



## (Vor-) Geschichte von UNIX

- Betriebssystem UNIX wurde von Sysadmins entwickelt
- Timesharing und Multiuserfähigkeit waren Ziele
- erste Versionen in Assembler
- zwecks einfacher Portierung wurde C entwickelt



## Systemadministration heute

- Bash/Perl/Python
- Mächtigkeit durch alle CLI-Tools
- Verknüpfungen über Pipes, Command Substitutions und Programmlogik
- Komplexe Funktionalität trotz kleinen Source Codes
- Langsam (Bash)
- manchmal hohe Load



## Vorteile C vs. Bash 1

- hohe Geschwindigkeit, niedrige Load
- Numerik besser
- bessere Stringverarbeitung (single chars)
- Systemaufrufe verwendbar, die in Bash nicht verfügbar sind (fork(), clone(), lseek() etc)
- (GNU-) Tools für spezielle Fälle unzureichend oder zu langsam
- kleine C-Programme können Bash-Scripts hervorragend ergänzen

## Vorteile C vs. Bash 2

Manchmal ist C-Code einfacher als Bash-Code. Hier eine beispielhafte Aufgabe:

Finde Dateien in einer Liste, deren Namen auf ".jpg" oder ".JPG" endet, an 4. Stelle kein "\_" hat und eine Namenslänge von 11 bis 14 Zeichen hat.

## sleep < 1s, msleep250.c

58 Bytes C-Code für ein Sleep von 250 ms sind einfach:

```
linux:/root # cat msleep250.c
#include <unistd.h>
int main()
{ return(usleep(250000)); }
```

## sleep coreutils

/bin/sleep aus neueren coreutils kennt Sekundenbruchteile (/bin/sleep .25):

```
linux:/root # du -b /bin/sleep
26000 /bin/sleep
linux:/root # ldd /bin/sleep
linux-gate.so.1 => (0x008ce000)
libc.so.6 => /lib/libc.so.6
(0x00a5d000)
/lib/ld-linux.so.2 (0x00a3b000)
```

## RedHat usleep

RedHat usleep:

```
linux:/root # du -b /bin/usleep  
7096 /bin/usleep  
linux:/root # ldd /bin/usleep  
linux-gate.so.1 => (0x00d4d000)  
libpopt.so.0 => /lib/libpopt.so.0  
(0x02a16000)  
libc.so.6 => /lib/libc.so.6  
(0x00a5d000)  
/lib/ld-linux.so.2 (0x00a3b000)
```

9

UNijenhuis 10/2003

## eigenes \*sleep

```
linux:/root # du -b *sleep*  
845 usleep.c  
564 usleep  
58 msleep250.c  
344 msleep250  
619 msleep100.asm  
118 msleep100  
linux:/root # ldd usleep msleep*0  
usleep: not a dynamic executable  
msleep250: not a dynamic executable  
msleep100: not a dynamic executable
```

10

UNijenhuis 10/2003

## daemonize 1

Nach Helmut Herold hat ein Daemon folgende Eigenschaften:

- Der init-Prozess ist der Vaterprozess.
- Der Prozess hat kein Kontrollterminal.
- Der Prozess ist Prozessgruppenführer.

(Die UID ist unwichtig.)

11

UNijenhuis 10/2003

## daemonize 2

```
pid = fork();  
if (pid < 0) { /* Fehler */  
    _exit(1);  
} else if (pid != 0) { /* Vater */  
    _exit(0);  
}  
setsid();  
umask(027);  
chdir("/");  
close(0); close(1); close(2);  
execve(argv[1], argv+1, 0);
```

12

UNijenhuis 10/2003

## dir entries 1

Ursachen für volle Verzeichnisse:

- Softwarefehler
- Konfigurationsfehler
- Unachtsamkeit (webcam-Verzeichnis)

Auswirkungen:

Bei ext2/3 FS werden Zugriffe überproportional langsam. I/O-Wait und damit Load steigt.

13

UNijenhuis 10/2009

## dir entries 3

```
linux:/root # time direntries  
500002  
real 0m0.631s  
linux:/root # time find .|wc -l  
500001  
real 0m1.632s  
linux:/root # time ls >/dev/null  
real 0m8.084s
```

15

UNijenhuis 10/2009

## dir entries 2

Symptome, beispielsweise "hängendes" ls:

```
linux:/root # ls  
^C^C^C^C^Z^Z  
^Z^Z^C^Z^C  
^C^Z
```

Prüfen mit "direntries":

```
linux:/root # time direntries  
400004  
real 0m0.572s
```

14

UNijenhuis 10/2009

## direntries.c 1

Zählen:

```
dptr = opendir(dir);  
if (dptr == 0) my_exit(1);  
for(count=0;  
    (eptr=readdir(dptr));  
    count++);  
closedir(dptr);  
putlong(count);
```

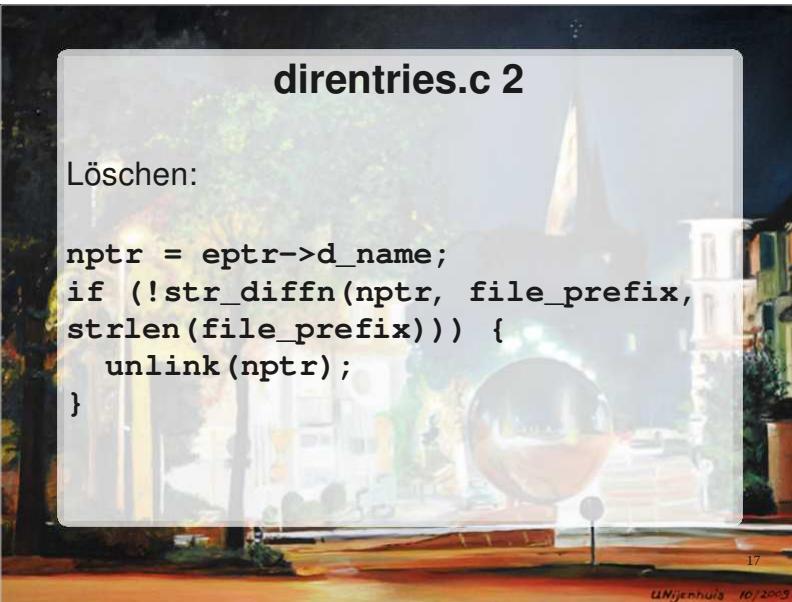
16

UNijenhuis 10/2009

## direntries.c 2

Löschen:

```
nptr = eptr->d_name;
if (!str_diffn(nptr, file_prefix,
strlen(file_prefix))) {
    unlink(nptr);
}
```



17

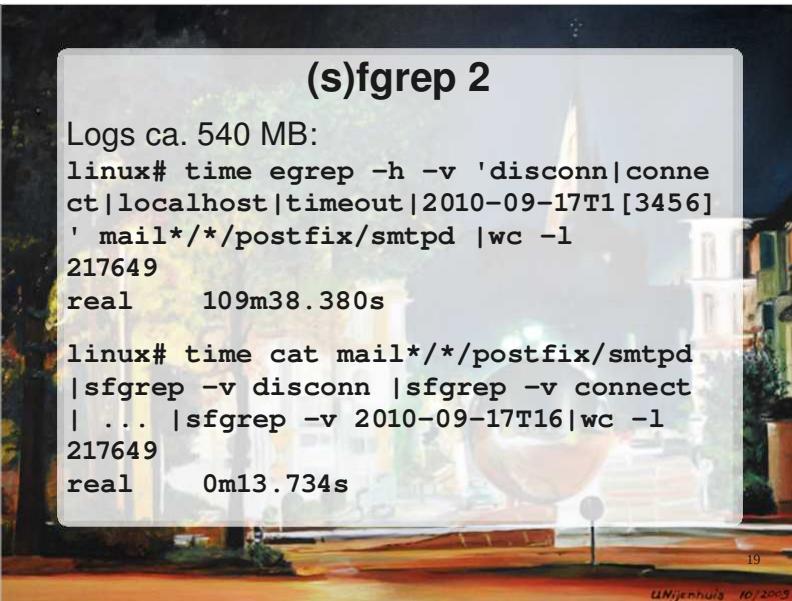
UHjehuis 10/2009

## (s)fgrep 2

Logs ca. 540 MB:

```
linux# time egrep -h -v 'disconn|connect|localhost|timeout|2010-09-17T1[3456]' mail*/*/postfix/smtpd |wc -l
217649
real    109m38.380s

linux# time cat mail*/*/postfix/smtpd |sfgrep -v disconn |sfgrep -v connect | ... |sfgrep -v 2010-09-17T16|wc -l
217649
real    0m13.734s
```



19

UHjehuis 10/2009

## (s)fgrep

noqueue-20091118 ca. 1,4 GB durchsuchen:

```
linux:/root # time fgrep
"bildmhiqbwl@braeviewrealty.com>
to=<romano" noqueue-20091118
real    1m51.225s

linux:/root # time sfgrep "..."
noqueue-20091118
real    0m4.882s
```



18

UHjehuis 10/2009

## Systemadministration mit C

ENDE

Danke für's Dabeisein.

Noch Fragen?

 tuxad.com  
Linux Systemhaus Herford

**OPEN  
RHEIN  
RUHR**  
ein Platz voll Software

 open  
source  
PRESS



20

UHjehuis 10/2009