|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | |  |
|  | | |  |
|  | | |
|  |  | |
|  |  | |
| **user版本下的各种调试方法修改**   |  | | --- | | 文档版本 V1.0 | | 发布日期 2019-12-11 | | | | | |
|  |  | |  |  |





紫光展锐科技有限公司

请参照交付物中说明文档对紫光展锐交付物进行使用，任何人对紫光展锐交付物的修改、定制化或违反说明文档的指引对紫光展锐交付物进行使用造成的任何损失由其自行承担。紫光展锐交付物中的性能指标、测试结果和参数等，均为在紫光展锐内部研发和测试系统中获得的，仅供参考，若任何人需要对交付物进行商用或量产，需要结合自身的软硬件测试环境进行全面的测试和调试。非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

本文件所含数据和信息都属于紫光展锐所有的机密信息，紫光展锐保留所有相关权利。本文件仅为信息参考之目的提供，不包含任何明示或默示的知识产权许可，也不表示有任何明示或默示的保证，包括但不限于满足任何特殊目的、不侵权或性能。当您接受这份文件时，即表示您同意本文件中内容和信息属于紫光展锐机密信息，且同意在未获得紫光展锐书面同意前，不使用或复制本文件的整体或部分，也不向任何其他方披露本文件内容。紫光展锐有权在未经事先通知的情况下，在任何时候对本文件做任何修改。紫光展锐对本文件所含数据和信息不做任何保证，在任何情况下，紫光展锐均不负责任何与本文件相关的直接或间接的、任何伤害或损失。

**版权所有 © 紫光展锐科技有限公司2019。 保留一切权利。**

————————————————————————————————————————————————————————————————————————————————————

—————————————————————

前 言

概述

为了在user版本下获得更多的调试信息.这里做一个统一的整理,相关的修改包括:

1.adb root；

2.打开user下的corefile；

3.默认打开user下的ylog；

4.将logcat输出到串口；

5.打开user下的sysdump；

6.打开hprof保存。

7. user 打开版本打开sprdruntimeinfo

8. user版本下导出dropbox修改意见

读者对象

稳定性调试

缩略语

|  |  |  |
| --- | --- | --- |
| 缩略语 | 英文全名 | 中文解释 |

符号约定

在本文中可能出现下列标志，它所代表的含义如下。

**（本模块“可选”，由各个写作模板单独定义。）**

|  |  |
| --- | --- |
| 符号 | 说明 |
|  | 用于突出重要/关键信息、 补充信息和小窍门等。  “说明”不是安全警示信息，不涉及人身、设备及环境伤害。 |

修改记录

| 文档版本 | 发布日期 | 作者 | 修改说明 |
| --- | --- | --- | --- |
| V1.0 | 2020-07-13 | Hursion,zhang | Temp |

作者信息

|  |  |  |  |
| --- | --- | --- | --- |
| 作者 | XXX | 日期 | 2019-11-08 |
| 审核 | XXX | 日期 | 2019-11-09 |
| 测试 | XXX | 日期 | 2019-11-10 |
| 签发 | XXX | 日期 | 2019-11-11 |

目 录

[1 user版本获取更多调试信息修改 5](#_Toc45889616)

[1.1 adb root 5](#_Toc45889617)

[1.1.1 android 8.1 5](#_Toc45889618)

[1.1.2 android 9.0 8](#_Toc45889619)

[1.2 打开user下的corefile 11](#_Toc45889620)

[1.2.1 android 8.1 11](#_Toc45889621)

[1.2.2 android 9.0 12](#_Toc45889622)

[1.3 默认打开user下的ylog 14](#_Toc45889623)

[1.3.1 android 8.1 14](#_Toc45889624)

[1.3.2 android 9.0 15](#_Toc45889625)

[1.4 将logcat输出到串口 15](#_Toc45889626)

[1.4.1 android 8.1 15](#_Toc45889627)

[1.4.2 android 9.0 16](#_Toc45889628)

[1.5 打开user下的sysdump 18](#_Toc45889629)

[1.5.1 android 8.1 18](#_Toc45889630)

[1.5.2 android 9.0 18](#_Toc45889631)

[1.6 打开hprof保存 19](#_Toc45889632)

[1.6.1 android 8.1 19](#_Toc45889633)

[1.6.2 android 9.0 19](#_Toc45889634)

[1.7 user 打开版本打开sprdruntimeinfo 20](#_Toc45889635)

[1.7.1 android 9.0 20](#_Toc45889636)

[1.8 user版本下导出dropbox修改意见 22](#_Toc45889637)

[1.8.1 android 9.0 22](#_Toc45889638)

# user版本获取更多调试信息修改

## adb root

### android 8.1

* 1. build\make\core\main.mk

修改代码:

ifneq (,$(user\_variant))

# Target is secure in user builds.

- ADDITIONAL\_DEFAULT\_PROPERTIES += ro.secure=1

+ ADDITIONAL\_DEFAULT\_PROPERTIES += ro.secure=0

ADDITIONAL\_DEFAULT\_PROPERTIES += security.perf\_harden=1

ifeq ($(user\_variant),user)

- ADDITIONAL\_DEFAULT\_PROPERTIES += ro.adb.secure=1

+ ADDITIONAL\_DEFAULT\_PROPERTIES += ro.adb.secure=0

endif

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/595825>



注意,在595825的提交里还打开了ro.debuggable，这是给user版本缺省打开ylog使用的。这里的修改会体现在文件：/system/etc/prop.default，但init是通过default.prop -> system/etc/prop.default使用的，default.prop是boot.img根目录下的symlink文件。所以这个修改需要下载systemimage，而如果是boot是签名的，boot\_sign.img也需要下载。

* 1. system\core\adb\Android.mk

修改代码:

- LOCAL\_CFLAGS += -DALLOW\_ADBD\_NO\_AUTH=$(if $(filter userdebug eng,$(TARGET\_BUILD\_VARIANT)),1,0)

+ LOCAL\_CFLAGS += -DALLOW\_ADBD\_NO\_AUTH=$(if $(filter user userdebug eng,$(TARGET\_BUILD\_VARIANT)),1,0)

- ifneq (,$(filter user userdebug eng,$(TARGET\_BUILD\_VARIANT)))

+ #ifneq (,$(filter user userdebug eng,$(TARGET\_BUILD\_VARIANT)))

LOCAL\_CFLAGS += -DALLOW\_ADBD\_DISABLE\_VERITY=1

LOCAL\_CFLAGS += -DALLOW\_ADBD\_ROOT=1

- endif

+ #endif

对应gerrit提交:<http://review.source.unisoc.com/gerrit/#/c/595828>

* 1. system\core\adb\daemon\main.cpp

修改代码:

static bool should\_drop\_privileges() {

+ #if 0

#if defined(ALLOW\_ADBD\_ROOT)

// The properties that affect `adb root` and `adb unroot` are ro.secure and

// ro.debuggable. In this context the names don't make the expected behavior

// particularly obvious.

//

// ro.debuggable:

// Allowed to become root, but not necessarily the default. Set to 1 on

// eng and userdebug builds.

//

// ro.secure:

// Drop privileges by default. Set to 1 on userdebug and user builds.

bool ro\_secure = android::base::GetBoolProperty("ro.secure", true);

bool ro\_debuggable = \_\_android\_log\_is\_debuggable();

// Drop privileges if ro.secure is set...

bool drop = ro\_secure;

// ... except "adb root" lets you keep privileges in a debuggable build.

std::string prop = android::base::GetProperty("service.adb.root", "");

bool adb\_root = (prop == "1");

bool adb\_unroot = (prop == "0");

if (ro\_debuggable && adb\_root) {

drop = false;

}

// ... and "adb unroot" lets you explicitly drop privileges.

if (adb\_unroot) {

drop = true;

}

return drop;

#else

return true; // "adb root" not allowed, always drop privileges.

#endif // ALLOW\_ADBD\_ROOT

+ #endif

+ return false;

}

注意；3 这段代码其实也可以不修改，因为service.adb.root这个property缺省是空的，当adb root;adb remount后，这个property会被置为1，直到重启。

所以这里prop = android::base::GetProperty("service.adb.root", "")执行后prop在user版本下，一直都是"",这样的话，这个函数should\_drop\_privileges其实一直都返回false;

这个函数其实只有一个地方使用，也可以在调用的地方直接使用false;

// Don't run as root if running in secure mode.

+ //if (should\_drop\_privileges()) {

+ if (should\_drop\_privileges()&&false) {

drop\_capabilities\_bounding\_set\_if\_needed(jail.get());

* 1. system\core\init\init.cpp

修改代码：

static bool selinux\_is\_enforcing(void)

{

+ return false;

if (ALLOW\_PERMISSIVE\_SELINUX) {

return selinux\_status\_from\_cmdline() == SELINUX\_ENFORCING;

}

return true;

}

* 1. system\core\init\property\_service.cpp

修改代码：

static void update\_sys\_usb\_config() {

- bool is\_debuggable = android::base::GetBoolProperty("ro.debuggable", false);

+ bool is\_debuggable = true;//android::base::GetBoolProperty("ro.debuggable", false);

std::string config = android::base::GetProperty("persist.sys.usb.config", "");

2/3/4/5->对应gerrit提交http://review.source.unisoc.com/gerrit/#/c/595828

代码修改，体现在 system.img中。

### android 9.0

* 1. \build\make\core\main.mk

修改代码:

ifneq (,$(user\_variant))

# Target is secure in user builds.

- ADDITIONAL\_DEFAULT\_PROPERTIES += ro.secure=1

+ ADDITIONAL\_DEFAULT\_PROPERTIES += ro.secure=0

ADDITIONAL\_DEFAULT\_PROPERTIES += security.perf\_harden=1

ifeq ($(user\_variant),user)

- ADDITIONAL\_DEFAULT\_PROPERTIES += ro.adb.secure=1

+ ADDITIONAL\_DEFAULT\_PROPERTIES += ro.adb.secure=0

endif

对应gerrit提交:<http://review.source.unisoc.com/gerrit/#/c/594255>

* 1. system\core\adb\Android.mk

修改代码:

- LOCAL\_CFLAGS += -DALLOW\_ADBD\_NO\_AUTH=$(if $(filter userdebug eng,$(TARGET\_BUILD\_VARIANT)),1,0)

+ LOCAL\_CFLAGS += -DALLOW\_ADBD\_NO\_AUTH=$(if $(filter user userdebug eng,$(TARGET\_BUILD\_VARIANT)),1,0)

- ifneq (,$(filter userdebug eng,$(TARGET\_BUILD\_VARIANT)))

+ #ifneq (,$(filter userdebug eng,$(TARGET\_BUILD\_VARIANT)))

LOCAL\_CFLAGS += -DALLOW\_ADBD\_DISABLE\_VERITY=1

LOCAL\_CFLAGS += -DALLOW\_ADBD\_ROOT=1

- endif

+ #endif

对应gerrit提交:<http://review.source.unisoc.com/gerrit/#/c/594232/>

* 1. \system\core\adb\daemon\main.cpp

修改代码:

static bool should\_drop\_privileges() {

+ #if 0

#if defined(ALLOW\_ADBD\_ROOT)

// The properties that affect `adb root` and `adb unroot` are ro.secure and

// ro.debuggable. In this context the names don't make the expected behavior

// particularly obvious.

//

// ro.debuggable:

// Allowed to become root, but not necessarily the default. Set to 1 on

// eng and userdebug builds.

//

// ro.secure:

// Drop privileges by default. Set to 1 on userdebug and user builds.

bool ro\_secure = android::base::GetBoolProperty("ro.secure", true);

bool ro\_debuggable = \_\_android\_log\_is\_debuggable();

// Drop privileges if ro.secure is set...

bool drop = ro\_secure;

// ... except "adb root" lets you keep privileges in a debuggable build.

std::string prop = android::base::GetProperty("service.adb.root", "");

bool adb\_root = (prop == "1");

bool adb\_unroot = (prop == "0");

if (ro\_debuggable && adb\_root) {

drop = false;

}

// ... and "adb unroot" lets you explicitly drop privileges.

if (adb\_unroot) {

drop = true;

}

return drop;

#else

return true; // "adb root" not allowed, always drop privileges.

#endif // ALLOW\_ADBD\_ROOT

+ #endif

+ return false;

}

对应gerrit提交:<http://review.source.unisoc.com/gerrit/#/c/594232/>

* 1. \system\core\init\selinux.cpp

关闭selinux

修改代码:

bool IsEnforcing() {

+ return false;

if (ALLOW\_PERMISSIVE\_SELINUX) {

return StatusFromCmdline() == SELINUX\_ENFORCING;

}

return true;

}

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/594232>

* 1. \system\core\init\property\_service.cpp

打开开发者模式里的adb 调试

修改代码:

static void update\_sys\_usb\_config() {

- bool is\_debuggable = android::base::GetBoolProperty("ro.debuggable", false);

+ bool is\_debuggable = true;//android::base::GetBoolProperty("ro.debuggable", false);

std::string config = android::base::GetProperty("persist.sys.usb.config", "");

if (config.empty()) {

property\_set("persist.sys.usb.config", is\_debuggable ? "adb" : "none");

} else if (is\_debuggable && config.find("adb") == std::string::npos &&

config.length() + 4 < PROP\_VALUE\_MAX) {

config.append(",adb");

property\_set("persist.sys.usb.config", config);

}

}

修改体现在system.img里

## 打开user下的corefile

### android 8.1

修改代码：

* 1. device\sprd\sharkle\common\DeviceCommon.mk

ifeq ($(TARGET\_BUILD\_VARIANT),user)

PRODUCT\_PROPERTY\_OVERRIDES += \

persist.sys.sprd.wcnreset=1 \

persist.sys.apr.enabled=0 \

persist.sys.apr.timechanged=180 \

persist.sys.apr.rlchanged=800 \

persist.sys.apr.lifetime=0 \

persist.sys.apr.reload=0 \

persist.sys.apr.reportlevel=2 \

persist.sys.apr.exceptionnode=0 \

- persist.sys.start\_udpdatastall=0

+ persist.sys.start\_udpdatastall=0 \

+ persist.vendor.sys.core.enabled = 1 \

+ vendor.debug.sysdump.enabled = true \

+ persist.vendor.eng.reset = 0

对应gerrit提交：<http://review.source.unisoc.com/gerrit/#/c/595832>

　这里同时将sysdump也打开了。

修改会体现在：vendor/build.prop中，所以需要下载vendor.img

* 1. system\core\init\init.cpp

关闭selinux

修改代码：

static bool selinux\_is\_enforcing(void)

{

+ return false;

if (ALLOW\_PERMISSIVE\_SELINUX) {

return selinux\_status\_from\_cmdline() == SELINUX\_ENFORCING;

}

return true;

}

　　对应gerrit提交：<http://review.source.unisoc.com/gerrit/#/c/595828/>

需要下载system.img

### android 9.0

* 1. \device\sprd\sharkle\common\DeviceCommon.mk

增加corefile开关:vendor.debug.sysdump.enabled

修改代码:

ifeq ($(TARGET\_BUILD\_VARIANT),user)

PRODUCT\_PROPERTY\_OVERRIDES += \

persist.vendor.sys.wcnreset=1 \

persist.vendor.aprservice.enabled=0 \

persist.sys.apr.enabled=0 \

persist.sys.apr.timechanged=180 \

persist.sys.apr.rlchanged=800 \

persist.sys.apr.lifetime=0 \

persist.sys.apr.reload=0 \

persist.sys.apr.reportlevel=2 \

- persist.sys.apr.exceptionnode=0

+ persist.sys.apr.exceptionnode=0 \

+ persist.vendor.sys.core.enabled = 1

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/594236>

注意,在这里的这个文件提交还同时打开了sysdump,如果不需要,可以忽略.

* 1. \system\core\init\selinux.cpp

关闭selinux

修改代码:

bool IsEnforcing() {

+ return false;

if (ALLOW\_PERMISSIVE\_SELINUX) {

return StatusFromCmdline() == SELINUX\_ENFORCING;

}

return true;

}

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/594232>

* 1. \kernel4.4\fs\coredump.c

允许system进程coredump

修改代码:

audit\_core\_dumps(siginfo->si\_signo);

binfmt = mm->binfmt;

if (!binfmt || !binfmt->core\_dump)

goto fail;

+// if (!\_\_get\_dumpable(cprm.mm\_flags))

+// goto fail;

cred = prepare\_creds();

if (!cred)

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/594247>

这个修改的文件比较杂,涉及的img比较多:

1的修改对应vendor.img

2的修改对应system.img

3的修改对应boot.img

## 默认打开user下的ylog

### android 8.1

* 1. build\make\core\main.mk

修改代码：

else # !enable\_target\_debugging

# Target is less debuggable and adbd is off by default

- ADDITIONAL\_DEFAULT\_PROPERTIES += ro.debuggable=0

+ ADDITIONAL\_DEFAULT\_PROPERTIES += ro.debuggable=1

endif # !enable\_target\_debugging

对应gerrit提交：<http://review.source.unisoc.com/gerrit/#/c/595825>

注意：这个修改是打开了ro.debuggable，这个影响比较大，很多log也同步开启了。

这里的修改会体现在文件：/system/etc/prop.default，但init是通过default.prop -> system/etc/prop.default使用的，default.prop是boot.img根目录下的symlink文件。

所以这个修改需要下载systemimage，而如果是boot是签名的，boot\_sign.img也需要下载。

* 1. device\sprd\sharkle\common\rootdir\root\init.common.rc

修改代码：

chmod 0755 /sys/kernel/debug/tracing

+ on property:init.svc.logd=running

+ setprop persist.sys.ylog.enabled 1

+ start ylog

on post-fs-data

mkdir /data/local/englog 0777

mkdir /data/local/slogmodem 0770 shell system

mkdir /data/local/mcue 0777 media system

setprop vold.post\_fs\_data\_done 1

mkdir /data/sprdinfo 0771 system system

mkdir /data/apr 0777 system system

chown system system /data/apr.conf

mkdir /data/misc 01771 system misc

mkdir /data/ylog 0777

mkdir /cache/ylog 0777

mkdir /data/ylog\_lite 0777

mkdir /data/corefile 0777

+ setprop persist.sys.ylog.enabled 1

start ylog

#start ylog\_lite

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/595832>

这个修改在init.common.rc中，这个文件在8.1上打包在bootimage的根目录下(和9.0不同)

所以需要下载boot.img.

### android 9.0

* 1. \vendor\sprd\tools\ylog\_common\ylog.rc

修改perporty打开ylog

修改代码:

on post-fs-data

mkdir /cache/ylog 0777 system system

mkdir /data/ylog 0777 shell shell

mkdir /data/ylog/dropbox 0777 shell shell

+ setprop persist.sys.ylog.enabled 1

start ylog

on property:init.svc.logd=running

+ setprop persist.sys.ylog.enabled 1

start ylog

对应gerrit提交:<http://review.source.unisoc.com/gerrit/#/c/594258/>

修改打包在system.img

## 将logcat输出到串口

### android 8.1

* 1. system\core\init\init.cpp

关闭selinux

修改代码：

static bool selinux\_is\_enforcing(void)

{

+ return false;

if (ALLOW\_PERMISSIVE\_SELINUX) {

return selinux\_status\_from\_cmdline() == SELINUX\_ENFORCING;

}

return true;

}

　　对应gerrit提交：<http://review.source.unisoc.com/gerrit/#/c/595828/>

需要下载system.img

* 1. device\sprd\sharkle\common\rootdir\root\init.common.rc

修改代码：

start aprd

+ service logcat /system/bin/logcat -f /dev/kmsg \*:w

+ class main

+ user root

+ group log

+ oneshot

service p2p\_supplicant /vendor/bin/hw/wpa\_supplicant \

　　对应gerrit提交：<http://review.source.unisoc.com/gerrit/#/c/595832>

init.common.rc会打包在bootimage.

### android 9.0

* 1. system\core\init\selinux.cpp

关闭selinux

修改代码:

bool IsEnforcing() {

+ return false;

if (ALLOW\_PERMISSIVE\_SELINUX) {

return StatusFromCmdline() == SELINUX\_ENFORCING;

}

return true;

}

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/594232>

* 1. device\sprd\sharkle\common\rootdir\root\init.common.rc

增加logcat服务

修改代码：

# Support audio

mkdir /data/vendor/local/media 0770 audioserver system

#start aprd

+ service logcat /system/bin/logcat -f /dev/kmsg \*:w

+ class main

+ user root

+ group log

+ oneshot

service p2p\_supplicant /vendor/bin/hw/wpa\_supplicant \

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/594236>

* 1. (另一种方法)system\core\rootdir\init.rc

增加logcat服务

修改代码：

#setprop vold.post\_fs\_data\_done 1

+ service logcat /system/bin/logcat -f /dev/kmsg \*:w

+ class main

+ user root

+ group log

+ oneshot

# It is recommended to put unnecessary data/ initialization from post-fs-data

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/594232>

2和3任选一种就可以，两者不同在于下载的image不同

1 -> 需要下载system.img

2 -> 需要下载vendor.img

3 -> 需要下载system.img

所以 1+2 -> 需要下载system/vendor

1+3 -> 只需下载system

## 打开user下的sysdump

### android 8.1

同1.2打开user下的corefile 8.1节

### android 9.0

* 1. device\sprd\sharkle\common\DeviceCommon.mk

修改代码：

persist.sys.apr.reportlevel=2 \

- persist.sys.apr.exceptionnode=0

+ persist.sys.apr.exceptionnode=0 \

+ vendor.debug.sysdump.enabled = true \

+ persist.vendor.eng.reset = 0

else

PRODUCT\_PROPERTY\_OVERRIDES += \

persist.vendor.sys.wcnreset=0 \

persist.vendor.aprservice.enabled=1 \

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/594236>

这个提交中还包括corefile的修改。

事实上，如果adb root了，进入后可以可以setprop将这两个property人工写入，也可以打开，并且直接生效。

## 打开hprof保存

### android 8.1

如果已经修改了1.3.1-1，那么这里不需要修改，否则作如下修改：

* 1. frameworks/base/services/core/java/com/android/server/am/ActivityManagerService.java

修改代码：

- boolean isDebuggable ="1".equals(SystemProperties.get(SYSTEM\_DEBUGGABLE, "0"));

+ boolean isDebuggable =true;// "1".equals(SystemProperties.get(SYSTEM\_DEBUGGABLE, "0"));

if (!isDebuggable) {

if ((proc.info.flags&ApplicationInfo.FLAG\_DEBUGGABLE) == 0) {

throw new SecurityException("Process not debuggable: " + proc);

}

}

修改在system.img中。

### android 9.0

打开user下的hprof功能

* 1. frameworks/base/services/core/java/com/android/server/am/ActivityManagerService.java

修改代码：

ProcessRecord proc = findProcessLocked(process, userId, "dumpHeap");

if (proc == null || proc.thread == null) {

throw new IllegalArgumentException("Unknown process: " + process);

}

- boolean isDebuggable = "1".equals(SystemProperties.get(SYSTEM\_DEBUGGABLE, "0"));

+ boolean isDebuggable = true;//"1".equals(SystemProperties.get(SYSTEM\_DEBUGGABLE, "0"));

if (!isDebuggable) {

if ((proc.info.flags&ApplicationInfo.FLAG\_DEBUGGABLE) == 0) {

throw new SecurityException("Process not debuggable: " + proc);

}

}

对应gerrit提交: <http://review.source.unisoc.com/gerrit/#/c/594267>

需要下载system.img

## user 打开版本打开sprdruntimeinfo

### android 9.0

* 1. 修改kernel

<http://review.source.unisoc.com/gerrit/#/c/576619/>

diff --git a/drivers/soc/sprd/Makefile b/drivers/soc/sprd/Makefile

index 2fb0f65..ed1d7bd 100644

--- a/drivers/soc/sprd/Makefile

+++ b/drivers/soc/sprd/Makefile

@@ -22,8 +22,10 @@

obj-$(CONFIG\_SPRD\_DEEPSLEEP\_PIKE2) += pm-pike2.o

obj-$(CONFIG\_SPRD\_DEEPSLEEP\_SHARKL3) += pm-sharkl3.o

obj-$(CONFIG\_SPRD\_ARM32\_SMP\_WITH\_PSCI) += sprd\_psci\_smp.o

+obj-$(CONFIG\_DEBUG\_FS) += debug/sprd\_debugfs\_frame.o

obj-$(CONFIG\_SPRD\_SYSDUMP) += debug/sysdump/

obj-$(CONFIG\_SPRD\_DMC\_DRV) += debug/dmc\_drv/

+obj-$(CONFIG\_SPRD\_CPU\_USAGE) += debug/cpu\_usage/

ifdef CONFIG\_ARM\_CPUIDLE

obj-$(CONFIG\_SPRD\_ARM32\_SMP\_WITH\_PSCI) += sprd\_aarch32\_cpuidle.o

endif

diff --git a/drivers/soc/sprd/debug/Kconfig b/drivers/soc/sprd/debug/Kconfig

index 209fa8e..ddf4c09 100644

--- a/drivers/soc/sprd/debug/Kconfig

+++ b/drivers/soc/sprd/debug/Kconfig

@@ -23,7 +23,7 @@

config SPRD\_CPU\_USAGE

bool "open sprd cpu rate compute"

- depends on DEBUG\_FS && SPRD\_DEBUG

+ depends on DEBUG\_FS

default y

help

CPU usage statistics

diff --git a/drivers/soc/sprd/debug/Makefile b/drivers/soc/sprd/debug/Makefile

index 4534575..88f9182 100644

--- a/drivers/soc/sprd/debug/Makefile

+++ b/drivers/soc/sprd/debug/Makefile

@@ -1,7 +1,5 @@

-obj-$(CONFIG\_SPRD\_DEBUG) += sprd\_debugfs\_frame.o

obj-$(CONFIG\_LOOK\_AT) += sprd\_lookat.o

obj-y += busmonitor/

-obj-$(CONFIG\_SPRD\_CPU\_USAGE) += cpu\_usage/

obj-y += dmc\_mpu/

obj-y += djtag/

obj-y += irq/

* 1. 修改 arch

<http://review.source.unisoc.com/gerrit/#/c/576618/>

diff --git a/arch/arm/include/asm/switch\_to.h b/arch/arm/include/asm/switch\_to.h

index ed11912..08c2f49 100644

--- a/arch/arm/include/asm/switch\_to.h

+++ b/arch/arm/include/asm/switch\_to.h

@@ -22,7 +22,7 @@

\*/

extern struct task\_struct \*\_\_switch\_to(struct task\_struct \*, struct thread\_info \*, struct thread\_info \*);

-#if defined(CONFIG\_SPRD\_CPU\_USAGE) && defined(CONFIG\_SPRD\_DEBUG)

+#if defined(CONFIG\_SPRD\_CPU\_USAGE)

extern void sprd\_update\_cpu\_usage(struct task\_struct \*prev,

struct task\_struct \*next);

#else

diff --git a/arch/arm64/kernel/process.c b/arch/arm64/kernel/process.c

index a88dbd5..0ef0788 100644

--- a/arch/arm64/kernel/process.c

+++ b/arch/arm64/kernel/process.c

@@ -385,7 +385,7 @@

write\_sysreg(\*task\_user\_tls(next), tpidr\_el0);

}

-#if defined(CONFIG\_SPRD\_CPU\_USAGE) && defined(CONFIG\_SPRD\_DEBUG)

+#if defined(CONFIG\_SPRD\_CPU\_USAGE)

extern void sprd\_update\_cpu\_usage(struct task\_struct \*prev,

struct task\_struct \*next);

#else

## user版本下导出dropbox修改意见

### android 9.0

[Bug #1062270](https://bugzilla.unisoc.com/bugzilla/show_bug.cgi?id=1062270)

* 1. services/core/java/com/android/server/DropBoxManagerService.java

<http://review.source.unisoc.com/gerrit/#/c/589955/>

diff --git a/services/core/java/com/android/server/DropBoxManagerService.java b/services/core/java/com/android/server/DropBoxManagerService.java

index 887de74..46121af 100644

--- a/services/core/java/com/android/server/DropBoxManagerService.java

+++ b/services/core/java/com/android/server/DropBoxManagerService.java

@@ -27,6 +27,7 @@

import android.net.Uri;

import android.os.Binder;

import android.os.Debug;

+import android.os.Build;

import android.os.DropBoxManager;

import android.os.FileUtils;

import android.os.Handler;

@@ -155,6 +156,15 @@

}

};

+ private static Boolean IS\_DEBUG\_BUILD = null;

+

+ public static boolean isDebug() {

+ if (IS\_DEBUG\_BUILD == null) {

+ IS\_DEBUG\_BUILD = Build.TYPE.equals("eng") || Build.TYPE.equals("userdebug");

+ }

+ return IS\_DEBUG\_BUILD;

+ }

+

/\*\*

\* Creates an instance of managed drop box storage using the default dropbox

\* directory.

@@ -175,7 +185,11 @@

@VisibleForTesting

public DropBoxManagerService(final Context context, File path, Looper looper) {

super(context);

- mDropBoxDir = path;

+ if (isDebug()){

+ mDropBoxDir = path;

+ }else{

+ mDropBoxDir = new File("/data/ylog/dropbox");

+ }

mContentResolver = getContext().getContentResolver();

mHandler = new Handler(looper) {

@Override

@@ -574,6 +588,7 @@

if (!temp.renameTo(file)) {

throw new IOException("Can't rename " + temp + " to " + file);

}

+ FileUtils.setPermissions(this.getFile(dir), 0644, -1, -1);

this.blocks = (int) ((file.length() + blockSize - 1) / blockSize);

}

* 1. device/sprd/xxx/common/sepolicy

<http://review.source.unisoc.com/gerrit/#/c/590032/>

diff --git a/common/sepolicy/priv\_app.te b/common/sepolicy/priv\_app.te

index fd0d158..17a9aea 100644

--- a/common/sepolicy/priv\_app.te

+++ b/common/sepolicy/priv\_app.te

@@ -37,3 +37,7 @@

#add for GPU

get\_prop(priv\_app, vendor\_gpu\_prop);

+

+#add for ylog\_file

+allow priv\_app ylog\_file:file { open read getattr };

+

diff --git a/common/sepolicy/system\_server.te b/common/sepolicy/system\_server.te

index 9a529f0..b47fa68 100644

--- a/common/sepolicy/system\_server.te

+++ b/common/sepolicy/system\_server.te

@@ -110,3 +110,8 @@

get\_prop(system\_server, vendor\_gpu\_prop);

allow system\_server media\_prop:file { read };

+

+#add system\_server rw ylog\_file

+allow system\_server ylog\_file:dir { create getattr open read rename write unlink search add\_name remove\_name };

+allow system\_server ylog\_file:file { create getattr open read rename write unlink setattr };

+

diff --git a/common/sepolicy/untrusted\_app.te b/common/sepolicy/untrusted\_app.te

index 7d08705..2a0b95f 100644

--- a/common/sepolicy/untrusted\_app.te

+++ b/common/sepolicy/untrusted\_app.te

@@ -31,3 +31,7 @@

# add for coredump

allow untrusted\_app coredump\_file:dir { write add\_name };

get\_prop(untrusted\_app, vendor\_sys\_prop);

+

+#add for ylog\_file

+allow untrusted\_app ylog\_file:file { open read getattr };

+

* 1. vendor/sprd/tools/ylog\_common/ylog.rc

<http://review.source.unisoc.com/gerrit/#/c/589958/>

diff --git a/ylog.rc b/

index 749e91d..9db6d4d 100755

--- a/ylog.rc

+++ b/ylog.rc

@@ -14,7 +14,8 @@

on post-fs-data

mkdir /cache/ylog 0777 system system

- mkdir /data/ylog 0777

+ mkdir /data/ylog 0777 shell shell

+ mkdir /data/ylog/dropbox 0777 shell shell

start ylog

on property:init.svc.logd=running