

Praktikum angewandte Systemsoftwaretechnik

Aufgabe 3: ~~sehr verspätete~~ Ostereier- Kernelbug-Suche

Alexander Würstlein, Moritz Strübe, Sebastian Maier

Lehrstuhl Informatik 4

30. Oktober 2014

Debuggen des Linux-Kernels

- kgdb nicht für alle Fehlertypen der beste Ansatz
 - Was tun, wenn der Kernel sehr früh Oopst?
 - Wie vorgehen, wenn die serielle Schnittstelle debugged werden soll?
 - Wenn alles „optimized out“ ist, wie sehe ich trotzdem was passiert?
- Lösung: printk()-Ausgaben auf der Konsole und im Kernel-Log

Prototyp:

```
int printk(const char *s, ...)
```

Beispiel aus linux-3.0/init/main.c:

```
printk(KERN_NOTICE "Kernel command line: %s\n", boot_command_line);
```

Meldungen „nachlesen“:

```
passt [~]> dmesg
...
[    0.000000] Kernel command line: root=/dev/vda1 console=ttyS0
...
```

Debuggen mit printk()

- Alle Ausgaben haben eine Priorität (<n> am Stringanfang)
- Kernel Log-Level muss für Ausgabe mindestens auf $n + 1$ gesetzt sein
- Log-Level wird standardmäßig mit 7 initialisiert
- Anpassung über Kommandozeile (debug, loglevel) und klogd(8)

Mögliche Prioritäten (linux-3.0/include/linux/printk.h):

```
#define KERN_EMERG      "<0>" /* system is unusable */
#define KERN_ALERT     "<1>" /* action must be taken immediately */
#define KERN_CRIT      "<2>" /* critical conditions */
#define KERN_ERR        "<3>" /* error conditions */
#define KERN_WARNING    "<4>" /* warning conditions */
#define KERN_NOTICE     "<5>" /* normal but significant condition */
#define KERN_INFO       "<6>" /* informational */
#define KERN_DEBUG      "<7>" /* debug-level messages */
```

Beispiel aus linux-3.0/kernel/panic.c:

```
printk(KERN_EMERG "Kernel panic - not syncing: %s\n",buf);
```

Code-Navigation mit cscope

```
make cscope  
cscope -d
```

- Funktionen finden
- Aufrufer und Aufgerufene
- Integration in Editoren
- ähnliche Funktionalität per Eclipse, etc.

Übungsaufgabe #3: Fehler finden und beheben

- Vorgegebene Kernelquellen mit injizierten Fehlern:
/proj/i4passt/kernel/ws14/passt-linux-borked.git
/proj/i4passt/kernel/ws14/passt-WS14-linux-borked.tar.gz
- Verschiedene Fehlertypen, *nicht nur* Systemabstürze
 - System muss „normal“ benutzt werden um alle Fehler zu finden
 - kgdb nicht immer das optimale Werkzeug
- Insgesamt zwölf verschiedene Fehler
- Patches an `linux-kernel@i4.cs.fau.de`

Bearbeitungszeit bis: 2014-11-28