Übung zu Betriebssysteme

Test- & Entwicklungsumgebung

Wintersemester 2020/21

Bernhard Heinloth & Christian Eichler

Lehrstuhl für Informatik 4
Friedrich-Alexander-Universität Erlangen-Nürnberg

Lehrstuhl für Verteilte Systeme
und Betriebssysteme

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Testumgebung

- Virtualisiert
  - **QEMU** Softwareemulation
  - **KVM** Hardware-Virtualisierung
- Nackte Hardware (*bare-metal*)
  - vier (identische) Testrechner
    - Intel© Core® i5 CPU (2 Kerne @ 3.2 GHz / 4 Threads (HT))
    - 8 GB Arbeitsspeicher
    - COM1 verbunden mit `cip6d0.cip.cs.fau.de` (`/dev/ttyBSX`)
  - Boot via Netzwerk (PXE)
  - Entfernter Zugriff (passwortgeschützt)
    - VNC: `i4stubs.cs.fau.de:5901 - 5904`
    - Weboberfläche: `https://i4stubs.cs.fau.de`
  - Koordination via IRC: `#i4stubs` (im IRCnet)
  - weitere Testrechner (mit 8 Threads) verfügbar
  - *privater PC (theoretisch) auch möglich*

Systemvoraussetzung

Minimal:

- Internet
- SSH Zugang in CIP
- *oder* Webbrowser für `remote.cip.cs.fau.de`

Optimal:

- PC mit x64 Architektur
- Unixoides System
  - Referenzsysteme: **Debian 10** und **Ubuntu 20.04**
  - Unter Windows **WSL** oder **VirtualBox** möglich
- Möglichkeit Softwarepakete zu installieren

Entwicklungsumgebung

- Aktueller Übersetzer (für x64)
  - Bevorzugt GCC (≥ 7)
  - Alternativ LLVM/CLANG (≥ 7)
  - Via Docker verfügbar: `inf4/stubs`
- Assemblierer NETWIDE ASSEMBLER (NASM)
- Buildtools (u.a. MAKE, `objcopy`)
- Emulator QEMU/KVM (für x86_64 Gast)
- Debugger GDB
- *Optional:* Python für `cpplint`
- *Optional:* GNU GRUB & GNU XORRISO für ISO

Wichtige Makefile Targets

make qemu	QEMU **ohne** Hardware-Virtualisierung
make kvm	QEMU **mit** Hardware-Virtualisierung
make qemu-gdb	starte in QEMU und verbinde zu integrierten GDB-Stub ➔ Fehlersuche mit gdb
make kvm-gdb	selbiges mit Hardware-Virtualisierung
make netboot	für Boot am Test-Rechner ins NFS kopieren

Suffix **-dbg** für Debugtransparente Optimierungen (`-Og` & Framepointer)
Suffix **-noopt** um Optimierungen auszuschalten (sonst `-O3`)
Suffix **-opt** für aggressive Optimierungen (`-Ofast` und LTO)
Suffix **-verbose** aktiviert zusätzliche Ausgaben (`DBG_VERBOSE`)

Weitere Makefile Targets

make iso	erstelle ein bootfähiges ISO-Abbild
make qemu-iso	boote das Abbild in QEMU
make usb-*dev*	bootfähigen USB-Stick auf ***dev*** (z.B. *sdb* - aber Vorsicht!) erstellen (als Superuser)
make solution-#	Musterlösung zur Aufgabe **#** mit Hardware-Virtualisierung starten (auf Testrechner bereits installiert)
make lint	prüft den Konformität des Coding Styles
make help	zeige eine Beschreibung der verfügbaren Targets an

Arbeiten mit QEMU/KVM

- lokale Ausführung mit **GTK** Frontend: **Linke** Ctrl + Alt Tasten
  - Maus mittels Ctrl + Alt + g *befreien*
  - Vollbild umschalten mit Ctrl + Alt + f
  - Beenden mit Ctrl + Alt + q
- lokale Ausführung mit (altem) **SDL** Frontend: **Rechte** Ctrl + Alt Tasten
  - Maus mittels Ctrl + Alt Kombination *befreien*
  - Vollbild mit Ctrl + Alt + f umschalten
  - Beenden durch wechseln in den Monitor mit Ctrl + Alt + 2 und anschließendem q
- **Curses** Frontend (z.B. bei SSH): **Linke** Alt-Taste
  - Beenden durch wechseln in den Monitor mit Alt + 2 und anschließendem q
  - Zurück zur Ausgabe mit Alt + 1

Passwortlose SSH-Verbindung

Öffentlichen & privaten Schlüssel mit `ssh-keygen` generieren:

```
heinloth:~/beispiel$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/heinloth/.ssh/id_rsa):
Created directory '/home/heinloth/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/heinloth/.ssh/id_rsa
Your public key has been saved in /home/heinloth/.ssh/id_rsa.pub
```

Öffentlichen Schlüssel mit `ssh-copy-id` in den CIP kopieren:

```
heinloth:~/beispiel$ ssh-copy-id uj66ojab@cip6d0.cip.cs.fau.de
```

(Fügt den Inhalt der lokalen Datei `.ssh/id_rsa.pub` auf dem Zielrechner an das Ende von `.ssh/authorized_keys` an)

Konfiguration für SSH-Verbindung

Benutzerkonfiguration liegt in `$HOME/.ssh/config`:

```
heinloth:~/beispiel$ cat ~/.ssh./config
Host *.cip.cs.fau.de
    User uj66ojab
    IdentityFile ~/.ssh/id_rsa
    ConnectTimeout 5
    Protocol 2
    HashKnownHosts no
    TCPKeepAlive yes
    ControlPath ~/.ssh/controlmasters/%r@%h:%p
    ControlMaster auto
    ControlPersist 10m
    ServerAliveInterval 10
```

Datei anlegen, `User` an den eigenen Login anpassen, Details in der Manpage:

```
heinloth:~/beispiel$ man ssh_config
```

Konfigurationshilfsskript `setup-env.sh` auf der Lehrveranstaltungswebseite unter „Entwicklungsumgebung“