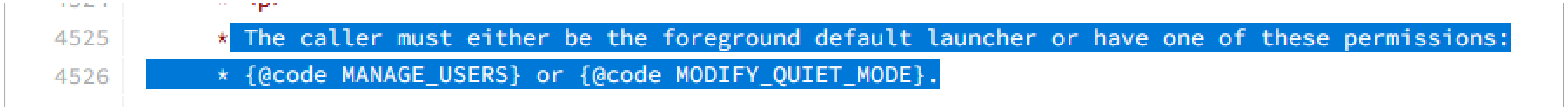
- Provision to lock the work profile by turning it “Off”
  - Deletes the CE Key
  - Thus, the directories for that user is encrypted
  - Can we still get access with root? – Yes, but not by directly accessing the keys!

```
redfin:/data/user/10 # ls
+dd9VDAAAAQpXFsgeQCzxnPofY5mxGlL1NXhuP5ta4E,OH87JpeV2+fNnplEazKwxz9yimaRoCF
+lMOQCAAAAQpXFsgeQCzxnPofY5mxGlLKPnSrPoeNj2qKOdqFmrsq4CawXG+XKaY6WFOOQNCDBf5+s+AVOz2Q1upNuu3p+Lz
+q9mqDAAAAQpXFsgeQCzxnPofY5mxGlLEB27Xll2d018NdW6CTaLz9e7Fe2h28e09R7iWnB5n,G
+upC8CAAAAQcQUfoobRF+yFZooBLOOVsleVwC6BJMHf+g+hlbGbUuA
,1PB4DAAAAwpsx,vZXJk8BIH6oA+3mf0uaCQFEHbPsH7B2Lj1344BzNcF,ZNIxHT1QPuxr1lNrK
,JULgCAAAAQ+AzX,IEVpg1BYbZ0vp4g5FuPv0yxQN70MDSM83kGlqA
,VFVDCAAAAQpXFsgeQCzxnPofY5mxGlLPisRa+BTQtw2l2aU,Rhf3+fNnplEazKwxz9yimaRoCF
,YFETCAAAAQpXFsgeQCzxnPofY5mxGlLZ0F8rhDmCzh3jYUMTe8n2QdYO,NT9KYNkD8c5h,E74F
,bN7aBAAAAwkmvPSVX0KGxnVZ,R1A60UleVwC6BJMHf+g+hlbGbUuA
,eNKzAAAAAA5OCEHsb5rsBlnPRnJZC0snXzAjdr5dqYgVSvBB3z0QA
0GjBwCAAAAAfKV8kZkg6KZRbOP5+snzAleVwC6BJMHf+g+hlbGbUuA
```

Trace the calls

- DE Key for user is already installed, but CE key is not.
- LockSettingsService -> StorageManagerService -> Vold -> Install CE key

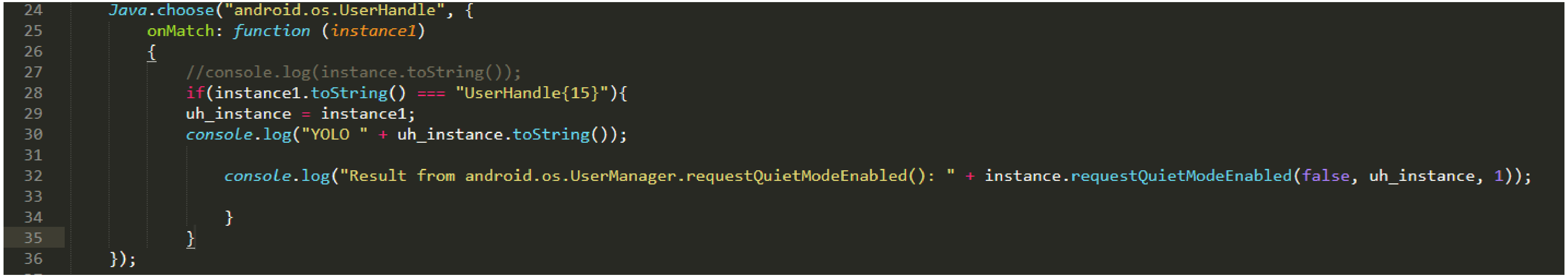
- Easier path -> Call [requestQuietModeEnabled()]()



```
     * <p>
     * The caller must either be the foreground default launcher or have one of these permissions:
     * {@code MANAGE_USERS} or {@code MODIFY_QUIET_MODE}.
```

Get hold of the launcher!

Hook the launcher process to initiate the CE key installation -> Works!



```
    Java.choose("android.os.UserHandle", {
        onMatch: function (instance1)
        {
            //console.log(instance.toString());
            if(instance1.toString() === "UserHandle{15}"){
            uh_instance = instance1;
            console.log("YOLO " + uh_instance.toString());

                console.log("Result from android.os.UserManager.requestQuietModeEnabled(): " + instance.requestQuietModeEnabled(false, uh_instance, 1));

                }
            }
    });
```

Even in the **background** state, when the device is **locked** – Reported to Google -> “Not an Issue”.

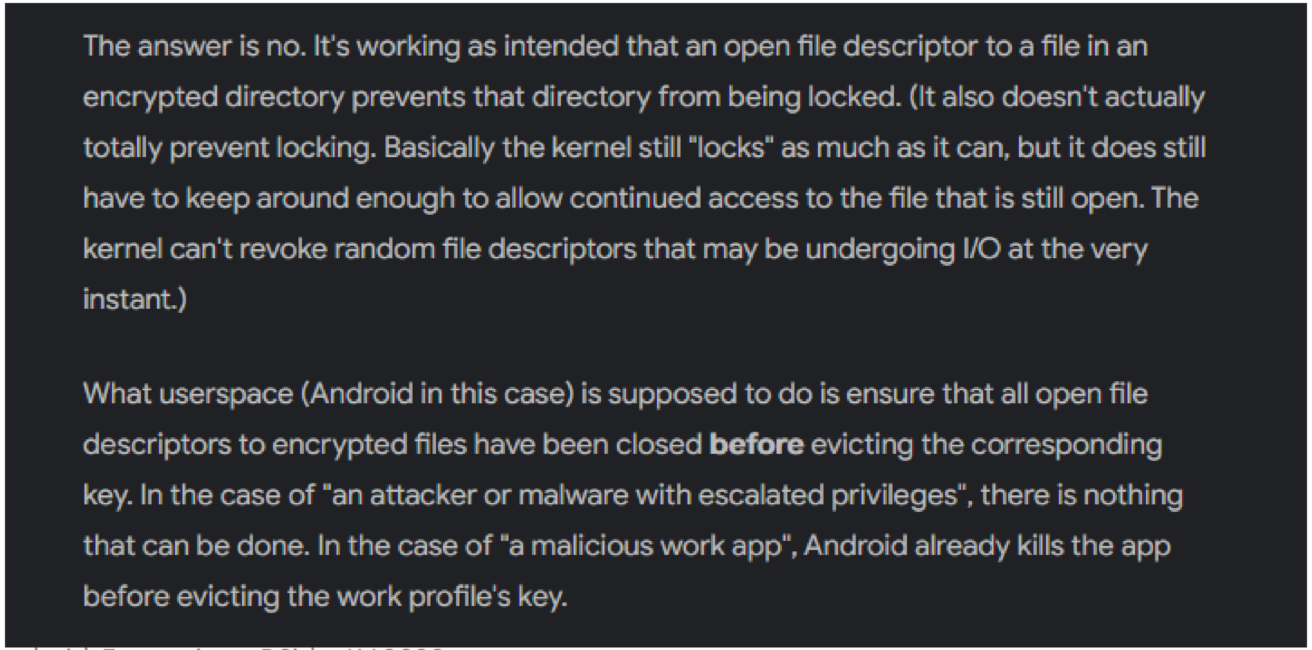
Reason? *“**foreground** default launcher does not mean the launcher has to be in the **foreground**”* !!!

I don’t know what it means then!

Published later at https://github.com/priyankn

Rootkits – Prevent actual locking

- Device is reported as compromised and the admin initiated a “lock”
- Get handle on work profile /data dir
- Note that the FBE for metadata does not have any affect on the top level directory .
- Just the way fscrypt works.
- Reported to Google – Cant fix



The answer is no. It's working as intended that an open file descriptor to a file in an encrypted directory prevents that directory from being locked. (It also doesn't actually totally prevent locking. Basically the kernel still "locks" as much as it can, but it does still have to keep around enough to allow continued access to the file that is still open. The kernel can't revoke random file descriptors that may be undergoing I/O at the very instant.)

What userspace (Android in this case) is supposed to do is ensure that all open file descriptors to encrypted files have been closed **before** evicting the corresponding key. In the case of "an attacker or malware with escalated privileges", there is nothing that can be done. In the case of "a malicious work app", Android already kills the app before evicting the work profile's key.

If separate locks for parent and work profiles is enabled

- Observation – the CE key is installed on first unlock of the work profile
- The work apps are still running in background.
- If a user on default profile has managed to escalate their privileges, they can still attach a debugger (eg Frida) to the running work apps and interact with them without knowing the work profile unlock key.
- **Threat** – If the device with a weak/no device password but a strong work profile password is stolen, the attacker can still interact with the work apps and access sensitive data (either via adb or via a debugger)
- Fix for a related issue in progress.
- Conclusion – **Don't** enforce **only** work passcode, or specify weaker requirements for device passcode.

Work Profile Lock vs Device lock – Know the difference

- Set password complexity on parent profile vs just the work profile (device owner vs profile owner)
- For profile owners, ACTION_SET_NEW_PASSWORD prompts the user to set a work challenge, and ACTION_SET_NEW_PARENT_PROFILE_PASSWORD prompts the user to set a device lock.

Work lock enabled – Can you interact with the work apps?

- # am start --user 10 -d instagram://user?username=instagramPageName

```
BiometricService: handleAuthenticate: modality(1), status(0), preAuthInfo: BiometricRequested: true, StrengthRequested: 255, CredentialRequested:
true, Eligible:{}, Ineligible:{ID(0), oemStrength: 15, updatedStrength: 15, modality 2, state: 0, cookie: 0:7 }, CredentialAvailable: true, requestId: 45
promptInfo.isIgnoreEnrollmentState: false

ActivityTaskManager: Displayed com.android.settings/.password.ConfirmDeviceCredentialActivity$InternalActivity: +53ms
```

When the phone is locked:

```
W ActivityManager: Background start not allowed: service Intent { act=Orca.START
cmp=com.instagram.android/com.facebook.rti.push.service.FbnsService (has extras) } to
com.instagram.android/com.facebook.rti.push.service.FbnsService from pid=22518 uid=1010242 pkg=com.instagram.android startFg?=false
```

Not Interactively!

But probably there is no need, if you already have access to the filesystem steal the token and interact with the APIs

Disable App ?

- An end-user or a malicious app on the user's device can effectively disable a work app whose uninstallation is effectively blocked by the IT admin's policies
- DPM can enforce this via setUninstallBlocked()
- Reported to Google – They expect USB Debugging to be disabled and marked it as “**Won’t fix**”

```
pm disable --user 10 com.reddit.frontpage

Exception occurred while executing 'disable':

java.lang.SecurityException: Cannot disable a protected package: com.reddit.frontpage at
com.android.server.pm.PackageManagerService.setEnabledSettings(PackageManagerService.java:3
711)
```

This Doesn’t work!

..but this does!

```
$ pm disable-user --user 10 com.reddit.frontpage
Package com.reddit.frontpage new state: disabled-user
```

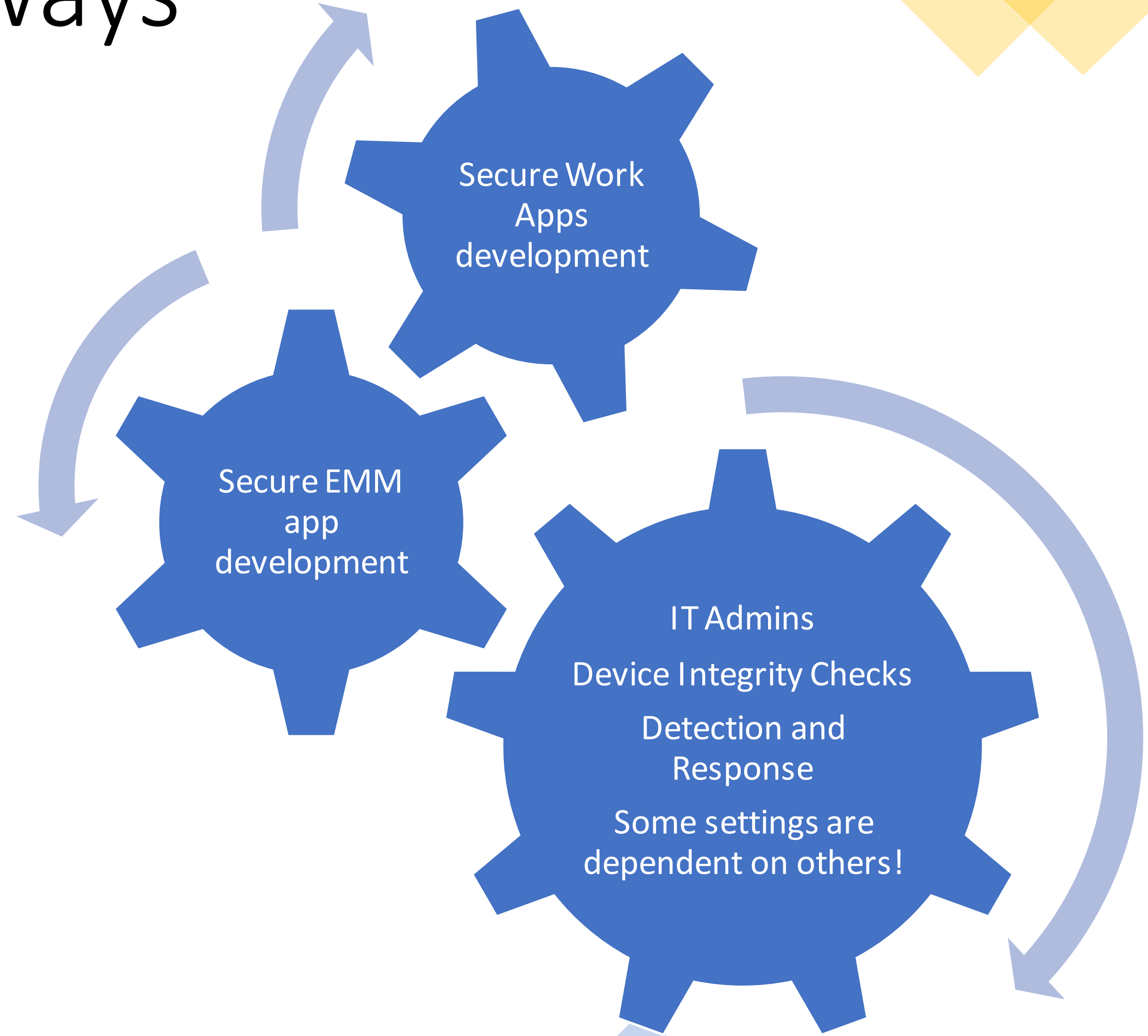
[Privacy] Is ANDROID_ID[1] queried? (Android 8.0 and less)

- A 64-bit number (expressed as a hexadecimal string), unique to each combination of app-signing key, user, and device
- The value may change if a factory reset is performed on the device or if an APK signing key changes.
- Post Android 8.0, apps with different signing keys running on the same device no longer see the same Android ID (even for the same user)

```
console.log(instance.getMAMSafetyNetAndroidID(context))
6233513b4fXXXX
```

https://developer.android.com/reference/android/provider/Settings.Secure.html#ANDROID_ID

Mitigations & Takeaways



Secure Work Apps development

Secure EMM app development

IT Admins

Device Integrity Checks

Detection and Response

Some settings are dependent on others!

Defaults

Policy	Microsoft Intune	Samsung knox Manage	Ivanti MobileIron
USB Debugging	Disabled (Enabled by default in Level 2 Enhanced security mode)	Allowed[1]	Enabled
Detect Rooted Devices	Not Configured	Not set	Not Configured
Google Play Integrity Protect	Not configured		
Require antivirus, antispyware, and antimalware	None	None	None
Microphone Access by work apps	Disabled (i.e. prompt user)	Not set	Enabled
Minimum security patch level	Not configured	Not set	None (5.0+)

https://learn.microsoft.com/en-us/mem/intune/enrollment/android-work-profile-security-settings

https://docs.samsungknox.com/dev/knox-sdk/wpc-knox-apis-allowed.htm

https://help.ivanti.com/mi/help/en_US/core/11.0.0.0/gsg/Content/CoreGettingStarted/Lockdn_Android%20and%20AE%20devices.htm

Questions?

@Rev_Octo

@priyankn