Pon-AP log模块

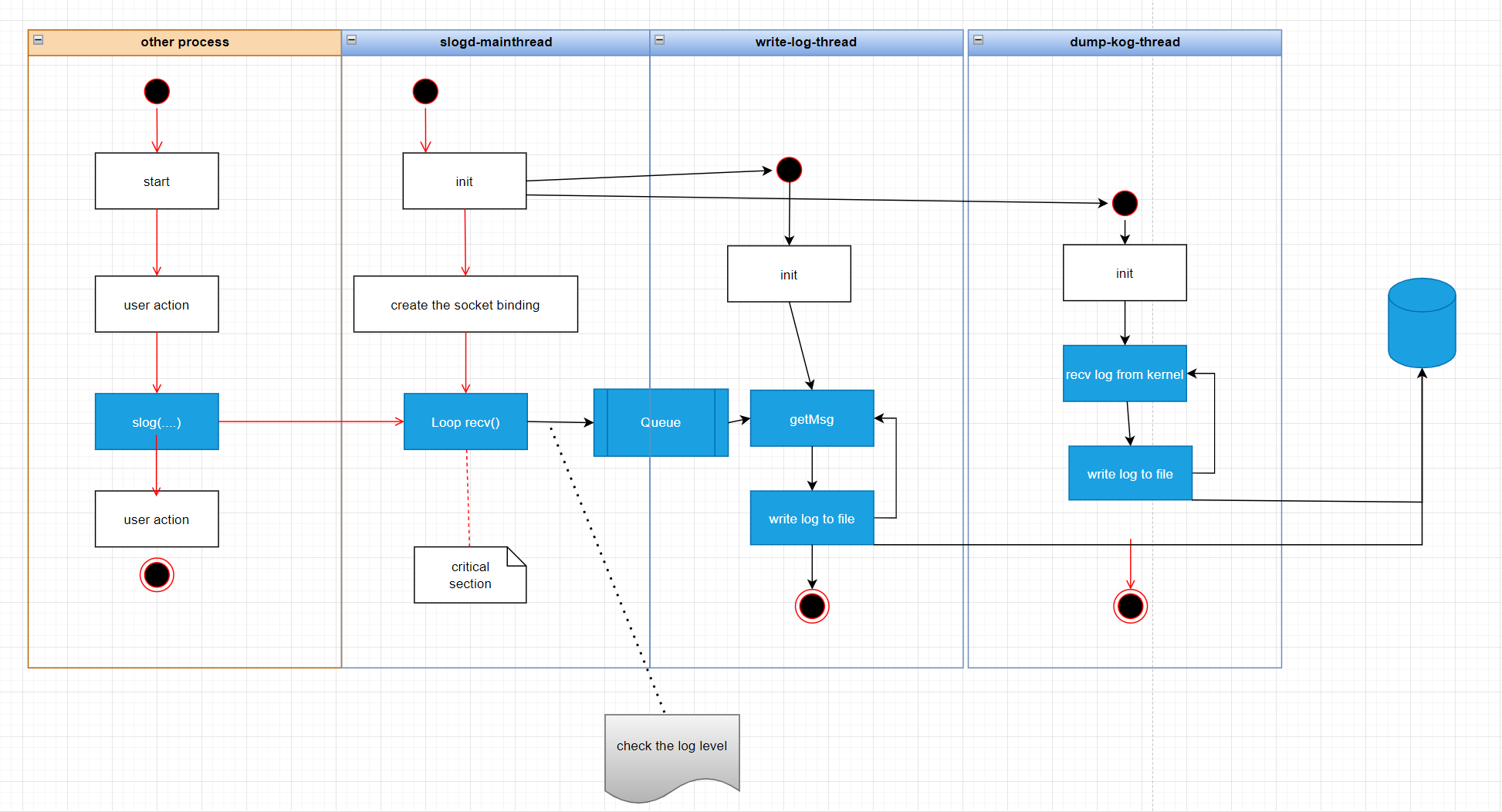
## 背景：

方便不同模块日志收集和导出；

方便各个模块调试，最好支持不同模块的调试级别调整

快速log记录和保存，在多线程多进程安全的前提下减少锁的开销

## 基本结构



1. 提供一个slogd主程序，用户接受各个模块的log上报，集中收集上报的log，根据log所示模块设置的级别，决定是否需要保存。
2. Slogd主程序可以根据配置确定日志保存的地方和保存日志的最大文件
3. Slogd主程序收集kernel日志和应用程序的日志一起保存在一个日志文件里面
4. Slogd和syslogd最大的区别是**对日志的级别和模块动态调整**
5. Slogd会根据当前的配置来处理和保存log到ramfile或者flashfile
6. Slogd会根据文件的大小来保存回滚日志，防止内存或者flash空间占用过多。
7. 提供一个slog(…) API,其他模块直接调用这个API对应的so文件就可以发送log给slogd。

## 主要模块

* Libslog.so：

提供给各个模块调用，只有一个调用api，格式和printf类似，使用简单。如果有新的模块，需要模块开发人定义新的模块LOG\_TAG编号和名字，需要定义模块的配置。

* 主线程

接受各个模块得log，根据当前的具体log配置来发送log数据到log写文件线程

* Kernel log处理线程：

从/dev/kmsg获取kernel的log，然后调用slog\_printf()发送给slogd。

* log写文件线程：

从队列里接受数据然后写入文件，根据配置写入ramfile或者flashfile。如果文件大小大于设定大小，备份已满文件为xxx.log.bk,然后清空现有文件继续开始记录新的log。

## 主要数据结构

1. LOG\_TAG：

typedef enum {

SLOG\_MOD\_KERN = 0, /\* kernel messages \*/

SLOG\_MOD\_FWUPGRADE = 1, /\* firmware upgrade messages \*/

SLOG\_MOD\_CAPWAP = 2, /\* capwap messages \*/

    SLOG\_MOD\_EVENTMGR       = 3,    /\* event manager \*/

    SLOG\_MOD\_LANHOST        = 4,    /\* lan host \*/

/\*THE END.... \*/

SLOG\_MOD\_MAX /\* just define it to other \*/

} SLOG\_MOD\_NAME\_E;

static char \*pmode2str[SLOG\_MOD\_MAX+1] = {

"KERNEL",

"FWUPGRADE",

"CAPWAP",

     "EVENTMGR",

     "LANHOST",

/\*THE END.... \*/

"other"

};

1. LOG\_LEVEL:

typedef enum {

SLOG\_LEVEL\_EMERG = 0, /\* system is unusable \*/

SLOG\_LEVEL\_ALERT = 1, /\* action must be taken immediately \*/

SLOG\_LEVEL\_CRIT = 2, /\* critical conditions \*/

SLOG\_LEVEL\_ERR = 3, /\* error conditions \*/

SLOG\_LEVEL\_WARNING = 4, /\* warning conditions \*/

SLOG\_LEVEL\_NOTICE = 5, /\* normal but significant condition \*/

SLOG\_LEVEL\_INFO = 6, /\* informational \*/

SLOG\_LEVEL\_DEBUG = 7, /\* debug-level messages \*/

    ATOS\_LOG\_DEBUG      = 7,

/\*THE END.... \*/

SLOG\_LEVEL\_MAX

} SLOG\_LEVEL\_E;

static char \*pleve2str[SLOG\_LEVEL\_MAX+1] = {

"EMERG",

"ALERT",

"CRIT",

"ERR",

"WARNING",

"NOTICE",

"INFO",

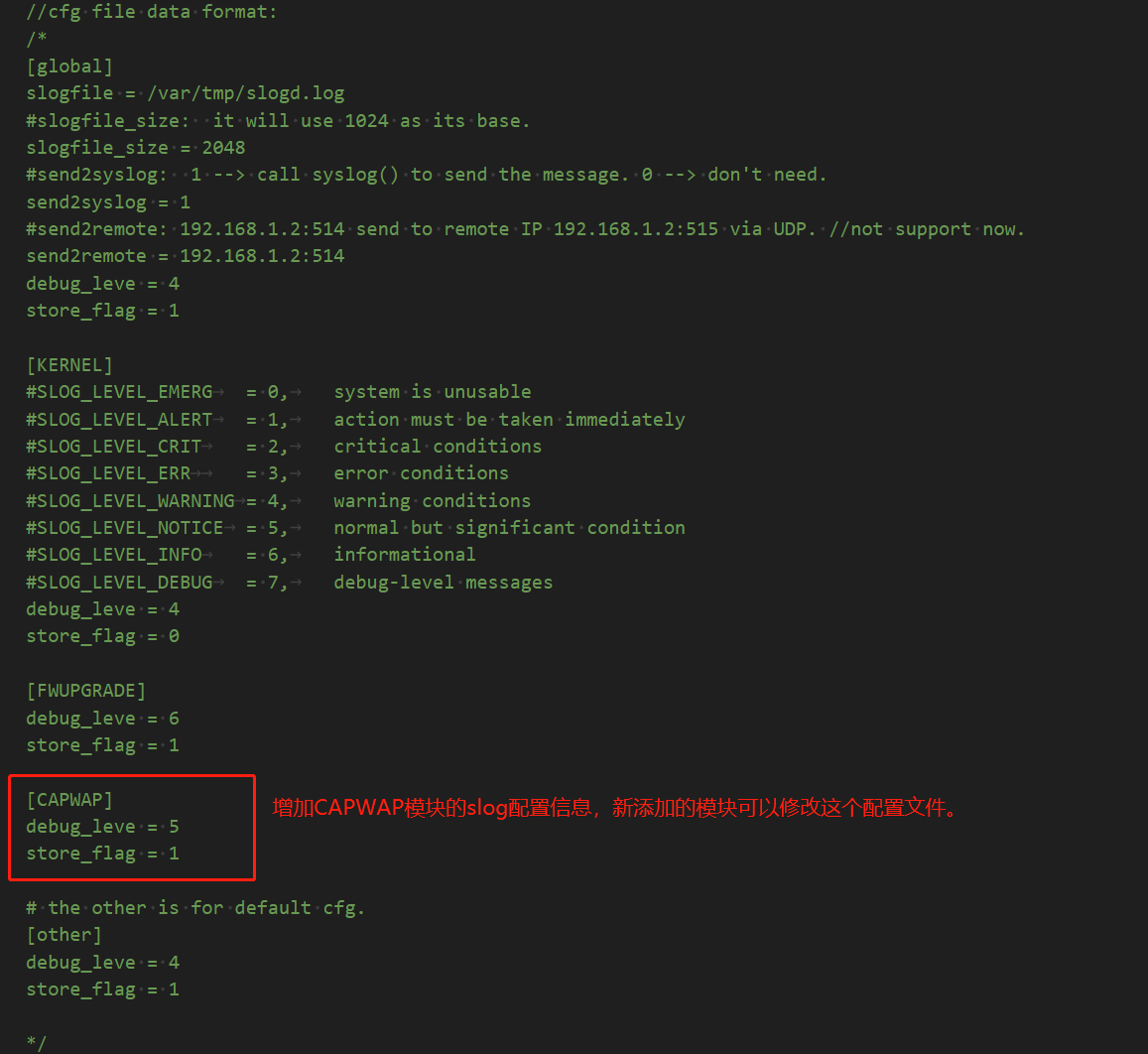
"DEBUG",

""

};

1. Slogd的配置文件：

缺省配置文件在/etc/slogd.cfg, 如果你需要保存修改，需要把修改后的文件复制到/data/目录，这样系统断电重启就用保持你的修改配置。



## 用户使用API

1. API如下所示，用户只需要在编译是添加libslog.so和头文件libslog.h即可使用

//if you don't know your mod\_index, you can use SLOG\_MOD\_MAX which will be match do it's program name.

void slog\_printf(SLOG\_MOD\_NAME\_E mod\_index, SLOG\_LEVEL\_E pri\_level, const char \*format,...);

void slog\_ctl(slogtype\_e type, SLOG\_MOD\_NAME\_E mod\_index, SLOG\_LEVEL\_E level, const char \*format,...);

我们可以根据具体的情况来调用slog\_printf(…), 或者在每个模块里面使用define的方式调用这个api。

1. 示例如下：

#define lanhost\_log(level, fmt, args...)  \

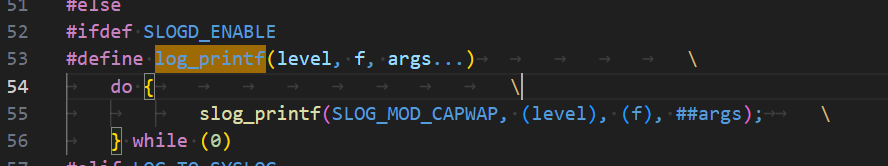
        slog\_printf(SLOG\_MOD\_LANHOST, (level), (fmt), ##args)

#define lanhost\_log\_error(fmt, args...)   \

        slog\_printf(SLOG\_MOD\_LANHOST, ATOS\_LOG\_ERR, "%s(%d): "fmt, \_\_FUNCTION\_\_, \_\_LINE\_\_, ##args)

#define lanhost\_log\_debug(fmt, args...)   \

        slog\_printf(SLOG\_MOD\_LANHOST, ATOS\_LOG\_DEBUG, "%s(%d): "fmt, \_\_FUNCTION\_\_, \_\_LINE\_\_, ##args)



新的模块需要添加模块索引，如SLOG\_MOD\_LANHOST和对应的字符串。一般的debug ldeve都是相同的1~7个级别，如果名称不一样，也可以使用define的方式保存你的代码不变。

