Patch rom Developer Guide

1. Introduction

This tutorial is suit for the customization base on original factory rom. Patchrom doesn't require the android source code to build. But it will be better to understand and have the android source code for reference. See <http://source.android.com/source/index.html> for how to get and build the source code. Before we start to customize Lewa rom we need to set up development environment.

2. Setup Develop Environment

2.1 OS

You can run patchrom project on Windows, Linux and Mac OS. However, we recommend to use Ubuntu (10.10 version or above is fine).

2.2 Set up Android SDK

Refer to <http://developer.android.com/sdk/installing.html> for how to install Android SDK.

After ANDROID SDK installation is complete, modify the PATH environment variable to

Provide access to SDK's tools/and platform-tools.

Edit your~/.bash\_profile or ~/.bashrc file. Look for a line that sets the PATH environment variable and add the full path to the tools/and platform-tools/directories to it. If you don't see a line setting the path, you can add one:<sdk>is your android SDK installation directory.

Export PATH=${PATH}:<sdk>/tools:<sdk>/platform-tools

3. Synchronous Lewa code

Create a patchrom project：

$ mkdir patchrom

Install repo：

$ mkdir ~/bin

$ vim ~/.bashrc

Set path:

PATH=~/bin:$PATH

$ curl https://dl-ssl.google.com/dl/googlesource/git-repo/repo > ~/bin/repo

$ chmod a+x ~/bin/repo

Switch to patchrom catalog：

$ cd patchrom

$ repo init -u git://github.com/LeWaCode/patchrom.git -b jellybean

$ repo sync

4. Pathrom project

Now, let me introduce the directory structure and usage for pathrom project.

Those files in android folder are the smail files which modified by Lewa : framework.jar.out, android.policy.jar.out, services.jar.out. There is a one-to-one correspondence between those three files and files in google-framework folder. Google-framework is made by aosp（google original code）without any change。

 build：Some compile script, none change recommends.

 lewa：There are three sub catalogs in this folder: data is used to deposit the pre-installed applications； The system catalog in HDPI folder store all the files compiled by lewa code, and those files are HDPI resource files which you will need when you modifies Lewa rom; Src restore overlay resource.

 tools：This folder includes all the scripts and tool applications. You will need those applications when compiling.

5. Familiar with your device

The first step is to familiar with your device, includes how to flash ROM, how to root and try some ROMs aimed at your device. The main work of this step is to surf some android forums

Which have rich information for your device. Here is a forum you definitely should look at: http://forum.xda-developers.com/

6. Choose suitable ROM

After getting acquainted with your device, be prepared to choose a suitable ROM for patchrom project. How to decide if a ROM is suitable, here are some guidelines:

(1) Android version must be in minor difference.

(2) Stock ROM

The stockroom which is released by device manufacturer is preferred. Therefore, the stock ROM is the most stable rom.

(3) Root

Generally speaking there are 2 kinds of root: root privilege and kernel root. The root privilege means that will be a Superuser app which will pop up a dialog to ask if you approve the request when other apps require root privilege. The kernel root means you can directly manipulate this device in root privilege via adb. The kernel root is preferred.

(4) Recovery

Since patchrom generates a recovery-ROM, so make sure that your device can enter recovery mode. The CWM recovery and ext4 recovery are good choices.

7. Generate base recovery-ROM

Firstly, we should create a standalone directory for the device. Let's use an imaginary device called xblade and assume current directory is patchrom.

$.build/envsetup.sh

$mkdirxblade

$cdxblade

Secondly, we will run a tool to automatically generate a recovery-ROM from the running device. You may ask” why we bother to generate a recovery-ROM if we already find a suitable ROM?” There are two reasons:

(1) Sometimes we can't find a stock recovery-ROM. But we requires a recovery-ROM as the basis.

(2) Most importantly, running this tool first is critical to automatically generate Lewa recovery-ROM and incremental OTA which makes lewa so special.

Now connect your device to PC, ensure the USB Debug mode is on.

$adb reboot recovery (if this command doesn't work, manually reboot into recovery mode)

$../tools/ota\_target\_from\_phone -r (-r means run this tool in recovery mode, we can also run this tool in normal mode, please refer to the source of this tool for more information)

This tool will generate a stockrom.zip and a metadata directory which will be used to generate our Lewa recovery-ROM.

8. makefile

We need a makefile to let make work. Here is a makefile to begin with:

local-zip-file :=stockrom.zip

local-out-zip-file :=

local-miui-modifed-apps :=

local-modified-apps :=

local-miui-removed-apps :=

local-phone-apps :=

local-pre-zip :=local-zip-misc

include$(PORT\_BUILD)/porting.mk

local-zip-misc:

We will explain some definitions in this file, for thorough understanding, you definitely should look at the whole build system.

* local-zip-file: the recovery-ROM generated in previous section, must be specified for each device.
* local-out-zip-file: the output recovery-ROM file name when you run "make zipfile"
* local-lewa-modified-apps: all the lewa apps are listed in the file patchrom/build/lewaapps.mk. This variable defines those lewa apps which we modified.
* local-modified-apps: This variable defines those apps from local-zip-file which we modified.
* local-lewa-removed-apps: Normally not all the lewa apps are suitable for our device(e.g. some
* device will not want Phone.apk).This variable defines those lewa apps which we don't want.
* local-phone-apps: This variable is used to remove some apps from local-zip-file. As you can see, we use blacklist for lewa apps and whitelist for stock ROM apps. This variable defines those stock ROM apps which we want to keep.
* local-zip-misc: This target permits you to do some customization before we generate the final recovery-ROM.

9. Workspace

The first make command we will run is "make workspace". This will prepare the workspace for you. The operation is simple; it will extract framework/android.policy/services.jar and framework-res.apk from stockrom.zip and run apk tool to disassemble them.

10. First patch

The second make command we will run is "make firstpatch". Before diving into this command, we need to talk about smali.

We use apk tool to disassemble the jar/apk which generates smali code. We will use framework.jar.out(which is the disassembled smali code from framework.jar) as an example to discuss this:

There are 3 smali code directories involved:

(1) old framework.jar.out

(2) new framework.jar.out

(3) target framework.jar.out

At first time, the old framework.jar.out will be the disassembled framework.jar which is compiled from google released source code, the new framework.jar.out will be the disassembled framework.jar which is compiled from lewa's source code. The target framework.jar.out will be the disassembled framework.jar from target device which is already prepared in "make workspace".

In order to facilitate this process, we try the best to make minimum change to the google released source code. You can see these changes by compare lewa/src and android/src. But most often there will be some conflicts when applying the patch.

After running this command, there will be a temp directory. There will be 5 child directories:

(1) old\_smali: the smali code from old framework.jar.out with .line removed

(2) new\_smali: the smali code from new framework.jar.out with .line removed

(3) dst\_smali\_orig: the smali code from target framework.jar.out with .line removed

(4) dst\_smali\_patched: the smali code after applying the patch into target framework.jar.out with .line removed

(5) reject: the rejected patch. We should use this directory to resolve the conflicts.

You may raise a question: what if lewa makes further change to the source code, how can we track these changes.

Firstly, please record the newest commit number of android/framework.jar.out when you run "make firstpatch". Assume the commit number is abcdefg.

Then at some time, the android/framework.jar.out will be updated and assume the newest commit number is hijklmn. Run "tools/git.patchabcdefg..hijklmn>patch.file".

Finally at your device directory, run "tools/git.apply<../android/patch.file". Now you can resolve any conflict.

11. fullota

After resolved the conflicts, you can run "make fullota" to build the final recovery-ROM. This command will generate fullota.zip under out directory. Flash this file in recovery mode. Now your device may boot successfully and see the Lewa lockscreen. However, most often things are not going so well, don't worry."adb logcat" will be the great tool to help us. From now on, there will be a problem-solving loop. Solve the problem or exception we encountered.