kmod-Linux内核模块工具

1. 项目背景分析

kmod 是为了能够操作 Linux 内核模块而推出的一系列工具集，这些操作包括 插入（insert），删除（remove），列出（list），检查属性（check properties）和解决依赖关系（dependencies）。

这些工具在底层都需要用到 libkmod 这个库，相关的源码也会跟着 kmod 项目一同发布。这个项目的目标是能够实现与在此之前 module-init-tools 项目所提供的工具兼容。

项目建立时间

从 git.kernel.org 上的 commit log 分析，该项目的建立时间是 2011-11-21。最初的项目是通过继承了 libabc 的框架开始逐步演变而来。2011-12-15 发布了 kmod 1 版本。

参考项目主页
http://git.kernel.org/cgit/utils/kernel/kmod/kmod.git

项目创建者和维护者

创建者是 Lucas De Marchi。这个人就职于巴西 Brazil Campinas 的一家公司ProFUSION Embedded Systems（该公司的主页http://profusion.mobi/），从他在 github 个人项目的帐号创建时间看是 2008年10月30号，应该是属于比较早期的 github 用户。

参考个人主页
https://github.com/lucasdemarchi

项目更新记录

项目最近一次提交 commit log 表明，该项目近期处于一个比较活跃的状态。从 2013-4-9 发布了最新的 kmod 13 版本之后，该项目几乎每隔1，2天有一次或多次提交。最近的一次提交是 2013-4-17，主要的贡献者仍然是 Lucas De Marchi。

参考提交记录
http://git.kernel.org/cgit/utils/kernel/kmod/kmod.git/log/

项目版本情况

第一个可以下载的软件包 kmod-1.tar.gz 是2012-2-24 上传的，最新的软件包 kmod-13.tar.gz 是 2013-4-9 上传的。

目前 kmod 已经发布到了第13个版本，从项目 NEWS 中可以看到，项目从版本 1 就开始支持原来的 insmod/rmmod/lsmod/modprobe 这几个常用命令，发展至今libkmod 库已经提供了100多个函数接口用于方便用户管理内核模块。

项目资源汇总

- 代码下载
  https://www.kernel.org/pub/linux/utils/kernel/kmod
- 邮件列表
  linux-modules@vger.kernel.org
- Git项目仓库
  git://git.kernel.org/pub/scm/utils/kernel/kmod/kmod.git
  https://git.kernel.org/pub/scm/utils/kernel/kmod/kmod.git
- Gitweb页面
  http://git.kernel.org/?p=utils/kernel/kmod/kmod.git

2. 项目技术分析

开发环境准备

- 首先需要安装如下的软件工具
  - GCC compiler
  - GNU C library
  - autoconf
  - shtool
  - libtool
  - xsltproc
- 可选的依赖关系:
  - ZLIB library
  - LZMA library

编译和安装

```
$ sudo apt-get install autoconf
$ sudo apt-get install shtool
$ sudo apt-get install libtool
$ sudo apt-get install xsltproc

$ aclocal
$ autoconf
$ ./configure CFLAGS="-g -O2" --prefix=/usr --sysconfdir=/etc --
libdir=/usr/lib
$ make && make install
```

错误及解决

```
$ autoconf
configure.ac:10: error: possibly undefined macro: AM_INIT_AUTOMAKE
    If this token and others are legitimate, please use m4_pattern_allow.
      See the Autoconf documentation.
configure.ac:28: error: possibly undefined macro: AM_PROG_CC_C_O
configure.ac:89: error: possibly undefined macro: AM_CONDITIONAL
$ aclocal
 (          aclocal.m4

$ ./configure CFLAGS="-g -O2" --prefix=/usr --sysconfdir=/etc --
libdir=/usr/lib
configure: error: cannot find install-sh, install.sh, or shtool in build-
aux "."/build-aux
$ autoreconf -f -i -Wall,no-obsolete
Can't exec "libtoolize": No such file or directory at /usr/bin/
autoreconf line 196.
Use of uninitialized value in pattern match (m//) at /usr/bin/
autoreconf line 196.
$ sudo apt-get install libtool

$ ./configure CFLAGS="-g -O2" --prefix=/usr --sysconfdir=/etc --
libdir=/usr/lib
configure: error: xsltproc command not found, try ./configure --
disable-manpages
$ sudo apt-get install xsltproc
         Makefile
```

编译过程

```
$ make
make --no-print-directory all-recursive
Making all in .
  CC     libkmod/libkmod.lo
  CC     libkmod/libkmod-list.lo
  CC     libkmod/libkmod-config.lo
  CC     libkmod/libkmod-index.lo
  CC     libkmod/libkmod-module.lo
  CC     libkmod/libkmod-file.lo
  CC     libkmod/libkmod-elf.lo
  CC     libkmod/libkmod-signature.lo
  CC     libkmod/libkmod-hash.lo
  CC     libkmod/libkmod-array.lo
  CC     libkmod/libkmod-util.lo
  CCLD   libkmod/libkmod-util.la
  CCLD   libkmod/libkmod.la
  CCLD   libkmod/libkmod-private.la
  CC     tools/kmod.o
  CC     tools/lsmod.o
  CC     tools/rmmod.o
  CC     tools/insmod.o
  CC     tools/modinfo.o
  CC     tools/modprobe.o
  CC     tools/depmod.o
  CC     tools/log.o
  CC     tools/static-nodes.o
  CCLD   tools/kmod
  CCLD   tools/kmod-nolib
  GEN    tools/insmod
  GEN    tools/rmmod
  GEN    tools/lsmod
  GEN    tools/modprobe
  GEN    tools/modinfo
  GEN    tools/depmod
  GEN    libkmod/libkmod.pc
Making all in libkmod/docs
make[2]: Nothing to be done for `all'.
Making all in man
  GEN    depmod.d.5
  GEN    modprobe.d.5
  GEN    modules.dep.5
  GEN    depmod.8
```

```
  GEN     insmod.8
  GEN     lsmod.8
  GEN     rmmod.8
  GEN     modprobe.8
  GEN     modinfo.8
```

生成文件

```
$ ls tools/ -l | grep x
lrwxrwxrwx 1 akaedu akaedu     10 Apr 17 04:43 depmod -> kmod-
nolib
lrwxrwxrwx 1 akaedu akaedu     10 Apr 17 04:43 insmod -> kmod-
nolib
-rwxrwxr-x 1 akaedu akaedu   8385 Apr 17 04:43 kmod
-rwxrwxr-x 1 akaedu akaedu 488644 Apr 17 04:43 kmod-nolib
lrwxrwxrwx 1 akaedu akaedu     10 Apr 17 04:43 lsmod -> kmod-
nolib
lrwxrwxrwx 1 akaedu akaedu     10 Apr 17 04:43 modinfo -> kmod-
nolib
lrwxrwxrwx 1 akaedu akaedu     10 Apr 17 04:43 modprobe -> kmod-
nolib
lrwxrwxrwx 1 akaedu akaedu     10 Apr 17 04:43 rmmod -> kmod-
nolib

$ ls libkmod/ -l | grep lo
-rw-rw-r-- 1 akaedu akaedu   308 Apr 17 04:43 libkmod-array.lo
-rw-rw-r-- 1 akaedu akaedu   310 Apr 17 04:43 libkmod-config.lo
-rw-rw-r-- 1 akaedu akaedu   304 Apr 17 04:43 libkmod-elf.lo
-rw-rw-r-- 1 akaedu akaedu   306 Apr 17 04:43 libkmod-file.lo
-rw-rw-r-- 1 akaedu akaedu   306 Apr 17 04:43 libkmod-hash.lo
-rw-rw-r-- 1 akaedu akaedu   308 Apr 17 04:43 libkmod-index.lo
-rw-rw-r-- 1 akaedu akaedu   306 Apr 17 04:43 libkmod-list.lo
-rw-rw-r-- 1 akaedu akaedu   296 Apr 17 04:43 libkmod.lo
-rw-rw-r-- 1 akaedu akaedu   310 Apr 17 04:43 libkmod-module.lo
-rw-rw-r-- 1 akaedu akaedu  316 Apr 17 04:43 libkmod-signature.lo
-rw-rw-r-- 1 akaedu akaedu   306 Apr 17 04:43 libkmod-util.lo

$ ls libkmod/ -l | grep la
-rw-rw-r-- 1 akaedu akaedu   923 Apr 17 04:43 libkmod.la
-rw-rw-r-- 1 akaedu akaedu   893 Apr 17 04:43 libkmod-private.la
-rw-rw-r-- 1 akaedu akaedu   884 Apr 17 04:43 libkmod-util.la

$ ls libkmod/ -l | grep pc
```

```
-rw-rw-r-- 1 akaedu akaedu   210 Apr 17 04:43 libkmod.pc
-rw-rw-r-- 1 akaedu akaedu   255 Apr 17 00:53 libkmod.pc.in
```

安装过程

```
$ make && make install
make --no-print-directory all-recursive
Making all in .
Making all in libkmod/docs
make[2]: Nothing to be done for `all'.
Making all in man
make[2]: Nothing to be done for `all'.
Making install in .
test -z "/usr/lib" || /bin/mkdir -p "/usr/lib"
  /bin/bash ./libtool     --mode=install /usr/bin/install -
c   libkmod/libkmod.la '/usr/lib'
libtool: install: /usr/bin/install -c libkmod/.libs/libkmod.so.
2.2.3 /usr/lib/libkmod.so.2.2.3
/usr/bin/install: cannot create regular file `/usr/lib/
libkmod.so.2.2.3': Permission denied
make[2]: *** [install-libLTLIBRARIES] Error 1
make[1]: *** [install-am] Error 2
make: *** [install-recursive] Error 1

$ sudo make install
Making install in .
test -z "/usr/lib" || /bin/mkdir -p "/usr/lib"
  /bin/bash ./libtool     --mode=install /usr/bin/install -
c   libkmod/libkmod.la '/usr/lib'
libtool: install: /usr/bin/install -c libkmod/.libs/libkmod.so.
2.2.3 /usr/lib/libkmod.so.2.2.3
libtool: install: (cd /usr/lib && { ln -s -f libkmod.so.
2.2.3 libkmod.so.2 || { rm -f libkmod.so.2 && ln -s libkmod.so.
2.2.3 libkmod.so.2; }; })
libtool: install: (cd /usr/lib && { ln -s -f libkmod.so.
2.2.3 libkmod.so || { rm -f libkmod.so && ln -s libkmod.so.
2.2.3 libkmod.so; }; })
libtool: install: /usr/bin/install -c libkmod/.libs/libkmod.lai /
usr/lib/libkmod.la
libtool: finish: PATH="/usr/local/sbin:/usr/local/bin:/usr/
sbin:/usr/bin:/sbin:/bin:/sbin" ldconfig -n /usr/lib
----------------------------------------------------------------------
Libraries have been installed in:
```

```
   /usr/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,-rpath -Wl,LIBDIR' linker flag
     - have your system administrator add LIBDIR to `/etc/
ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
test -z "/usr/bin" || /bin/mkdir -p "/usr/bin"
  /bin/bash ./libtool  --mode=install /usr/bin/install -c tools/
kmod '/usr/bin'
libtool: install: /usr/bin/install -c tools/.libs/kmod /usr/
bin/kmod
make --no-print-directory install-exec-hook
if test "/usr/lib" != "/usr/lib"; then \
        /bin/mkdir -p /usr/lib && \
        so_img_name=$(readlink /usr/lib/libkmod.so) && \
         so_img_rel_target_prefix=$(echo /usr/lib | sed 's,\(^/
\|\)[^/][^/]*,..,g') && \
       ln -sf $so_img_rel_target_prefix/usr/lib/$so_img_name /
usr/lib/libkmod.so && \
        mv /usr/lib/libkmod.so.* /usr/lib; \
    fi
test -z "/usr/include" || /bin/mkdir -p "/usr/include"
 /usr/bin/install -c -m 644 libkmod/libkmod.h '/usr/include'
test  -z  "/usr/lib/pkgconfig"  ||  /bin/mkdir  -p  "/usr/lib/
pkgconfig"
  /usr/bin/install  -c  -m  644  libkmod/libkmod.pc  '/usr/lib/
pkgconfig'
Making install in libkmod/docs
make[2]: Nothing to be done for `install-exec-am'.
make[2]: Nothing to be done for `install-data-am'.
Making install in man
make[2]: Nothing to be done for `install-exec-am'.
test -z "/usr/share/man/man5" || /bin/mkdir -p "/usr/share/man/
```

```
man5"
 /usr/bin/install -c -m 644 depmod.d.5 modprobe.d.5 modules.dep.
5 modules.dep.bin.5 '/usr/share/man/man5'
test -z "/usr/share/man/man8" || /bin/mkdir -p "/usr/share/man/
man8"
 /usr/bin/install -c -m 644 depmod.8 insmod.8 lsmod.8 rmmod.
8 modprobe.8 modinfo.8 '/usr/share/man/man8'
$ sudo make install
```

安装文件

```
$ ls /usr/lib/libkmod.so
libkmod.so        libkmod.so.2      libkmod.so.2.2.3
$ ls /usr/lib/libkmod.so* -l
lrwxrwxrwx  1  root  root         16  Apr  17  04:55  /usr/lib/
libkmod.so -> libkmod.so.2.2.3
lrwxrwxrwx 1 root root     16 Apr 17 04:55 /usr/lib/libkmod.so.
2 -> libkmod.so.2.2.3
-rwxr-xr-x 1 root root 313349 Apr 17 04:55 /usr/lib/libkmod.so.
2.2.3

$ ls /usr/lib/libkmod.l* -l
-rwxr-xr-x 1 root root 924 Apr 17 04:55 /usr/lib/libkmod.la
$ ls /usr/lib/libkmod.* -l
-rwxr-xr-x 1 root root    924 Apr 17 04:55 /usr/lib/libkmod.la
lrwxrwxrwx  1  root  root         16  Apr  17  04:55  /usr/lib/
libkmod.so -> libkmod.so.2.2.3
lrwxrwxrwx 1 root root     16 Apr 17 04:55 /usr/lib/libkmod.so.
2 -> libkmod.so.2.2.3
-rwxr-xr-x 1 root root 313349 Apr 17 04:55 /usr/lib/libkmod.so.
2.2.3

$ ls /usr/bin/kmod  -l
-rwxr-xr-x 1 root root 233584 Apr 17 04:55 /usr/bin/kmod
$ file /usr/bin/kmod
/usr/bin/kmod: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linke
Linux 2.6.24, BuildID[sha1]=0x9d4131d1eb78b1e1852cc5ad44f06417ae3caa3c, not stripped
$ kmod
missing command
kmod - Manage kernel modules: list, load, unload, etc
Usage:
    kmod [options] command [command_options]
```

```
Options:
    -V, --version     show version
    -h, --help        show this help

Commands:
  help         Show help message
  list         list currently loaded modules
 static-nodes outputs the static-node information installed with the currently running ker

kmod also handles gracefully if called from following symlinks:
  lsmod        compat lsmod command
  rmmod        compat rmmod command
  insmod       compat insmod command
  modinfo      compat modinfo command
  modprobe     compat modprobe command
  depmod       compat depmod command


$ ls /usr/include/libkmod.h -l
-rw-r--r-- 1 root root 9429 Apr 17 04:55 /usr/include/libkmod.h
$

$ ls /usr/lib/pkgconfig/libkmod.pc -l
-rw-r--r-- 1 root root 210 Apr 17 04:55 /usr/lib/pkgconfig/
libkmod.pc
$ cat /usr/lib/pkgconfig/libkmod.pc
prefix=/usr
exec_prefix=/usr
libdir=/usr/lib
includedir=/usr/include

Name: libkmod
Description: Library to deal with kernel modules
Version: 13
Libs: -L${libdir} -lkmod
Libs.private:
Cflags: -I${includedir}
$

$ ls /usr/share/man/man5/ -l | grep "Apr 17"
-rw-r--r-- 1 root root  3969 Apr 17 04:55 depmod.d.5
-rw-r--r-- 1 root root  9306 Apr 17  2012 fonts-conf.5.gz
-rw-r--r-- 1 root root  1599 Apr 17  2012 initramfs.conf.5.gz
-rw-r--r-- 1 root root  8059 Apr 17 04:55 modprobe.d.5
```

```
-rw-r--r-- 1 root root  2494 Apr 17 04:55 modules.dep.5
-rw-r--r-- 1 root root    18 Apr 17 04:55 modules.dep.bin.5
-rw-r--r-- 1 root root   585 Apr 17  2012 update-initramfs.conf.
5.gz
$ ls /usr/share/man/man8/ -l | grep "Apr 17"
-rw-r--r-- 1 root root  6398 Apr 17 04:55 depmod.8
-rw-r--r-- 1 root root  5170 Apr 17  2012 initramfs-tools.8.gz
-rw-r--r-- 1 root root  2151 Apr 17 04:55 insmod.8
-rw-r--r-- 1 root root   526 Apr 17  2012 lsinitramfs.8.gz
-rw-r--r-- 1 root root  1839 Apr 17 04:55 lsmod.8
-rw-r--r-- 1 root root  1570 Apr 17  2012 mkinitramfs.8.gz
-rw-r--r-- 1 root root  4009 Apr 17 04:55 modinfo.8
-rw-r--r-- 1 root root 10618 Apr 17 04:55 modprobe.8
-rw-r--r-- 1 root root  3058 Apr 17 04:55 rmmod.8
-rw-r--r-- 1 root root  1016 Apr 17  2012 update-initramfs.8.gz
$
```

功能简介

- libkmod.so
  - kmod 库的共享库文件，用于动态链接。
- libkmod.la
  - 用 libtool 工具生成的库文件，其实就是一个文本文件，记录同名共享库的相关信息
  - libtool 工具的作用，是在编译大型软件的过程中解决了库的依赖问题。
  - 特别是在交叉编译的条件下，解决动态链接器如何去寻找共享库的问题。
- kmod
  - 一个管理内核模块的工具，提供列表list，加载load，卸载unload等功能。
  - 目前的版本似乎只支持 help, list, static_nodes 三条命令
  - help 列出帮助信息
  - list 列出当前加载模块
  - static-nodes 输出当前内核加载的 static-node 信息，包括设备节点文件名，类型，主设备号和次设备号。
- libkmod.h

  - 使用 libkmod 库所需要包含的头文件，详细接口定义见下节--项目代码分析。

- libkmod.pc
  - 文本文件，包含了使用 libkmod 库所需要了解的一些信息，例如 安装目录，头文件所在目录，库名称，描述等。

- man5 & man8
  - 提供通过类似 man 8 insmod 命令来查看帮助的源文件 inssmod.8
  - 提供通过类似 man 5 depmod.d 命令来查看帮助的源文件 depmod.d.5

3. 项目代码分析

源码目录结构

- tools
  - insmod.c
  - rmmod.c
  - lsmod.c
  - depmod.c
  - modinfo.c
  - modprobe.c
  - kmod.c
  - kmod.h
  - log.c
  - log.h
  - static-nodes.c

- libkmod
  - COPYING
  - docs
  - libkmod-array.c
  - libkmod-array.h
  - libkmod.c
  - libkmod-config.c

- libkmod-elf.c
- libkmod-file.c
- libkmod.h
- libkmod-hash.c
- libkmod-hash.h
- libkmod-index.c
- libkmod-index.h
- libkmod-list.c
- libkmod-module.c
- libkmod.pc.in
- libkmod-private.h
- libkmod-signature.c
- libkmod.sym
- libkmod-util.c
- libkmod-util.h
- macro.h
- missing.h
- README

• testsuite

- COPYING
- delete_module.c
- init_module.c
- mkdir.c
- mkdir.h
- path.c
- README
- rootfs-pristine
- stripped-module.h
- test-alias.c
- test-blacklist.c
- test-dependencies.c
- test-depmod.c
- test-init.c
- test-loaded.c

- test-modinfo.c
- test-modprobe.c
- test-new-module.c
- testsuite.c
- testsuite.h
- test-testsuite.c
- uname.c

- m4
  - attributes.m4

- man
  - depmod.d.xml
  - depmod.xml
  - insmod.xml
  - lsmod.xml
  - Makefile.am
  - modinfo.xml
  - modprobe.d.xml
  - modprobe.xml
  - modules.dep.xml
  - rmmod.xml

头文件分析

```
$ cat /usr/include/libkmod.h

/*
 * libkmod - interface to kernel module operations
 *
 * Copyright (C) 2011-2013  ProFUSION embedded systems
 *
 * This library is free software; you can redistribute it and/
or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.
 *
```

```
 * This library is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with this library; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301  USA
 */

#pragma once
#ifndef _LIBKMOD_H_
#define _LIBKMOD_H_

#include <fcntl.h>
#include <stdarg.h>
#include <stdbool.h>
#include <inttypes.h>

#ifdef __cplusplus
extern "C" {
#endif

/*
 * kmod_ctx
 *
 * library user context - reads the config and system
 * environment, user variables, allows custom logging
 */
struct kmod_ctx;
struct kmod_ctx *kmod_new(const char *dirname, const char * const *config_paths);
struct kmod_ctx *kmod_ref(struct kmod_ctx *ctx);
struct kmod_ctx *kmod_unref(struct kmod_ctx *ctx);
void kmod_set_log_fn(struct kmod_ctx *ctx,
            void (*log_fn)(void *log_data,
                    int priority, const char *file, int line,
                    const char *fn, const char *format,
                    va_list args),
            const void *data);
int kmod_get_log_priority(const struct kmod_ctx *ctx);
void kmod_set_log_priority(struct kmod_ctx *ctx, int priority);
void *kmod_get_userdata(const struct kmod_ctx *ctx);
void kmod_set_userdata(struct kmod_ctx *ctx, const void *userdata);
```

```
/*
 * Management of libkmod's resources
 */
int kmod_load_resources(struct kmod_ctx *ctx);
void kmod_unload_resources(struct kmod_ctx *ctx);

enum kmod_resources {
    KMOD_RESOURCES_OK = 0,
    KMOD_RESOURCES_MUST_RELOAD = 1,
    KMOD_RESOURCES_MUST_RECREATE = 2,
};
int kmod_validate_resources(struct kmod_ctx *ctx);

enum kmod_index {
    KMOD_INDEX_MODULES_DEP = 0,
    KMOD_INDEX_MODULES_ALIAS,
    KMOD_INDEX_MODULES_SYMBOL,
    KMOD_INDEX_MODULES_BUILTIN,
    /* Padding to make sure enum is not mapped to char */
    _KMOD_INDEX_PAD = (1 << 31),
};
int kmod_dump_index(struct kmod_ctx *ctx, enum kmod_index type, int fd);

/*
 * kmod_list
 *
 * access to kmod generated lists
 */
struct kmod_list;
struct kmod_list *kmod_list_next(const struct kmod_list *list,
                         const struct kmod_list *curr);
struct kmod_list *kmod_list_prev(const struct kmod_list *list,
                         const struct kmod_list *curr);
struct kmod_list *kmod_list_last(const struct kmod_list *list);

#define kmod_list_foreach(list_entry, first_entry) \
    for (list_entry = first_entry; \
        list_entry != NULL; \
        list_entry = kmod_list_next(first_entry, list_entry))

#define kmod_list_foreach_reverse(list_entry, first_entry) \
    for (list_entry = kmod_list_last(first_entry); \
        list_entry != NULL; \
```

```
        list_entry = kmod_list_prev(first_entry, list_entry))

/*
 * kmod_config_iter
 *
 * access to configuration lists - it allows to get each configuration's
 * key/value stored by kmod
 */
struct kmod_config_iter;
struct kmod_config_iter *kmod_config_get_blacklists(const struct kmod_ctx *ctx);
struct kmod_config_iter *kmod_config_get_install_commands(const struct kmod_ctx *ctx);
struct kmod_config_iter *kmod_config_get_remove_commands(const struct kmod_ctx *ctx);
struct kmod_config_iter *kmod_config_get_aliases(const struct kmod_ctx *ctx);
struct kmod_config_iter *kmod_config_get_options(const struct kmod_ctx *ctx);
struct kmod_config_iter *kmod_config_get_softdeps(const struct kmod_ctx *ctx);
const char *kmod_config_iter_get_key(const struct kmod_config_iter *iter);
const char *kmod_config_iter_get_value(const struct kmod_config_iter *iter);
bool kmod_config_iter_next(struct kmod_config_iter *iter);
void kmod_config_iter_free_iter(struct kmod_config_iter *iter);

/*
 * kmod_module
 *
 * Operate on kernel modules
 */
struct kmod_module;
int kmod_module_new_from_name(struct kmod_ctx *ctx, const char *name,
                          struct kmod_module **mod);
int kmod_module_new_from_path(struct kmod_ctx *ctx, const char *path,
                          struct kmod_module **mod);
int kmod_module_new_from_lookup(struct kmod_ctx *ctx, const char *given_alias,
                          struct kmod_list **list);
int kmod_module_new_from_loaded(struct kmod_ctx *ctx,
                          struct kmod_list **list);

struct kmod_module *kmod_module_ref(struct kmod_module *mod);
struct kmod_module *kmod_module_unref(struct kmod_module *mod);
int kmod_module_unref_list(struct kmod_list *list);
struct kmod_module *kmod_module_get_module(const struct kmod_list *entry);


/* Removal flags */
enum kmod_remove {
    KMOD_REMOVE_FORCE = O_TRUNC,
```

```
    KMOD_REMOVE_NOWAIT = O_NONBLOCK,
};

/* Insertion flags */
enum kmod_insert {
    KMOD_INSERT_FORCE_VERMAGIC = 0x1,
    KMOD_INSERT_FORCE_MODVERSION = 0x2,
};

/* Flags to kmod_module_probe_insert_module() */
enum kmod_probe {
    KMOD_PROBE_FORCE_VERMAGIC =     0x00001,
    KMOD_PROBE_FORCE_MODVERSION =       0x00002,
    KMOD_PROBE_IGNORE_COMMAND =     0x00004,
    KMOD_PROBE_IGNORE_LOADED =      0x00008,
    KMOD_PROBE_DRY_RUN =            0x00010,
    KMOD_PROBE_FAIL_ON_LOADED =     0x00020,

    /* codes below can be used in return value, too */
    KMOD_PROBE_APPLY_BLACKLIST_ALL =    0x10000,
    KMOD_PROBE_APPLY_BLACKLIST =        0x20000,
    KMOD_PROBE_APPLY_BLACKLIST_ALIAS_ONLY = 0x40000,
};

/* Flags to kmod_module_apply_filter() */
enum kmod_filter {
    KMOD_FILTER_BLACKLIST = 0x00001,
    KMOD_FILTER_BUILTIN = 0x00002,
};

int kmod_module_remove_module(struct kmod_module *mod, unsigned int flags);
int kmod_module_insert_module(struct kmod_module *mod, unsigned int flags,
                              const char *options);
int kmod_module_probe_insert_module(struct kmod_module *mod,
            unsigned int flags, const char *extra_options,
            int (*run_install)(struct kmod_module *m,
                               const char *cmdline, void *data),
            const void *data,
        void (*print_action)(struct kmod_module *m, bool install,
                               const char *options));


const char *kmod_module_get_name(const struct kmod_module *mod);
const char *kmod_module_get_path(const struct kmod_module *mod);
```

```
const char *kmod_module_get_options(const struct kmod_module *mod);
const char *kmod_module_get_install_commands(const struct kmod_module *mod);
const char *kmod_module_get_remove_commands(const struct kmod_module *mod);
struct kmod_list *kmod_module_get_dependencies(const struct kmod_module *mod);
int kmod_module_get_softdeps(const struct kmod_module *mod,
              struct kmod_list **pre, struct kmod_list **post);
int kmod_module_get_filtered_blacklist(const struct kmod_ctx *ctx,
                    const struct kmod_list *input,
             struct kmod_list **output) __attribute__ ((deprecated));
int kmod_module_apply_filter(const struct kmod_ctx *ctx,
                    enum kmod_filter filter_type,
                    const struct kmod_list *input,
                    struct kmod_list **output);



/*
 * Information regarding "live information" from module's state, as returned
 * by kernel
 */

enum kmod_module_initstate {
    KMOD_MODULE_BUILTIN = 0,
    KMOD_MODULE_LIVE,
    KMOD_MODULE_COMING,
    KMOD_MODULE_GOING,
    /* Padding to make sure enum is not mapped to char */
    _KMOD_MODULE_PAD = (1 << 31),
};
const char *kmod_module_initstate_str(enum kmod_module_initstate state);
int kmod_module_get_initstate(const struct kmod_module *mod);
int kmod_module_get_refcnt(const struct kmod_module *mod);
struct kmod_list *kmod_module_get_holders(const struct kmod_module *mod);
struct kmod_list *kmod_module_get_sections(const struct kmod_module *mod);
const char *kmod_module_section_get_name(const struct kmod_list *entry);
unsigned long kmod_module_section_get_address(const struct kmod_list *entry);
void kmod_module_section_free_list(struct kmod_list *list);
long kmod_module_get_size(const struct kmod_module *mod);



/*
 * Information retrieved from ELF headers and sections
 */
```

```
int kmod_module_get_info(const struct kmod_module *mod, struct kmod_list **list);
const char *kmod_module_info_get_key(const struct kmod_list *entry);
const char *kmod_module_info_get_value(const struct kmod_list *entry);
void kmod_module_info_free_list(struct kmod_list *list);

int kmod_module_get_versions(const struct kmod_module *mod, struct kmod_list **list);
const char *kmod_module_version_get_symbol(const struct kmod_list *entry);
uint64_t kmod_module_version_get_crc(const struct kmod_list *entry);
void kmod_module_versions_free_list(struct kmod_list *list);

int kmod_module_get_symbols(const struct kmod_module *mod, struct kmod_list **list);
const char *kmod_module_symbol_get_symbol(const struct kmod_list *entry);
uint64_t kmod_module_symbol_get_crc(const struct kmod_list *entry);
void kmod_module_symbols_free_list(struct kmod_list *list);

enum kmod_symbol_bind {
    KMOD_SYMBOL_NONE = '\0',
    KMOD_SYMBOL_LOCAL = 'L',
    KMOD_SYMBOL_GLOBAL = 'G',
    KMOD_SYMBOL_WEAK = 'W',
    KMOD_SYMBOL_UNDEF = 'U'
};

int kmod_module_get_dependency_symbols(const struct kmod_module *mod, struct kmod_list **
const char *kmod_module_dependency_symbol_get_symbol(const struct kmod_list *entry);
int kmod_module_dependency_symbol_get_bind(const struct kmod_list *entry);
uint64_t kmod_module_dependency_symbol_get_crc(const struct kmod_list *entry);
void kmod_module_dependency_symbols_free_list(struct kmod_list *list);

#ifdef __cplusplus
} /* extern "C" */
#endif
#endif
$
```

接口设计

重要接口实现

调用流程图