ROMs

- Redmi Note 8
- Sauvegarde des cellulaires
- ADB Configuration script
- Play Integrity/SafetyNet

Redmi Note 8

Codename: gingko

```
To find product name : fastboot getvar product
```

Telegram groups:

t.me/rn8discussions
t.me/RedmiNote8Official

Forum XDA

https://forum.xda-developers.com/redmi-note-8

TWRP

https://forum.xda-developers.com/redmi-note-8/how-to/guide-unlocking-bootloader-twrp-rooting-t4031831

- TWRP Official : https://forum.xda-developers.com/t/recovery-official-twrp-teamwin-recovery-redmi-note-8.4301393/

Unlocking phone

https://forum.xda-developers.com/t/guide-all-about-unlocking-bootloader-twrp-and-rooting-redmi-note-8-8t.4031831/

USB Debugging (with security)

This requires root, shell access with su, and a way to edit text files as root. I used Termux and and android root file manager that can edit text files.

First, run the following command (su first, to switch to root):

```
setprop persist.security.adbinstall 1
```

If you also want to enable "USB debugging (Security settings)", for granting permissions, simulating UI input, etc. via ADB, run the following command too:

```
setprop persist.security.adbinput 1
```

Next, we need to edit the edit the file

```
/data/data/com.miui.securitycenter/shared_prefs/remote_provider_preferences.xml
```

so that MIUI actually lets us do these things:

Add the following line to the XML file (with all the other ones, before the closing tag):

```
<boolean name="security_adb_install_enable" value="true" />
```

TWRP

Was in TWRP 3.3.1-9
To install the correct TWRP version, check the "base" of the TWRP indicated after the version number. E.G. :
`twrp-3.6.2_9-0-ginkgo.img` is a base Android 9-0
`twrp-3.5.2_10-2-ginkgo.img` is a base Adndroid 10-2
`twrp-3.7.0_12-1-ginkgo.img` is a base Adndroid 12-1

Some Android versions cannot decrypt your data folder because of incompatibility issues. Simply downgrade to a lower Android base TWRP.

- Enter Fastboot to install latest TWRP

```
# To check current recovery tool (with phone powered up)
adb reboot recovery


# Boot to recovery
adb reboot bootloader
```

```
# Confirm correctly in fastboot
fastboot devices
# Install latest TWRP
cd "/home/wes/Documents/Cellulaires-ROMs/Redmi Note 8"
fastboot flash recovery twrp-3.7.0_12-1-ginkgo.img
# Shutdown phone then "Volume Up + Power" to enter recovery
fastboot reboot
```

First time installation

You might need to use Lineage OS recovery the first time to initialize a factory reset. Simply follow the instruction here : https://wiki.lineageos.org/devices/ginkgo/install/variant1#installing-a-custom-recovery-using-fastboot
Sideload the custom ROM rather than Lineage.

ArrowOS

- Boot in recovery
- Wipe data, cache, dalvik cache
- Flash latest android-11 firmware (Ginkgo | Willow)

  ```
  adb push Arrow-v13.1-ginkgo-OFFICIAL-20230829-GAPPS.zip /sdcard
  ```

- Flash ROM zip
- Format data (if encrypted)
- Reboot
- Configure system manually
- Install Magisk for root

  ```
  adb reboot recovery
  adb sideload Magisk.v26.3.apk
  ```

`No need to follow this, simply install by sideloading + app within OS`
Since Redmi Note 8 has no ramdisk, a special guide is required.

- Install the Magisk app from the official git repo.
- Push installed recovery locally : `adb push twrp-3.7.0_12-1-ginkgo.img /sdcard/Documents/`
- Execute Magisk app
- Press the Install button on the Magisk card.

- Enable "Recovery Mode" under Options
- Choose "Select and Patch a File" under Method, and select the local recovery image.
- Retrieve the patched image locally : adb pull /sdcard/Download/magisk_patched_[random_strings].img
- Reboot in fastboot and flash recovery : fastboot flash recovery /path/to/magisk_patched.img
- Configure Magisk
  - Settings/Systemless hosts
  - Settings/Zygisk
- Install Magisk Modules

  microG installer : https://github.com/nift4/microg_installer_revived

  safetynet fix : https://github.com/kdrag0n/safetynet-fix

  ```
  # microG
  adb push microG_Installer_Revived.zip /sdcard/Documents/
  # Google safetynet fix
  adb push safetynet-fix-v2.4.0.zip /sdcard/Documents/
  ```

- Install EasySSHFS

  https://github.com/bobrofon/easysshfs/releases
  adb push easysshfs-0.5.10-release-signed.apk /sdcard/Documents/
- Install neo-backup

  https://github.com/NeoApplications/Neo-Backup/releases adb push Neo_Backup_release_832.apk /sdcard/Documents/

Reconfigure some apps

Some apps require to re-login after restoration :

- Banks (Wise ; Desjardins)
- eMails (Proton ; Tutanota ; etc)
- Social apps (Instagram ; Viber ; Signal ; Telegram ; etc)
- Music (Deezer)
- xMatters

Some interesting modules

https://github.com/Magisk-Modules-Repo/ssh

Lineage OS

https://wiki.lineageos.org/devices/ginkgo/variant1/

Boot to Fastboot/bootloader

```
adb reboot bootloader
```

Flash vbmeta file (to disable boot security feature)

```
fastboot --disable-verity --disable-verification flash vbmeta vbmeta.img
```

Flash and boot to recovery TWRP

```
fastboot flash recovery twrp-3.3.1-9-ginkgo-mauronofrio.img
fastboot boot twrp-3.3.1-9-ginkgo-mauronofrio.img
```

Format Data

```
TWRP > Wipe > Format Data (not in advanced mode) - No reboot to system
```

Reboot in recovery

```
TWRP > Reboot > Recovery
```

Wipe again

```
TWRP > Wipe > Advanced > Cache, data, dalvik, system
```

Flash Disable_DM-Verity_ForceEncrypt

```
https://forum.xda-developers.com/android/software/universal-dm-verity-forceencrypt-t3817389/
```

Flash ROM

Flash Magisk

Reboot to ROM

Reboot to recovery

Generate and Install Smali Patcher module in Magisk

Flash nanodroid setup (to configure nanodroid) - doesn't works.

Flash nanodroid

Source:
https://forum.xda-developers.com/showpost.php?p=81403899&postcount=37

Install smali patcher for signature spoofing

https://forum.xda-developers.com/apps/magisk/module-smali-patcher-0-7-t3680053

Baseband

Version

Firmware	Baseband (BAR_GEN_PACK)
V11.0.8.0.PCOMIXM	1.252721.1.260983.1
V11.0.7.0.PCOMIXM	1.246073.2.252721.1
20.1.16 (China)	1.241207.1.242935.1
V11.0.6.0.PCOMIXM	1.239860.1.241207.1
V11.0.3.0.PCOMIXM	1.223797.1.225109.3

Sauvegarde des cellulaires

Sauvegarde cellulaires

https://sebsauvage.net/wiki/doku.php?id=backup-telephone

https://alexandre.deverteuil.net/post/synchronizing-files-across-phone-laptop-nas-with-nextcloud-and-syncthing/

- SMS : https://play.google.com/store/apps/details?id=com.idea.backup.smscontacts
- Apps : https://f-droid.org/en/packages/com.machiav3lli.backup/
- Accès au systèmes de fichiers :
  - EasySSHFS : https://f-droid.org/en/packages/ru.nsu.bobrofon.easysshfs/
  - primitive ftpd : https://f-droid.org/fr/packages/org.primftpd/

Rapport

Valider le contenu du stockage:

```
# Vue globale du stockage
df -h
cd /data/
du -m |  awk '$1 >= 100' |  sort -g # List directories above 100 MB
du -am -d1 |  awk '$1 >= 100' |  sort -g # List all above 100 MB at depth level 1
du -am -d2 |  awk '$1 >= 100' |  sort -g # List all above 100 MB at depth level 2
```

Étapes

```
# Pré-requis - récup du cell id
adb devices -l
```

1. Backup des apps avec neobackups
2. Backup Signal
   Settings/Chats/Chats backup/Create backup
   (passphrase in keepass)

   ```
   adb -s <cellulaire id> pull -a /sdcard/Documents/Backup_signal ~/Documents/sauvegarde
   ```

3. Backup Markor

   ```
   adb -s <cellulaire id> pull -a /sdcard/Documents/markor ~/Documents/sauvegarde
   ```

4. Sauvegarde des settings

   ```
   mkdir /sdcard/Documents/backup_settings
   su - # passe en root
   cp /data/system/users/0/* /sdcard/Documents/backup_settings/
   ```

5. Copie local de la sauvegarde

   ```
   adb -s <cellulaire id> pull -a /sdcard/backup_neobackup ~/Documents/sauvegarde
   ```

6. Copie de la sauvegarde vers le nouveau cell

   ```
   adb -s <cellulaire id> push ~/Documents/sauvegarde/backup_neobackup /sdcard/
   ```

7. Installation des apps minimums

   ```
   adb -s <cellulaire id> install ~/Documents/Cellulaires-ROMs/magisk-v28.apk
   adb -s <cellulaire id> install ~/Documents/Cellulaires-ROMs/droid-ify-v0.6.3.apk
   adb -s <cellulaire id> install ~/Documents/Cellulaires-ROMs/auroraStore-4.6.2.apk
   adb -s <cellulaire id> install ~/Documents/Cellulaires-
   ROMs/neo_Backup_838_release.apk
   ```

8. Restauration des apps via Neobackup
9. Configuration des connexions pour les apps
   - Signal
   - Viber
   - Whatsapp
   - Facebook (Messenger)
   - Desjardins
   - Wise
   - Picard
   - Proton Mail
   - Proton VPN
   - Tuta
   - Manuvie
   - ...etc...

Configuration EasySSHFS

wes
192.168.66.103
remote path : ~/Documents/sauvegarde
(create folder in Android home named "sauvegarde")

Ressources

https://github.com/newhinton/Round-Sync
Outil pour sync 1-direction basé sur rclone

ADB Configuration script

ADB Config

Check logs using logcat

- Find name of app
- filter with the name

```
$ pm list packages |  grep -i seed
package: com. stevesoltys. seedvault
$ logcat -e seedvault
# or use a simple grep
logcat |  grep ' vault'
```

```
# This is specific to SeedVault (https: //github. com/seedvault-
app/seedvault/blob/android15/logcat- verbose. sh)
set -ex

adb shell setprop log. tag. BackupManagerService VERBOSE
adb shell setprop log. tag. BackupManagerConstants VERBOSE
adb shell setprop log. tag. BackupTransportManager VERBOSE
adb shell setprop log. tag. KeyValueBackupJob VERBOSE
adb shell setprop log. tag. KeyValueBackupTask VERBOSE
adb shell setprop log. tag. TransportClient VERBOSE
adb shell setprop log. tag. BackupAgent VERBOSE
adb shell setprop log. tag. PMBA VERBOSE
```

Navigate through the slots

To flash permanently, first switch/select any slot you want or it will **automatically flashed to the active slot**.

```
# Check current active slot
adb shell getprop ro.boot.slot_suffix
fastboot getvar current-slot

# Switching slots
fastboot set_active other  //switch to inactive slot
fastboot set_active (a or b)  //switch to specified slot
```

Check current slot and baseband version

```
adb shell getprop |  grep -Ei '(gsm.*baseband|slot_suffix)'
```

Have both slots (A/B) with same verity/verification configuration

```
fastboot flash vbmeta --disable-verity --disable-verification --slot=all vbmeta.img
```

Secure phone (to review)

```
#! /bin/bash
adb root
adb remount
adb pull /system/build.prop .
sed -i "s/ro.secure=0/ro.secure=1/" build.prop
sed -i "s/ro.debuggable=1/ro.debuggable=0/" build.prop
adb push build.prop /system/build.prop
rm build.prop
```

Play Integrity/SafetyNet

Play Integrity

- ROM dummps (for vendor data file) : https://dumps.tadiphone.dev/dumps