Xen

Vortrag im Proseminar Konzepte von Betriebssystemkomponenten

Johannes Schlumberger
spjsschl@cip.informatik.uni-erlangen.de

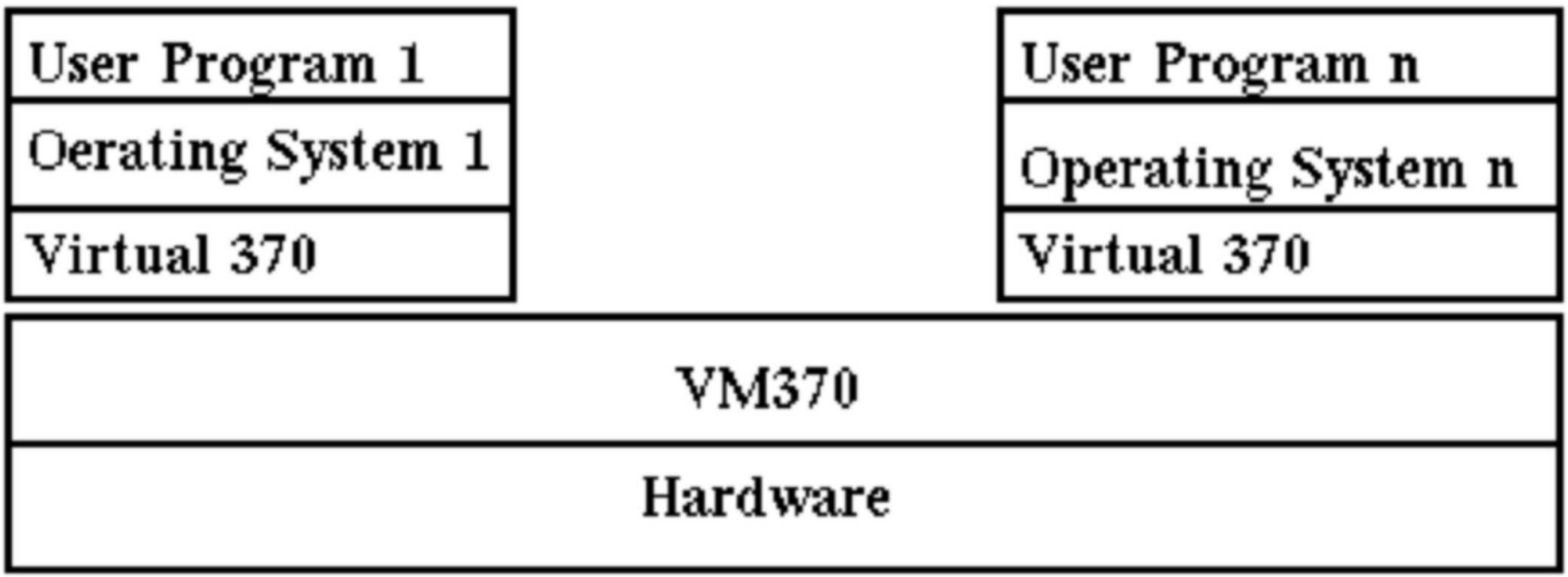
Friedrich-Alexander-Universität Erlangen/Nürnberg

19. Mai 2006

Was ist Virtualisierung?

- schwer aussagekräftig aber prägnant zu formulieren
- virtuelle Maschine
- eine einzige echte Maschine
- verschiedene simultan laufende Betriebssysteme
- jedes Betriebssystem glaubt es laufe alleine auf der echten Maschine

IBM VM/370



VIRTUAL MACHINE ARCHITECTURE (VM370)

Virtualisierung

- Sicherheit
- Flexibilität
- Herkömmliche Virtualisierungsverfahren sehr langsam (VMWare, etc.)

Überblick

Was ist Xen?

- virtuelle Maschine für x86
- entworfen für gängige Betriebssysteme
  - Linux
  - BSD
  - Win XP
- sicher
- Ressourcenisolation

Was ist Xen nicht?

- funktional schwächer
- wesentlich langsamer als nichtvirtualisierte Maschinen

Virtualisierungsarten

- Vollvirtualisierung
  - virtuelle Hardware entspricht genau der echten Hardware
  - keine Änderungen an den Gastbetriebssystemen nötig
- Teilvirtualisierung
  - virtuelle Hardware entspricht nicht genau der echten Hardware
  - Änderungen an den Gastbetriebssystemen nötig
  - geprägt druch Xen

Vorteile der Teilvirtualisierung

- umgeht viele Schwierigkeiten die bei Virtualisierung auf x86-Architektur auftreten (z.B. MMU)
- sowohl echte als auch virtuelle Ressourcen nutzbar (z.B. Zeit)
- Zugriff auf echte Speicheradressen möglich

Ziele

Ziele beim Design von Xen

- ABI darf nicht verändert werden
- keine Einschränkungen im Mehrbenutzerbetrieb
- hohe Performanz
- starke Ressourcenisolation
- Paravirtualisierung soll Performanzverluste auffangen

Umsetzung

Speicher

x86-Speicher

- kein softwaregesteuerter TLB
- TLB-Fehlschläge werden automatisch vom Prozessor über Hardware-Seitentabellen verarbeitet (sehr schwer, und v.a. teuer virtualisierbar)

$\Rightarrow$ Alle gerade gültigen Adressen müssen aus Gründen der Performanz in den Hardware-Seitentabellen bereit gehalten werden

Designentscheidungen

- Gastbetriebssysteme verwalten und allokieren ihre Seitentabellen soweit irgend möglich selber; Xen kümmert sich ausschließlich um Sicherheit und Isolation
- Um einen kompletten TLB-Flush beim Kontextwechsel zu vermeiden, residiert Xen fest in den oberen 64 MB Hauptspeicher, der von allen gängigen x86-Betriebssystem sowieso nicht genutzt wird

Seitenanforderung

1. Gastbetriebssystem braucht eine neue Seite
2. allokieren und initialisieren aus eigenem Speicher
3. neue Seite bei Xen bekanntmachen
4. gleichzeitige Abgabe der Schreibrechte für Seitentabellen an Xen
5. Xen validiert im folgenden alle Updateanfragen

Kontrolle durch Xen

- Schreibverbot in Seitentabellen
- Verbot der Einsicht in fremde Seitentabellenteile
- Verbot des Zugriffs auf 64 MB am oberen Speicherende

Batchupdates: Gastbetriebssysteme können Tabellenupdates gesammelt bei Xen abgeben, um die Kosten für den Kontextwechsel zu Xen zu minimieren.

Segmentierung

Xen kontrolliert alle Änderungsanfragen für die Segmentdeskriptortabelle.

- Verbot von Deskriptoren die Xens Privilegienlevel erreichen
- Verbot von Segmenten die Xens Speicher beinhalten

Prozessor

x86 Prozessor

- 4 Ringe ($Ring0 - 3$) in Hardware
- $Ring0$ Betriebssystem (erlaubt privilegierte Anweisungen)
- $Ring3$ Anwendungen
- $Ring1$ und $Ring2$ üblicherweise ungenutzt (Ausnahme: OS/2)

$\Rightarrow$ Das Gastbetriebssystem muss so modifiziert werden, dass es in $Ring1$ läuft, damit $Ring0$ für Xen frei wird.

Privilegierte Anweisungen

- Gastbetriebssystem ist sicher isoliert von Anwendungen
- Xen ist sicher isoliert von Gastbetriebssystem
- Gastbetriebssystem kann keine privilegierten Anweisungen mehr ausführen
- Jeder solche Versuch wird vom Prozessor stillschweigend ignoriert oder mit fault quittiert

$\Rightarrow$ Privilegierte Anweisungen müssen paravirtualisiert von Xen validiert und ausgeführt werden.

Ausnahmen

- Xen wird eine Tabelle mit einem Eintrag für jeden Exceptionhandler bekannt gemacht
- Handler sind identisch zu denen für normale x86-Architektur, da Xen die Exception-stack-frames nicht verändert (außer Seitenfehler)
- Ausnahme tritt auf:
  1. Xen kopiert Exceptionstack in Gastbetriebssystem
  2. Kontrollübergabe an den entsprechenden Handler

Seitenfehler

- x86: Fehlerhafte Adresse steht in privilegiertem Register `CR2`
- von Gastbetriebssystem nicht lesbar
- Inhalt von `CR2` wird von Xen in einen erweiterten Exception-stack-frame kopiert
- Anpassung des Seitenfehler-handlers nötig

fast Handler

- Verbesserte Systemaufrufperformanz durch direkte Ausführung eines Handlers ohne Indirektion über Xen.
- Handler werden von Xen validiert, ehe sie in die Handler-tabelle eingetragen werden. (Handler darf nicht verlangen in *Ring*0 zu laufen)

E/A

E/A

- Keine Virtualisierung von existierender Hardware (Festplatten, o.ä.)
- neue Abstraktionsschicht für Geräte

$\Rightarrow$ Gerätetreiber müssen teilweise angepasst werden

E/A

- E/A-Daten werden vertikal über shared-memory-Segmente ausgetauscht
  - hochperfomant
  - Xen kann Daten validieren
- Hardware-Unterbrechungen werden durch ein leichtgewichtiges, asynchrones Ereignissystem nachgebildet
  - Bitmap mit ausstehenden Ereignis
  - optional ruft Xen bei Ereigniseintritt festgelegten Handler auf
  - Handler können vom Gastbetriebssystem gesperrt werden um Extrakosten (durch wiederholtes Aufwecken) zu sparen

Zeit

Zeit

Xen stellt über sein Ereignissystem drei Arten von Zeit zur Verfügung:

- reale Zeit (in Nanosekunden seit Boot)
- virtuelle Zeit (wird nur gezählt wenn die aktuelle Domain läuft)
- Uhrzeit (Offset der auf die reale Zeit gezählt wird)

⇒Es ist damit möglich jeweils die korrekte Zeit für die Synchronisation zu Xen, für die Scheduler im Gastbetriebssystem und für lokalisierte Desktopuhren zu erhalten.

Zusammenfassung

Speicher

Paging	Direktzugriff auf Hardware-Seitentabellen Kontrolle durch Xen bei Schreibzugriff Batchupdates
Segmentierung	keine vollprivilegierten Segmentdeskriptoren keine Segmente die in den Xen-Adressraum reichen

Prozessor

Sicherheit	Xen läuft auf höherer Stufe
Ausnahmen	Deskriptortabelle für Exceptionhandler Seitenfehler-Handler wird verändert
Systemaufrufe	Laufen über Xen ab fast Signalhandler für Aufrufe aus Anwendungen
Unterbrechungen	ersetzt durch ein leichtgewichtiges Ereignissystem

E/A und Zeit

Netzwerk	einfach ansprechbare virtuelle Geräte
Festplatten	asynchroner Datentransport
	Ereignismechanismus statt Hardwareunterbrechungen
Zeit	reale und virtuelle Zeit über Xen Schnittstelle

Portabilität

- Linux: 0.04% des Quellcodes (etwa 1400 Zeilen)
- Windows XP: 1.36% des Quellcodes (etwa 5000 Zeilen)

Relative Performanz

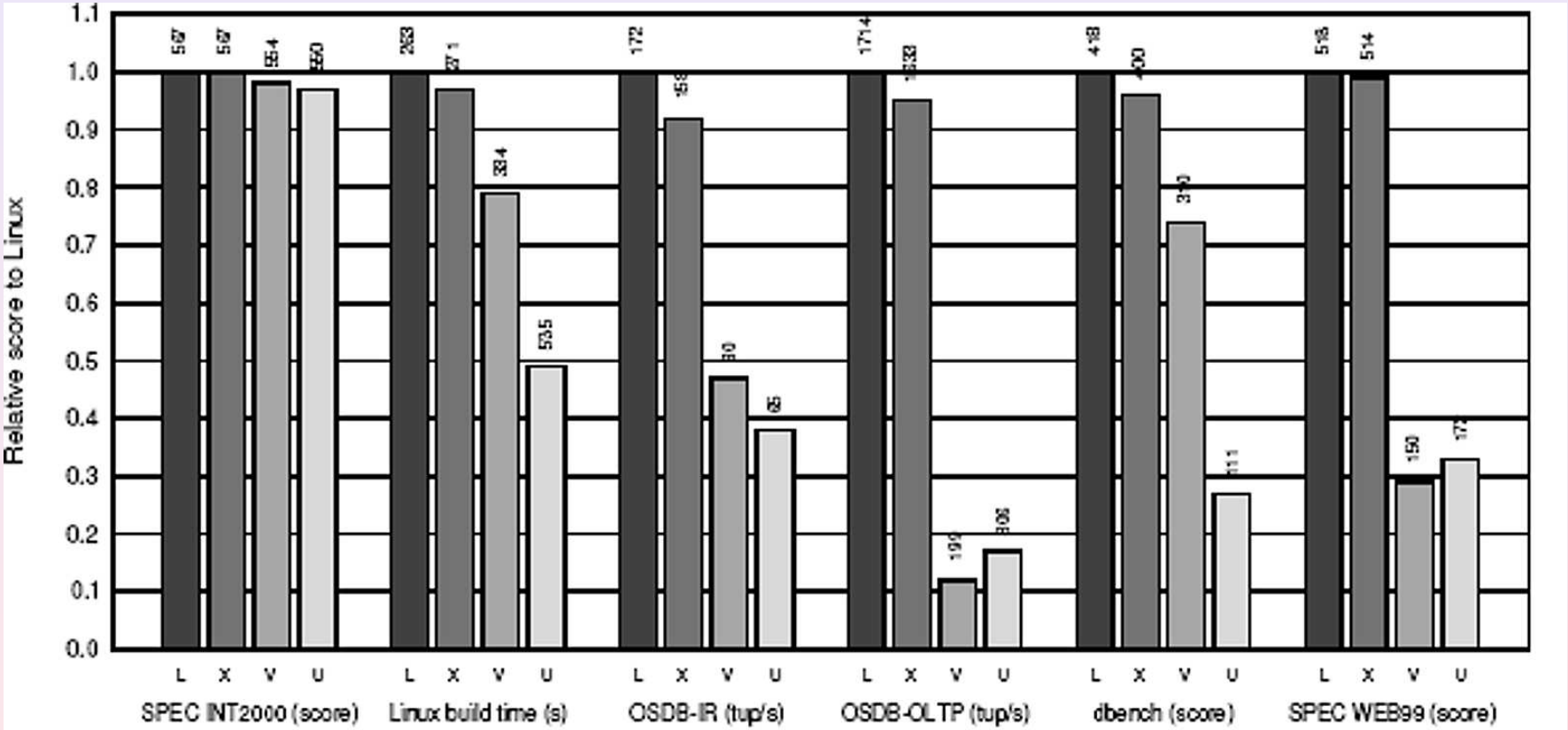


Figure 3: Relative performance of native Linux (L), XenoLinux (X), VMware workstation 3.2 (V) and User-Mode Linux (U).

Skalierbarkeit

- Eine Domäne mit Standard Redhat Installation

  6.2MB RAM ohne Swap

  4.2MB RAM mit Swap
- 100 Domänen

  600 MB RAM

  kein Problem für moderne Serverhardware

Der Arbeitsspeicher wird je nach Bedarf von Xen an die Domänen zugeteilt

Latenz

mittlere Antwortzeit auf ein UDP-Paket

- 128 Domänen unter Vollast

  an eine Domäne 147ms
- 128 Domänen unter Vollast, eine Domäne ohne Last

  an unbelastetet Domäne: 5.4ms

Wenn man die Zeischeiben die der Scheduler zuteilt verkleinert, kann man diese Zeiten noch verbessern.

Zusammenfassung

Xen ist eine Virtualisierungssoftware, die sich durch

- eine Reihe gut getroffener Designentscheidungen
- hohe Performanz
- sehr gute Ressourcenisolation und
- gute Skalierbarkeit

auszeichnet.

Danke...

für eure Aufmerksamkeit!

Alles klar?

Gibts Fragen?