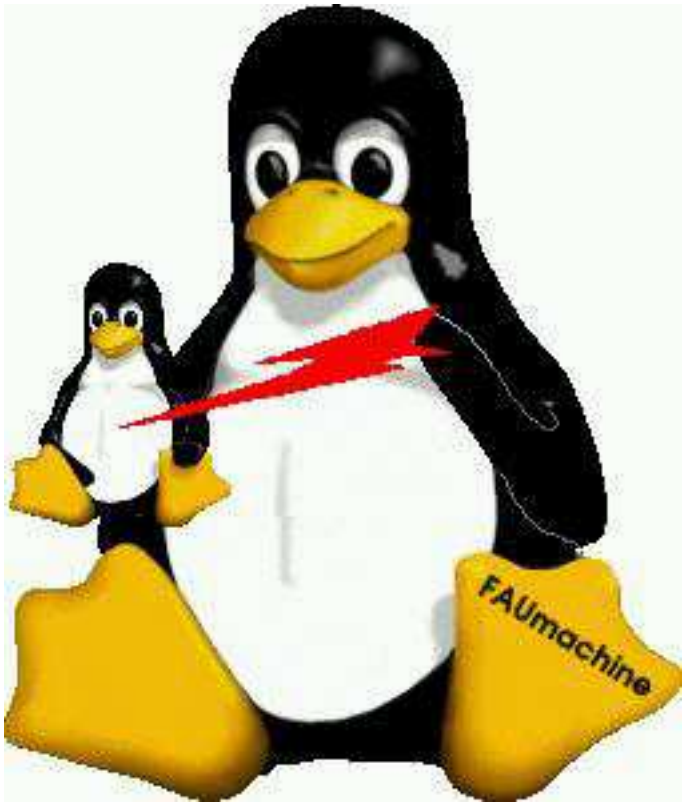
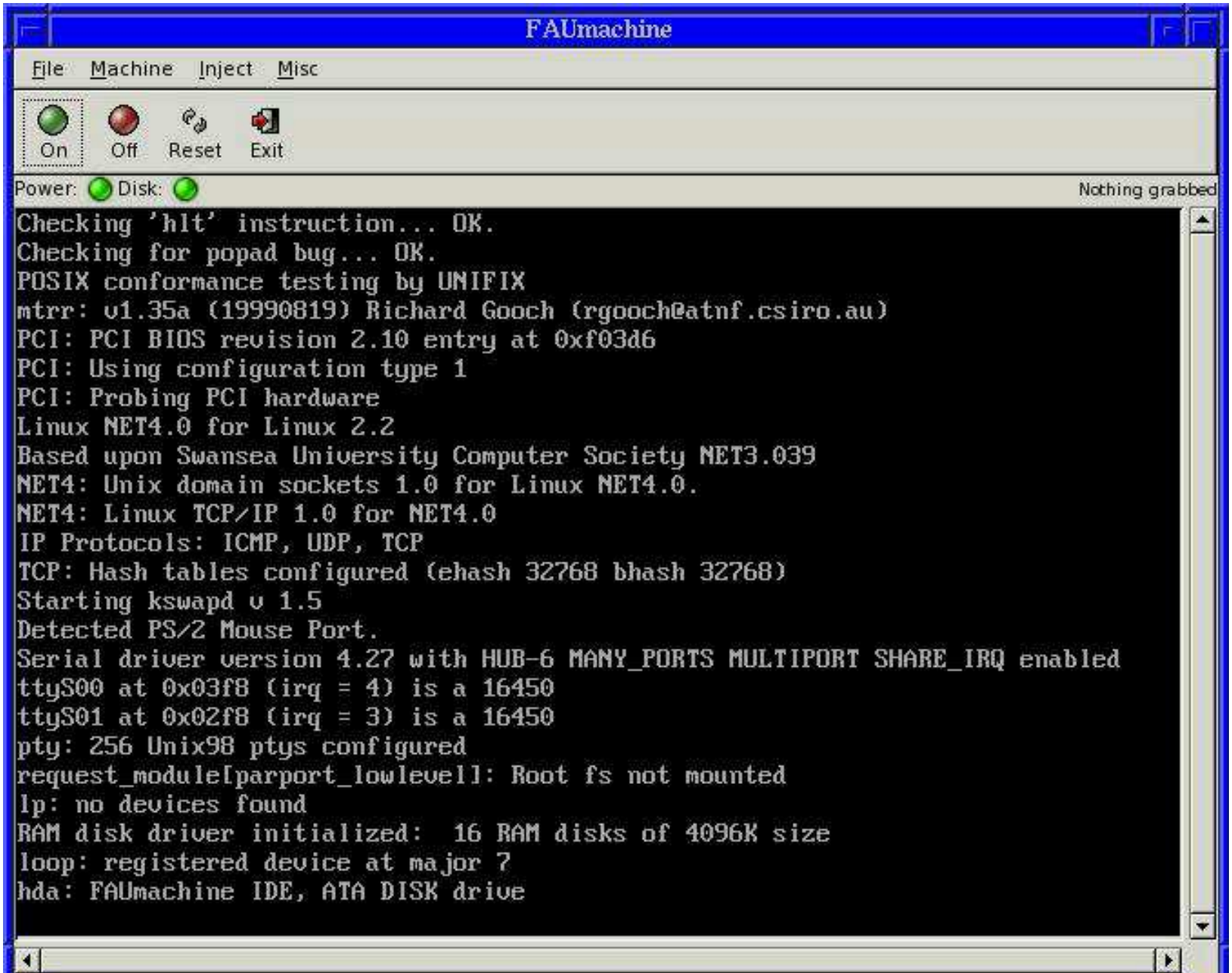
Virtuelle PCs und Netzwerke mit FAUmachine



Hans-Jörg Höxer <Hans-Joerg.Hoexer@yerbouti.franken.de>

Einführung FAUmachine



Übersicht

- Einführung FAUmachine
- Funktionsweise
- Simulierte Hardware
- Fehlerinjektion & Fernsteuern
- Einsatzmöglichkeiten von FAUmachine
- Performance
- Ausblick

Einführung FAUmachine

Was ist FAUmachine?

- virtueller PC mit i386 Prozessor ("PC im Fenster")
- Hardwaresimulator für CPU und Peripherie
- binärcompatibel zum i386
- Linux Kern und Linux-Anwendungen sind lauffähig
- viele wichtige Linux-Distributionen sind installierbar
- andere Gast-Betriebssysteme in Vorbereitung bzw. schon teilweise unterstützt (OpenBSD, DOS & Co., Windows, ...)

Einführung FAUmachine

FAUmachine läuft auf

- Linux ab Version 2.2
  - stabil
- OpenBSD ab Version 3.6
  - noch nicht ganz stabil
- Cygwin/Windows2000
  - in Arbeit, noch sehr experimentell

Quellcode veröffentlicht unter GPL

- siehe http://www.faumachine.org/

Einführung FAUmachine

Motivation

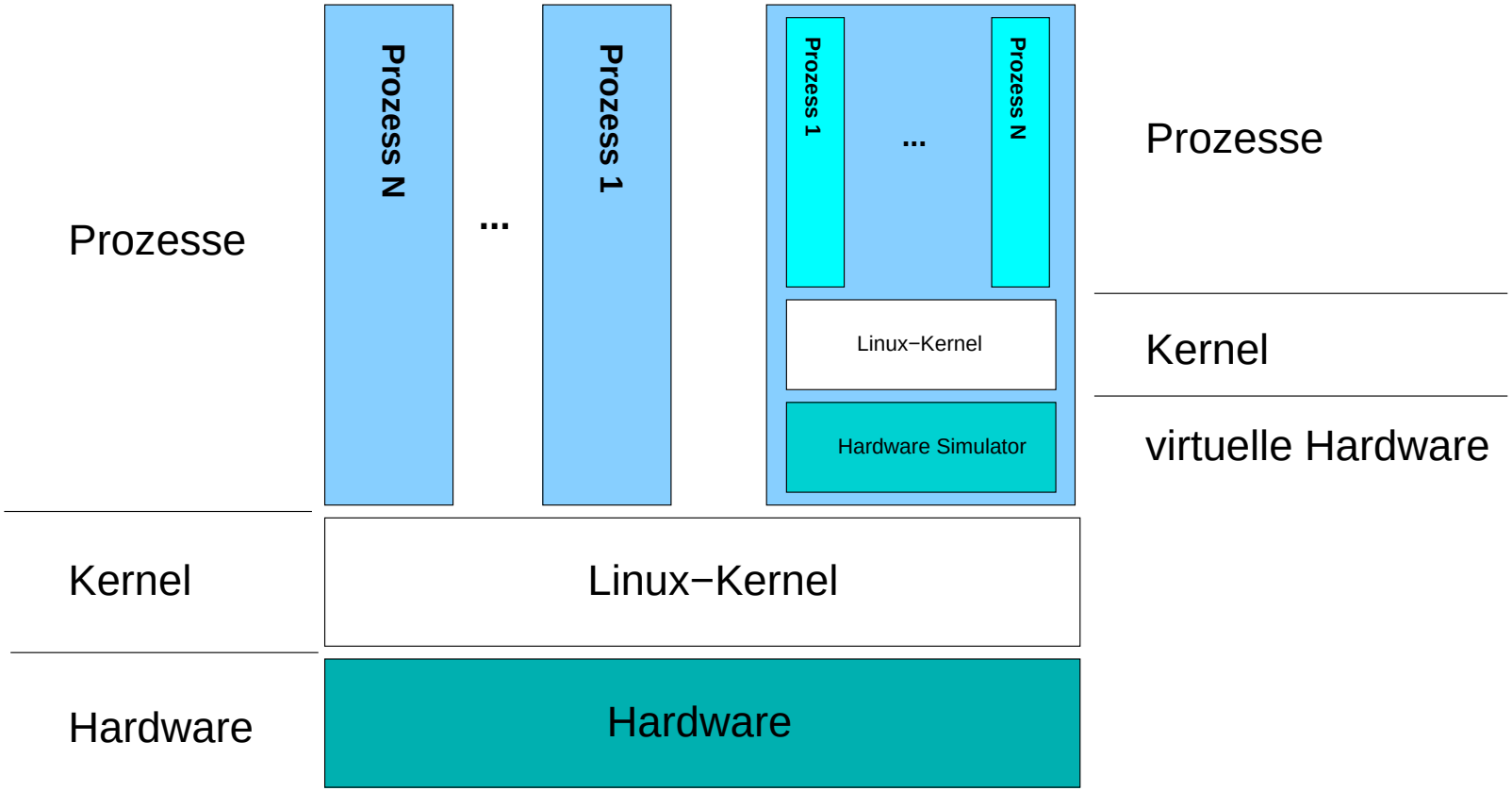
- entwickelt am Lehrstuhl für Informatik 3, FAU Erlangen-Nürnberg
- virtuelle Rechner für Forschungsprojekt benötigt
- Test von fehlertoleranten (vernetzten) Systemen

verwandte Ansätze

- VMware, VirtualPC: kommerziell, kein Quellcode
- Bochs: CPU-Simulation, sehr langsam
- Plex86: beta?
- User-Mode-Linux von Jeff Dike
- Qemu
- BSD Jails

Funktionsweise

Linux auf FAUmachine auf Linux



Funktionsweise

Gast-CPU

- durch Unix-Prozess nachgebildet
- Instruktionen werden direkt von Gastgeber-CPU ausgeführt --- 100% Performance
- nur spezielle Befehle werden durch Simulatoraufrufe ersetzt und vom Hardwaresimulator simuliert

- betrifft aber nur Systemsoftware (z.B. Linux-Kern)

Funktionsweise

Ersetzen und Simulieren von Befehlen

- direkt im Sourcecode
  - kritische Instruktionen identifizieren und mit Präfix (Aufruf in den Simulator) versehen
  - Kontextdiff für Linuxkern etwa 500 Zeilen (XXX)
  - gute Performance
  - Sourcecode nötig
- zur Laufzeit
  - Just-In-Time-Compiler (JIT) erkennt kritische Befehle und fügt Präfix automatisch ein
  - funktioniert auch mit binary-only (System-)Software
  - durch Caching immer noch gute Performance

Funktionsweise

Optimierung des Gastgeber-Kerns

- reduziert Overhead bei der Simulation von Software-Traps (int 0x80, Systemcall)
- bei Linux optional
  - ansonsten wird ptrace(2) verwendet
  - Patch für Vanilla-Kernel
- bei OpenBSD notwendig (macht aber nix)
  - ptrace(2) fehlt PTRACE_SYSCALL
  - ist als LKM implementiert
  - kann in den Kernel der Standardinstallation geladen werden

Funktionsweise

Speicher:

- Datei als physikalischer Speicher
- MMU mit mmap(2) und munmap(2) simuliert
- Adressraum des Prozesses als virtueller Speicher
- Achtung: noch keine Protection implementiert

Peripherie:

- Dateien als Festplatten
- X-Programme als "Monitor/Tastatur"
- Netzwerkkarte kommuniziert über Sockets bzw. Tap-Interface

Simulierte Hardware

- alle nötigen Chipsatzkomponenten
- Tastatur und Maus (PS/2)
- SVGA-Karte (Framebuffer, funktioniert auch mit X)
- IDE (Festplatte, CD, DVD) --- Copy-On-Write
- NE2000 (ISA)
- EEPro100 (PCI)
- Soundblaster und PC-Speaker
- serielle und parallele Schnittstellen
- serielles Terminal
- Modem

- Floppy ist in Arbeit

Fehlerinjektion & automatisches Testen

Simulation von Hardwarefehlern

- Netzwerkkarte verliert Pakete
- Bitflips in CPU/Speicher
- Blöcke auf der Platte defekt

Fernsteuern mittels Skriptsprache

- basierend auf VHDL und expect
- Mustererkenner im Bildschirm und an serieller Schnittstelle
- Systeme können z.B. automatisch installiert werden

FAUmachine Einsatzgebiete

- Softwareentwicklung
- Simulation kompletter Netzwerke
- Testen von Distributionen
- Schulungen
- Sandbox

FAUmachine Einsatzgebiete

Softwareentwicklung

- gleichzeitiges Testen auf verschiedenen Systemen und Distributionen
  - unterschiedliche Versionen von gcc auf verschiedenen Distributionen
  - Abhänigkeiten für RPMs
- Kernel-Entwicklung (Kernel-Replacement)
  - Filesysteme
  - Netzwerkprotokolle
- Assemblerprogrammieren unter DOS ohne umständliches Booten

FAUmachine Einsatzgebiete

Simulation von Netzwerken

- mehrere Rechner (Nodes) auf einem oder mehreren Gastgebern
- Nodes über virtuelles Netz verbunden
- Bridgen in physikalisches Netz möglich
- Node erscheint im Netz wie ein physikalischer Rechner

FAUmachine Einsatzgebiete

Testen von Distributionen

- mehrere Distributionen auf einem Rechner
- kein Umpartitionieren nötig
- leichtes wiederherstellen nach Fehlkonfigurationen
- Ausprobieren von Linux unter OpenBSD :-)

FAUmachine Einsatzgebiete

Schulungen

- jeder Teilnehmer hat eigenes Netzwerk
- root-Rechte auf virtuellem Rechner
- automatisches Erstellen von einheitlichen Setups
- gefahrlos Netzwerkdienste installieren, hacken, etc.

FAUmachine Einsatzgebiete

Sandbox

- Partitionierung eines physikalischen Rechners
  - Dienste auf virtuelle Maschinen verteilen
  - User bekommt eigene virtuelle Maschine
- virtuelle Maschinen sind unprivilegierte Prozesse
  - können als noboy laufen
- Angreifer hackt eine virtuelle Maschine
  - kein unmittelbarer Einfluss auf andere Maschinen
- Achtung: Derzeit noch keine Protection!

Performance

Kernel-Bauen um die Wette

- reale Maschine 86 s

- FAUmachine 635 s
- FAUmachine (JIT) 943 s (*)
- FAUmachine (beschleunigt) 274 s
- FAUmachine (beschleunigt/JIT) 387 s (*)

- VMware 250 s
- Bochs 239 min

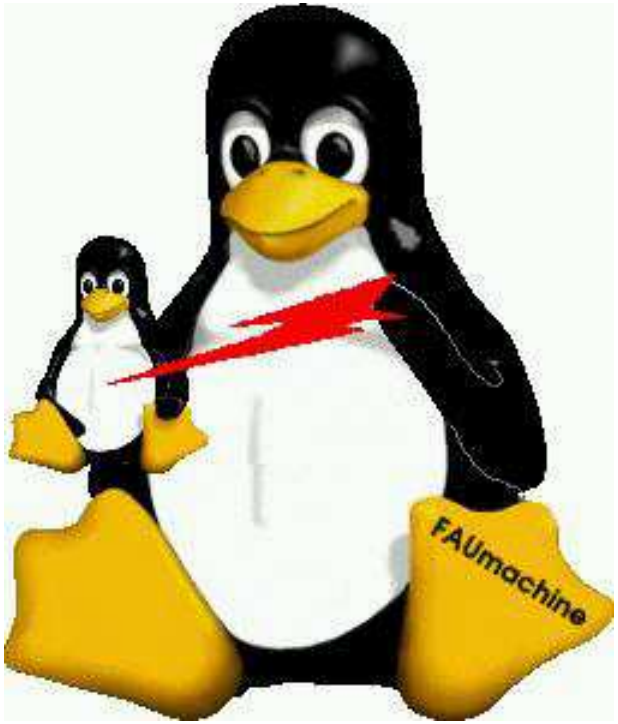
(*) extrapoliert basierend auf einer anderen Messung

Ausblick

Todo

- vollständigere Implementation der CPU-Simulation
  - nötig für Windows, FreeBSD, etc.
- BIOS verbessern
- 4G Adressraum für Simulator
- Protection
- weitere Gastgeber
  - Windows (in Arbeit)
  - NetBSD
  - FreeBSD
  - Solaris x86
- Just-In-Time-Crosscompiler (Qemu)?
- ...
  - PowerPC

Fragen?



http://www.faumachine.org/