Kernel Debugging User-Space API Libarary (KDUAL)

John Livingston
TJHSST Computer Systems Research 2005

Abstract

- Creation of a library to emulate the Linux kernel for software development
- Current development cycle of kernel modules is inefficient; running kernels are hard to debug
- Extensive research into the structure of the Linux kernel and its development
- Programs can now be compiled against the library

Background

- Debugging kernel modules is especially tedious because a problem can often result in a total system crash
- Saving crash information is difficult when your operating system isn't running anymore
- Improving the module development process will be a great benefit to kernel developers

Background

- The scope of the project is to create a debugging library that can simulate sections of a running kernel
- Cannot simulate hardware interactions
- Library will allow “sandbox” testing of module code without risk of a system crash

KDUAL

- Emulates some functions of the Linux kernel
- High-speed atomic mathematics
- Locking
- Virtual Filesystem Switch (VFS)
- Extensible structure

KDUAL Examples

Atomic Addition:

```
static __inline__ void kc_atomic_add
    (int i, atomic_t *v) {
    __asm__ __volatile__(
        LOCK "addl %1,%0"
        :"=m" (v->counter)
        :"ir" (i), "m" (v->counter));
}
```

KDUAL Examples

Making coffee:

```
ifneq ($(strip $(filter coffee, $(MAKECMDGOALS))),)
override COFFEE_MSG_NUM=$(shell expr “$$RANDOM” '%' '4')
coffee:
ifeq ($(COFFEE_MSG_NUM),0)
	@echo EBORK: Coffee maker is broken!
else
ifeq ($(COFFEE_MSG_NUM),1)
	@echo ENOCFE: Out of coffee!
else
ifeq ($(COFFEE_MSG_NUM),2)
	@echo ENOMY: Insufficient payment!
else
	@echo ETOMCH: Cup overflow error!
endif
endif
endif
endif
```

Conclusion

- Design and base implementation successful
- Broader functionality needed