FODA to Android 9 integration guide

1. Development environment pre-requisites.

1. Download and install the Ubuntu 16.04 LTS.
2. Prepare at least the 250 GB of the free memory and 16 GB of the RAM.
3. Download FODA source code
4. Install the adb and fastboot utilities

2. Development environment setup

1. Install the required packages

   ```
   sudo apt-get install git-core gnupg flex bison gperf build-essential zip curl zlib1g-dev gcc-multilib g++-multilib libc6-dev-i386 lib32ncurses5-dev x11proto-core-dev libx11-dev lib32z-dev libgl1-mesa-dev libxml2-utils xsltproc unzip
   ```

2. Install the Repo utility
   - Create an empty directory to hold your working files. Give it any name you like:

     ```
     mkdir WORKING_DIRECTORY
     cd WORKING_DIRECTORY
     ```

   - Download the Repo tool and ensure that it is executable:

     ```
     curl https://storage.googleapis.com/git-repo-downloads/repo >repo
     chmod a+x repo
     ```

3. Download the branch with the Android 9 GSI
   - Init repo to bring down the needed branch specified with `-b` (for a list of branches, see Source Code Tags and Builds):

     ```
     repo init -b pie-gsi -u https://android.googlesource.com/platform/manifest
     ```

   - To pull down the Android source tree to your working directory run

     ```
     ./repo sync -c  -j 4 --no-tags --no-clone-bundle
     ```

   - The Android source files will be located in your working directory. The initial sync operation will take an hour or more to complete.
4. Prepare shell environment:

```
. build/envsetup.sh
lunch aosp_arm64_ab-userdebug
```

3. FODA integration

Install FODA source code

1. Copy FODA sources from `app/src/main` to `WORKING_DIRECTORY/packages/apps/FODA`.
2. Add app name "FODA" to "PRODUCT_PACKAGES" variable of `WORKING_DIRECTORY/build/target/product/core.mk` file.
3. Change `AndroidManifest.xml`: add `android:sharedUserId="android.uid.system"` to the `<manifest>` tag in header
4. Copy `app/sharedLibs` to `WORKING_DIRECTORY/packages/apps/FODA`.
5. Add `Android.mk`.
6. Check paths in `ConfigUtils.java`

Overcome SELinux problems

SELinux could block the access of FODA to the various folders and places set in the `ConfigUtils.java`. Overcome this may take a lot of time and vary from the system-to-system basis.

Example - MEMOR20 device

1. In Datalogic MEMOR20 Q10 device the problem appears with storing of the FODA's data into the persist folder (`/mnt/vendor/persist/misc/com.friendly.foda`). As it starts from `/mnt/vendor` - it is treated by SELinux as a vendor-related dir.
2. As FODA needs a high-level of permissions, it should be run by the system user and signed by platform key. Due this, FODA has been treated by SELinux as a "system_app".
3. Starting from Android 9 SELinux security rules ("neverallow") deny writing to the vendor-related dirs for system apps.
4. To overcome this issue and give the FODA access to the persistent storage we use the workaround: create a separate dir (`/mnt/persist/misc`) and bind mount `/mnt/vendor/persist/misc/com.friendly.foda` dir into it during the boot stage.
5. To implement this, the next files have been modified:
   - `init.target.rc` - startup script, performs creation of the folders and bind mount
   - `android/device/qcom/sepolicy/vendor/common/vendor_init.te` - allow rules for startup script
   - `android/device/datalogic/datalogic-common/sepolicy/system_app.te` - allow rules for FODA

4. Building firmware image from source code

1. Build the Android 9 GSI code - it may take up to 12 hours: `USE_CCACHE=1 CCACHE_DIR=ccache make -j 4`
2. To build only FODA use: `USE_CCACHE=1 CCACHE_DIR=ccache make FODA`
3. **Warning!** Watch out that GSI branch builds only three images: vbmeta-, cache- and system.img files So, if you need work on other partitions, yo may to download the it from https://developers.google.com/android/images

5. Flashinf firmware into device

1. Connect the Pixel 3XL the PC via the USB cable
2. Prepare the device for flashing
    1. *Enable the bootloader's access and USB-debug mode* In Settings, tap About phone, then tap Build number seven (7) times. When you see the message `You are a developer`, tap the back button. Tap Developer options and enable OEM unlocking and USB debugging.
    2. *Enable the USB connection between the PC and the Android* Connect/reconnect the USB to the Android and switch the transfer mode to enable the files transfer and confirm the connection to the PC.
    3. *Send the device into the fastboot mode* In the command prompt: `adb reboot bootloader` or Press and hold Volume Down, then press and hold Power, during the cold boot - for the PIXEL 3XL crosshatch.
3. Unlock the bootloader In the command prompt: `fastboot flashing unlock` On the smartphone - using the volume keys tap to the Unlock bootloader and press power button to confirm, if not - try to restart the bootloader.
4. To flash a device with a GSI system image: Use the sudo privilege and be sure that system.img and vbmeta.img files are in `cd ~/out/target/product/generic_arm64/` folder, then use:

```
    fastboot flash system system.img
    fastboot flash vbmeta vbmeta.img
    fastboot -w reboot
```