Live-Migration in virtuellen Umgebungen

Tim Felgentreff und Tobias Pape

***Zusammenfassung*— Beim Betrieb von Infrastrukturdiensten im Umfeld von Cloud-Computing – Infrastructure as a Service (IAAS) – gibt es einige Herausforderungen wie Ausfallzeitminimierung, Hardware-Änderungen im laufenden Betrieb und Skaliereung nach Last. Mit Hilfe von Live-Migration von virtuellen Maschinen ist es möglich, einige davon anzugehen. Es gibt diverse Virtualisierungs-Platformen unterschiedlicher Hersteller, die Live-Migration auch zu einem hohen Grad unterstützen. Es mangelt noch an Interoperabilität zwischen den Systemen und Standardisierung ist in diesem Bereich noch nicht weit fortgeschritten.**

***Schlüsselworte*—Cloud-Computing, Virtualisierung, Live-Migration, IaaS, Interoperabilität**

I. EINFÜHRUNG

HEUTZUTAGE betreiben große Firmen eine Vielzahl von Diensten auf einer großen Anzahl von Servern, möglicherweise in einem oder mehreren Rechenzentren. Oft laufen Dienste physisch getrennt auf unterschiedlichen Servern, um die Ausfallsicherheit zu verbessern, außerdem wird Hardware vorgehalten, um fehlerhafte Komponenten zu ersetzen, Systeme zur unterbrechungsfreien Stromversorgung werden eingesetzt, um auch im Falle eines Stromausfalls nicht offline zu gehen.

Virtuelle Maschinen (VMs) spielen hier eine große Rolle. Die Technologie zur Virtualisierung von Hardware existiert seit über 40 Jahren [**?**], die Fähigkeit, mehrere Maschinen auf einem physikalischen System auszuführen, und Sicherungen ganzer Betriebszustände zu erstellen ermöglichen es Dienstanbietern, Ausfälle zu isolieren und schnell zu beheben und deshalb können Versprechen von 99.9% Verfügbarkeit an alle Kunden gegeben werden [**?**].

Mit dem Aufkommen von Cloud-Anbietern hat sich ein Dienstleistungssektor etabliert, der sich voll und ganz mit dem Ziel „100% Uptime“ beschäftigt. Diese Clouds, privat oder öffentlich, sind für Unternehmen zunehmend interessant, da sie die physische Hardware von der Ausführungsumgebung von Diensten zunehmend entkoppelt. Für Unternehmen ist der Schritt schon heute aufgrund eines potentiell geringeren administrativen Aufwands interessant, Rechenleistung kann bereitgestellt werden. Doch egal wie abstrahiert, auch VMs laufen letzten Endes auf Hardware, und diese kann ausfallen. Um dem „100% Uptime“-Ziel nahe zu kommen, müssen Dienste nicht nur virtualisiert werden, sondern auch frei zwischen Servern und Rechenzentren bewegt werden können. Hierfür sind in den letzten Jahren mehrere Lösungen auf den Markt gekommen, nachdem im Jahr 2002 erstmals erfolgreich eine virtualisiert laufendes System ohne herunterfahren auf einen anderen Server migriert wurde.

Die Autoren studieren am Hasso-Plattner-Institut, Potsdam.
Email: {vorname.nachname}@student.hpi.uni-potsdam.de

A. Der Status in Rechenzentren

Das Aufstocken von Hardware oder reagieren auf Hardware-Probleme ist Tagesgeschäft in Rechenzentren. Nichtsdestoweniger sind *Ausfallzeiten* bei solchen Arbeiten unvermeidbar, sofern man nicht mit Grid- oder Clustertechnologie arbeitet.

Ein Server ist dann für den Betreiber am effizientesten, wenn er möglichst ausgelastet ist [**?**]. Nun ist bei vielen Anwendungen, die auf solchen Servern laufen mit Lastspitzen aus den verschiedensten Gründen zu rechnen, wie das Verarbeiten von Batch-Jobs am Monatsende, das bei vielen Enterprise resource planning (ERP)-Systemen üblich ist. Auch das plötzliche Bekanntwerden vorher eher unbekannter Anwendungen (insbesondere Webseiten), bekannt als *Slashdot effect* [**?**], kann solche Lastspitzen auslösen. Während erstere planbar sind, können letztere unerwartet auftreten. Aus diesem Grund muss dafür Sorge getragen werden, dass für Lastspitzen ausreichend Pufferkapazitäten vorhanden sind. Eine *gleichmäßig hohe Auslastung* ist damit traditionell kaum möglich, selbst wenn Anwendungen in einem Cluster betrieben werden.

Das Betreiben von Servern zieht neben den Beschaffungskosten noch *laufende Kosten* nach sich, abgesehen von Software sind das insbesondere Strom und gegebenenfalls Kosten aus Wartungsverträgen. Auch das Personal, das die Server betreibt, ist zu bezahlen.

Anbieter von Cloud-Computing-Platformen argumentieren, dass die genannten Probleme durch ihre Platformen zumindest in großen Teilen gelöst wird [**?**], [**?**], [**?**]. Insbesondere wenn es um Infrastruktur, also Betriebssysteme etc., als Dienst geht, ist die Virtualisierung dieser Infrastruktur die Grundlage, um einen solchen Cloud-Dienst bereitzustellen. [**?**].

B. Cloud-Computing und Virtualisierung

Wenn es um Cloud-Computing und Virtualisierung geht, stellt sich die Frage, was an sich virtualisiert werden soll und überhaupt mit Cloud-Computing unterstützbar ist. Neben der Frage nach dem Was, ist auch interessant, wo zwischen Systemplatine und Endbenutzeranwendung Virtualisierung stattfinden kann. Dies soll im Folgenden beleuchtet werden.

1) Level der Virtualisierung: Virtualisierung kann auf verschiedenen Ebenen stattfinden. Je näher an der Hardware die Virtualisierung stattfindet, desto mehr kann die Hardware auch unterstützend wirken. Je ferner der Hardware, desto weniger ist die Virtualisierung von der unterliegenden Hardware abhängig. [**?**] In diesem Zusammenhang beschreiben *Konsolidierung* (insb. Server-Konsolidierung), *Separation* und *Virtualisierung* verschiedene Perspektiven auf den selben Sachverhalt: *Das Nebeneinanderbetreiben von Ausführungsumgebungen auf ein und derselben physischen Recheneinheit ohne (oder mit minimaler) gegenseitige(r) Kenntnis.*

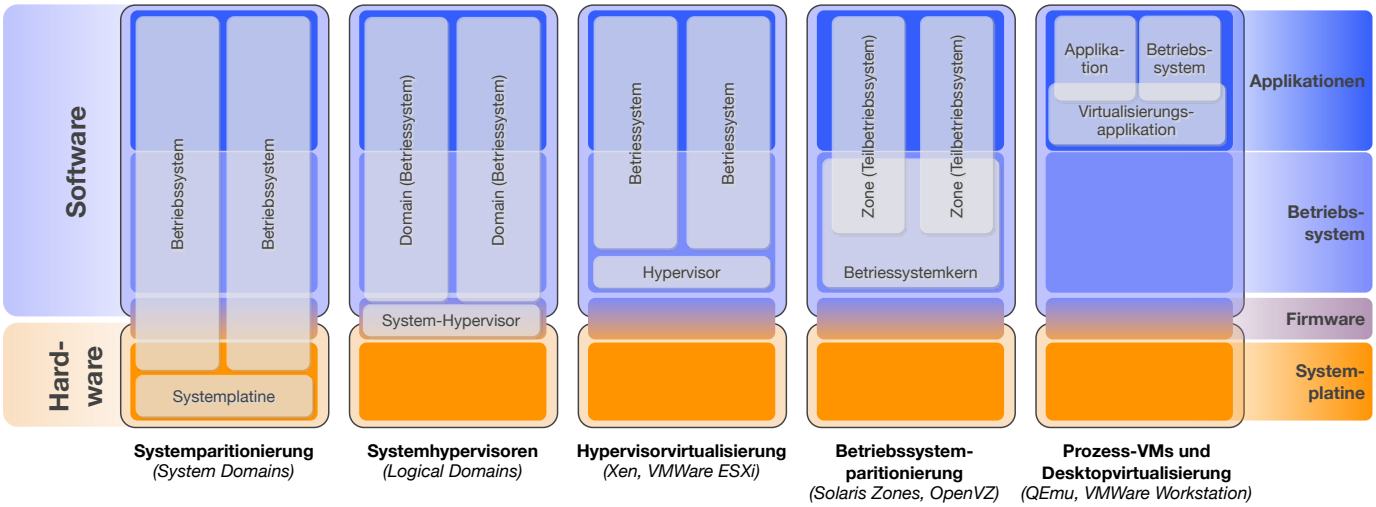

Abbildung 1. Virtualisierung kann auf verschiedenen Ebenen angesetzt werden. Im Kontext von Live-Migration sind die vertikal eingezeichneten Elemente, das heißt, die zu virtualisierenden Betriebssysteme oder Applikationen von Interesse.

In der untersten Schicht kann die so genannte *Systempartitionierung* betrachtet werden (Siehe zu den Ebenen Abbildung 1). Die „Sun Fire Dynamic System Domains“ [**?**], die auf Sun Fire-Servern zur Verfügung steht, ist ein Beispiel für diesen Virtualisierungsansatz. Die Hardware dieser Server ist in der Lage, verschiedene Ausführungsumgebungen, das heißt, Betriebssysteme, elektronisch isoliert zu betreiben. Es bedarf keinerlei Unterstützung in den darüberliegenen Softwareschichten. Diese Art Virtualisierung kommt der Ausführung auf physikalisch getrennten Rechnern am nächsten.

Eine Ebene höher kann Virtualisierung mit Hilfe von *Hypervisoren* betrieben werden. Virtuelle Maschinen (VMs) laufen auf physikalischen Maschinen durch einen *Hypervisor*, ein minimales Betriebssystem, dass den Zugriff auf die physikalische Hardware regelt und Routing zwischen dem physikalischen und virtuellen Netzwerk bereitstellt [**?**]. Wenn eine VM ausfällt, bleiben die anderen davon unbeeinflusst. Da des weiteren alle Operationen innerhalb einer VM virtualisiert sind, kann der Hypervisor den Zustand eines virtuellen Servers in regelmäßigen Abständen sichern, und so bei Ausfall die VM von einer früheren Sicherung wiederherstellen. Beispiele für Hypervisorvirtualisierung sind VMWare ESXi und damit auch vSphere [**?**] und Xen [**?**].

Ein Speziallfall der Virtualisierung mit Hypervisoren sind die *Logical Domains (*LDOMs*)*, die auf Rechnern mit Prozessoren der SPARC T-Serie zur Verfügung stehen und unter dem Namen „Oracle VM Server for SPARC“ [**?**] von Oracle vermarktet werden. Dabei können die Ressourcen des Rechners in logische Bereiche (Logical Domains) eingeteilt werden, die gleichzeitig jeweils ein Betriebssystem zur Verfügung stellen können. Damit ist es möglich, ohne direkte Software-Unterstützung eine virtualisierte Ausführung von Betriebssystemen zu haben: die Firmware der SPARC-Rechner stellt Überwachungs- und Steuerungsfunktionalität bereit, die Hypervisoren nicht unähnlich ist.

Auf Betriebssystemebene ist ebenfalls eine Art Virtualisierung möglich. Dabei ist das Ziel, mehrere Exemplare der selben Betriebssysteminstallation auszuführen. Ein Beispiel für diese *Betriebssystempartitionierung* (auch *Servervirtualisierung auf Betriebssystemebene*) sind die „Solaris Zones“ [**?**]. Sie erlauben, eine Solaris-Installation mehrfach ausführen zu lassen, und zu einem gewissen, einstellbaren Grad die installierten Betriebssystemkomponenten nachzunutzen respektive zu ersetzen. Dabei wird der Betriebssystemkern einmal ausgeführt, trägt aber Sorge dafür, den einzelnen Zonen als „ihr“ Kern zu erscheinen. Eine vergleichbare Technologie für Linux-Systeme ist das OpenVZ-Projekt [**?**].

Auf Applikationsebene gibt es zwei unterschiedliche Virtualisierungsansätze. Während *Desktopvirtualisierung*, vergleichbar zur Hypervisorvirtualisierung, Betriebssysteme virtualisiert und somit innerhalb eines anderen Betriebssystem bereit stellt, haben Prozess-VMs das Ziel, eine Ausführungsumgebung für bestimmte Programmiersprachen oder Applikationen bereit zu stellen. Beispiele für ersteres sind VMWare Workstation [**?**] oder Oracle VirtualBox [**?**] (vormals Sun xVM), für letzteres VMs wie die JVM [**?**] oder Smalltalk-Implementierungen [**?**].

2) Cloud-Computingansätze: Bei Betrachtung der verschiedenen Ebenen, auf denen Virtualisierung von Ausführungsumgebungen möglich ist, stellt sich die Frage, welche davon sich als „Cloud-Ebene“ eignen. [**?**]

SaaS Auf Applikationsebene (vgl. Abbildung 1) können Anwendungen in „Cloud-Manier“ bereit gestellt werden. Der Anbieter der *Software as a Service* (SAAS)-Lösung hält dafür (zumeist mehrere) Installationen einer Anwendung bereit, die von Kunden genutzt werden können, ohne irgendeinen Installationsaufwand für die Anwendung beim Kunden zu haben. Ein Beispiel für solche Clouds ist Google Apps [**?**], womit z.B. Büroapplikationen (neben anderen) als Dienst angeboten werden.
Im Zusammenhang mit Software as a Service (SAAS) ist Live-Migration zumeist uninteressant; wo eine Anwendung läuft spielt in diesem Zusammenhang eine untergeordnete Rolle. Auf Anbieterseite kann Migration und Live-Migration ein Thema sein, ist aber von Anwendung zu Anwendung gesondert zu Betrachten. Für einzelne Anwendungen kommt gegebenenfalls

Prozessmigration in Frage, bei komplexeren, viele Prozesse umspannenden Anwendungen wird zumeist die Migration der unterliegenden Ausführungsumgebung, insbesondere Betriebssystem, vorgezogen (vgl. IaaS).

PaaS Auch auf Applikationsebene, ist bei *Platform as a Service* (PAAS) nicht eine Anwendung, sondern eine Platform Gegenstand. Nutzer dieser Clouds bekommen so eine Art Middleware oder Ausführungsumgebungen bestimmter Programmiersprachen bereit gestellt, zum Teil auch Entwicklungsumgebungen. Beispielsweise stellt Google mit App Engine [**?**] eine JVM-basierte Ausführungsumgebung bereit, auf die die Nutzer ihre eigenen Webanwendungen ausliefern können.
An sich ist Platform as a Service (PAAS) eine spezialisierte Form von SAAS, nur der bereitgestellte Dienst ist insbesondere eine Ausführungsumgebung. Auch hier ist Live-Migration weniger relevant; Nutzer, die ihre Anwendungen bei einem PAAS-Anbieter laufen lassen, unterscheiden sich kaum von Nutzern, die eine SAAS-Anwendung nutzen: wo die Anwendung läuft ist kaum interessant. Für Anbieter bietet sich wieder Prozessmigration an oder, je nach angebotener Platform, traditionelle ausfalltolerante, lastverteilungsbasierte Konfigurationen der Platform.

IaaS Die Idee hinter *Infrastructure as a Service* (IAAS) ist, im Gegensatz zu den vorigen Ansätzen, komplette Ausführungsumgebungen im Sinne von Betriebssystemen auf Nachfrage zur Verfügung zu stellen. Damit ist IAAS die Ebene, auf der Virtualisierung im hier vorgestellten Sinn genutzt wird. Anbieter und Kunden von IAAS-Diensten können ein Interesse daran haben, den Ort, an dem der Dienst / das Betriebssystem, ausgeführt wird, zu wechseln. Damit kann einerseits die Hardware gemeint sein, auf der das Betriebssystem läuft, andererseits auch der geografische Ort an dem die Hardware steht. Anforderungen wie
  - schnelles Bereitstellen von neuen Betriebssysteminstanzen,
  - Vorsorge für Lastspitzen und
  - schnelle, dynamische Hardwareanpassung

  machen IAAS zur eigentlichen Schnittstelle zwischen Virtualisierung und Cloud-Computing.
  Interessanterweise ist für das Bereitstellen von Infrastrukturdiensten mit Hilfe von virtualisierten Betriebssystemen vom Ansatz her nicht relevant, welches Virtualisierungslevel (vgl. Abbildung 1) dafür genutzt wird. Praktisch hat es jedoch schon Auswirkungen, ob ein IAAS-Dienst auf Grundlage von VirtualBox, VMWare ESXi oder Oracle Dynamic System Domains betrieben wird.

Live-Migration im Kontext von Cloud-Computing ist somit hauptsächlich bei IAAS anzutreffen. Entsprechend wird im Folgenden vorrangig IAAS besprochen.

C. Herausforderungen des Cloud-Computing

Oft wird in letzter Zeit, wenn der Begriff Cloud-Computing benutzt wird, zwischen *public cloud* und *private cloud* unterschieden. Zur *public cloud* gehören alle Anbieter von Cloud-Diensten, von SAAS bis IAAS, die auf dem Markt auftreten; im Gegensatz dazu sind *private clouds* an ein Unternehmen – oder Betreiber – gebunden und im Normalfall der Öffentlichkeit nicht zugänglich [**?**]. Zumeist geht es bei dieser Weise, Cloud-Computing zu betreiben, um IAAS. Die Idee der *private cloud* hat sich in den letzten Jahren aus dem bis dahin im Allgemeinen öffentlichen, das heißt, durch nich-unternehmenseigene Firmen betrieben, Cloud-Computing gebildet: Unternehmen, die die Vorteile davon nutzen wollten, hatten gewisse Herausforderungen zu bewältigen.

Anbieterbindung: Wählt ein Unternehmen einen bestimmten Cloud-Computing-Anbieter, heißt das bis dato, sich langfristig an diesen zu binden. Proprietäre oder zumindest inkompatible Cloud-Computing-Platformen machen ein zukünftiges Wechseln schwierig, von Live-Migration von Anbieter zu Anbieter ganz zu Schweigen.

Anforderungen an den Standort: Aufgrund gesetzlicher Bestimmungen ist es bisweilen notwendig, sicherzustellen, das die Verarbeitung bestimmter Daten an einem gestimmten geografischen Ort stattfindet, z.B. einem bestimmten Staatsgebiet. Bei *public cloud*-Betreibern ist es zumeist nicht möglich das zu gewährleisten. Sollte das doch der Fall sein, kann ein Unternehmen trotzdem Interesse daran haben, das zumindest bestimmte Teile der Datenverarbeitung „innerhalb“ des Unternehmens stattfinden.

Das einrichten von in Cloud-Manier betriebenen Rechenzentren in Unternehmen, führt nun dazu, dass diese beiden Probleme angegangen werden können, das Unternehmen betreibt nun eine *private cloud*. Es bleiben jedoch noch Herausforderungen, die der privaten und der öffentlichen Variante gemein sind oder gar durch das private Betreiben hinzukommen. [**?**]

Skalierungsplanung: Bei den meist als Entwicklungsprojekten betriebenen Cloud-Computing-Anwendungen ist für Unternehmen oft im Vorhinein nicht klar, wie stark eine Skalierung später notwendig ist. Zumeist IAAS-Projekte, birgt eine zu sparsame Konfiguration das Risiko, nicht schnell genug auf Lastspitzen reagieren zu können, wenn das Projekt plötzlich sehr stark genutzt wird [**?**]. Andersherum kann eine zu großzügige Konfiguration zu kostenintensiv sein, ob nun *private* oder *public*. Anpassbare IAAS-Konfigurationen sind daher wünschenswert. Im Rahmen von Betriebssystemvirtualisierung mit Virtuelle Maschinen (VMS) ist das auch durchführbar, stößt aber auch an seine Grenzen: benötigt eine VM plötzlich weit mehr Speicher als vorher, weil z.B. die Datenbank des betriebenen Projektes stark vergrößert wurde, ist die Grenze der Speicher, der physisch verfügbar ist. Wachsen die Anforderungen darüber hinaus, kann das Verschieben der VM auf Hardware mit mehr Speicher Abhilfe schaffen. Live-Migration kann hier helfen, Ausfallzeiten gering zu halten.

Auslastungsplanung: Eine Herausforderung, auf die Cloud-Computing konzeptuell überhaupt reagieren sollte, ist, die vorhandene Hardware bestmöglich, das heißt, gleichmäßig hoch auszulasten. Nutz man öffentliche Cloud-Anbieter, ist das für ein Unternehmen als Kunden kein Problem, bei privat betriebenen Clouds muss das jedoch bedacht werden. Das Konsolidieren von sonst einzelnen Betriebssystem auf

einen Server mittels Virtualisierung ist eine Möglichkeit, eine verbesserte Auslastung von Hardware zu ermöglichen. Auslastungsplanung und Konsolidierung ist jedoch auch im Kontext von Skalierungsplanung zu betrachten. Das heißt, dass hier normalerweise eine Abwägung zwischen guter Auslastung und Lastspitzensicherheit geschehen muss. Live-Migration von VMs kann helfen, beides in gewisser Weise zu ermöglichen: Wird eine VM unter Normalbedingungen zusammen mit anderen auf einer Hardware betrieben, ist es doch möglich, bei Lastspitzen die VM auf eine andere, gegebenenfalls eigene, Hardware zu verschieben.

Wartung: In jedem Rechenzentrum gehört der Ausfall von Hardware, aber auch das Aufstocken oder Austauschen dieser zum Tagesgeschäft. Dies ist im Normalfall mit Ausfallzeiten verbunden. Das Verschieben von VMs auf andere Hardware und nachfolgende Wartungsarbeiten an der Vorherigen hilft, Ausfallzeiten zu minimieren.

II. Live-Migration als Teil der Lösung

Hansen präsentierte das NomadBIOS [**?**] als Live-Migrations-Lösung als Teil der Lösung für oben genannte Probleme der Clouds. Im Folgenden wird auf mögliche Anwendungsfälle dafür eingegangen.

A. Verlässlichkeit von Virtualisierung

In jedem verteilten System muss man mit der Realität umgehen, dass Hardware und Software Fehler enthält, die zu Ausfällen führen können. Da scheint die Idee, viele VMs auf wenigen physikalischen Systemen auszuführen zunächst wie ein Schritt in die falsche Richtung: Wenn dann ein physikalischer Server ausfällt, oder die Virtualisierungssoftware einen Fehler enthält, fallen potentiell eine ganze Reihe von Diensten auf einmal aus. In der Realität entstehen die meisten Fehler in Datencentern allerdings durch Softwarefehler [**?**]. Softwarefehler können umso häufiger zutage treten, je mehr unterschiedliche Systeme miteinander interagieren müssen, und je größer die Code-Basis der einzelnen Systeme ist [**?**]. Durch Virtualisierung kann man Softwaresysteme voneinander isolieren, ohne für jedes System einen eigenen, physikalischen Server mit eigenem Betriebssystem bereitzustellen. Der Hypervisor selbst ist in solch einem Fall die einzige Software, die im Kernel Modus läuft, und dieser hat um Größenordnungen weniger Zeilen Quelltext als ein aktuelles Betriebssystem [**?**]. Folglich enthält der Hypervisor wahrscheinlich weniger Bugs [**?**].

Kritische Dienste können so in kompletter Isolation ausgeführt um das Risiko eines Ausfalls durch inkompatible Software auf demselben Server zu minimieren.

Dessen ungeachtet müssen auch Vorkehrungen gegen Hardwarefehler getroffen werden. In klassischen Hostingumgebungen kann man bei immanenten Ausfällen von physikalischen Systemen relevante Prozesse auf neue VMs migrieren [**?**]. Für eine gegebene VM kann eine Ähnliche (z.B. aus einem früheren, gesicherten Zustand) auf einem neuen Server gestartet werden. Der Zustand der relevanten Prozesse wird soweit als möglich gemäß den Gegebenheiten des Betriebssystems kopiert und neue Dienstanfragen dann auf den neuen Server umgeleitet. Die alte VM muss allerdings zunächst online bleiben, um Abhängigkeiten von Dienstanfragen bereitzustellen, die sich noch in Ausführung befinden, wenn der Hardwarealarm ausgelöst wird [**?**]. Das ist bekannt als „residual dependencies problem“ (Problem mit verbleibenden Abhängigkeiten).

Diese Vorgehensweise zur Migration auf Prozessebene hat also den wesentlichen Nachteil, dass laufende Prozesse nicht verschoben werden können, sondern noch auf der bestehenden Hardware zum Ende laufen müssen. Die meisten Applikationen verlassen sich auf viele verschiedene Weisen auf die Fähigkeiten des Betriebsystems, die Hardware anzusteueren und in geeignete Zustände zu überführen, die den Ablauf der Applikation erst möglich machen. Das heißt, dass ein laufender Prozess meistens nicht einzeln verschoben werden kann [**?**]. Im Falle eines Webservers, bei dem jede Anfrage auch an einen neuen Prozess gehen kann, ist das nicht weiter problematisch. Schwieriger ist die Situation bei Diensten wie z.B. Online-Spieleplattformen, bei denen ein Serverprozess TCP-Verbindungen zu mehreren Clients teilweise über Stunden aufrecht erhalten muss. Wenn ein Hardwareausfall kurz bevor steht, bleibt hier meist nichts anderes übrig, als das Spiel zu unterbrechen: Das bedeutet mindestens unzufriedene Kunden und vielleicht sogar verringerten Umsatz.

Wenn man aber stattdessen den Prozess und die virtuelle Maschine als Einheit betrachtet [**?**], und geschlossen live migriert, kann auf Hardwarefehler angemessener reagiert werden: Zuerst kann die VM bis aufs letzte Detail vom aktuellen Stand kopiert werden, das heißt, dass Änderungen, die seit der letzten Checkpoint-Sicherung geschehen sind, nicht verloren gehen. Wichtiger allerdings ist, dass auch Prozesse, die sich gerade in Ausführung befinden, migriert werden können. Im Beispiel der Spieleplattform bedeutet dies, dass Spiele nicht unterbrochen werden müssen, sondern, für den Nutzer transparent, auf den neuen Server migriert werden.

B. Interoparibilität von Virtualisierungs-Lösungen

Für Unternehmen, die mit öffentlichem Cloud-Hosting liebäugeln, kann die Bindung an ein bestimmtes System schwerwiegende Konsequenzen nach sich ziehen. Um mit der Flexibilität vom eigenen Rechenzentrum mitzuhalten, darf die Virtualisierungstechnologie des Providers nicht die Verwendung des Systems innerhalb das Unternehmens einschränken. Insbesondere muss für den Kunden die Möglichkeit bestehen, auf eine andere Virtualisierungstechnologie umzusteigen, um z.B. vorhandene VMs auf die Technologie eines Cloud-Anbieters zu bewegen oder umgekehrt auch VMs aus der Cloud wieder in das eigene Rechenzentrum oder auf die Rechner der Entwickler umzuziehen. Die Interoparibilität der verwendeten Virtualisierungslösungen spielt hier eine wichtige Rolle.

Virtuelle Maschinen variieren stark zwischen verschiedenen Virtualisierungslösungen. Das fängt mit den Formaten der virtuellen Dateisysteme an und geht bis hinunter zu den spezifischen Arten der virtualisierten Hardwarekomponenten [**?**]. Das bedeutet im Einzelnen, dass der Wechsel zwischen Hypervisor Technologien von außen eine Konvertierung zwischen den (oft proprietären) Formaten bedeutet, aber auch im virtualisierten Host unter Umständen Treiber neu installiert werden müssen. Solch tiefgreifende Änderungen, die oftmals gezwungenerma-

ßen mit Neustarts der virtualisierten Betriebssysteme einhergehen, sind kaum ohne Ausfallzeiten zu unternehmen. Folglich hat ein Cloud-Hosting-Provider allein durch die Auswahl seiner Virtualisierungslösung die Möglichkeit, Kunden sehr stark zu binden.

Live-Migration verspricht hier Abhilfe, wenn nur die Migrationsprotokolle der Hypervisoren kompatibel sind. Dann nämlich kann, ohne Umweg über die serialisierten Formate der Virtualisierungslösungen, das virtualisierte System direkt zwischen zwei VMs kopiert werden, und sozusagen zur Laufzeit konvertiert werden.

C. Inter-Cloud Verschiebungen

Probleme durch inkompatible Virtualisierungen werden im Rahmen des Outsourcings in die *public cloud* noch verstärkt, da die Entscheidung für einen Cloudprovider unter Umständen bedeutet, dass die Dienste nur unter erheblichem Aufwand wieder in eine andere Cloud verschiebbar sind. Das bedeutet für den Cloud-Kunden eine starke Bindung an den Hosting Provider, eine Bindung die für Unternehmen so problematisch sein kann, dass auf das Hosting bei einem fremden Provider ganz verzichtet werden muss.

Zur Migration von Systemen auf andere Maschinen können, wie bereits beschrieben, Prozessemigrationssystem, wie sie in den 1980er Jahren entwickelt wurden, verwendet werden. Dazu zählen z.B. das Sprite oder das MOSIX System [**?**]. Diese benutzten Fähigkeiten des Hypervisors sowie der darauf laufenden Betriebssysteme in Konjunktion und sind damit nicht auf beliebigen Kombinationen der beiden ausführbar. Die Notwendigkeit von spezifischen Anpassungen and verschiedenen Teilen des Softwarestacks und proprietäre Protokolle schränken machen die Anwendung von Prozessmigration über die Grenzen von Unternehmen und Organisationen hinweg defacto unmöglich. Diese Art System skaliert folglich nicht auf Verschiebungen zwischen Clouds mit inkompatiblen Stacks.

Migration ganzer VMs bietet hier einen Ausweg durch kleinere Schnittstellen und Wegfallen der Kommunikation zwischen den migrierten Systemen nachdem die neue VM in Betrieb geht. Da kein kompliziertes Routing zwischen der neuen und der alten VM vonstatten gehen muss, beschränkt sich die Kommunikation (und damit der Anspruch an die Schnittstellen der Providers) auf die initiale Synchronisationsphase.

Selbst wenn der Migrationspfad zwischen Virtualisierungslösungen vorhanden ist, ist das Bewegen einer VM aus einer fremden Cloud trotzdem oft nicht trivial. Cloud-Provider bieten derzeit noch verschiedene, teilweise inkompatible und in ihrer Mächtigkeit schwankende Zugriffsmöglichkeiten auf die gehosteten Systeme. Kleine Provider wie Heroku [**?**] bieten üblicherweise nur Zugriff auf einen Teil der gehosteten Software und erlauben Maschinenkonfiguration nach dem Baukastenprinzip. Ein vollständiges VM-Abbild ist nicht verfügbar. Im Vergleich dazu bieten Engine Yard [**?**] und VMWare ausgefeilte Tools [**?**] um VMs mit Meta-Konfigurationen [**?**] vollständig selbst einzurichten und bei Problemen auch komplette Abbilder dieser Maschinen zu erhalten. Wieder muss allerdings gesagt werden, dass diese Schnittstellen inkompatibel sind, und es unseres Wissens nicht ohne Ausfallzeit möglich ist, Dienste von einer in die andere Cloud zu bewegen. Zwar bieten die meisten Provider eine VM-Importfunktion, dazu muss die VM am Ursprung aber natürlich erst heruntergefahren werden [**?**].

Es gibt Bemühungen diese Verschiedenen Aspekte des Cloud-Hostings zu standardisieren [**?**]. Darin inbegriffen sind die Beschreibungen von VMs, sowie Application Programming Interfaces (APIs), die die Hypervisoren bereitstellen. Dieser Prozess ist allerdings noch nicht abgeschlossen, und obwohl große Hosting-Anbieter wie Rackspace, VMWare und Engine Yard ihre Unterstützung bekennen [**?**], fehlt z.B. noch Amazon als der größte Hosting-Provider in Europa.

Die Fortentwicklung der VM-Migrationstechnologie, wie sie Hansen beschrieben hat, könnte eine einfache Lösung für dieses Problem bieten, indem sie die Fähigkeiten zur Live-Migration vollständig in das Betriebssystem verschiebt. Mittels standardisierter Hypervisor-APIs könnte dann vom virtuellen System aus der Migrationsprozess angestoßen werden, und der Umzug in eine neue Cloud auch ohne Mitwirken des Hosting-Providers möglich werden. Das bedeutet effektiv eine Reduzierung der Bindung an eine bestimmte Cloud und vermindert das Risiko von *public clouds* aus Sicht des Unternehmes.

D. Hardware-Aufstockung ohne Ausfallzeigen

Moderne IT-Unternehmen müssen auf Marktänderungen, die Skalierung erfordern, schnell und mit geringen Kosten reagieren können. Neue Märkte werden oft erst erschlossen, indem Dienste auf günstiger Hardware erpropt werden [**?**]. Wenn sich ein Marktsegment als lukrativ erweist, muss man in der Lage sein, schnell die Anzahl und Menge der Server zu skalieren. Diese Fähigkeit ist entscheidend für den Erfolg eines modernen Web-Unternehmens. Studien zeigen, dass gerade am Anfang eines Online-Angebots lange Ladezeiten die Annahmebereitschaft beeinträchtigen [**?**].

Oftmals können diese Vorrausetzungen nur durch großzügige Planung der kommendenen Hardwareanforderungen erfüllt werden, was mit einem größeren finanziellen Risiko einhergeht. Gerade kleine Unternehmen und *Start-Ups* können sich dies oft nicht leisten. Die Fähigkeit, Dienste in der Cloud anzubieten und *bei Bedarf* mehr gleichartige VMs zu starten, oder VMs auf stärker ausgebaute Hardware zu verschieben, ohne dafür Ausfallzeit in Kauf nehmen zu müssen, kann den Unterschied machen zwischen einem erfolgreichen Dienst und einer Seite, zu der Kunden nicht mehr zurückkehren, weil sie zu langsam ist. Der Cloud-Hosting Provider *Heroku* hat sich auf ebenso solche, kleine, Webdienste spezialisiert, und versteckt den Prozess des Skalierens für den Kunden vollständig hinter einem einfach Schieberegler im Web-Interface: Der benötigte Leistungsindex wird erhöht und Heroku kümmert sich um das Spawning und Verschieben von VMs innerhalb seiner Infrastruktur.

Verwandt hierzu sind auch Dienste wie Google App Engine, wenngleich sie eher PAAS zuzurechnen sind, bieten auch sie kleineren Unternehmen die Möglichkeit, nur den Umfang an Leistung zu zahlen, der auch tatsächlich genutzt wird, und bei Bedarf mehr Leistung „einzukaufen“.

E. Adaptive Auslastung

Neben der Notwendigkeit, bei Marktwachstum schnell skalieren zu können, gibt es auch die Anforderung an viele

Systeme, in Stoßzeiten, oder bei unvorhergesehenen Nachfrageschwankungen adaptiv skalieren zu können. Ein Beispiel ist der bei *Heroku* gehostete Dienst *FlightCaster* [**?**]: Wenn Flüge verspätet sind, passiert das üblicherweise in Clustern, was zu sehr hohen Zugriffszahlen in kurzer Zeit führt [**?**]. In solch einem Fall ist es für den Dienstanbieter wichtig, weiterhin einen stabilen und schnellen Service zu leisten, um die Kundenzufriedenheit nicht zu gefährden, oder gar Kunden zu verlieren. In klassischen Rechenzentren kann man, wie im Vortrag von [**?**] besprochen, die verfügbare Kapazität auf die erwarteten Maximalauslastungen optimieren. Das ist jedoch tendentiell teuer und gerade in Situationen, wo die maximale Abweichung von der durchschnittlichen Auslastung sehr groß ist, oder Lastspitzen sehr selten auftreten, ist die Investition oft nicht gerechtfertigt.

Live-Migration kann auch hier zum schnellen Verschieben von Maschienen zwischen leistungsstarker und -schwachere Hardware genutzt werden.

Ein Gegenargument, dass man für diesen Anwendungsfall im Besonderen hervorbringen kann, ist, dass VM-Migration mit hoher Netzwerklast einhergeht. In Situationen, in denen der Durchsatz ohnehin gefärdet ist, kann man argumentieren, dass die zusätzliche Netzwerklast durch eine laufende Migration den Dienst vorübergehend ganz zum Erliegen bringen könnte. Die erhöhte Last liegt, je nach Anwendung, für mehrere Minuten auf dem Netzwerk, da die Umzugszeiten von der Netzwerklatenz zwischen den Servern abhängen. Tests von VMWare haben allerdings gezeigt, das sogar unter Last bei der angemessenen Migrationslösung eine schnelle Migration gewährleistet werden kann (Abbildung 2). Zu lösen wäre dieses Problem in jedem Fall durch ein parallel betriebenes Netzwerk, dass speziell für diesen Anwendung reserviert bleibt. Es steht zu vermuten, dass die Kosten für einen solchen Parallelbetrieb zweier Netzwerke weit hinter den Kosten für das vorhalten von ausreichender, aber meist ungenutzter, Rechenkapazität zurückbleiben.

F. Keine Ausfallzeit durch erzwungene Verschiebungen

Neben den Anwendungsfällen zur Skalierung und Aufrechterhaltung von (Cloud-)Services, gibt es ein weiteres Problem, mit dem international agierende Anbieter umgehen müssen, und das sind die sich wandelnden gesetzlichen Vorschriften zur Datenverarbeitung, die gerade in Deutschland noch vielfach in Veränderung begriffen sind [**?**]. Ein Beispiel ist Vorschrift, dass personenbezogene Daten von deutschen Behörden nur auf deutschem Boden verarbeitet werden dürfen. Diese Art von Gesetz schließt Dienste, die in anderen Ländern große Datenverarbeitungsdienste bieten, oft von Aufträgen deutscher Behörden aus. Rechenleistung in anderen Ländern als Deutschland ist unter Umständen wesentlich billiger für ein Unternehmen mit einem hohen Leistungsbedarf. Dauerhaft ein Rechenzentrum in Deutschland zu betreiben, für die Möglichkeit, ab und an einen Auftrag von einer deutschen Behörde zu erhalten, wiegt unter Umständen nicht den Ertrag solcher Aufträge auf.

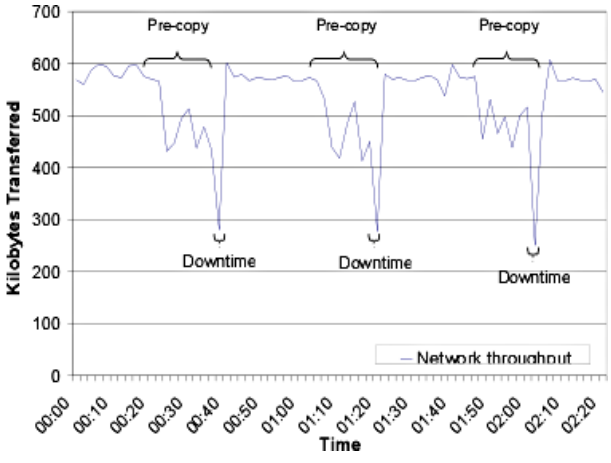

Abbildung 2. Effekt von Live-Migraton auf das Netzwerk. Quelle: [**?**]

Durch die Möglichkeit, ohne Verlust des aktuellen Dienstzustandes die Virtuelle Maschine umzuziehen, ist es jedoch möglich, solchen gesetzlichen Vorschriften nichtsdestoweniger zu genügen. Indem der Datenverarbeiter bei einem Cloud-Hosting-Anbieter auf deutschem Boden Rechenzeit einkauft, kann er den gestzlichen Vorschriften gerecht werden. Die VMs, die für den Auftrag bereitgestellt sind, können einfach in die Cloud auf deutschem Boden migriert werden. Migrationen über Ländergrenzen oder Kontinente hinweg, z.B. durch den Atlantik, sind zwar notwendigerweise langsamer als zwischen Rechenzentren mit höherer Lokalität, bei großen, lang laufenden Aufträgen ist das jedoch unter Umständen zu vernachlässigen.

Auf diese Weise ist es Datenverarbeitungsdiensten möglich, landesspezifischen Einschränkungen bezüglich der geografischen Lage der physikalischen Nodes gerecht zu werden, und Services „on-demand“ anzubieten, ohne die Flexibilität von lokal vorbereiteten Maschinen aufzugeben.

Die Amazon EC2 bietet diese Möglichkeit bereits zum Teil, indem man solche Einschränkungen beim Kauf der Rechenzeit angibt. Derzeit müssen dazu allerdings die VMs neu gestartet werden. Mit Hilfe von Live-Migration könnte man hier einen Dienst anbieten, bei dem es möglich ist, zur Laufzeit solche Einschränkungen zu konfigurieren, ohne Ausfallzeiten in Kauf nehmen zu müssen.

G. Langzeit-Lauffähigkeit

Das Versprechen der Cloud ist, Informationen und Dienste jederzeit verfügbar zu haben, unabhängig von der physikalischen Verfügbarkeit einzelner physikalischer Server oder Netzwerke respektive Teilnetzwerke. Der Aufwand und damit die Kosten, die durch die Umstellung bestehender Systeme (und der zugehörigen Prozesse) entstehen, werden gerechtfertigt dadurch, dass der Zugang zu kritischen Diensten und Informationen nicht alle paar Monate unterbrochen wird, um mit den Folgen von

- Hardware-Ausfällen,
- Turnusmäßigen Hardware-Aufstockung,
- Schwankenden Anforderungen,
- Ausweitung des Kerngeschäfts, oder
- Hosting-Wechsel

umzugehen.

In den obigen Anwendungsbeispielen kann diesen Problemen mit Live-Migration-Technologie auf VM-Ebene begegnet werden. Folglich ist die Flexibilität von Cloud-Hosting in Kombination mit dieser Technologie, im Vergleich zum klassischen Rechenzentrum, in diesen Bereichen höher, und besser an wechselnde Gegebenheiten anzupassen. Ein Anbieter kann daher längerfristig davon ausgehen, dass Dienste und Informationen verfügbar sind. Das vermindert den Druck und die Kosten, Dienste „In-House“ und Datensicherungen ausschließlich „vor Ort“ vornehmen zu müssen. Geschäftsmodelle können daraufhin für größere Marktschwankungen ausgelegt werden, ohne die Kosten für Hosting und Hardware unnötig in die Höhe zu treiben. Weiterhin können strategische Entscheidungen langfristiger getroffen werden, da die dauerhafte Verfügbarkeit einfacher zu gewährleisten ist, und nicht mit längerer Laufzeit rapide schwieriger und damit teurer wird.

III. Live-Migration-Systeme

Im Folgenden wollen wir einige konkrete Implementierungen der oben vorgestellten Techniken vorstellen, und deren Grenzen und Anwendbarkeit für die oben genannten Anwendugsfälle betrachten.

A. NomadBIOS und vMotion

Wie im Vortrag von Hansen beschrieben, war die erste vollständige Implementierung zur Live-Migration einer kompletten VM der NomadBIOS-Hypervisor. Asger Jensen und Jacob Gorm Hansen entwickelten diesen Hypervisor auf dem L4-Mikrokern und demonstrierten als erste erfolgreich Live-Migration von virtuellen Maschinen über ein lokales Netzwerk. Die Motivation dieser Arbeit kam aus dem Grid-Computing: Für jeden Job sollte eine VM gestartet werden, die dann, mit Prioritäten ausgestattet, durch das Grid verschoben werden konnte, auf der Suche nach freien Ressourcen. Daher rührt auch der Name *NomadBIOS*, ein „nomadisches“ System.

Im Rahmen ihrer Masterarbeit implementierten Hansen und Jensen den Hypervisor zunächst auf dem L4-Kern. Gastsysteme mussten angepasst werden, um auf diesem teilvirtualisierten System als Userspace-Prozess ausgeführt zu werden. Im Umkehrschluss konnten viele der Prinzipien hinter Prozessmigration verwendet werden, um Betriebssystemprozesse zu verschieben. Außerdem verwendet das NomadBIOS *Checkpointing*, den Akt, den Speicherzustand von Prozessen einzufrieren um sie später fortsetzen zu können.

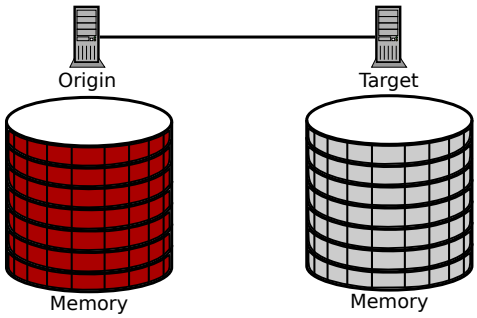

Abbildung 3. NomadBIOS: 1. Migrationsphase

1) Implementierungsdetails des NomadBIOS: Im NomadBIOS werden mehrere Userspace-Module für den L4-Kern implementiert, die dazu dienen, Gastbetriebssysteme von der Hardware zu isolieren [**?**]. Dazu gehören *Paging*, für die Isolation des Speichermanagements, *Interrupt Multiplexing* um Zugriff auf Hardware zu verweben und *Address Space Translation*, um einfache Kommunikation zwischen dem Gast- und dem Hostbetriebssystem zu ermöglichen. Letzteres wird genutzt, um Unterstützung für den Migrationsvorgang im Gastsystem implementieren zu können.

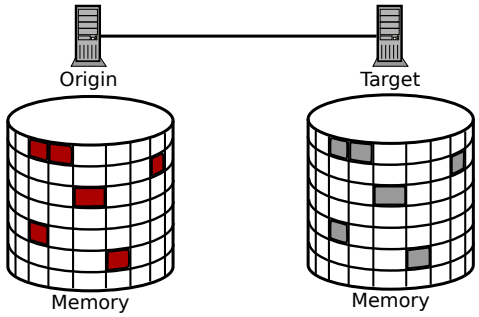

Abbildung 4. NomadBIOS: 2. Migrationsphase

Für die eigentliche Migration implementiert das NomadBIOS ein *Precopy*-Schema, wie es bereits aus der Prozessmigration bekannt ist [**?**].

1) In Abbildung 3 ist der Startzustand der Migration dargestellt. Initial ist der Zustand allen Speichers auf dem Zielsystem unbekannt (helles Grau). Der gesamte Speicher vom Ursprungssystem muss zum Kopieren ausgewählt werden (rot). Der Speicher auf dem Ursprung, der dem Gastsystem gehört, wird zunächst für Schreibzugriffe gesperrt und der Kopiervorgang auf den neuen Host wird gestartet. Wenn währenddessen Speicher geschrieben wird, wird ein Page-Fault ausgelöst, und das NomadBIOS kann die entsprechende Speicherseite erneut zum Kopieren markieren. Danach wird die Seite als für das Gastsystem beschreibbar markiert und geschrieben.
2) Nach einem ersten Kopierdurchlauf werden so eine Reihe von Speicherseiten noch zum Kopieren markiert worden sein. Es ist nicht klar ob diese Seiten vor oder nach der letzten Manipulation zum Zielsystem kopiert wurden (Abbildung 4). Diese können nun in einem weiteren, kürzeren Kopierdurchlauf mit dem Zielsystem

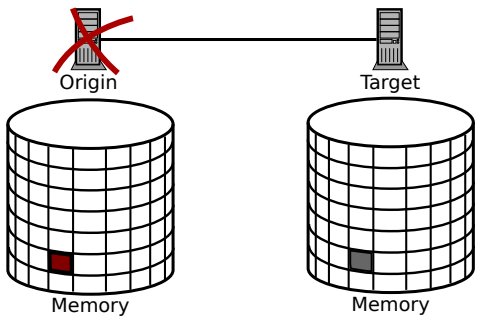

Abbildung 5. NomadBIOS: 3. Migrationsphase

synchronisiert werden. Da dieser zweite Kopiervorgang schneller vonstatten geht, werden währenddessen weniger Seiten wieder vom Gastsystem beschrieben werden.

3) Nach mehreren Iterationen ist das Delta zwischen dem Gast und Zielsystem so weit geschrumpft, dass die verbleibende Differenz in wenigen Millisekunden kopiert werden kann. In diesem Moment wird der Gastprozess auf dem Ursprungssystem eingefroren, die restlichen Seiten kopiert, und auf dem Zielsystem gestartet (Abbildung 5). Wenn im gleichen Moment auch das Routing angepasst wird, so kann die VM direkt weiterarbeiten, ohne das Benutzer mehr als einen kurzen Latenzanstieg bemerken.

Hansen arbeitet seit 2007 bei VMWare. VMWare bietet inzwischen ein kommerzielles Produkt zur Live-Migration names *vMotion*. Der primäre Fokus dieses Produktes liegt, ähnlich wie beim NomadBIOS, darin VMs innerhalb eines Netzwerkes mit gemeinsamem Speicher wandern zu lassen, um größtmögliche Ressourcennutzung zu erzielen. Dazu bietet VMWare vMotion die Möglichkeit, sogenannte *Resource Pools* zu definieren, in denen VMs fortwährend automatisiert wandern können.

Um höhere Verfügbarkeit zu gewährleisten, weicht vMotion vom oben etablierten Precopy-Schema ab [**?**]. Im letzten Schritt wird die Ziel-VM direkt gestartet, wenn die Ursprungs-VM eingefroren wird. Damit wird vermieden, dass beim Übertragen des letzten Deltas eventuell offene TCP Verbindungen einen Timeout erfahren. Die noch zum Kopieren ausstehenden Speicherseiten sind markiert, und werden noch in die Ziel-VM kopiert. Wenn zwischenzeitlich darauf zugegriffen werden soll, werden sie als nächstes über das Netzwerk zum Ziel übertragen. Das bedeutet, dass zu Beginn der Laufzeit der Ziel-VM noch eine Abhängigkeit zum Ursprungssystem besteht, und die Verbindung zwischen beiden Systemen noch kurze Zeit weiterbestehen muss.

Eine weitere Besonderheit an der VMWare Lösung ist, dass der VMWare ESXi-Hypervisor über Para-Virtualisierung hinausgeht, und deshalb die Migration von beliebigen Betriebsystemen zwischen beliebigen (von VMWare unterstützten) Hardware-Architekturen unterstützt [**?**]. Dafür wird Netzwerkhardware voll virtualisiert, außerdem wird davon ausgegangen, dass Verbindungen zu Speicher ausschließlich über Storage Area Network (SAN) oder Network Attached Storage (NAS) bereitgestellt werden, und dass alle VMs mit denselben SAN- bzw. NAS-Servern verbunden sind.

B. IBM

IBM unterstützt die Linux Community in großem Umfang [**?**]. IBM hat zur Live-Migration von Linux-VMs sowohl beim Xen-Projekt, als auch dem Kernvirtualisierungsmodul Kernel Virtual Machine (KVM) im Rahmen des RESERVOIR-Projektes wesentliche Beiträge geleistet [**?**]. Dabei lag der Fokus der angestrebten Lösung explizit auf Interoperabilität zwischen verschiedenen Anbietern und der Möglichkeit, Migration über Distanzen hinweg zu ermöglichen. Beides zusammen soll die Kosten für benötigte Infrastruktur (wie z.B. SAN-/NAS-Server) senken und die Skalierbarkeit über die Grenzen eines einzelnen Cluster Netzwerks hinweg erhöhen.

Die Lösung hierfür sind Virtual Execution Environment Manager (VEEMs). Ein VEEM verteilt VMs (in diesem Kontext Virtual Execution Environments (VEEs) genannt) über verfügbare Host-Systeme, unter Berücksichtigung bestimmter Einschränkungen, dazu können z.B. auch gesetzliche Bedingungen oder Service Level Agreements (SLAs) zählen. Außerdem können mehrere VEEs zu einer VEE-Gruppe zusammengefasst werden, um bestimmte Dienste nah beieinander zu halten und nur gemeinsam zu verwalten. IBM hat hierbei speziell darauf geachtet, offene Protokolle zu erstellen, um die Struktur der Lösung unabhängig von den verwendeten Virtualisierungsplatformen (z.B. Hypervisoren) zu machen. Auf diese Weise wird ein Migrationspfad für Cloud Anbieter geöffnet, der es ermöglicht, bestehende Infrastruktur inkrementell mit den nötigen APIs auszustatten und so interoperabel mit vielen verschiedenen Cloud-Lösungen zu werden.

C. HP

HP bietet Virtualisierungsdienste auf Basis von Microsoft's *HyperV*. *HyperV* unterstützt in Windows Server 2008 R2 bereits Live-Migration von virtuellen Maschinen, der Migrationsprozess läuft dabei analog zum NomadBIOS ab. Eine wesentliche Limitierung des HyperV ist, dass alle physikalischen Maschinen Zugriff auf gemeinsamen Speicher (NAS / SAN) haben müssen, um VM-Daten auszutauschen [**?**]. Das kann beim Umziehen zwischen geografisch entfernten Orten oft nicht gewährleistet werden. HP hat hierfür sein Cluster-Extension (CLX)-Produkt erweitert. CLX ist zunächst eine Software zur Replikation von Daten über große Distanzen. Mit den HyperV spezifischen Erweiterungen kann CLX nun dazu genutzt werden, den Zustand von VMs zwischen verschiedenen Rechenzentren zu synchronisieren. Dabei ist CLX voll in den HyperV-Migrationsprozess integriert. Mit der CLX-HyperV Kombination kann sichergestellt werden, das Replikation über Datencenter hinweg VM-Migrationen unterstützt, und nicht behindert, indem während der Migration dieser mehr Bandbreite zugestanden wird, damit die finale Speicherseitensynchronisation (während der die Ursprungs-VM eingefroren ist) so schnell wie möglich vonstatten geht. Wenn die Migration abgeschlossen ist, macht CLX den neuen physikalischen Standort der VM automatisch zum Master für die weitere Replikation.

Im März 2010 aktualisierte HP außerdem seine UNIX Distribution *HP-UX*. Teil dieses Updates war die Version 4.2 des hauseigenen *Integrity Virtual Machines* Hypervisors. Dieser enthält eine *Online Migration* Funktion, die ebenso mit CLX zusammen eingesetzt werden kann [**?**].

D. Oracle

Oracle bietet in Solaris (vormals bei Sun) eine betriebssystemintegrierte Form der Virtualisierung an: Solaris Zones. Im Gegensatz zu den vorher vorgestellten Lösungen werden in den Zones keine vollständigen Betriebsysteme ausgeführt, vielmehr gibt es eine Betriebssysteminstanz, genannt *global zone*, in der der Betriebssystemkern läuft und mehrere *non-global zones*, die diesen Kern mitbenutzen. Den *non-global zones* erscheint der Kern jedoch als ihr „eigener". Solaris Zones

werden zumeist zur Separierung von Applikationen genutzt und mit Resourcenmanagement verbunden; die Kombination wird *Solaris Container* genannt [**?**].

Im Kontext von Live-Migration sind Solaris Zones (und damit auch Container) jedoch ungeeignet. Es existieren keinerlei Vorrichtungen dafür im Kern von Solaris, geschweige denn Werkzeuge. Auch nicht-Live-Migrationen sind mit hohem Manuellen aufwand verbunden und entsprechen eher dem Prozess „Herunterfahren, exportieren, kopieren, importieren, hochfahren“ [**?**]. Die Migration von Zones als Standard-Anwendungsfall ist in Solaris nicht vorgesehen.

Oracle Logical Domains (LDOMs) (jetzt VM Server for SPARC), die auf SPARC-Servern der T-Serie und einigen anderen Systemen zur Verfügung stehen, unterstützen seit der Version 1.2 aus dem Jahr 2009 Live-Migration [**?**]. Diese hardwaregestützte Virtualisierungsvariante bietet physisch getrennte Gast-Systeme (Domänen) und dabei die Rekonfigurierbarkeit dieser im eingeschalteten Zustand. Zur Live-Migration von Domänen wird auf beiden Seiten nicht nur die selbe Software (in dem Fall auch Firmware) genutzt, auch die Hardware muss in Prozessortyp und -frequenz übereinstimmen. Es ist auch nicht möglich, von Vornherein bestimmte Auslastungsgrenzen zu bestimmen, ab denen eine Live-Migration vorgenommen wird. Vielmehr wird bei Auslastungsändereungen direkt auf dem Server reagiert, indem z.B. Prozessoren dazu- oder abgeschaltet werden oder der für die Domäne verfügbarer Speicher verändert wird [**?**]. Als Betriebssystem in den LDOMs sind praktisch nur Solaris (und – dank Linux-Kern-Emulation in bestimmten Solaris Zones – ältere Linux) bekannt, theoretisch ist es aber möglich, beliebige Betriebssysteme, die SPARC-Tx-Prozessoren unterstützen, zu nutzen; so z.B. natives Linux [**?**].

E. Vergleich

Wir kehren nun zurück zu den zuvor beschriebenen Anwendungsfällen für VM-Live-Migration (Abschnitt II). Die Fähigkeiten und Ziele der in Abschnitt III beschriebenen Lösungen machen sie unterschiedlich gut für die beschriebenen Zwecke einsetzbar (Tabelle I). Im Folgenden werden wir die Systeme gemäß ihrer Ziele und Beschreibungen kurz mit den entsprechenden Beispielanwendungen in Beziehung setzen. Dieser Vergleich basiert zum größten Teil auf den Informationen der Hersteller.

Verlässlichkeit: Hardware-Fehler sind ein Problem das sich nicht in absehbarer Zeit lösen lässt. Virtualisierung bietet allerdings die Möglichkeit, angebotene Dienste von physikalischer Hardware zu entkoppeln. Mit Live-Migration kann mit einigen Hardwarefehlern umgegangen werden, ohne dass deshalb der oder die Dienste, die gerade auf der Hardware laufen, abgeschaltet werden müssen.

Alle vorgestellten Systeme können mit diesem Szenario umgehen. Nach den LDOMs von Oracle hat VMWare mit vMotion out-of-the-box hier wohl die engsten Begrenzungen, da SAN bzw. NAS Server gemeinsam von Quell- und Zielsystemen erreichbar sein müssen. Das heißt, dass der Ausfall eines gesamten Rechenzentrums nicht ohne Mehraufwand kompensierbar ist, da hierfür Replikation des Netzwerkspeichers getrennt gelöst werden muss. Hier bietet HP mit CLX und HyperV die integrierte Lösung, und macht sogar explizit Werbung mit der Migration von Streaming-Services sogar bei katastrophalen Fehlern in einem Rechenzentrum. Die von IBM erstellten APIs wurden ebenfalls explizit mit dem Gedanken erstellt, Migration über große Entfernungen zu ermöglichen, und sind deshalb *per se* geeignet, um Dienste gegen weitreichende Hardware-Ausfälle abzusichern.

Interoperabilität: Virtualisierungslösungen existieren unabhängig von der Cloud (Abschnitt I-B1) und es besteht Interesse, die Abhängigkeit von einem bestimmten Hersteller so gering wie möglich zu halten, indem bei den verwendeten Systemen auf Interoperabilität geachtet wird. Live-Migration ist hier interessant, damit beim Wechsel der Virtualisierungslösung nicht Dienste offline gehen müssen.

Hier sind die para-virtualisierenden Systeme (Xen, KVM) eingeschränkter, da am Gast-System Anpassungen vorgenommen werden müssen, die unter Umständen nicht im Betrieb zu ändern sind, weshalb man nicht einfach ein System in ein anderes übertragen kann. Bei Solaris Zones ist ein Gastsystem meist gar kein vollständiges System, sondern teilt wesentliche Teile der POSIX Umgebung mit den anderen Gästen und dem Host. Ein Extrahieren eines einzelnen Gastes ist gar nicht ohne weiteres möglich. Bei den LDOMs besteht wiederum das Problem, dass die virtualisierten System zwar volle Betriebssysteme sind, die Virtualisierung jedoch zu weiten Teilen aus der Firmware kommt und damit unportabel ist. VMWare ESXi, als das vorgestellte System mit dem höchsten Level der Virtualisierung, macht das Umwandeln in andere Systeme derzeit am leichtesten, da Speicher im Netzwerk liegt und weite Teile der Hardware virtualisiert sind, weshalb Gastssysteme hier auch ohne Anpassungen laufen können. Aber auch hier ist eine Live-Migration in ein anderes System unseres Wissens nicht möglich.

Auf Dauer ist das RESERVOIR-Projekt von IBM am ehesten geeignet, die Beschränkungen beim Umstieg von einem System auf das andere Aufzuheben, und tatsächlich Live-Migration zwischen den Systemen zu bieten. Die Abstraktion von verwendeten Hypervisor-Technologien und dem verwendeten Speicher wird es ermöglichen, zwischen zu den beschriebenen Schnittstellen kompatiblen Systemen dynamisch zu migrieren.

Inter-Cloud Verschiebungen: Ebenso wie die Abhängigkeit von einer Virtualisierungslösung, soll die Bindung an einen Cloud-Anbieter möglichst schwach bleiben. Deshalb sollte es möglich sein, Dienste auch zwischen Clouds zu verschieben.

Derzeit gibt es keinen einheitlichen Standard für Live-Migration und die verwendeten Protokolle. Wie bereits angemerkt unterstützen mit Ausnahme von HP und IBM die Systeme von sich aus gar keine Verschiebung zwischen Netzwerken mit getrenntem Speicher. Weiterhin sind die Schnittstellen der Hypervisoren nicht einheitlich, was eine Migration zwischen Clouds mit verschiedenen Virtualisierungstechnologien erschwert. Derzeit ist unter Umständen möglich über Umwege wie selbst eingerichtete Replikation zwischen Cloud-Anbietern mit denselben Technologien Live-Migration durchzuführen. Wirklich möglich wird auch das allerdings erst, wenn die RESERVOIR-Schnittstellen oder die vorgeschlagenen Cloud-Standards von vielen Lösungen implementiert werden.

Tabelle I
ANWENDUNG-ANBIETER-MATRIX

	VMWare	IBM	HP	Oracle
Verlässlichkeit	✓	✓	✓	
Interoperabilität	✓	✓		
Inter-Cloud Verschiebungen		✓	(✓)	
Hardware Maintenance	✓	(✓)	✓	✓
Adaptive Auslastung	✓	✓	✓	(✓)
Erzwungene Verschiebungen		(✓)	(✓)	

Hardware-Maintenance: Die Kosten, alte Hardware am Laufen zu halten, übersteigen nach einiger Zeit immer die Anschaffungskosten für neue, bessere Hardware. Deshalb ist es im Interesse eines Unternehmens, neue Hardware nutzen zu können, ohne Verfügbarkeit einschränken zu müssen.

Hier punkten die integrierten Lösungen von HP, Oracle und VMWare, da sie bereits die Abstraktion von Ressourcen ermöglichen, und Verwaltung von Hardware als *Resource Pools* vereinheitlichen. Dabei kann neue Hardware hinzugefügt und alte entfernt werden, und das System konfiguriert sich selbst neu und verteilt die Last automatisch auf die neu verfügbare Hardware. Auch IBM kennt diese Abstraktion in Form von VEEs, allerdings muss diese Lösung noch entsprechend weitreichend implementiert werden. Auch bei der LDOM-basierten Lösung ist es möglich, dass zwar Hardware eines Systems aufgestockt und ausgetauscht werden kann, ohne Gast-VMs herunterfahren zu müssen.

Adaptive Auslastung: Dienste, die sich auf bestimmte Zielgruppen, Länder oder Themengebiete spezialisieren, sehen oft stark schwankende Nutzerzahlen, sei es, weil bestimmte Ereignisse Zugriffe in die Höhe treiben, oder schlicht weil die Zielgruppe generell im selben Land zu finden ist, und tendenziell zu ähnlichen Zeiten den Dienst in Anspruch nimmt. Als Anbieter hat man hier ein Interesse daran, nur die Menge an Hosting-Kosten zu zahlen, die zu einem gegebenen Zeitpunkt benötigt wird. Dazu kann Live-Migration beitragen, indem Dienste je nach Last auf schnellere oder langsamere Systeme verschoben werden, und so mehr oder weniger CPU-Zeit bekommen.

Hier sind alle vorgestellten Systeme einsetzbar, der Umfang der Schwankungen sollte über die Auswahl entscheiden. Bei vMotion und CLX sind relativ weitreichende Änderungen der physikalischen Hardware beim Migrieren möglich, was in bei einer hohen Varianz der verwendeten Hardware nützlich ist. Bei den von IBM unterstützten Lösungen Xen und KVM müssen die unterliegenden Systeme relativ ähnlich sein, was die Hardwareauswahl einschränkt. Bei LDOM-basierten Lösungen, schließlich, muss die Hardware nahezu identisch sein.

Erzwungene Verschiebungen: In manchem Anwendungsgebieten müssen Dienste an bestimmten geografischen Orten ausgeführt werden, sei es aus Sicherheitsgründen oder weil ein Gesetz es vorschreibt. Ein Anbieter von Datenverarbeitungsdiensten möchte nicht jedes mal, wenn eine solche Einschränkung in Kraft tritt, bestehende, und möglicherweise lang laufende Jobs abbrechen, um sie an einem anderen Ort neu zu starten.

Die LDOM-Lösung fällt hier durch, da Migration über geografische Entfernungen nicht angemessen möglich sind. Auch vMotion ist ungeeignet, da nicht ohne weiteres zwischen entfernten Netzwerken migriert werden kann. HP bietet hier zwar mit der CLX eine Lösung, allerdings steht zumindest zu vermuten, dass die reine Ausführung an anderer Stelle nicht eventuellen Sicherheits- oder Gesetzesrichtlinien entspricht, wenn es eine Replikation der Daten an verschiedenen Orten gibt. Trotzdem scheint das derzeit die einzige verfügbare Lösung zu sein. Einzig die von IBM vorgeschlagenen VEEMs wären hier besser geeignet, da sie, laut Spezifikation, die Möglichkeit bieten sollen, derlei Einschränkungen an den Ort der VM-Daten zu konfigurieren. Damit ließe sich eine allgemeine Lösung auch automatisieren.

IV. SCHLUSSFOLGERUNGEN

Keines der betrachteten Systeme kann die gestellen Anforderungen in vollem Umfang erfüllen. Gleichwohl bietet die Kombination der von IBM vorgestellten Techniken, insbesondere durch die Implementierung der vorgeschlagenen Standards, schon ein hohes Maß gut handhabbarer und konfigurierbarer Live-Migration von VMs. Während sich die Lösungen von VMWare eher in Richtung dynamischer Anpassung von VM-Infrastruktur orientieren, sind die von HP bereitgestellten Techniken eher auf Ausfallsicherheit und eine sehr hohe Verfügbarkeit ausgelegt. Beide lassen aber bei der Interoperabilität miteinander und auch mit anderen Live-Migrationssystemen noch Spielraum. Es ist zu wünschen, dass sich auf Standards dafür geeinigt wird, gegebenenfalls auch die von IBM vorgeschlagenen. Die LDOM-Lösung von Oracle hat einen hohen Hardwarebezug, wodurch der Vergleich mit anderen Anbietern schwierig ist.

Momentan bietet es sich an, wenn schon eine Virtualisieungslösung einer der Anbieter im Einsatz ist, dessen Live-Migrationsfähigkeiten zu prüfen und zu nutzen. Zum aktuellen Zeitpunkt ist es jedoch nicht möglich, eine allgemeine Empfehlung für oder gegen eine bestimmte Lösung auszusprechen. Das Wechseln der Virtualisierungslösung nur zum Zweck, die entsprechende Live-Migrationsfähigkeit zu nutzen, lohnt momentan nicht.

In Zukunft ist mit einer weitergehenden Standardisierung der an Live-Migration beteiligten Techniken zu Rechnen, die Anbieter scheinen auch daran Interesse zu haben. Es ist damit zu rechnen, dass Live-Migration, genau wie Virtualisierung, eine Normalität beim Serverbetrieb werden wird.

LITERATUR

[1] TANENBAUM, A.S. ; TANNENBAUM, A.: *Modern operating systems*. Bd. 271. Prentice Hall Englewood Cliffs, NJ, 1992

[2] *Google Apps Service Level Agreement*. http://www.google.com/apps/intl/en/terms/sla.html. Version: 31. März 2011

[3] SCHWARZER, Ingo: *Cloud Hype? Trend? Fake?: Einordnung in die Unternehmensstrategie*. 3. Februar 2011; Wintersemester 2010. – Vortrag im Rahmen des Industrieseminars „Cloud-Computing“, Studiengang IT-Systems Engineering, Hasso-Plattner-Institut, Universität Potsdam

[4] HALAVAIS, A.M.C.: *The Slashdot Effect: Analysis of a large-scale public conversation on the world wide web*, UMI, Ann Arbor, Mich., Diss., 2001

[5] *Heroku*. http://www.heroku.com/. Version: 21. März 2011

[6] *Engine Yard*. http://www.engineyard.com/. Version: 21. März 2011

[7] *OpenStack Cloud Software*. http://www.rackspace.com/cloud/openstack/. Version: 30. März 2011

[8] HANSEN, Jacob G.: *Virtualization as a foundation for the Cloud*. 09. Dezember; Wintersemester 2010. – Vortrag im Rahmen des Industrieseminars „Cloud-Computing“, Studiengang IT-Systems Engineering, Hasso-Plattner-Institut, Universität Potsdam

[9] GIESEKUS, Andreas: *Virtualisierung und Konsolidierung von Unix-Plattformen bei der gkv informatik*. 25. November; Wintersemester 2010. – Vortrag im Rahmen des Industrieseminars „Cloud-Computing“, Studiengang IT-Systems Engineering, Hasso-Plattner-Institut, Universität Potsdam

[10] SHOUMACK, Tony: *Beginners Guid to LDoms*. 2.1. Santa Clara, CA: Sun Microsystems, Inc., Juli 2007

[11] *Free VMware vSphere Hypervisor: Bare Metal Hypervisor (Based on VMware ESXi)*. http://www.vmware.com/products/vsphere-hypervisor/overview.html

[12] BARHAM, Paul ; DRAGOVIC, Boris ; FRASER, Keir ; HAND, Steven ; HARRIS, Tim ; HO, Alex ; NEUGEBAUER, Rolf ; PRATT, Ian ; WARFIELD, Andrew: Xen and the art of virtualization. In: *Proceedings of the nineteenth ACM symposium on Operating systems principles*. New York, NY, USA : ACM, 2003 (SOSP '03). – ISBN 1–58113–757–5, 164–177

[13] ORACLE CORPORATION: *Oracle VM Server for SPARC*. http://www.oracle.com/us/technologies/virtualization/oraclevm/oracle-vm-server-for-sparc-068923.html. Version: 3. März 2011, Abruf: 28. März 2011

[14] PRICE, D. ; TUCKER, A.: Solaris zones: Operating system support for consolidating commercial workloads. In: *Proceedings of the 18th USENIX conference on System administration* USENIX Association, 2004, S. 241–254

[15] AHMED, M. ; ZAHDA, S. ; ABBAS, M.: Server consolidation using OpenVZ: Performance evaluation. In: *Computer and Information Technology, 2008. ICCIT 2008. 11th International Conference on* IEEE, 2008, S. 341–346

[16] VMWARE, INC: *VMware Workstation: Run Multiple OS Including Linux & Windows7, on Virtual Machines*. http://www.vmware.com/products/workstation/

[17] ORACLE CORPORATION: *VirtualBox*. http://www.virtualbox.org/

[18] SUN DEVELOPER NETWORK: *The Java Virtual Machine Specification*. http://java.sun.com/docs/books/jvms/. Version: 04 2003

[19] Norm INCITS 319-1998 1998. *American National Standard for Information Systems – Programming Languages – Smalltalk*

[20] GEBHART, Alexander: *Virtualization and Cloud Computing @ TIP Core*. 11. November; Wintersemester 2010. – Vortrag im Rahmen des Industrieseminars „Cloud-Computing“, Studiengang IT-Systems Engineering, Hasso-Plattner-Institut, Universität Potsdam

[21] *Google Apps*. http://www.google.com/apps/

[22] *Google App Engine: Run your web applications on Google's infrastructure.* http://appengine.google.com

[23] WAGENER, Walfried: *Cloud Computing@BIOTRONIK*. 4. November; Wintersemester 2010. – Vortrag im Rahmen des Industrieseminars „Cloud-Computing“, Studiengang IT-Systems Engineering, Hasso-Plattner-Institut, Universität Potsdam

[24] *Flight Caster Success Story*. http://http://success.heroku.com/flightcaster. Version: 29. März 2011

[25] ZELLER, A.: *Why programs fail*. Elsevier/Morgan Kaufmann, 2006. – ISBN 1558608664

[26] HANSEN, J.G. ; JUL, E.: Self-migration of operating systems. In: *Proceedings of the 11th workshop on ACM SIGOPS European workshop* ACM, 2004, S. 23

[27] CLARK, C. ; FRASER, K. ; HAND, S. ; HANSEN, J.G. ; JUL, E. ; LIMPACH, C. ; PRATT, I. ; WARFIELD, A.: Live migration of virtual machines. In: *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2* USENIX Association, 2005, S. 273–286

[28] HANSEN, J.G. ; HENRIKSEN, A.K.: Nomadic operating systems. In: *Master's thesis, Dept. of Computer Science, University of Copenhagen, Denmark* (2002)

[29] NELSON, M. ; LIM, B.H. ; HUTCHINS, G.: Fast transparent migration for virtual machines. In: *Proceedings of the annual conference on USENIX Annual Technical Conference*, 2005, S. 25–25

[30] *Engine Yard CLI User Guide*. http://docs.engineyard.com/appcloud/guides/deployment/home. Version: 21. März 2011

[31] *Chef*. http://www.opscode.com/. Version: 21. März 2011

[32] *Amazon EC2 FAQ*. http://aws.amazon.com/ec2/faqs/#What_is_VM_Import. Version: 31. März 2011

[33] *Cloud Standards*. http://www.cloud-standards.org. Version: 4. März 2011

[34] KOHAVI, R. ; LONGBOTHAM, R.: Online Experiments: Lessons Learned. In: *Computer* 40 (2007), Nr. 9, S. 103–105. – ISSN 0018–9162

[35] *FlightCaster*. http://www.flightcaster.com/. Version: 29. März 2011

[36] FOLKERTS, Enno: *Benchmarking the Cloud*. 20. Januar; Wintersemester 2011. – Vortrag im Rahmen des Industrieseminars „Cloud-Computing“, Studiengang IT-Systems Engineering, Hasso-Plattner-Institut, Universität Potsdam

[37] DEMAL GMBH: *Änderungen des BDSG – eine Zusammenfassung*. http://www.demal-gmbh.de/datenschutz/info/Aenderungen-des-BDSG.pdf. Version: 01. September 2009

[38] KROAH-HARTMAN, G. u. a.: Linux kernel development. In: *Linux Symposium*, 2007, S. 239–244

[39] ROCHWERGER, B. ; BREITGAND, D. ; LEVY, E. ; GALIS, A. ; NAGIN, K. ; LLORENTE, IM ; MONTERO, R. ; WOLFSTHAL, Y. ; ELMROTH, E. ; CACERES, J. u. a.: The reservoir model and architecture for open federated cloud computing. In: *IBM Journal of Research and Development* 53 (2009), Nr. 4, S. 4–1. – ISSN 0018–8646

[40] HEWLETT-PACKARD DEVELOPMENT COMPANY: *Live Migration across data centers and disaster tolerant virtualization architecture with HP StorageWorks Cluster Extension and Microsoft Hyper-V*. 1. Palo Alto, CA: HP, 2010

[41] HEWLETT-PACKARD COMPANY: *HP Integrity Virtual Machines 4.2: Release Notes*. 11. Palo Alto, CA: HP, 2010

[42] KIMCHI, Orgad: *Solaris Zones migration with ZFS*. http://blogs.sun.com/vreality/entry/solaris_zone_migration_with_zfs

[43] LAURENT, Jim: *Answering a customer's LDOMs security questions*. http://blogs.sun.com/jimlaurent/entry/answering_a_customer_s_ldoms

[44] ORACLE CORPORATION (Hrsg.): *Oracle VM Server for SPARC 2.0 Administration Guide*. 2. Redwood Shores, CA: Oracle Corporation, September 2010

[45] CHARTRE, Alexandre: *Linux with LDoms*. http://blogs.sun.com/achartre/entry/linux_with_ldoms