Die approbierte Originalversion dieser Diplom-/Masterarbeit ist an der Hauptbibliothek der Technischen Universität Wien aufgestellt (http://www.ub.tuwien.ac.at).

The approved original version of this diploma or master thesis is available at the main library of the Vienna University of Technology (http://www.ub.tuwien.ac.at/englweb/).

FAKULTÄT FÜR !NFORMATIK

Virtualisierung in Bildungseinrichtungen

MAGISTERARBEIT

zur Erlangung des akademischen Grades

Mag.rer.soc.oec.

im Rahmen des Studiums

Informatikmanagement

eingereicht von

DI(FH) Martin Adler
0727701

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung:
Betreuerin: DI Dr. Monika Di Angelo

Wien, 23.07.2009 _____ _____
(Unterschrift Verfasser) (Unterschrift Betreuerin)

KURZFASSUNG

Die Diplomarbeit beschäftigt sich mit der Implementierung und Anwendung von virtuellen Umgebungen im Bildungsbereich. Dabei werden sowohl technische, finanzielle als auch didaktische Aspekte berücksichtigt um in weiterer Folge einen möglichen Einfluss auf Unterrichtskonzepte und Lehrpläne zu ermitteln und diesen zu interpretieren.

Eine der Kernfragen im Zusammenhang mit Virtualisierung ist, wie der Benutzer, in diesem Fall der Schüler, tatsächlich mit der virtuellen Maschine arbeiten kann. Um diese Frage diskutieren zu können werden grundsätzliche Ansätze ermittelt und in weiterer Folge aktuelle Technologien zur Virtualisierung hinsichtlich ihrer Tauglichkeit evaluiert.

Nachdem die technischen Grenzen ermittelt sind, wird ein speziell auf den schulischen Einsatz zugeschnittenes Konzept erstellt. Dieses soll aber dennoch soweit generisch sein, dass es einfach an die jeweilige Situation angepasst werden kann.

Das Konzept wird im Anschluss exemplarischer aufgebaut. Diese Beispielimplementierung soll im Wesentlichen zwei Ziele verfolgen, die Demonstration der Funktion sowie als Leitfaden der im Fall einer Implementierung die grundsätzlichen Schritte auflistet.

Danach wird basierend auf dem generischen Konzept sowie der exemplarischen Realisierung der Einfluss auf die Didaktik diskutiert. Für diese Analyse werden Schultypen sowie die Lehr- und Unterrichtspläne dieser herangezogen.

Die Erkenntnisse beziehungsweise Erfahrungen die im Zuge der Arbeit gewonnen wurden sind in einem abschließenden Kapitel zusammengefasst.

ABSTRACT

This thesis deals with the implementation as well as the usage of a virtual infrastructure in an educational environment. In order to determine the impact on didactics a comprehensive analysis concerning technical financial as well as didactical aspects will be provided.

One of the main questions in virtualization is how the user can work with his virtual machine. Therefore this thesis will discuss different technologies and will give a final statement how to use state of the art approaches.

Once the technical limitations are established, a concept will be developed. The main goal is to provide something generic that can be easily adjusted to local needs.

In order to have some guideline and a best practice feeling this concept will be realized, including all necessary steps from setting up the operating system to configuring the applications.

In one of the final chapters the concept and the exemplary deployment will be tested against didactical needs.

ABKÜRZUNGSVERZEICHNIS

VM	Virtuelle Maschine
KVM	Kernel Virtualization Module
CPU	Central Processing Unit

SCHLÜSSELBEGRIFFE

Virtualisierung

KVM

XEN

INHALTSVERZEICHNIS

1. EINLEITUNG 1

1.1 AUSGANGSLAGE/AKTUELLE SITUATION 1
1.2 MOTIVATION UND FRAGESTELLUNG DER ARBEIT 4
1.3 ZIEL DER ARBEIT 5

2. VIRTUALISIERUNG 7

2.1 WAS IST VIRTUALISIERUNG 7
2.2 ANSÄTZE ZUR VIRTUALISIERUNG 7
2.3 VIRTUALISIERUNGSSOFTWARE 12
2.4 NACHTEILE 18
2.5 VORTEILE 20

3. TECHNISCHE EVALUIERUNG 25

3.1 ANSÄTZE ZU EINEM KONZEPT 25
3.2 ZENTRALE STEUERUNG DER EIN- UND AUSGABE GERÄTE 26
3.2.1 Virtualisierungssoftware 26
3.2.2 Gast Betriebssystem Evaluierung 27
3.2.3 Hardware 28
3.2.4 Konfiguration 31
3.2.5 Fazit 33
3.3 DEZENTRALE STEUERUNG DER EIN- UND AUSGABE GERÄTE 35
3.3.1 Ansteuerung der Ein- und Ausgabegeräte 35
3.3.2 Virtualisierungssoftware 38
3.3.3 Fazit 38

4. TECHNISCHES KONZEPT 40

4.1 KOMPONENTEN 40
4.1.1 Dispatch 40
4.1.2 Virtualisierung 41
4.1.3 LDAP 42
4.1.4 PXE Server 43
4.1.5 Management 43
4.1.6 ThinClient sowie Ein- und Ausgabegeräte 44
4.2 INTERAKTION DER KOMPONENTEN 44
4.3 SKALIERUNG UND ADAPTIERUNG 51
4.4 FINANZIELLE ASPEKTE 53
4.4.1 Anschaffungskosten 53
4.4.2 Operative Kosten 57
4.5 INTEGRATION IN BESTEHENDE SCHULNETZWERKE 61

5. TECHNISCHE IMPLEMENTIERUNG 63

5.1 UMSETZUNG DES KONZEPTES 63
5.2 INSTALLATION VON CENTOS 65

5.3 VIRTUALISIERUNG KVM 69
5.4 OPENTHINCLIENT SERVER 71
5.5 LDAP 72
5.6 DISPATCH 73
5.7 MANAGEMENT INTERFACE 74

6. ANWENDUNG/FALLBEISPIEL DIDAKTISCHE ASPEKTE 77

6.1 NOTEBOOK KLASSEN 79
6.2 UNTERSCHIEDLICHE BETRIEBSSYSTEME IM UNTERRICHT 81

7. FAZIT/SCHLUSSWORT 88

7.1 REFLEXION DER ZIELE 88
7.2 AUSBLICK 89

1. EINLEITUNG

1.1 Ausgangslage/Aktuelle Situation

Im Jahr 2007 hat die Donauuniversität Krems im Auftrag des Bundesministeriums für Unterricht und Kultur den Einsatz und die Tauglichkeit von Linux an Schulen im Rahmen einer Studie untersucht. [SCHW06]

Dies mag zwar nun nicht das Thema per se sein, allerdings lassen sich durchaus einige grundsätzliche Gemeinsamkeiten finden.

Im Jahr 2003 verwendeten etwa 98% aller PCs im österreichischen Schulbereich Microsoft Windows als Betriebssystem und Microsoft Office als „die Büroanwendung“. Laut den Verfassern sind hinsichtlich der folgenden Punkte immer wieder Bedenken und Probleme geäußert worden: [SCHW06]

Hohe Kosten

Durch die Verwendung von lizenzpflichtiger Software sind jährlich nicht unerhebliche Beträge für Lizenzen zu bezahlen.

Sicherheitsprobleme (Viren, Abstürze, Hacker, Trojaner, …)

Schulcomputer sind wesentlich anfälliger auf diverse Viren und Schadsoftware. USB-Sticks oder sonstige portable Speichermedien werden verwenden um diverse Daten zu transportieren. Nachdem diese Daten auch durchaus schulischer Natur sein können, kann diese ohne Alternative auch nur sehr schwer verhindert werden. Diese Verwendung ermöglicht aber das Einschleusen von Viren, Trojanern oder sonstiger anderer Schadsoftware. Zwar kann hier mit spezieller Arbeitsplatzüberwachungssoftware Abhilfe

geschaffen werden, nur ist diese für den schulischen Betrieb meist unerschwinglich, setzt einschlägiges Know-how voraus und ist äußerstet wartungsintensiv. Die Aufbewahrung beziehungsweise Speicherung von Schülerdaten könnte natürlich auch auf Netzwerklaufwerken geschehen. Dies würde den Schülern erlauben ihre eigenen Daten zu sichern ohne externe Speichermedien verwenden zu müssen. Allerdings müsste eine Möglichkeit geschaffen werden auf diese Daten auch von zum Beispiel zuhause zugreifen zu können.

Hoher Wartungsaufwand

Ein großes Problem an Schulen ist auch der ständige Wechsel der Benutzer. Dies liegt prinzipiell in der Natur der Sache, da ein Informatiksaal einfach Ressourcen zur Verfügung stellt, die von mehreren Schülern unterschiedlich genutzt werden. Schüler oder allgemeiner Benutzer, verfügen allerdings über unterschiedliche Wissensstände beziehungsweise sind versierter oder weniger versiert im Umgang mit Computern.

All diese Herausforderungen verlangen somit eine ständige Überwachung, Wartung, Wiederherstellung und Modifikation an berstender EDV-Saal Infrastruktur. Dies benötigt viel Zeit, und da die Säle tagsüber vermutlich ausgelastet sein werden muss sich jemand findet der dies außerhalb des normalen Schulbetriebes macht. Aber auch dies skaliert nur bis zu einem gewissen Grad, denn sollte „ein Experiment“ eines Schülers in der ersten Stunde schief gehen, und der Arbeitsplatz nicht mehr verwendbar sein, so fehlt den restlichen Tag dieser Platz.

In diesem Zusammenhang sei vielleicht noch angemerkt, dass oftmals Lehrer die Aufgaben eines IT-Administrators übernehmen. Selbst wenn der besagte Lehrer über das nötige Know-How verfügt, muss er sich dennoch einen großen Teil seiner Arbeitszeit anderen Aufgaben widmen, wodurch für die IT-Infrastruktur, sei dies nun eine neue Implementierung oder die Lösung eines bestehenden Problems, oft nicht genug Zeit aufgebracht werden kann.

Die Problematik, dass Lehrer oft als IT-Experten herhalten müssen ist weitläufig bekannt und so hat das Land Tirol hat im Jahr 2002 ein Konzept für zukünftige IT-Landschaften in Schulen ausgearbeitet. Dieses Konzept sieht vor, dass „wirkliche“ IT-Experten zur Verfügung stehen. [TIROL02]

Hohe Hardwareanforderungen

Aus der Studie geht nicht eindeutig hervor ob sich dieser Punkt nun an die Software wendet, welche bestimmte Vorrausetzungen erfüllt haben muss oder an die Anzahl an verfügbaren Geräten.

Da häufig lediglich eine überschaubare Anzahl an Standardanwendungen (klassische Bürosoftware, Programmierentwicklungsumgebungen, usw.) verwendet wird, kann davon ausgegangen werden, dass die Anforderungen dieser Software sehr schnell und einfach auch durch etwas ältere Hardware erfüllt werden können. Somit kann diese Aussage quantitativ betrachtet werden.

Applikationen für Schüler

Schüler benötigen zuhause die gleichen Applikationen wie in der Schule. Durch die Verwendung von lizenzpflichtiger Software beziehungsweise der fehlenden Möglichkeit die Software der Schule auch zuhause zu verwenden begeben sich Schüler immer wieder in illegale Situationen indem sie Raubkopien anfertigen.

Sollte IT nötig sein um die im Lehrplan vorgegeben Lehrziele zu erreichen können diese mitunter nicht vollständig erfüllt werden wenn die verwendete Infrastruktur die nötigen Anforderungen nicht erfüllt.

Mangelnde Alternativen

Werden im Unterricht immer dieselben Applikationen/Hersteller verwendet, können den Schülern keine Alternativen aufgezeigt werden.
Weiteres soll speziell in der Schule auch experimentiert werden können, Dinge erprobt und dadurch erlernt werden (Entdeckendes Lernen). „Probieren“ kann allerdings neben dem „guten“ Ende auch ein „schlechtes“ finden, was in diesem Fall mit unter das Betriebssystem eines EDV Saal Arbeitsplatzes zerstört. Dadurch entstehen zwangsläufig Konflikte mit den IT-Verantwortlichen.

Oft wird als so genannte Büroanwendungen Microsoft Office, das wiederum Microsoft Windows als Betriebssystem voraussetzt, verwendet. Dies macht auch durchaus Sinn, da Microsofts Office auch im Berufsleben meist vorgefunden wird. Allerdings lassen sich speziell in technischen Schulen (zum Beispiel höhere technische Lehranstalten) viele Aspekte und Techniken wesentlich leichter mit Hilfe von Linux darstellen und in weiterer Folge ausprobieren.

Durch die nicht homogene Umgebung in EDV Sälen ist es auch sehr schwer die Systeme speziell auf den Nutzen einzelner Klassen oder Abteilungen zu konfigurieren. Zwar lässt sich das optische Auftreten des Betriebssystems meist auf Benutzerebene modifizieren, doch die installierten Applikationen sind irgendwie doch immer für jeden verfügbar. Dies wirkt sich zwangsläufig negativ auf die Stabilität des Gesamtsystems aus, da jede installierte Applikation einer zusätzlichen potentiellen Fehlerquelle darstellt.

1.2 Motivation und Fragestellung der Arbeit

Ein interessanter Aspekt der Studie „Linux an Schulen“ [SCHW06] ist die Tatsache, dass die Hardware vielen Schulen aktualisiert werden muss. Auch deuten die in Kapitel 1.1 angeführten Punkte durchaus darauf hin, dass die die aktuelle Situation verbesserungswürdig ist.

Dadurch ergäbe sich die Möglichkeit in neue innovative Technologie zu investieren die dann über mehrere Jahre verwendet werden kann. Momentan ist in der IT ein sehr starker Trend in Richtung Virtualisierung zu verzeichnen und so stellt sich die Frage ob diese Technik auch in Bildungseinrichtungen eine brauchbare Möglichkeit wäre.

1.3 Ziel der Arbeit

Das primäre Ziel der Arbeit ist es ein Konzept zu entwickeln, welches den Einsatz von Virtualisierung an Schulen ermöglicht und die aktuellen Probleme löst.

Eine zentrale Frage in der Virtualisierung ist „wie bedient der Benutzer seine virtuelle Maschine“. Ziel ist es jedem Benutzer ein eigenes Set an Eingabegeräten, sprich Tastatur und Maus, sowie ein Ausgabegerät (Monitor) zur Verfügung stehen. Dies soll weitgehend unabhängig von den Systemen der anderen Schüler sein.

Weiters soll jedem Benutzer ein „eigenes Betriebssystem“ zur Verfügung stehen. Dieses kann speziell nach den eigenen Wünschen eingerichtet und modifiziert werden.

Das „eigene Betriebssystem“ soll von überall erreichbar sein, so dass es zum Beispiel auch von zuhause aus benutzbar ist.

Der operative Aufwand sowie das Management der Benutzerbetriebssysteme sollen so gering beziehungsweise so einfach wie möglich sein. Sollte zum Beispiel ein „eigenes Betriebssystem“ nicht mehr funktionsfähig sein, muss dieses sehr rasch und anwenderfreundlich entweder wiederhergestellt werden können oder zumindest ein neues verfügbar sein.

Das System soll flexibel auf Anforderungen reagieren können um neue Bedürfnisse schnell und unkompliziert zu befriedigen. Auch soll es, wie schon in [SCHW06] gefordert, Lehrern die Freiheit geben Applikationen selbst auszuwählen, und zwar ohne auf das nötige Betriebssystem Rücksicht nehmen zu müssen.

Es soll die Möglichkeit bestehen aktiv, kreativ und experimentell mit der IT umzugehen. [SCHW06]

Das Konzept soll eine lokale Adaptierung erlauben und mit Anforderungen wachsen können.

Das Gesamtsystem soll möglichst kostengünstig sein.

2. VIRTUALISIERUNG

2.1 Was ist Virtualisierung

Der Begriff Virtualisierung ist nicht eindeutig definiert, allerdings wird im Allgemeinen die Emulation beziehungsweise Simulation von Hardware verstanden [PICH09]. Technisch betrachtet kann diese logische Separation als Abstraktionsschicht zwischen Hardware und Software betrachtet werden.

Üblicherweise folgen traditionelle Computersysteme dem Ansatz, dass ein Betriebssystem auf einem Server, oder allgemeiner einem Computer, betrieben wird. Nachdem dies nun unter Umständen zur Verschwendung von Ressourcen führt, beruht die Idee der Virtualisierung darauf mehrere Betriebssysteme parallel auf einem physikalischen Computer zu betreiben. Die physikalischen Ressourcen, wie CPU, Arbeitsspeicher, Netzwerk, und so weiter werden nun zwischen den virtuellen Maschinen aufgeteilt. [VIRT09]

Die Idee zu Virtualisierung ist an sich alles andere als neu. Schon sehr früh wurden unterschiedliche Methoden entwickelt mehrere logische Komponenten auf ein und derselben Hardware unterzubringen. Speziell in der letzten Zeit kam aber erneut „Bewegung“ in den Virtualisierungsmarkt und Anforderrungen von der Industrie wurden in unterschiedlicher Art und Weise von den Herstellern umgesetzt. [PICH09]

2.2 Ansätze zur Virtualisierung

Picht klassifiziert Virtualisierung in die folgenden Ansätze: [PICH09]

- Programmiersprachen
- Interpretation und Simulation

- Systemvirtualisierung
- Prozessvirtualisierung
- Paravirtualisierung

Programmiersprachen

Um nicht den Eindruck zu wecken Virtualisierung sei nur im Betriebssystemumfeld zu finden sei hier ein Ausblick gegeben wo Virtualisierung auch verwendet wird, nämlich bei plattformunabhängigen Programmiersprachen, wie zum Beispiel Java.

Eines der Ziele bei der Entwicklung von Java war es eine Programmierumgebung zu erstellen, deren Code unabhängig vom Betriebssystem ausgeführt werden kann. Um dies zu erreichen wird mit der so genannten Java Virtual Machine (JVM) gearbeitet.

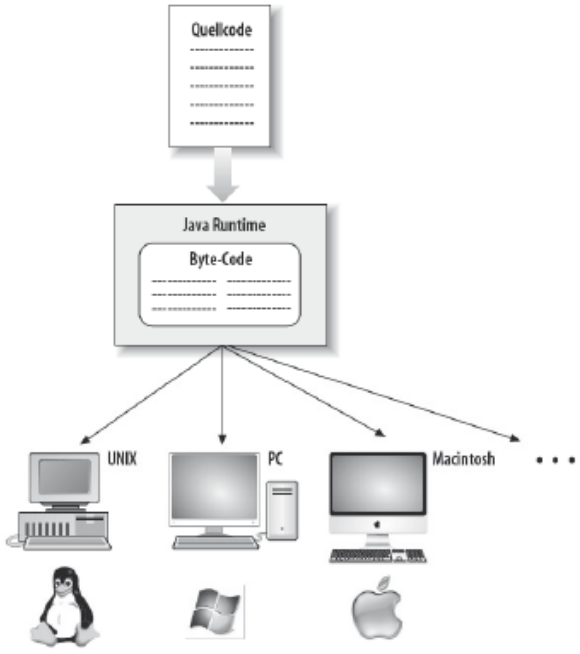

Abb. 2-1: Virtualisierung in Java [PICH09]

Der Trick liegt dabei in der Abstraktion beziehungsweise der Einführung einer weiteren Schicht zwischen Quellcode und Programm, dem Bytecode. Der

Bytecode benötigt dann nur noch die Java Virtual Machine und ist somit auf jedem System das diese Umgebung zur Verfügung stellt ausführbar.

Interpretation und Simulation

Bei diesem Ansatz wird ein komplettes System simuliert beziehungsweise emuliert. Dabei wird sich der generelle Aufbau von Computersystemen und die Trennung von Hard- und Software zu nutze gemacht. Traditionelle Systeme verwenden eine Abstraktionsebene zwischen Hardware und Software (Instruction Set Architecture - ISA). Diese Ebene fängt Instruktionen von Software in Richtung Hardware ab und bereitet diese in Hardware verständlichen Befehle auf. Die Emulation macht sich nun genau diesen Aspekt zu nutze indem eigene ISAs für virtuelle Instanzen verwendet werden. Dadurch ist es auch möglich, dass Betriebssysteme auf Computern betrieben werden die eine komplett andere Architektur in Hardware haben als das Betriebssystem verlangen würde. [PICH09]

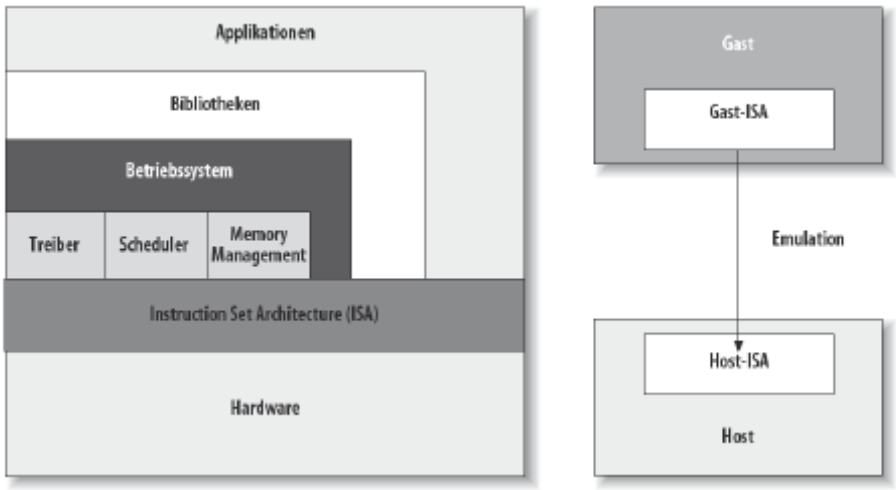

Abb. 2-2: Emulation [PICH09]

Einer der bekanntesten Vertreter dieses Ansatzes ist das unter der GPL frei verfügbare QEMU. QEMU hat prinzipiell zwei Operationsmodus, eine komplette Systememulation oder eine „teilweise“ Benutzerprozessemulation. [QEMU09]

Dadurch das QEMU quelloffen ist, wird es auch öfters in modifizierter Form angefunden. So verwendet zum Beispiel der Hypervisor KVM ein modifiziertes QEMU zur Emulation (näheres siehe in Kapitel 2.3).

Systemvirtualisierung

In diesem Ansatz wird der Begriff Virtual Machine Monitor (VMM) oder Hypervisor eingeführt. Dieses Stück Software ist für die Verwaltung und den Betrieb der virtuellen Maschinen zuständig und verantwortlich.

Abb. 2-3 zeigt exemplarisch einen VMM womit zwei unterschiedliche Betriebssysteme (in diesem Fall Windows und Linux) betrieben werden. Die VMM hat exklusiven Zugriff auf die Hardwareressourcen und verwaltet und reglementiert die Anfragen der virtuellen Maschinen auf die Hardware. Dies bedeutet, wenn ein virtuelles Betriebssystem direkten Zugriff auf Hardware ausführen möchte, wird dieser vom VMM abgefangen und modifiziert weitergeleitet. [SMIT05]

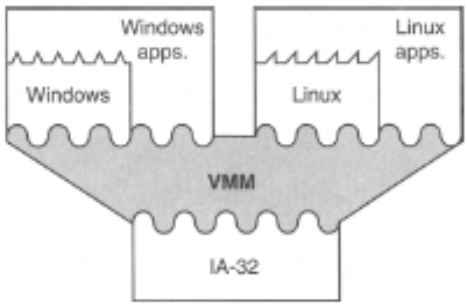

Abb. 2-3: Systemvirtualisierung [PICH09]

Es wird zwischen zwei Typen von VMM beziehungsweise Hypervisors unterschieden. Dem Hypervisor Typ-1 und Typ-2. [PICH09]

Abb. 2-4 zeigt den schematischen Aufbau eines **Hypervisor Typ-1**. Der Hypervisor/VMM läuft direkt auf der Hardware. Dies bedeutet allerdings, dass

das der Hypervisor auch für die Ansteuerung der physikalischen Hardware zuständig ist, sprich es müssen Treiber dafür zur Verfügung gestellt werden. [PICH09]

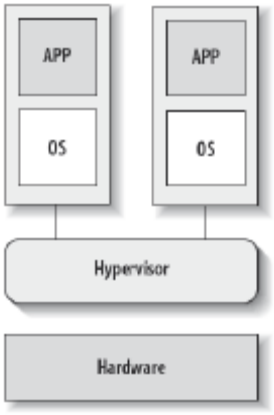

Abb. 2-4: Hypervisor Typ-1 [PICH09]

Ein Hypervisor Typ-2 (Abb. 2-5) setzt nicht direkt auf der Hardware sondern auf einem Betriebssystem auf. Dies bedeutet in weiterer Folge, dass der Hypervisor lediglich für die Verwaltung der Virtuellen Maschinen zuständig ist, da das Ansteuern der Hardware dem darunter liegendem Betriebssystem überlassen werden kann. [PICH09]

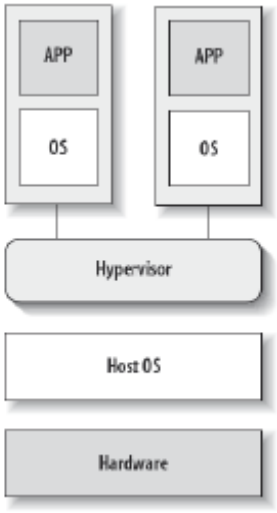

Abb. 2-5: Hypervisor Typ-2 [PICH09]

Prozessvirtualisierung/Betriebssystemvirtualisierung

Wie der Name schon vermuten lässt werden hier nur einzelne Prozesse virtualisiert oder besser gesagt voneinander abgeschirmt. Dies bedeutet, dass nicht das gesamte Betriebssystem mehrfach auf einer physikalischen Maschine ausgeführt wird sondern eben nur diese Prozesse. In weiterer Folge bedeutet dies allerdings, dass nicht unterschiedliche Betriebssysteme zum Einsatz kommen können. [PICH09]

Paravirtualisierung

In diesem Fall wird eine Schnittstelle zwischen der virtuellen Instanz und der physikalischen Hardware geschaffen. Dies bedeutet, dass das Gastsystem wissen muss, dass es virtualisiert ist, was zwangsläufig eine Modifikation an diesem verlangt. Nachdem diese Anpassungen nur auf Kernelebene erfolgen können, sind primär Quelloffene Betriebsysteme beziehungsweise jene Betriebssysteme die auf „offene“ Kernel setzten im Paravirtualisierungsumfeld zu finden.

2.3 Virtualisierungssoftware

Durch die vielen und doch durchaus unterschiedlichen Ansätze zur Virtualisierung hat sich zwangsläufig ein breites Spektrum an Virtualisierungssoftware entwickelt.

Eine sehr ausführliche Liste der Anbieter sowie deren unterschiedlicher Virtualisierungsapplikationen lässt sich unter der Webseite http://www.virtualization.info/radar/ finden.

Um den Rahmen der Arbeit nicht zu sprengen wird hier nur eine kleine Auswahl der verfügbaren Produkte vorgestellt. Die Wahl fiel auf VMWare, XEN und KVM.

VMWare

VMWare bietet im Wesentlichen zwei freie Applikationen zur Virtualisierung an, VMWare Server und VMWare ESXi. [VMWARE09a]

ESXi folgt dem Konzept eines Hypervisor-Typ1, und fungiert quasi als eigenes Hostbetriebssystem. ESXi bietet somit eine schlanke Grundlage um Virtuelle Maschinen zu erstellen, betreiben und zu warten. Dadurch das ESXi als Betriebssystem fungiert ist natürlich hier die Hardware Frage sehr entscheidend. Speziell bei sehr neuer oder etwas älterer Hardware ist mitunter nicht für jede Komponente der richtige und passende Treiber vorhanden. Dies liegt allerdings in der Natur der Sache, denn sollten alle erdenklichen oder zumindest sehr viele Treiber standardmäßig dabei sein, würde das ESXi System nicht mehr den Vorteil des einfachen, kleinen und dynamischen Betriebssystems genießen können. Da Virtualisierung hauptsächlich im Serverbereich ihre Anwendung findet beschränken sich die Treiber sinngemäß hauptsächlich auf bekannte und weit verbreitete Server Hardware. Dies schließt die Anwendung auf anderer Hardware selbstverständlich nicht gleich aus, da die Verwendung von so genannten Standardtreibern eventuell auch zum Ziel führt. Allerdings leidet vermutlich die Stabilität des Systems erheblich darunter. Neben der Stabilität wird vermutlich auch die Performance darunter leiden, da die Hardware Komponenten nicht optimal genutzt werden können. Um den Benutzer nicht „die Katze im Sack“ kaufen zu lassen bietet VMWare eine Kompatibilitätsliste (http://www.vmware.com/resources/guides.html) an. Sollte die Entscheidung ESXi zu verwenden gefallen sein, macht es sicherlich Sinn diese Liste zu konsultieren. [VMWare09b]

Das zweite kostenlos verfügbare Produkt ist der VMWare Server. Dieser basiert im Wesentlichen auf der Methode der vollen Virtualisierung, was bedeutet, dem Gastsystem wird ein kompletter Computer simuliert. Die Treiberthematik am Hostsystem ist hier weitgehend unerheblich da auf

Standardbetriebssysteme aus der Windows oder Linuxwelt zurückgegriffen wird. [VMWare09c]

Der Vorteil den VMWare im Moment genießt ist eine umfassende Treiberkompatibilität innerhalb der Gastbetriebssysteme. Dies ist vermutlich hauptsächlich durch das lange Bestehen am Markt und dem daraus resultierendem Status zu begründen. Durch diese Etablierung bietet beinah jedes Betriebssystem standardmäßig Treiber für die virtuelle VMWare Hardware an.

Xen

XEN wurde an der Universität Cambridge entwickelt und folgt ebenfalls dem Hypervisor Ansatz. Genauer gesagt handelt es sich um den Typ-2, sprich es setzt auf einem laufenden Betriebssystem auf. Die ursprüngliche Paravirtualisierung wurde im Laufe der Zeit auch mit der Option der Vollvirtualisierung ergänzt. Die Paravirtualisierung verspricht zwar mehr Performance innerhalb der virtuellen Instanzen hat allerdings per Definition mit dem Nachteil der Gastsystemmodifikation zu kämpfen. [XEN08]

XEN läuft ausschließlich auf Linux basierten Hostsystemen, und verlangt für die Paravirtualisierung auch Linuxgastsysteme, da sich nur dort der Standard Kernel gegen einen XEN Kernel (Modifizierter Linux Kernel) tauschen lässt. Es gibt zwar vereinzelt Opensource Projekte die sich mit der Paravirtualisierung von Windows beschäftigen, allerdings sind diese noch in einem sehr frühen Entwicklungsstadium und können bis Weilen noch nicht als stabil bezeichnet werden.

XEN bietet die Möglichkeit USB-Geräte an Gäste weiterzureichen. Dies funktioniert sowohl für paravirtualisierte als auch Vollvirtualisierte Gäste, also unabhängig vom Gast Betriebssystem. Weiters besteht die Möglichkeit ganze PCI(e)-Geräte zu überreichen. Dies ist primär aus der Historie im Serverumfeld entstanden, da es da einfach sinnvoll ist zum Beispiel eine

eigene Netzwerkkarte für eine Virtuelle Maschine zu haben. Gründe dafür sind vorwiegend in der Sicherheit zu finden. Das durchreichen von PCI-Hardware funktioniert ohne spezielle Hardware Unterstützung nur bei paravirtualisierten Gästen. Um dies auch bei vollvirtualisierten Gästen zu schaffen muss im Moment eine Addressabkopplung am Mainboard möglich sein. Dies kann im Moment durch Intel vt-d oder AMD IMMU erreicht werden. Diese Technologien sind allerdings meist nur in teurere Serverhardware zu finden und lassen sich nur vereinzelt in „Heimanwender Komponenten“ finden. Allerdings ist dies mit etwas Vorsicht zu genießen, da es sich dabei oftmals nur um eine teilweise Implementierung handelt, sprich es lassen sich nicht alle Komponenten sauber voneinander abtrennen, sondern nur bestimmte Teile davon. [XEN08]

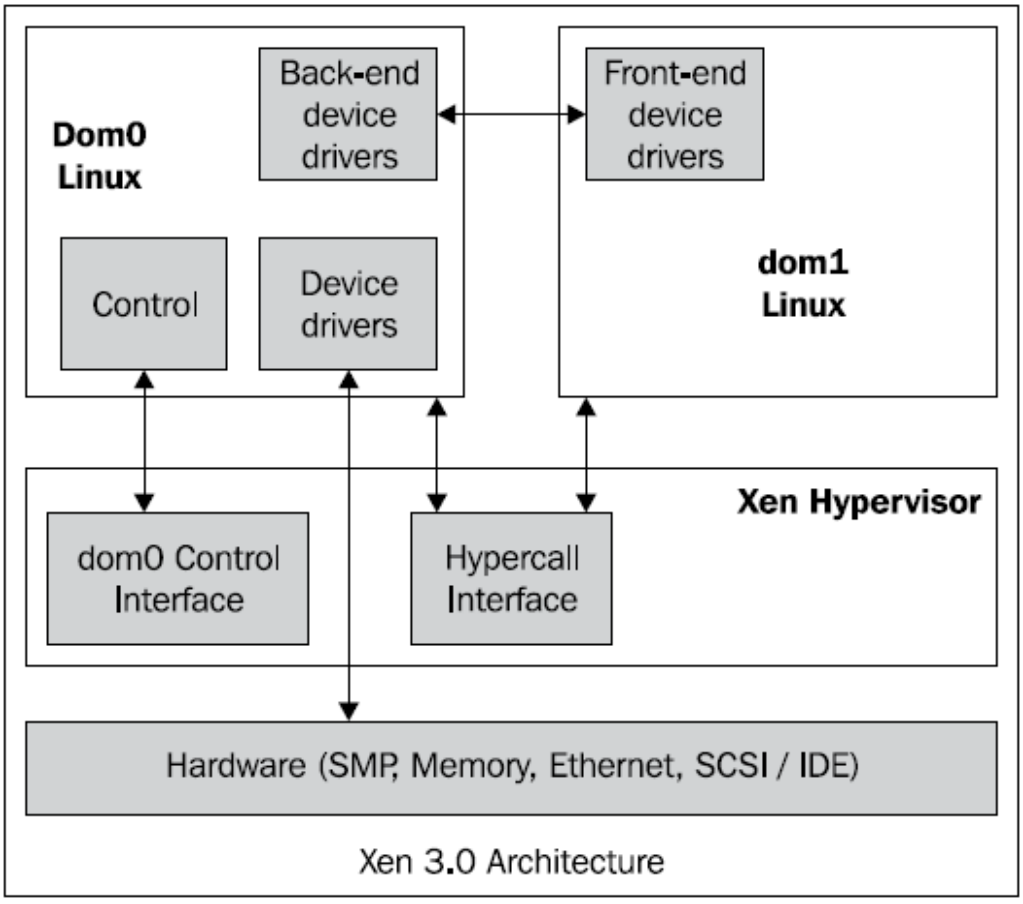

Abb. 2-6: XEN Architektur [CHAG01]

Abb. 2-6 zeigt den prinzipiellen Aufbau eines XEN Systems, und führt die Begriffe Dom0 und Dom1 an. Dom0 ist in der XEN Terminologie als die privilegierte Domain zu bezeichnen, sprich jene Instanz die direkten und „vollen“ Zugriff auf die Hypervisor Ressourcen hat. Dom1 (oder allgemeiner als DomU – „Unprivileged“) wird die unprivilegierte Domain bezeichnet. Diese hat lediglich limitierten Zugriff auf den Hypervisor sowie der darunter liegenden Hardware und Software. [CHAG01]

KVM (Kernel Virtualization Module)

KVM verwandelt den Standardlinuxkernel in einen Hypervisor. Dabei macht es sich den modularen Aufbau des Kernels zu nutze und bindet sich als Kernelmodul ein. Daraus ergeben sich folgende Vorteile [KVM06]:

- KVM kann de facto mit jeder Linuxdistribution verwendet werden
- Der Hypervisor (KVM) benötigt keine eigene Treiber für physikalische Hardware, da diese schon im Linuxkernel abgebildet sind
- Das Hostsystem/Der Hypervisor ist so stabil wie das verwendete Linux, sprich wenn gute Erfahrungen mit einer Distribution gemacht wurden können diese weitergenutzt werden.

KVM verwendet zur Emulation beziehungsweise zur Bereitstellung der virtuellen Ressourcen ein modifiziertes QEMU und kann dadurch jedes von QEMU unterstütze Betriebssystem virtualisieren. [KVM06]

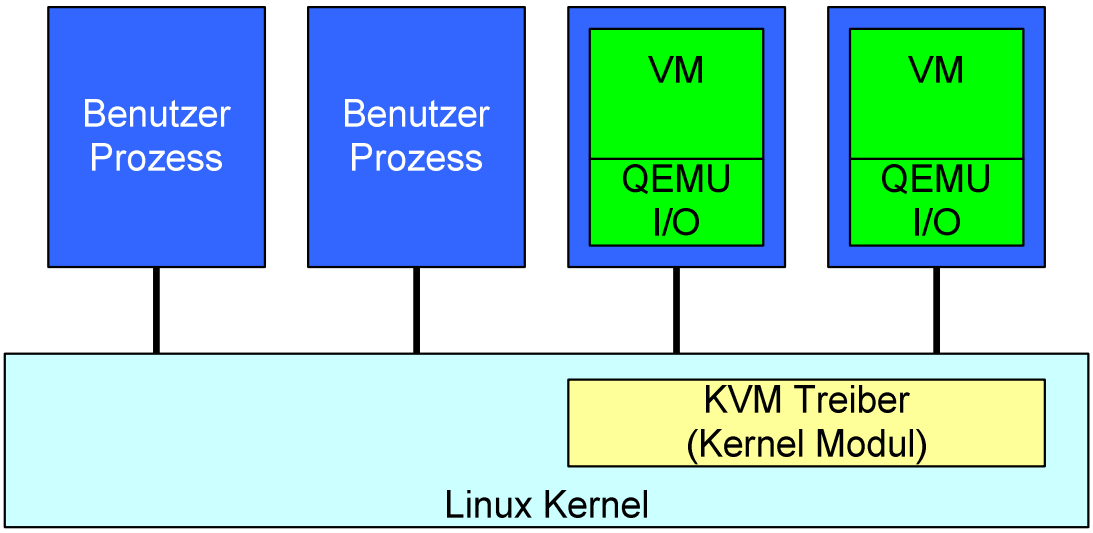

Abb. 2-7: Schematischer Aufbau von KVM [KVM06]

Abb. 2-7 zeigt nun den schematischen Aufbau eines KVM Systems. Wie oben beschrieben ist der KVM Treiber direkt in den Linuxkernel eingebunden und interagiert dabei mit der QEMU-Instanz um die Ein- und Ausgabe (I/O) Operationen durchzuführen. Die Darstellung zeigt auch sehr schön, dass die virtuellen Maschinen als „normaler“ Prozess neben anderen Prozessen laufen.

KVM erstellt eigenes “Gerät” namens `/dev/kvm` welches die folgenden für die Virtualisierung wichtigen Funktionen erfüllt: [KIVI07]

- Neue virtuelle Maschinen erstellen
- Allokieren von Speicher für virtuelle Maschinen
- Schreiben und Lesen von virtuellen CPU Register
- Interrupts für virtuelle CPUs
- Bereitstellung und Betrieb von virtuellen CPUs

Eine besonders „schöne“ Eigenschaft von KVM ist, dass jede virtuelle Maschine einen eigenen Speicherbereich innerhalb des userspaces erhält. Dies für zu „systemfreundlichen“ Prozessen, sprich der Prozess einer virtuellen Maschine kann selbst im Fehlerfall nicht den gesamten Host beeinträchtigen. [KIVI07]

2.4 Nachteile

Abb. 2-8 zeigt das Ergebnis einer von RAAD (www.raad.de) durchgeführten Studie. Dabei wurden im November/Dezember 2008 1000 Unternehmen zum Thema Virtualisierung befragt. Von diesen 1000 Unternehmen haben sich lediglich 430 über Nachteile geäußert, und davon gaben wieder rum 36% an, es gäbe keine Nachteile. [LINK01]

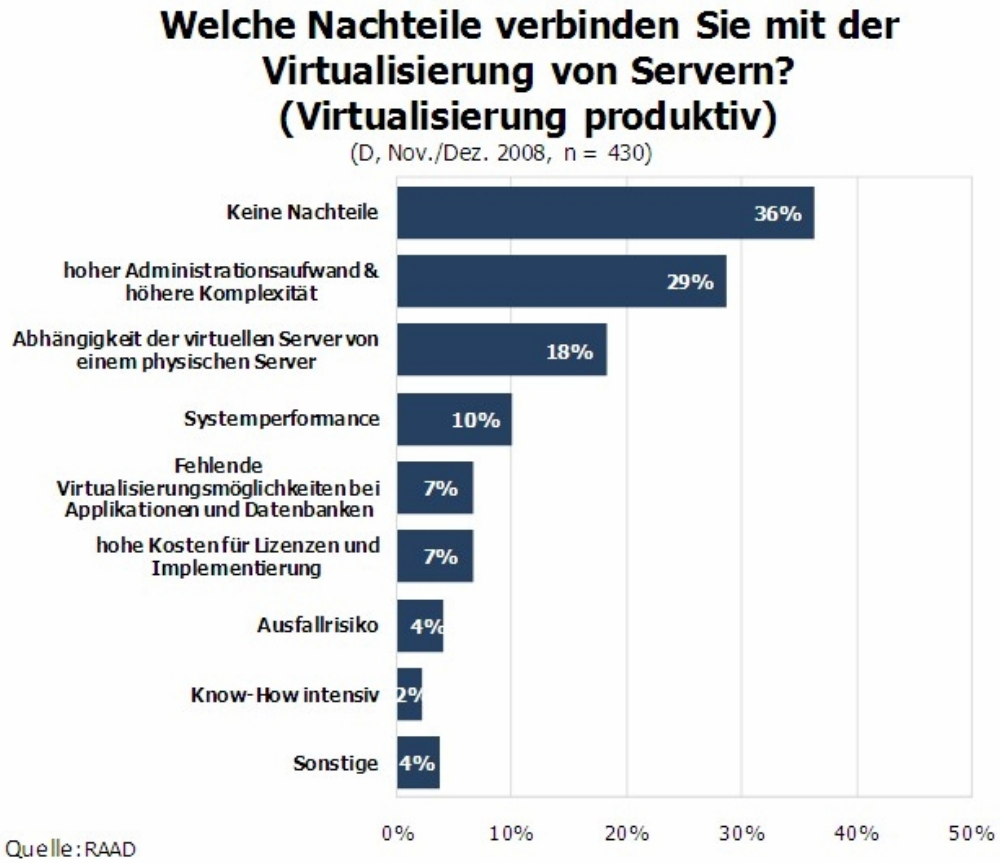

Abb. 2-8: Nachteile der Virtualisierung (Quelle: RAAD)

Komplex

Mit 29% der wohl größte „echte“ Nachteil. Dieses oft genannte Hauptargument der Virtualisierungsgegner und –kritiker lässt sich an sich auch nur schwer entkräften, da rein prinzipiell die Verwendung neuer Technologie im ersten Moment die Komplexität und den Wartungsaufwand erhöht. Auch bringt diese im ersten Anlauf meist Skepsis und Zurückhaltung

ans Tageslicht, vor allem wenn das bestehende System gut funktioniert, sich historisch entwickelt hat und Virtualisierung eine „Überarbeitung“ der IT-Strategien und Konzepte verlangt.

Abhängigkeiten

Laut der Umfrage besteht die Sorge, dass durch die Migration in logische Instanzen die sich einen physikalischen Computer teilen nicht gewünschte Abhängigkeiten entstehen. Auch diese Tatsache lässt sich schwer relativieren, da eben genau das der Sinn und Zweck der Virtualisierung ist.

Das Problem an dieser Stelle ist weniger die Virtualisierung, denn auch hier lassen sich Kriterien wie Hochverfügbarkeit, Redundanz, und so weiter realisieren, sondern die Anwendung von IT-Strategien die bei traditionellen Systemen verwendet wurden.

Ein weiterer Aspekt der ungewollte Abhängigkeiten bringt ist das wahllose virtualisieren, sprich die Migration in eine virtuelle Instanz wird durch das Ende eines Hardware Lebenszyklus ausgelöst. Dies bedeutet natürlich, dass sich unterschiedliche Dienste in logischen Instanzen auf einem physikalischen Gerät befinden, was dann sehr schnell zu ungewollten Abhängigkeiten führt.

Performance

Die dritte „Sorge“ rührt vorwiegend wieder aus der „falschen“ Anwendung der Technologie.

2.5 Vorteile

Im Zuge der Studie die Nachteile erhoben hat wurden auch Fragen bezüglich Vorteile die sich durch Virtualisierung ergeben gestellt. Abb. 2-9 zeigt das Ergebnis. [LINK01]

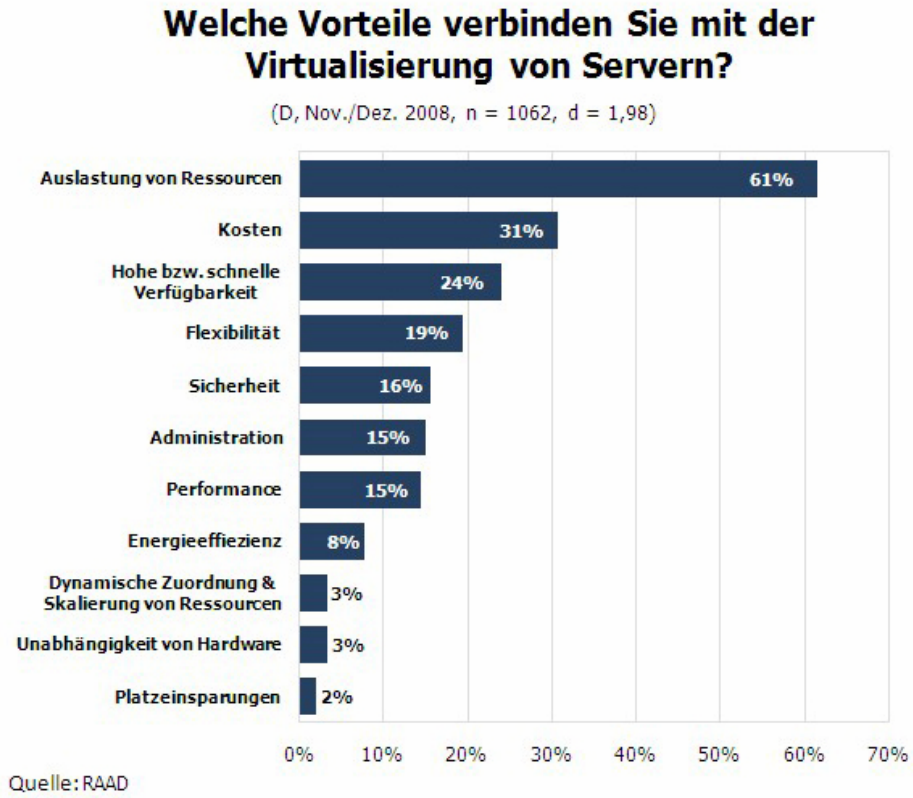

Abb. 2-9: Vorteile der Virtualisierung (Quelle: RAAD)

Kostenersparnis

Der wohl am häufigsten genannte Vorteil im Kontext von Virtualisierung ist die Kostenersparnis. Oftmals wird diese allerdings sehr plakativ in Rechnung gestellt und argumentiert, dass zum Beispiel fünf physikalische Computer gegen einen mit fünf virtuellen Instanzen getauscht werden kann. Dies mag unter bestimmten Vorraussetzungen zwar durchaus richtig sein nur liegt hierbei nicht zwingend die Kostenersparnis.

Hardware die voll ausgelastet wird produziert de facto auch keine unnötigen Kosten, wobei dies selbstverständlich voraussetzt, dass der angebotene

Dienst auch genutzt wird beziehungsweise einen gewissen Mehrwert bringt welcher den Aufwand rechtfertigt. Nachdem aber eine solche Maschine ohnehin schon ausgelastet ist werden keine Ressourcen verschwendet, und die Migration in eine virtuelle Umgebung ist fragwürdig und sollte genauestens analysiert werden. Eine der ersten Fragen sollte hier in Richtung der Hardware gestellt werden. Läuft der Dienst auf einer etwas älteren Hardware, die leistungsmäßig von neuer Hardware bei weitem übertroffen wird? Wenn dies mit „ja“ zu beantworten ist, steht der Virtualisierung an sich nichts mehr im Wege. Wird der Dienst allerdings von einer neuen Hardware betrieben, und es werden einfach sehr viele Ressourcen benötigt, dann würde eine Virtualisierung wenig bis keinen Sinn ergeben.

Also in erster Linie sollten Dienste konsolidiert werden, welche wenig Hardware Anforderungen stellen und aktuelle Hardware bei weitem nicht auslasten können und demnach operative Kosten ungenutzt verbrauchen.

Kosten lassen sich allerdings nicht nur bei der Anschaffung sparen. Vor allem da speziell die Anschaffungskosten auf mehrere Jahre abgeschrieben werden können, und meist durch das Budget von diversen Projekten gedeckt sind. Wesentlich interessanter sind jene Kosten die dann im Laufe des Betriebes anfallen, die operativen Kosten. Zu diesen zählen Stromkosten, Wartungskosten, Wartungsverträge, Lizenzkosten, Klimaanlage, und so weiter.

Umweltschutz

Das Thema Umweltschutz bestimmt schon seit langem die Medien im alltäglichen Leben und hat mittlerweile auch die IT erreicht. Hierbei geht es vor allem um das Senken von Stromverbrauch. Aber auch die Dimensionierung von Klimaanlagen und deren Kühlmittel und in weiterer Folge auch der ordnungsgemäßen Entsorgung von Elektromüll.

Durch die Zusammenfassung von unterschiedlichen IT-Diensten lassen sich natürlich auch hier im großen Stil Einsparungen erzielen.

Redundanz und Ausfallsicherheit

Durch die Definition von so genannten „Service Level Agreements“ (SLA), welche oftmals durch den Einzug von IT-Service Management ausreichend sowie weit reichend definiert werden, werden zwangsläufig Begriffe wie Redundanz und Ausfallsicherheit schlagend.

In der Vergangenheit wurde Ausfallssicherheit häufig durch Hardwareredundanz gelöst. Durch diverse Treiber, wie Kosteneinsparungen, knappe Budgets, Umweltschutz, etc., lässt sich Redundanz auch sehr schön durch Virtualisierung erzielen.

Datensicherheit und Wiederherstellung

Die Sicherheit der Daten sowie die rasche Wiederherstellung im Notfall ist heutzutage unbedingt nötig. Das Sichern und Widerherstellen von reinen Konfigurationsdaten ist mit modernen Backuptechnologien nicht wirklich ein Problem, nur setzt dies im Fall der Wiederherstellung ein funktionierendes Betriebssystem voraus und genau das ist der Knackpunkt beziehungsweise das Problem dieser Methode, da das Installieren das Systems viel Zeit und gegebenenfalls auch Probleme in Anspruch nimmt/mit sich führt.

Zwar würde sich auch das ganze Betriebssystem sichern lassen nur setzt dies im Wiederherstellungsfall exakt die gleiche Hardware voraus. Dies mag zwar vielleicht funktionieren wenn das System aufgrund eines Softwareproblems zum Erliegen kam, kann aber bei defekter Hardware zu einem unbrauchbaren Backup führen. Der Grund dafür ist, dass selbst gleiche Computermodelle oft unterschiedliche Versionen einzelner Komponenten verwenden und dadurch ist die Treiberkompatibilität nicht mehr gegeben.

In einer komplett virtuellen Umgebung ist dieses Problem de facto umgangen. Solange die gleiche Virtualisierungssoftware verwendet wird, werden virtuelle Maschinen dieselbe virtuellen Hardwarekomponenten emuliert. Auch muss ein Backup einer virtuellen Maschine nicht auf traditionellem Wege erfolgen. Die virtuelle Maschine wird letztlich auf dem Hostsystem als einzelne Datei abgebildet, welche eigentlich nur an eine sichere Stelle kopiert werden muss. Um die Datenkonsistenz zu gewährleisten empfiehlt es sich allerdings die virtuelle Maschine davor zu stoppen.

Skalierbarkeit

Oftmals ist es leider sehr schwer abzuschätzen wie sich ein Dienst „entwickelt“. Zwar können mit Hilfe von diversen Statistiken und eventuell vorhandenen historischen Daten Prognosen errechnet werden, aber eine sichere Vorhersage lässt sich nur schwer treffen. Es ist auch nicht sehr sinnvoll in Ressourcen zu investieren die dann doch nicht verwendet werden. Weiters ist eine Hardware Erweiterung mit unter auch mit erheblichen Konfigurationsaufwand verbunden. Sehr anschaulich können hier Hardwaretreiber genannt werden. Abhängig von der verbauten Hardware müssen diese passend ausgewählt, installiert und konfiguriert werden. Nun ist aber nicht gesagt, dass immer alle Modelle zum Beispiel die gleichen Netzwerkkarten verwenden. Allein dies würde eine 1:1 Migration von Systemen schon wesentlich schwieriger gestalten. Die Verwendung einer virtuellen Umgebung kann dahingehend helfen, dass virtuelle Hardware abstrahiert von der physikalischen ist. Eine virtuelle Maschine kann beliebig oft verschoben und kopiert werden und es wird für das Gastbetriebssystem keinen Unterschied machen. Das Hostsystem kann in aller Ruhe installiert, konfiguriert und mit allen nötigen Treibern bestückt werden.

Weiters können zum Beispiel mehrere unterschiedliche Dienste anfangs eine physikalische Maschine teilen. Sollten die Kapazitäten eng werden, können

die ressourcenintensivsten wie oben beschrieben de facto einfach auf eine neue Maschine verschoben werden. Dadurch wächst die nötige Infrastruktur praktisch dynamisch mit den eigenen Anforderungen mit.

3. TECHNISCHE EVALUIERUNG

3.1 Ansätze zu einem Konzept

Dieses Kapitel soll erste Ideen zur Realisierung der in Kapitel **Fehler! Verweisquelle konnte nicht gefunden werden.** definierten Anforderungen aufwerfen und konzeptionell diskutieren.

Um die Kostenanforderungen zu erfüllen sollte weitgehende freie Software verwendet werden, wodurch auf jeden Fall die Lizenzkosten eingespart werden können. Sollten kostenpflichtige Betriebssysteme verlangt oder gewünscht sein, so sollten sich diese lediglich in den virtuellen Instanzen befinden.

Wie schon eingangs erwähnt sind sämtliche Virtualisierungsapplikationen im Bereich der Servervirtualisierung schon sehr ausgereift. Servervirtualisierung ist im dem Konzept als die Emulation eines Servers zu verstehen, sprich sämtliche Hardwarekomponenten werden virtuell zur Verfügung gestellt und es ist nicht zwingend nötig direkt auf physikalische Ressourcen zuzugreifen.

Desktopvirtualisierung im Sinne der Bereitstellung dedizierter Ein- und Ausgabegeräte steckt momentan noch ziemlich in den Kinderschuhen. Zwar lässt sich ein gewisser Trend in diese Richtung erkennen, nur spezialisiert sich dies primär auf die Verwendung von lokalen Speichermedien in virtuellen Maschinen und anderer Peripheriegeräte, wie zum Beispiel Drucker oder Scanner. Die Bedienung der virtuellen Maschinen wird meist durch so genannte Konsolefenster realisiert.

Eine der wesentlichen Kernfragen die sich hier stellt ist, wie nun ein Benutzer tatsächlich mit der virtuellen Maschine arbeiten kann beziehungsweise wie

die virtuellen Instanzen mittels Ein- und Ausgabegeräten bedient werden können. Nachdem diese Funktionalität irgendwo zur Verfügung gestellt werden muss stellt sich auch hier die Frage ob das zentral oder dezentral gesehen soll. Dies ist allerdings auf Schnittstellenebene zu betrachten, sprich ob mehrere USB und VGA-Anschlüsse von einer physikalischen Maschine ausgehen oder ob es für jeden Arbeitsplatz eine eigene Komponente gibt.

Die folgenden Kapitel werden beide Optionen hinsichtlich ihrer technischen Machbarkeit evaluieren, analysieren und diskutieren.

3.2 Zentrale Steuerung der Ein- und Ausgabe Geräte

3.2.1 Virtualisierungssoftware

Um Desktopvirtualisierung mit dedizierten Ein- und Ausgabegeräten zu verwirklichen, ist es nötig physikalische Geräte direkt an ein Gastbetriebssystem durchzureichen. Dies ist momentan beschränkt mit VMware Server möglich, denn laut VMWare Server Spezifikation können manche USB-Geräte, wie USB-Sticks, USB-Festplatten, an virtuelle Maschinen weitergeleitet werden. Dies ist allerdings sowohl vom Gerätetyp als auch innerhalb der Typen vom Hersteller beziehungsweise Modell abhängig. Das Weitergeben von USB-Mäuse ist aber generell, also unabhängig von Hersteller und Modell, nicht möglich ist und daher kann der VMware Server nicht verwendet werden. [VMWARE09d]

Die speziell für Desktopvirtualisierung zugeschnittene Version VMWare Workstation bietet zwar wesentlich mehr Varianten und Möglichkeiten als die Server Version, ist allerdings nicht mehr kostenlos zu benutzen, und scheidet daher auch aus.

XEN unterstützt prinzipiell das Weiterleiten von PCI und USB-Geräten. Das Durchreichen von PCI und PCIe-Geräten ist für paravirtualisierte Gäste schon seit längerem möglich. Die Weitergabe von einzelnen USB-Geräten ist nur bei vollvirtualisierten Gästen möglich. Um hier sowohl Eingabegeräte (meist USB) als auch Ausgabegeräte (zumeist PCI oder PCIe) zu erreichen kann entweder eine gesamte PCI-USB-Karte weitergereicht werden, oder USB-Grafikkarten verwendet werden. [XEN08]

Die neuste Version von Xen (3.3.1) unterstützt bei geeigneter Hardware auch das Weiterleiten von PCI und PCIe-Geräten an vollvirtualisierte Gäste. Dadurch ist es erstmals auch möglich zum Beispiel einem virtuellen Windows XP dediziert eine PCI/PCIe-Karte zur Verfügung zu stellen. Dies stellt allerdings bestimmte Anforderungen an die verwendete Hardware. Diese werden in Kapitel 3.2.3 behandelt.

Somit wird für diesen Test XEN als Virtualisierungssoftware ausgewählt.

3.2.2 Gast Betriebssystem Evaluierung

Nachdem XEN zur Virtualisierung verwendet wird und dieses nur unter Linux beziehungsweise UNIX verfügbar ist können sämtliche Microsoft Produkte nicht mehr verwendet werden.

Beinah jede gängige Linuxdistribution bietet in irgendeiner Art und Weise Virtualisierung an. Oftmals ist diese sogar schon bei der Installation auszuwählen was den Vorteil hat, dass das ganze System (inklusiver Virtualisierung) schon getestet wurde. Im Testaufbau wurde CentOS 5.3 verwendet.

CentOS benutzt laut eigenen Angaben die „Source RPMs“ (SRPM) eines „prominenten Nordamerikanischen Enterprise Linux“ Herstellers. Dabei sehen die CentOS Entwickler ihre Hauptaufgabe in der Entfernung der Distributionsspezifischen Merkmale aus den Source RPMs. [CentOS09]

Auf der CentOS Webpräsenz ist nicht zu finden welche Distribution nun hinter der Bezeichnung „prominenter Nordamerikanischer Enterprise Linux“ Hersteller steckt, doch bei genauerem betrachten und arbeiten mit CentOS kann recht rasch erkannt werden, dass es sich hierbei um RedHat Linux handelt. Die RedHat Distribution ist nur durch das Abschließen eines Supportvertrages legal zu betreiben.

Der Vorteil von CentOS ist nun, dass de facto kostenlos auf das breite Erfahrungsspektrum von RedHat zurückgegriffen werden kann. Auch können für RedHat kompilierte Packete installiert und verwendet werden.

Wie auch RedHat bietet somit CentOS seit längerem Virtualisierung mittels XEN an. Der Support reicht von eigens entwickelten Applikationen zur einfacheren Wartung und Erstellung von virtuellen Maschinen bis hin zu einem eigens modifizierten, adaptierten und speziell auf das CentOS Betriebssystem kompilierten XEN-Kernel. Weiters stehen noch diverse Skripte zur Verfügung die die Einrichtung von Netzwerkverbindungen zwischen den virtuellen Maschinen sowie dem Internet ermöglichen und vereinfachen.

3.2.3 Hardware

Auch wenn dieses Vorhaben sehr softwarelastig klingt, ist speziell in diesem Punkt die Auswahl der Hardware entscheidend.

Mainboard und CPU

Im Wesentlichen gibt es zwei große CPU (Central Processing Unit) Hersteller, Intel und AMD. Zwangsläufig sind auch das oft die Hersteller der Chipsätze zur Ansteuerung der gesamten Mainboardkomponenten.

Um nun physikalische Geräte an eine vollvirtualisierte Maschine zu übergeben ist eine spezielle Technologie nötig um die Adressräume abzutrennen und somit I/O Virtualisierung zu ermöglichen. Diese Technologie wird allgemein IOMMU genannt.

Intels IOMMU Lösung nennt sich VT-d, und findet sich in unterschiedlichen Implementierungen in verschiedenen Chipsätzen. Leider bietet nicht jede Version von VT-d den vollen Funktionsumfang und so sind oft Chips zu finden welche zwar prinzipiell VT-d beherrschen, dieses Stück Hardware allerdings durch die Firmware beziehungsweise das BIOS nicht verwenden. Weiters gibt es auch Mainboards die zwar VT-d fähige Chips benutzen, allerdings wird VT-d nicht bei allen nötigen Komponenten (zum Beispiel I/O Controller) unterstützt, was wiederum eine eingeschränkte beziehungsweise gar keine Funktion bietet. [ABRA06]

Prinzipiell sind die Intel Chipsets Q35 und Q45 voll VT-d fähig. Die nächste entscheidende Komponente ist der I/O Controller. Speziell dieser muss IOMMU nun am I/O Bus anwenden können. Intel bietet hier ICH-9 (oftmals in Kombination mit Chipset Q35) und ICH-10 (in Kombination mit Chipset Q45) an. [INTEL09a] und [INTEL09b]

Weiters muss auch der Prozessor selbst Virtualisierung unterstützen, wobei allerdings die meisten Intel Core 2 Duo CPUs dies schon standardmäßig beinhalten. [INTEL09c]

Die Wahl für den Prototypen fiel hier auf das Intel Mainboard DQ45CB [INTEL08a], basierend auf dem Q45 Chipsatz, mit einem ICH-10 I/O Controller. Unterstützt wird dies von einer Intel Core2 Duo CPU mit 3GHz pro Kern. Das Intel Mainboard besitzt lediglich 2 PCIe und 1 PCIe16 Port für Grafikkarten. Ist es nötig mehrere Monitore zu betreiben müsste hier ein anderes Board verbaut werden. Zusätzlich werden 4 GB Arbeitsspeicher verbaut um ausreichend Ressourcen für die virtuellen Maschinen zur Verfügung zu stellen.

Grafikkarte

Moderne Mainboards stellen oftmals eine on-board Grafikkarte zur Verfügung. Aus Kostengründen wird hier oft nur der Chip verbaut, und gewisse Ressourcen werden mit dem restlichen System geteilt. Ein Paradebeispiel wäre hier der Speicher. Hier wird vom physikalischen Speicher ein bestimmter Bereich dediziert der Grafikkarte zugeordnet. Durch diese Speicherzuweisung ist eine Weiterleitung einer internen Grafikkarte an eine virtuelle Maschine nicht möglich, da diese im Gast keinen Arbeitsspeicher mehr hätte.

Somit sind PCI/PCIe-Grafikkarten mit eigenem Speicher eine Option. Im Wesentlichen gibt es zwei große Grafikchip Hersteller, ATI und NVIDIA. Durch die standardmäßige Bestückung mit on-board Grafikkarten ist speziell der Grafikkartenmarkt vorwiegend auf eine bestimmte Zielgruppe ausgerichtet, nämlich jene die spezielle Anwendungen, wie Spiele, betreiben. Nachdem nun die meisten Computerspiele ein Microsoft-Betriebsssytem voraussetzen sind die Grafikkartentreiber auch meist für Microsoft Windows programmiert und kompiliert. Zwar sind oft auch Linuxtreiber verfügbar, nur bieten diese nicht den vollen Funktionsumfang und sind veraltert. Zwar wird von der Opensource Gemeinde sehr fleißig an der Herstellung von Opensource Treibern gearbeitet, nur entstehen diese meist durch „reverse-engineering“. Dies bedeutet wiederum nur limitierte Funktionen und nicht die volle Leistung.

Da für diese Aufgabe allerdings weder 3D-Beschleunigende Techniken noch besondere Grafikmerkmale gefordert sind ist dies nicht das gravierende Problem. Vielmehr kommt bei der Verwendung von paravirtualisierten Gästen die Herausforderung des XEN Kernes zu tragen. Dieser basiert zum einen meist auf einer etwas älteren Kernelversion, welche neue Hardware oft nicht unterstützen kann. Weiters sind meist spezielle Funktionen zur Virtualisierung eingearbeitet und andere „Desktop“ Funktionen wurden

entfernt. Speziell die Linuxtreiber der Hersteller sind nur auf den „normalen“ Linuxkernel kompiliert, was ein Portieren nach XEN erforderlich macht.

Besonders seien hier noch externe USB-Grafikkarten anzumerken. Diese werden einfach an den USB-Port gesteckt und ermöglichen den Betrieb eines Monitors. Zwar sind diese oft nicht besonders leistungsstark und für grafikintensive Anwendungen wie Spiele gänzlich ungeeignet, allerdings für Bürotätigkeiten mehr als ausreichend. Im Wesentlichen basieren alle gängigen USB-Grafikkarten auf einem Chipsatz von Displaylink. Durch die relativ junge Technologie und der offensichtlich relativ geringen Nachfrage gibt es dafür leider keine Linux Treiber beziehungsweise auch keinen Support im aktuellen Linuxkernel. Somit ist dies im Moment keine Option, könnte aber in der Zukunft eine wesentliche Rolle in der Desktopvirtualisierung einnehmen.

3.2.4 Konfiguration

Das Weiterleiten von PCI(e)-Geräten an einen XEN Gast ist prinzipiell möglich. Die Herausforderung ist es allerdings zu verhindern, dass das Gastsystem die Hardware nicht für sich in Anspruch nimmt.

Die Idee ist nun mit der Linuxkernelfunktion pciback die Hardware schon an das System zu binden bevor der Treiber dies tun kann. Ist die Hardware an pciback gebunden kann diese im laufenden Betrieb an das Hostsystem übergeben und auch wieder weggenommen werden und gleiches gilt auch für die virtuellen Gäste. Der XEN Kernel sieht diese Funktion prinzipiell vor, nur ist sie in der aktuellen Version von CentOS nicht direkt im Kernel beinhaltet sondern lässt sich als Modul einbinden. Somit lässt sich beim Bootvorgang ohne Modifikationen das Gerät nicht vor dem Host verstecken. Um die Hardware dennoch an pciback zu geben bevor der eigentliche Treiber sie bindet kann zwischen den folgenden zwei Ansätzen gewählt werden.

Der CentOS XEN-Kernel wird **neu** mit **aktivierter PCIBACK** Option **kompiliert**. Dies hat allerdings zur Konsequenz, dass das Gesamtsystem davon betroffen sein kann beziehungsweise ist. Im Idealfall hat es keine Auswirkungen, nur lässt sich dies leider schwer voraussagen, da der eigens zusammengestellte Kernel natürlich nicht von der gleichen Anzahl an Benutzern verwendet wird.

Um den original Kernel zu behalten kann das PCIBACK Kernelmodul auch geladen werden bevor die PCI-Adressen in das Betriebssystem eingebunden werden. Die PCI-Geräte werden dadurch auch wiederum an die pciback Instanz gebunden und können von dort aus beliebig verwendet werden.
Dies verlangt folgende Modifikationen:

```
bash# lspci | grep –i vga

bash# echo “option pciback hide (01:00.0)”>>/etc/modprobe.conf

bash# mkinitrd -f --preload=pciback /boot/initrd-$(uname -r)
$(uname -r)

bash# reboot
```

Um nach dem Neustart festzustellen, dass ein bestimmtes PCI-Gerät tatsächlich an PCIBACK gebunden wurde, sollte das logfile des Starvorganges durchsucht werden. Dies lässt sich am einfachsten mit folgendem Befehl bewerkstelligen

```
bash# dmesg | grep -i pciback
```

Die Ausgabe sollte wie folgt aussehen:

```
pciback 0000:01:00.0: seizing device
```

3.2.5 Fazit

Im Zuge der Tests wurden unterschiedliche Versionen von XEN verwendet. Darunter die Version 3.1, 3.3 sowie die noch nicht als stabil freigegeben Version 3.4. Auch wurde unterschiedliche Hardware, speziell unterschiedliche Chipsätze sowie jeweils verschiedene BIOS Versionen verwendet. Trotz all dieser Bemühungen war es nicht möglich eine Grafikkarte funktionsfähig durchzureichen. Zwar sollte laut Herstellerspezifikationen jedes Teil für sich in der Lage sein PCI-Geräte an virtuelle Maschinen zu übergeben, aber in Kombination lässt sich dies nicht realisieren.

Die Tests wurden sowohl mit paravirtualisierten als auch mit vollvirtualisierten Gästen durchgeführt. Folgende Anmerkungen sind zu den Virtualisierungsmethoden relevant:

Vollvirtualisierte Gäste

- Obwohl die Hardware eine IOMMU (Intel vt-d) zur Verfügung stellt, und dies auch vom Betriebssystem erkannt und laut Protokolldatei verwendet wird, lässt sich keine einwandfreie Funktion erzielen. Zwar ist das Durchgereichte PCI-Gerät in der virtuellen Maschine sichtbar, doch fehlen bestimmte Speicherbereiche und dadurch kann es nicht verwendet werden.

- Durch das Durchreichen der PCI-Hardware wurden massive Stabilitätsprobleme am Hostcomputer verzeichnet. Es lässt sich nicht sagen ob diese Probleme aus der nicht korrekten Weitergabe oder aus der Weitergabe an sich rühren.

Paravirtualisierte Gäste

- Der für paravirtualisierte Gäste nötige XEN Kernel ist nicht oder zumindest nicht vollständig kompatibel mit den aktuellen Grafikkartentreibern der Hersteller.
- Obwohl die Abkopplung beziehungsweise das Management der Adressbereiche vom Hostsystem beziehungsweise dessen XEN Kernel abgearbeitet werden sollte, wird dies nicht richtig gemacht. Dadurch werden wieder nur „Teile“ des Gerätes durchgereicht was wiederum dazu führt, dass es nicht verwendet werden kann.

Ein Problem das sich unabhängig von der Art der Virtualisierung ergibt ist die Interrupt Verwaltung. Jedes Gerät eines Computers wird einem Interrupt zugewiesen, wobei es unter normalen Bedingungen kein Problem ist wenn sich mehrere Geräte einen Interrupt teilen. Doch im Falle von Virtualisierung wird genau das zu einem Problem, da es nicht möglich ist Geräte durchzureichen die sich einen Interrupt mit einem nicht durchgereichten Gerät teilen. Die Interrupt Verteilung, sprich die Zuordnung Interrupt/Gerät, wird völlig automatisiert vom Betriebssystem übernommen und es ist auch nicht möglich in diese einzugreifen. Der einzige anwendbare Trick der unter Umständen zum Erfolg führt ist das Ausschalten nicht benötigter on-board Geräte (wie zum Beispiel USB-Controller/Ports) beziehungsweise den PCI-Port zu tauschen. Dieser Ansatz mag vielleicht vereinzelt zum Erfolg führen lässt sich aber nicht pauschal anwenden, und schon gar nicht mit der Bedingung mehre Grafikkarten im Hostsystem zu haben und diese an unterschiedliche virtuelle Maschinen weiterzureichen.

Abgesehen davon, dass keine wesentlichen Erfolge beim Durchreichen erzielt werden konnten, lässt sich eine entscheidende Instabilität im System feststellen. Auch ist eine sehr starke Abhängigkeit von Hardware und Software und vor allem hinsichtlich der verwendeten Version gegeben, was zwangsläufig zu Problemen bei Aktualisierungen führen wird. Auch machte das System in den Versuchen und Tests keinen homogenen, stabilen und

vorhersagbaren Eindruck. So kam es durchaus vor, dass Konfigurationen auf unterschiedliche Art und Weise interpretiert wurden und plötzlich unerwartete Fehler auftraten.

All diese Aspekte machen diese Anwendung der Virtualisierung für Schulen im Moment unbrauchbar, da es wesentlich komplexer und fehleranfälliger ist als ein traditionelles Schulnetzwerk. Es gilt nun abzuwarten ob die technischen Entwicklungen der nächsten Zeit die Komponenten kompatibler als auch stabiler werden lassen. Sollte dies der Fall sein, würde ein erneuter Versuch mit Sicherheit Sinn machen.

3.3 Dezentrale Steuerung der Ein- und Ausgabe Geräte

Der wesentliche Unterschied zu der in Kapitel 3.2 evaluierten Lösung ist die Verlagerung der Ein- Ausgabe Intelligenz in ein eigenes Gerät. Dadurch wird das Problem, welches sich beim Durchreichen von physikalischen PCI(e)-Grafikkarten ergibt, de facto umgangen und dennoch kann die virtuelle Maschine dem Benutzer zur Verfügung gestellt werden. Der einzige Nachteil der sich daraus ergibt ist, dass ein weiteres Gerät verwendet werden muss. Dieses Gerät muss in der Lage sein mit einer virtuellen Instanz mittels standardisierten Protokolls zu kommunizieren, beziehungsweise die Benutzerschnittstelle darzustellen.

3.3.1 Ansteuerung der Ein- und Ausgabegeräte

Im einfachsten Fall ist dies eine „normaler“ Personal Computer, mit eigenständigem Betriebssystem. Sollte dies hier ausschließlich verwendet werden, würde das prinzipielle Ziel der einfachen Wartung nur unter bestimmten Umständen erreicht werden können. Auch würde der Sinn der Virtualisierung in eine starke Abhängigkeit in Richtung der Verwendung rutschen. Daher wird ein Gerät benötigt das möglichst zentral gesteuert und konfiguriert wird, aber dennoch die nötigsten Funktionen zur Verfügung stellt.

Je einfacher dieses aufgebaut ist, umso niedriger werden die Wartungskosten ausfallen.

Ein solches Gerät könnte zum Beispiel ein ThinClient sein, dessen Kosten durch die fallenden Hardwarepreise in den letzten Jahren nach und nach in Vergessenheit geraten. Im Wesentlichen ist ein ThinClient ein Computer der nur die wirklich nötigen Hardware Komponenten zur Verfügung stellt. Oft werden dabei auf sämtliche bewegliche Teile, wie zum Beispiel Festplatten, CD/DVD Laufwerke, etc. Verzichtet. Dies bedeutet allerdings dass ein ThinClient nicht aus „eigener Kraft“ ein Betriebssystem booten kann. Dazu ist eigener Server nötig, der dann die nötigen Daten in den Arbeitsspeicher des ThinClients kopiert und somit das Starten eines Systems ermöglicht. [KNER08]

Das prinzipielle Konzept eines ThinClient-Systems ist denkbar einfach. Ein zentraler Server versorgt mehrere ThinClients, und bietet je nach Benutzer oder Hardware (eindeutige Identifizierung anhand der MAC Adresse ist möglich) unterschiedliche Oberflächen und Applikationen. Ein- und Ausgabeoperationen werden über den ThinClient abgehandelt, und die eigentlichen „Rechenoperationen“ werden am Server erledigt. Durch die Auslagerung der Performanceintensiven Applikationen müssen ThinClients nicht über sonderliche Hardwarekapazitäten verfügen, und sind daher meist kostengünstiger und natürlich auch Strom sparender als herkömmliches PC.

Auch der Wartungsaufwand beschränkt sich auf die Initiale Konfiguration, welche ja zwangsläufig auch bei „normalen“ PCs nötig wäre. Der dafür nötige Aufwand hängt hier viel von der ThinClient Server Applikation beziehungsweise der dazugehörigen Management Schnittstelle ab. Ist dies intuitiv und Schemenbasierend lässt sich nach einer initialen Konfiguration sehr leicht und rasch ein weiterer Client hinzufügen.

ThinClient Boot Prozess

Wie der ThinClient nun zu den Dateien kommt ist prinzipiell „offen“. Hierbei gibt es sowohl völlig proprietaire als auch de facto Standardansätze. Beinah jedes auf der x86 oder x86_64bit Architektur basierendes Mainboard und in weiterer Folge BIOS der letzten Jahre unterstützt das Booten über ein Netzwerk. Hierbei wird auf das PXE Protokoll zurückgegriffen.

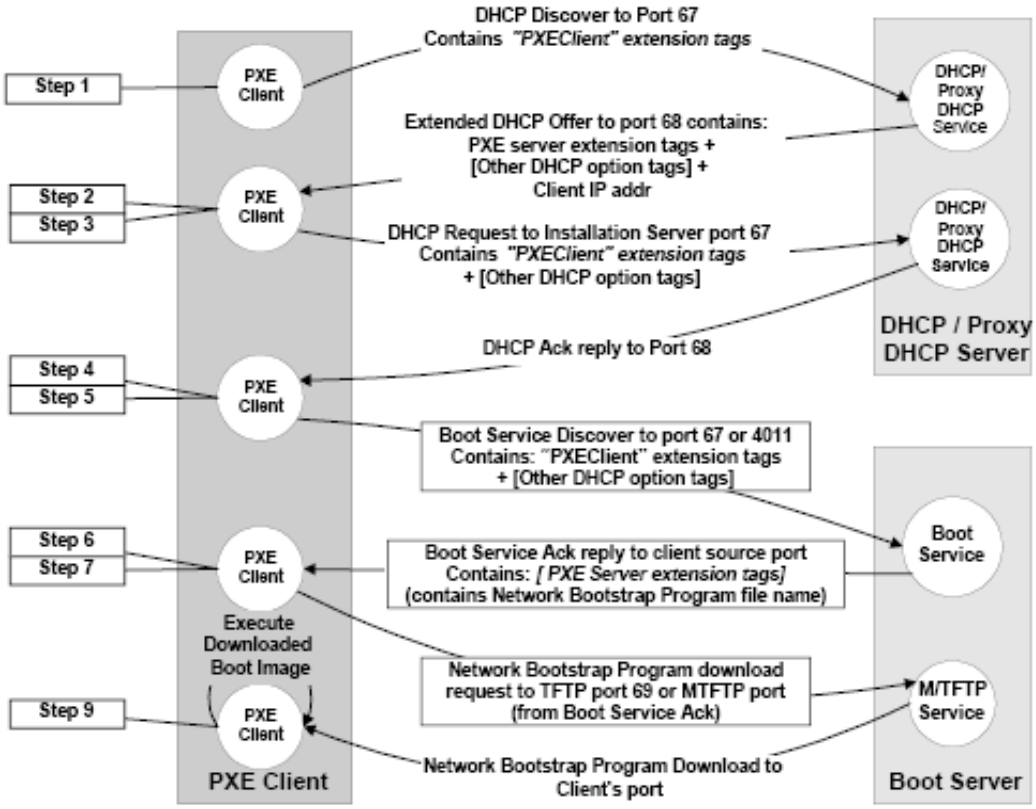

Abb. 3-1: PXE Startprozess [INTEL99]

Um einen ThinClients mittels PXE Boot starten zu können muss im BIOS die Option aktiviert sein. Abb. 3-1 zeigt welchen Ablauf das Protokoll beim Starten folgt. Kurz zusammen gefasst folgt dies in den Schritten 1-4 dem Standard DHCP Protokoll (IETF RFC 2131 [IETF97]) und erst die Schritte 4-9 sorgen für das Kopieren der nötigen Startdateien.

3.3.2 Virtualisierungssoftware

Die Auswahl der Virtualisierungssoftware ist in diesem Ansatz beziehungsweise in dieser Methode an sich unerheblich. Nachdem keine Geräte durchgereicht werden müssen können im Wesentlichen sämtliche Servervirtualisierungslösungen zur Anwendung kommen. In den Tests hat sich die relativ junge Virtualisierungsapplikation KVM bewährt. Die Vorteile von KVM in Vergleich zu XEN sind im Wesentlichen folgende:

- Virtuelle Maschinen laufen als „normaler“ Prozess im „Userspace“
- KVM läuft als Kernelmodul und es kann daher ein Standard Linuxkernel verwendet werden
- Durch den Standardlinuxkernel gibt es keine zusätzlichen Probleme mit Hardwaretreibern
- KVM verwendet zur Emulation ebenfalls QEMU, wenn auch in etwas modifizierter Form
- KVM ist „schlanker“ als XEN
- KVM benötigt keine eigene „Management Domain“ wie XEN

3.3.3 Fazit

Schon die ersten Versuche und Tests verliefen durchaus erfolgreich. Durch die Abkopplung und die zusätzliche Intelligenz zur Ansteuerung der Ein- und Ausgabegeräte lässt sich ein sehr komplexer Teil aus der Virtualisierungsschicht abkoppeln. Diese Zerlegung vereinfacht die Realisierung, Wartung und den Betrieb erheblich und führt letztlich zu einem transparenten und homogenen Gesamtsystem.

Auch werden keine Probleme bei Softwareaktualisierungen beziehungsweise Hardwareänderungen erwartet. Im Wesentlichen gibt es hier keinerlei zusätzliche Abhängigkeit zwischen Software und Hardware hinsichtlich der Virtualisierung. Dies schließt selbstverständlich die für den Betrieb der Hardware nötigen Treiber aus, denn diese bilden zwangsläufig eine

Abhängigkeit zwischen Hard- und Software. Um bei der Kommunikation zwischen den Schnittstellen keine unnötige Abhängigkeit einzuführen sollte hier soweit wie möglich auf etablierte Standards zurückgegriffen werden.

Es ist anzunehmen, dass die Wartung eines auf dieser evaluierten Methode basierendem System einfacher und weniger zeitintensiv ist. Diese Vermutung rührt primär aus der Tatsache, dass die Anzahl der komplexen Systeme, der Virtualisierungssysteme sowie der Serverkomponenten, überschaubar, zentral und homogen ist.

4. TECHNISCHES KONZEPT

Nachdem die technische Machbarkeitsanalyse in Kapitel 3.2 negativ für den Einsatz einer zentraleren Steuerung von Ein- und Ausgabegeräten ausfiel wird dieses Konzept nun den Ansatz der dezentralen Steuerung folgen.

Somit wird basierend auf den gestellten Anforderungen in Kapitel 1.3 und der technischen Evaluierung in Kapitel 3.3 ein generisches Konzept erstellt welches dann an die gegebene Situation angepasst werden kann und soll.

Das Konzept ist abstrahiert, sprich es stellt nur technische Anforderungen, bietet aber keine technischen Lösungen. Diese werden im Anschluss in Kapitel 5 exemplarisch gezeigt.

4.1 Komponenten

Um das System rund um die Virtualisierung sowie die Virtualisierung selbst möglichst flexibel und skalierbar zu gestalten wird „das Ganze“ in mehrere Einzelteile, so genannte Komponenten, zerlegt. Um möglichst zukunftsorientiert zu arbeiten sollte sämtliche Kommunikation zwischen den Komponenten weitgehend auf definierten und etablierten Standards basieren. Dies soll vor allem den Freiraum zur eigenen Softwareauswahl beziehungsweise in weiterer Folge die Funktionalität nach Upgrades einer einzelnen Komponente gewährleisten.

4.1.1 Dispatch

Die wesentliche Aufgabe des Dispatchers ist es den geeigneten VM-Server für eine virtuelle Maschine zu finden und diese dort zu starten. Um nun tatsächlich den geeigneten Server zu finden, können mehrere unterschiedliche Parameter ins Kalkül gezogen werden. Im einfachsten Fall

könnte zum Beispiel die Anzahl der schon laufenden virtuellen Maschinen den Entscheidungsprozess terminieren.

Die Anzahl der gestarteten virtuellen Maschinen gibt zwar einen groben Überblick, allerdings sagt dieser Wert nur bedingt etwas über die verfügbaren Ressourcen des VM-Servers aus, da die produzierte Last jeder virtuellen Maschine durchaus unterschiedlich sein kann. So wird ein Webserver mit nur einem Testbenutzer wesentlich weniger Ressourcen des VM-Servers benötigen wie ein rechenintensives selbst geschriebenes Programm. Eine Möglichkeit dies in den Dispatch Prozess mit einzubinden wäre die Überwachung der Server mittels SNMP. Dadurch könnten zum Beispiel Parameter wie CPU und Arbeitsspeicher Auslastung berücksichtigt werden.

Die Kommunikation von VM-Server zurück zum Dispatcher ist an sich nicht vorgesehen, da der Datenfluss von der Management Schnittstelle zum Dispatcher und von der zum VM-Server läuft. Jegliche Überprüfungen müssen somit immer vom Dispatcher initiiert werden. Somit ist ausgeschlossen, dass ein defekter VM-Server irreführende Meldungen schickt und den Dispatcher zu „falschen“ Entscheidungen „verleitet“. Durch die unidirektionale Kommunikation ergibt sich implizit die Anforderung an den Dispatch Algorithmus, bei jeder Entscheidung die konfigurierten VM-Server nach deren Status zu fragen. Nachdem der Auftrag zu Virtualisierung nur an einen antwortenden VM-Server gehen kann ist auch sichergestellt, dass nur ein funktionierender VM-Server zur Virtualisierung bewegt wird.

4.1.2 Virtualisierung

Diese Komponente besteht aus einer beliebigen Anzahl an physikalischen Computern, den so genannten VM-Servern. Die Menge wird im Wesentlichen durch die geplanten virtuellen Maschinen beziehungsweise durch die Hardware Kapazitäten vorgegeben.

Wie der Name schon vermuten lässt stellt die Virtualisierungskomponente die für die Virtualisierung nötigen Ressourcen zur Verfügung, wobei hier die „Aufträge“ vom Dispatcher erteilt und im Wesentlichen auch verteilt werden.

Um die Abfrage systemspezifischer beziehungsweise systemkritischer Parameter zu ermöglichen könnte ein SNMP Dienst betrieben werden. Diese Daten können dann im Dispatch Prozess als Entscheidungskriterium herangezogen werden.

Für die virtuellen Maschinen wird eine so genannte Network-Bridge verwendet. Diese erlaubt es mehrere MAC und IP-Adressen an eine physikalische Netzwerkschnittstelle zu binden. Werden hier höhere Netzwerkbandbreiten benötigt, können mehrere physikalische Schnittstellen zusammengefasst werden. Weiters könnte eine dedizierte Managementschnittstelle in Betracht gezogen werden.

4.1.3 LDAP

Sämtliche Benutzerdaten werden in einem LDAP-Verzeichnis-Server gespeichert. Alle anderen Komponenten können wenn nötig auf diesen zentralen Punkt zugreifen um dort die relevanten Informationen zu beziehen.

Die folgenden Parameter müssen zu jedem Benutzer verfügbar sein:

- Benutzername
- Passwort (in verschlüsselter)
- Image Dateien
- Virtuelle MAC und IP Adressen
- ThinClient Parameter

Es muss erlaubt sein, dass andere Komponenten auch schreibend auf den LDAP-Verzeichnisbaum zugreifen dürfen. Dies ist speziell dann sinnvoll

wenn einem Benutzer ein weiteres virtuelles Betriebssystem gegeben werden soll.

4.1.4 PXE Server

Der PXE Server ist im Wesentlichen der erste Ansprechpartner von jedem ThinClient beziehungsweise PXE Client. Die PXE Anfrage wird entsprechend dem PXE Standard behandelt und die nötigen Dateien werden zum Client kopiert.

Zusätzlich könnte der PXE Server einen DHCP Teil bekommen

- Konfiguration des OpenThinClient Betriebssystems
- Beantworten von PXE Requests
- Stellt OpenThinClient OS zur Verfügung

4.1.5 Management

Die Management Komponente teilt sich auf in mehrere Subkomponenten, die jeweils eine Funktion zu erfüllen haben. Diese sind:

Das **LDAP-Benutzer-Management**, stellt eine Schnittstelle zur Administration der Benutzerdaten zur Verfügung. Hier können sowohl neue Benutzer angelegt werden, bestehende Veränderung sowie gelöscht werden.

Eine intuitive Schnittstelle zum **Starten von virtuellen Maschinen**, wo basierend auf den Benutzerdaten alle assoziierten Instanzen angezeigt und wenn gewünscht gestartet werden können.

Das Betriebessystem beziehungsweise das Erkennen, Erlauben und Starten der ThinClients wird mittels der **PXE-Server-Management** Schnittstelle konfiguriert.

Werden neue virtuelle Betriebssysteme benötigt können diese durch eine **Duplizier Schnittstelle** generiert werden. Alle nötigen Daten werden dann in den LDAP-Baum geschrieben.

Die Zugriffsberechtigungen, sprich welcher Benutzer (Schüler, Lehrer oder IT-Administrator) welche Schnittstelle verwenden darf, sollte an die jeweilige Situation angepasst werden. Generell könnte die folgende Tabelle als Entscheidungsbasis dienen:

Schnittstelle	Zugriffsberechtigt
LDAP-Benutzermanagement	IT-Administrator
Starten von VM	Schüler, Lehrer und IT-Administrator
Management des PXE-Server	IT-Administrator
Duplizieren von VMs	IT-Administrator und Lehrer

4.1.6 ThinClient sowie Ein- und Ausgabegeräte

Diese Komponente umfasst im Prinzip alles was der Benutzer „sieht“ und „angreifen“ kann. Dies wäre neben dem Monitor, Tastatur und Maus auch der ThinClient selbst. Allerdings schließt das Konzept nicht aus auch „normale“ Personal Computer als Client System zu verwenden.

4.2 Interaktion der Komponenten

Die Interaktion der Komponenten lässt sich am besten und einfachsten durch ein Beispiel demonstrieren. Hierbei kann ganz plakativ auf den Ablauf der sich ergibt wenn ein Benutzer mit seiner virtuellen Maschine arbeiten möchte zurückgegriffen werden.

Der komplette Ablauf lässt sich im Wesentlichen in vier Sequenzen einteilen:

1. ThinClient starten
2. Auswahl des virtuellen Betriebssystems das gestartet werden soll
3. Starten des virtuellen Betriebssystems
4. Verbindungsaufbau zum virtuellen Betriebssystem

1. ThinClient starten

Der erste Teil (ThinClient starten) bezieht sich erwartungsgemäß lediglich auf den ThinClient Teil und ist somit nicht nötig wenn andere Zugriffsmethoden gewählt werden. Dies könnte zum Beispiel die Einwahl in das Schulnetzwerk von zuhause oder die Verwendung eines Notebooks sein.

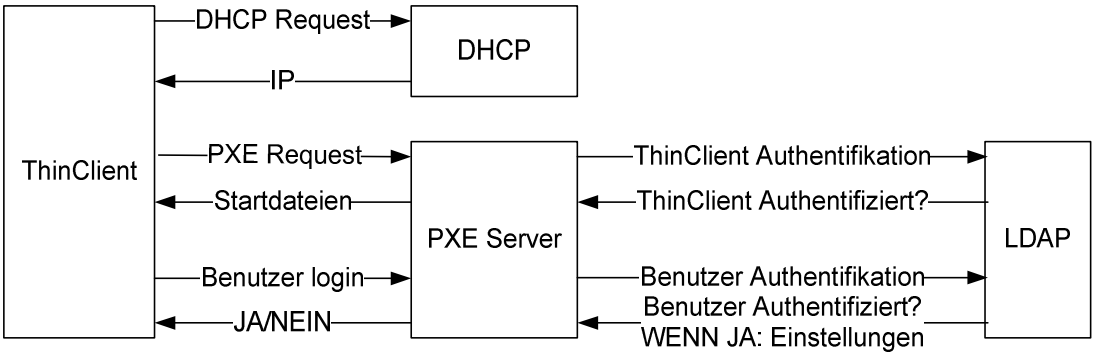

Abb. 4-1: ThinClient starten

Beim Starten des ThinClients folgt dieser dem PXE Standard, sendet DHCP Requests um eine IP Adresse vom DHCP Server zu beziehen sowie die PXE Request für die zum Starten nötigen Dateien. Die PXE Requests werden vom PXE Server empfangen und abhängig von der Konfiguration eventuell noch hinsichtlich der MAC Adresse überprüft (nachfragen im LDAP-Verzeichnisbaum). Der PXE-Server „erkennt“ den Client und kopiert die für den Start des OpenThinClient Betriebssystems nötigen Dateien in Richtung Client und dieser startet das ThinClient-System.

Der Benutzer gibt seine Daten ein, und diese werden mit den im LDAP-Verzeichnis gespeicherten verglichen und bei Erfolg kann die Anmeldung vollzogen werden. Die Informationen zu den Einstellungen beziehungsweise die für den Benutzer verfügbaren Programme werden ebenfalls aus dem LDAP-Verzeichnis ausgelesen.

2. Auswahl des virtuellen Betriebssystems das gestartet werden soll

Abb. 4-2 zeigt schematisch den Ablauf dieser Sequenz und beinhaltet abstrahiert die ausgetauschten Informationen.

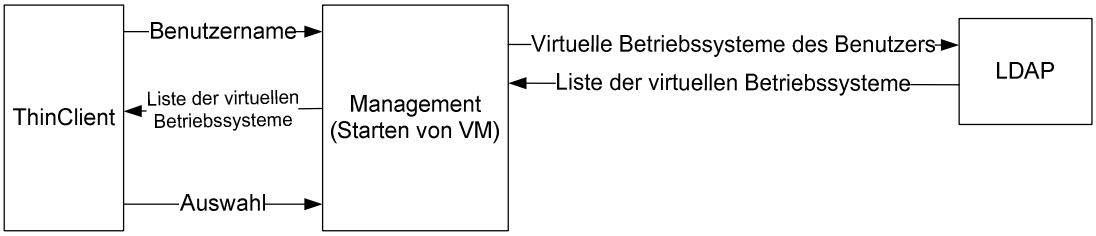

Abb. 4-2: zu startendes virtuelles Betriebssystem auswählen

Pauschal gesagt ist der nächste Schritt das Starten der virtuellen Maschine. Zu diesem Zweck wird das Managementportal geöffnet, der Benutzer authentifiziert sich und wählt die zu startenden Virtuellen Maschinen aus. Auch in dieser Sequenz spielt das LDAP-Verzeichnis eine tragende Rolle, da die Images wiederum von dort ausgelesen werden.

In welcher Art und Weise diese Informationen darin abgebildet sind beziehungsweise sein können obliegt dann der jeweiligen Implementierung, vorausgesetzt die Informationen sind vollständig (Image und MAC Adresse). Es ist allerdings empfehlenswert das Parameterfeld eines LDAP-Eintrages nicht mit übermäßig vielen Zeichen zu belegen. Der Grund dafür ist, dass viele LDAP-Clients bei der Suche dann nicht alle Informationen anzeigen können.

3. Starten des virtuellen Betriebssystems

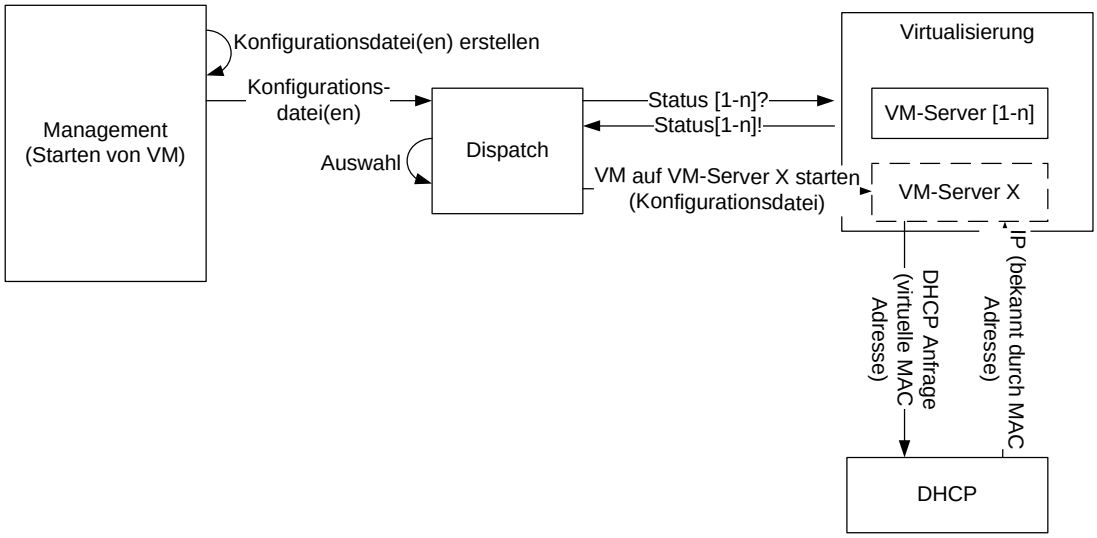

Abb. 4-3: Virtuelles Betriebssystem starten

Wie Abb. 4-3 zeigt werden die Daten aus dem im Schritt 2 (Auswahl des virtuellen Betriebssystems) ausgewählten Betriebssystem zu der für die Virtualisierung nötigen Konfigurationsdatei umgebaut.

Die Konfigurationsdatei wird anschließend der Dispatch Komponente übergeben wo der geeignete VM-Server gesucht wird. Dies folgt dem in Kapitel 4.1.1 beschriebenen Prozess. Ist der VM-Server identifiziert wird die Konfigurationsdatei kopiert und die virtuelle Maschine gestartet.

Die virtuelle Maschine startet und die virtuelle Netzwerkkarte schließt sich der Network-Bridge des VM-Servers an. Als MAC Adresse wird dabei die im LDAP-Verzeichnis definierte verwendet. Die IP Adresskonfiguration innerhalb der virtuellen Instanz muss dynamisch sein, um im Endeffekt vom DHCP Server eine „bekannte“ IP Adresse zu beziehen. Dieser muss mit statischen Reservierungen arbeiten können.

4. Verbindungsaufbau zum virtuellen Betriebssystem

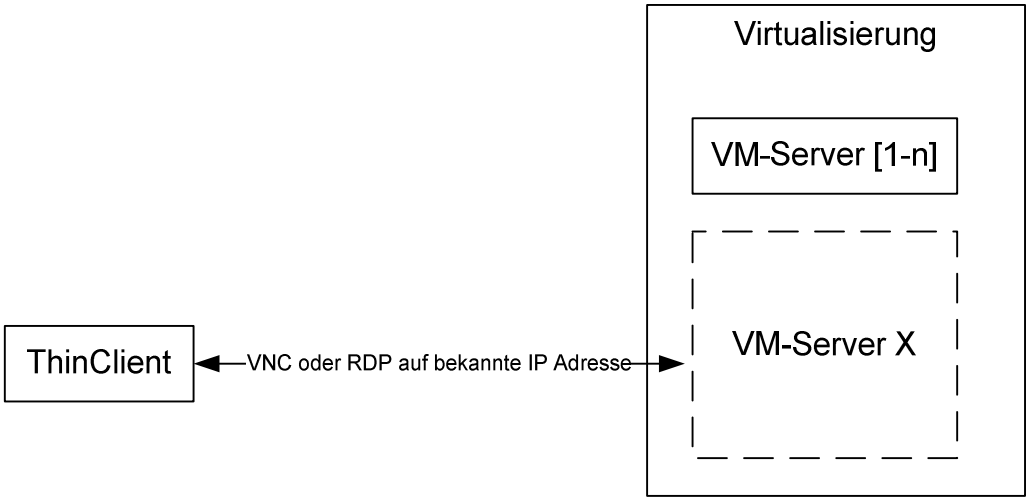

Abb. 4-4: Verbindung zu virtuellen Betriebssystem

Der letzte Schritt wird, wo der Benutzer sich nun mit seiner virtuellen Maschine verbinden kann wird in Abb. 4-4 dargestellt. Das Protokoll ist hier Abhängig vom Betriebssystem der virtuellen Maschine. So wäre dies das Remote Desktop Protokoll (RDP) für Microsoft Windows Gäste oder VNC für Linux Gäste.

Gesamter Ablauf

Der Vollständigkeit zuliebe zeigt Abb. 4-5 (nächste Seite) nun den kompletten Ablauf beginnend beim Starten des ThinClients bis zu dem Moment wo sich der Benutzer mit seiner eigenen virtuellen Maschine verbindet.

Gesamter Ablauf

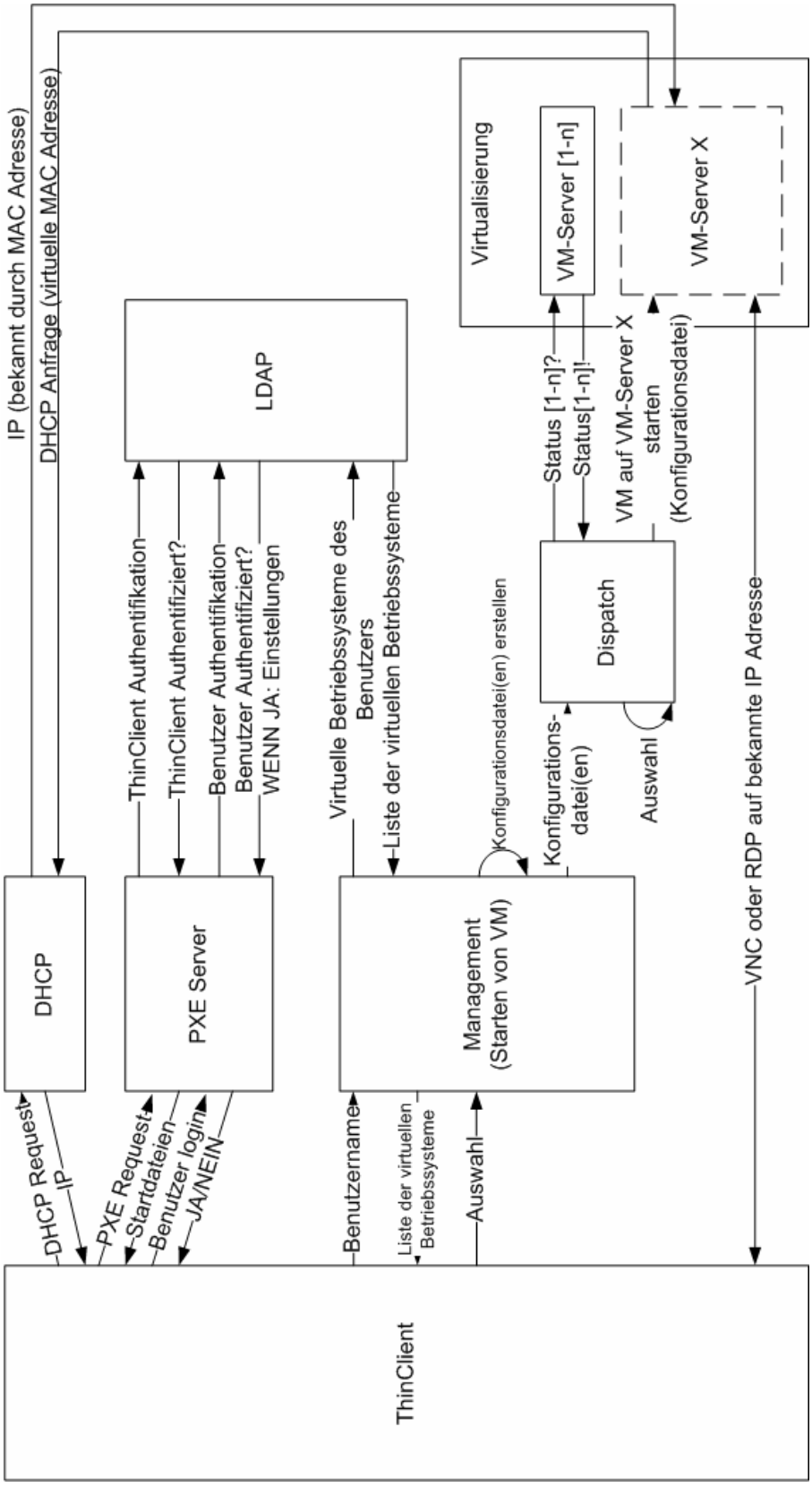

Abb. 4-5: Interaktion der Komponenten

Neues virtuelles Betriebssystem für einen Benutzer anlegen/erstellen

Eine weitere Funktion ist das Duplizieren von vorgefertigten Betriebssystemen. Dies ermöglicht ein sehr schnelles und einfaches Herstellen von Betriebssystemen sowie der Referenz zum jeweiligen LDAP-Benutzer. Diese Funktion kann allerdings auch verwendet werden wenn ein Benutzer sein virtuelles Betriebssystem zerstört hat oder für seine Arbeit ein zusätzliches benötigt.

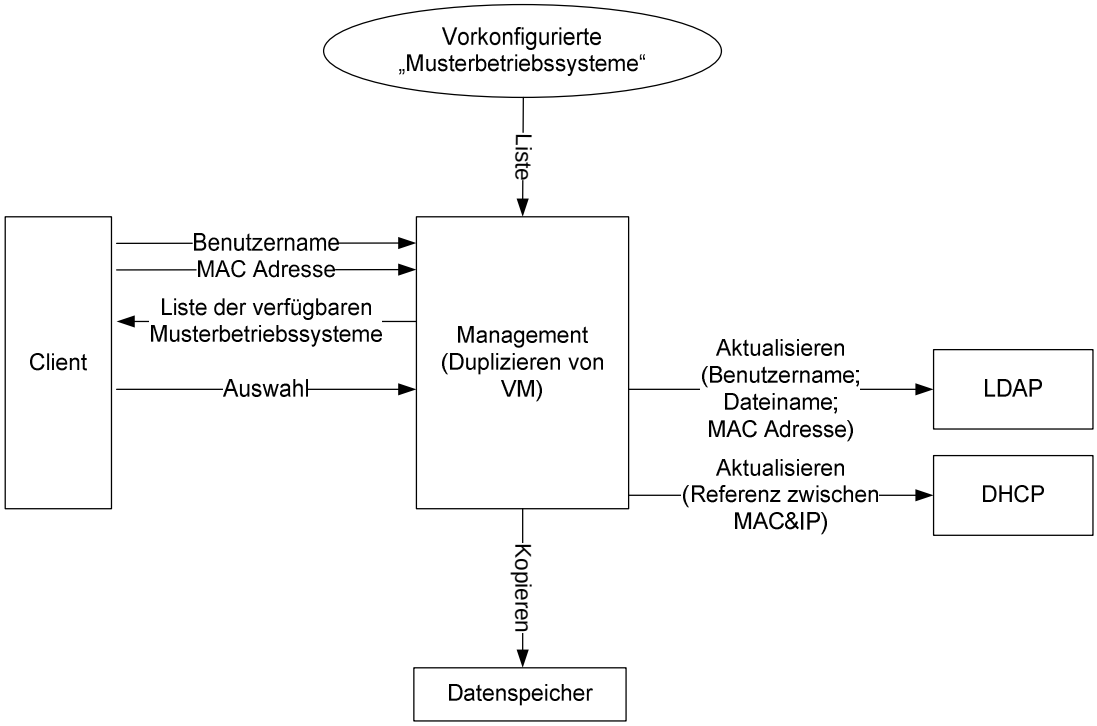

Abb. 4-6: Virtuelles Betriebssystem hinzufügen

Abbildung Abb. 4-6 zeigt nun schematisch den dafür nötigen Ablauf. Der Client (kann auch ein ThinClient sein) verbindet sich mit der dafür vorgesehen Managementschnittstelle. Als Parameter sind der Name des Benutzers der das Betriebssystem bekommen soll sowie eine eindeutige MAC Adresse zu übergeben. Die Managementinstanz liest nun alle verfügbaren Musterbetriebssysteme aus und bietet diese dem Administrator an, der das gewünschte auswählen kann. Ist diese Selektion getroffen sind alle nötigen Informationen vorhanden und das Skript aktualisiert den LDAP-

Verzeichnisbaum und initiiert das duplizieren der Musterbetriebssystemdatei am Datenspeicher.

4.3 Skalierung und Adaptierung

Ein wesentlicher Vorteil der sich durch die modulare Zusammensetzung aus unterschiedlichen Komponenten ergibt ist die Option der Skalierbarkeit. Dies bedeutet, dass in kleineren Anwendungsgebieten kostengünstiger und effizienter dimensioniert werden kann. Aber auch in Testumgebungen kann durch die Zusammenfassung mehrere Komponenten das System ausführlich und intensiv getestet werden, ohne hier Unmengen an Hardware anschaffen zu müssen.

Prinzipiell können alle Komponenten auf einer physikalischen Maschine zusammengefasst werden, allerdings wäre es vermutlich ratsam auch bei „kleineren“ Umgebungen auf mindestens zwei Computer zurückzugreifen. Die Idee dahinter ist relativ einfach, die Virtualisierungskomponente ist die mit den höchsten Kapazitäts- und Performance Ansprüchen. Auch ist diese Ebene jene die zwangsläufig als erstes nach einer Erweiterung verlangen wird, da alle anderen lediglich die Informationen für diese Ebene aggregieren und zur Verfügung stellen. Die eigentliche Rechenleistung wird aber auf den VM-Servern zur Virtualisierung benötigt.

Das gesamte Konzept ist darauf ausgelegt mit den Anforderungen zu wachsen. So können die Komponenten nicht nur zusammengefasst werden sondern auch auf mehrere physikalische Maschinen aufgeteilt werden. Auch lassen sich viele der Komponenten über mehrere Server verteilen, sollte die Leistung eines einzelnen nicht mehr ausreichen.

Diese Adaptierung verlangt allerdings die eine oder andere Modifikation am Netzwerk. So lässt sich die Gesamtarchitektur netzwerktechnisch einfacher implementieren bei zentraler Konsolidierung als bei dezentraler.

Wie auch schon bei der Beschreibung der Komponenten erwähnt ist die Virtualisierungskomponente jene die vermutlich zuerst in die „Breite“ skaliert werden muss. Im Prinzip kann sogar angenommen werden, dass auch bei einer kleinen Installation schon mehr als ein physikalischer VM-Server zur Anwendung kommt. Durch die Einführung der Dispatcher Komponente lässt sich dies allerdings ohne großen Aufwand leicht umsetzen.

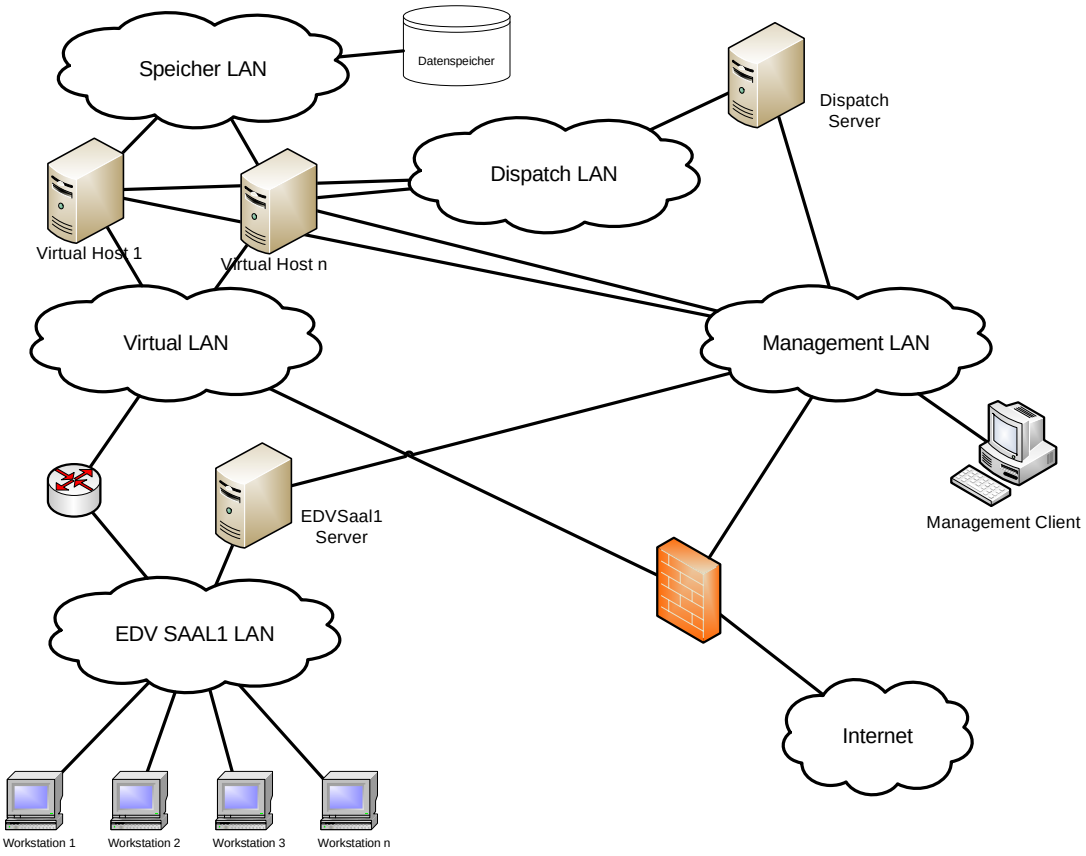

Abb. 4-7: Beispiel Architektur bei maximalem Ausbau

Abb. 4-7 zeigt nun exemplarisch den maximalen Ausbau des Systems. Jede Komponente wird auf einem eigenen physikalischen Server betrieben und die Kommunikation dazwischen bringt zwangsläufig eine gewisse Komplexität in den Netzwerkplan.

4.4 Finanzielle Aspekte

Die finanzielle Frage beziehungsweise ob nun eine virtuelle Umgebung kostengünstiger ist als eine herkömmliche traditionelle lässt sich pauschal nicht beantworten. Zwar können Kosten auf der Arbeitsplatzseite eingespart werden, nur ist stattdessen wesentlich mehr in die Serverseite zu investieren.

Um diesen Aspekt etwas zu verdeutlichen wurden exemplarisch diverse Komponenten zur Ausstattung der IT-Infrastruktur ausgewählt. Die Preise sind jeweils kongruent mit der unverbindlichen Preisempfehlung des Herstellers. Der Fairness halber wurden nur Komponenten von HP gewählt, da nur so wirklich verglichen werden kann, HP alle nötigen Teile anbietet und die Preise im Internet ausschreibt.

Vermutlich würde bei einer größeren Bestellung ein Rabatt möglich sein, vor allem da schulische Bestellungen ohnehin meist über einen zentralen Einkauf abgehandelt werden. Dieser ist allerdings in der Kalkulation nicht berücksichtigt, da die Bestimmung der Höhe nicht möglich ist.

4.4.1 Anschaffungskosten

Im folgenden Beispiel werden die für die Anschaffung einer virtuellen Umgebung Kosten aufgeschlüsselt und mit denen einer traditionellen Umgebung verglichen. Dies beinhaltet lediglich die Hardware Kosten, eventuell nötige Dienstleistungen zur Implementierung sind hier nicht berücksichtigt.

Als Rahmenbedingungen werden fünf Informatiksäle mit jeweils 20 Arbeitsplätzen definiert. In der Schule sind 400 Schüler angemeldet. Es wird angenommen, dass jedes virtuelle Betriebssystem 10GB benötigen wird.

Beginnend mit der **virtuellen Infrastruktur** wird vorerst ein geeigneter ThinClient ausgewählt. Die Wahl viel auf den HP T5145 (Stückliste kann in Tabelle 4-1 gefunden werden).

Anzahl	Bezeichnung	Preis	Gesamtpreis
100	HP t5145 ThinClient	€ 249,00	€ 24.900,00
		Gesamtpreis	€ 24.900,00

Tabelle 4-1: Stückliste Thinclients

Um die ThinClients nun auch noch betreiben zu können sind die folgenden Serverkomponenten nötig. Diese Posten dienen sowohl zur Virtualisierung als auch um die anderen nötigen Komponenten zu betreiben. Im Zuge der technischen Implementierung hat sich gezeigt, dass ein 3GHz Prozessor Kern in 3 virtuelle CPUs geteilt werden kann. Somit wird angenommen, dass auf der für die VM-Server ausgewählten Hardware mindestens 20 virtuelle Instanzen parallel betreiben werden können. (2 physikalische Quad Core CPUs; 3GHz pro Kern => 8 Kerne mit 3GHz => maximal 24 virtuelle Instanzen). Dies würde bei 100 Arbeitsplätzen genau fünf VM-Server ergeben. Um dabei aber auf der „sicheren“ Seite zu sein wird einer als Reserve genommen. Weiters ergibt sich durch die oben definierte Anzahl an Schülern und dem geschätzten Speicherbedarf pro Schüler von 10GB ein Gesamtbedarf von 4TB. Um diese Herausforderung zu bewältigen kann Tabelle 4-2 als Stückliste für die Serverseite aufgefasst werden.

Position	Stück	Bezeichnung	Preis	Gesamtpreis
1	6	HP ProLiant DL165 G5p	€ 2.838,15	€ 17.028,90
2	18	HP 4 GB Registered PC2-5300 Niederspannungs-DDR2-Speicherkit (2 x 2 GB)	€ 166,60	€ 2.998,8
3	1	HP ProLiant DL120	€ 815,15	€ 815,15
4	1	HP StorageWorks 20 Modular Smart Array	€ 2.546,60	€ 2.546,60
5	5	HP 1 TB 7.200 U/Min Hot-Plug-fähige SAS Midline-Festplatte mit 2 Ports, 1 Jahr Garantie	€ 702,10	€ 3.510,50
			Gesamtpreis	**€ 26.899,35**

Tabelle 4-2: Stückliste Server Infrastruktur

Somit würde sich die Gesamte virtuelle Lösung auf 51.799,35 Euro belaufen und würde serverseitig noch ein Wachstum um 20 Arbeitsplätze erlauben.

Zum Vergleich dazu werden nun die **nicht virtualisiert** Kosten aufgeschlüsselt beziehungsweise ermittelt. Durch die fehlende Serverkomponente ist dies wesentlich einfacher. Hier wurde ein Standard Desktopcomputer von HP gewählt. (siehe Tabelle 4-3)

Stück	Bezeichnung	Preis	Gesamtpreis
100	HP Compaq dc5700 SFF	€ 556,80	**€ 55.680,00**
		Gesamtpreis	**€ 55.680,00**

Tabelle 4-3: Stückliste Arbeitsplatzcomputer

Wie in dem Beispiel anschaulich demonstriert ist bei einer Anzahl von 100 Arbeitsplätzen die virtualisierte Infrastruktur schon billiger als die nicht virtualisierte. Interessant ist nun ab welcher Dimension, Größenordnung beziehungsweise Anzahl an Arbeitsplätzen, der virtuelle Ansatz kostengünstiger wird. Um dies zu ermitteln wurde die oben angeführt Infrastruktur in Abhängigkeit zur Anzahl der Arbeitsplätze gestellt. Diese Anzahl wurde nun in Schritten von fünf erhöht.

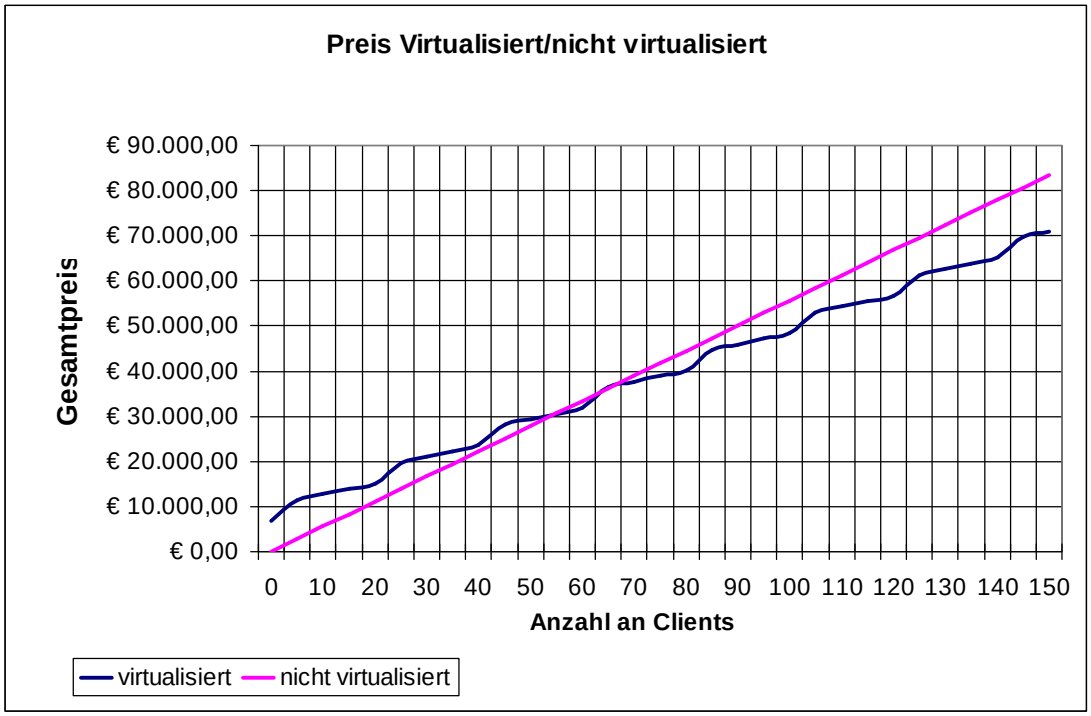

Abb. 4-8: Kosten in Abhängigkeit der Arbeitsplätze

Wie in Abb. 4-8 ersichtlich ist, kreuzen sich die Kurven mehrmals bis die virtualisierte Variante endgültig günstiger wird. Die mehrfachen Schnittpunkte können dadurch erklärt werden, dass die teure Anschaffung eines VM-Servers für 20 Arbeitsplätze gilt. Die verhältnismäßig günstigen ThinClients produzieren nur noch einen kleinen linearen Anstieg innerhalb einer „Stufe“.

Obwohl die virtualisierte Variante schon bei einer relativ geringen Anzahl an Arbeitsplätzen kostengünstiger ist, lässt sich hier noch weiteres Potential nutzen. Um die Kosten der ThinClients zu senken kann unter Umständen bestehende Hardware in das neue virtuelle Konzept integriert werden. Die Hardware Vorrausetzungen sind dabei im Wesentlichen auf die PXE Funktionalität beschränkt. Auch sind keine beweglichen Geräte, wie Festplatten oder optische Laufwerke nötig. Sollte keine PXE Funktionalität geboten sein, so sieht der PXE Standard an sich vor, die Funktionalität auf eine Netzwerkkarte auszulagern. Dies bedeutet, dass auch alte Hardware womöglich durch die Anschaffung neuer Netzwerkkarten weiterverwendet werden kann.

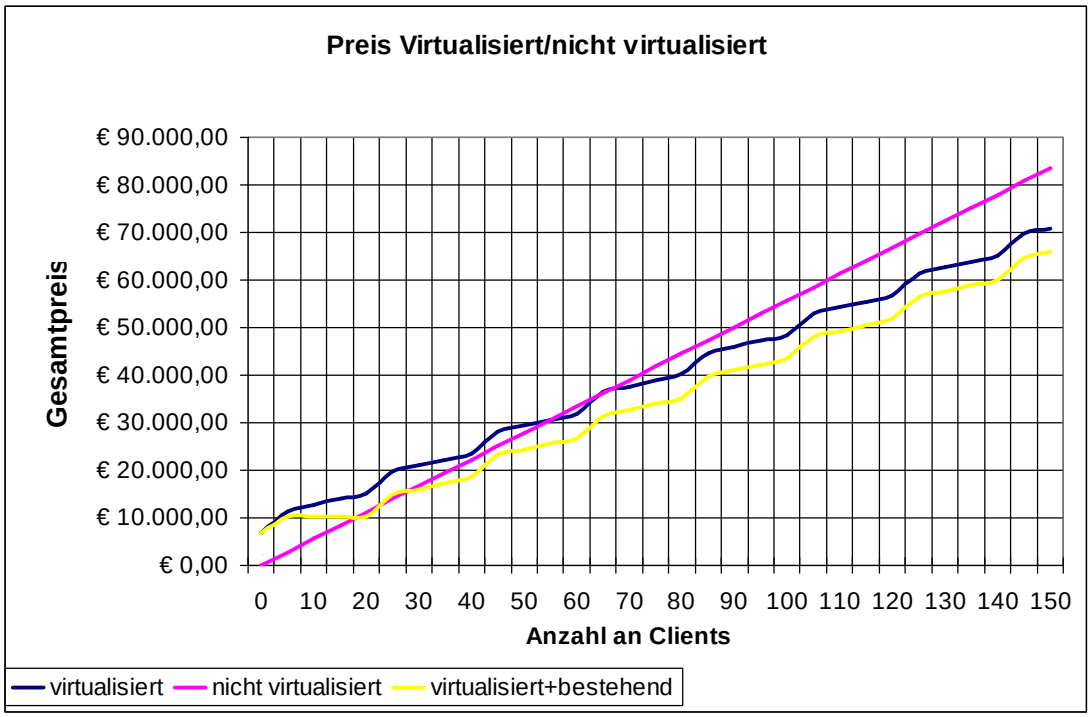

Abb. 4-9: Virtualisiert/nicht virtualisiert/mit bestehender HW in Abhängigkeit der Anzahl

Die in Abb. 4-9 dargestellte Grafik berücksichtigt nun auch diesen Aspekt und zeigt drei Geraden, virtualisiert, nicht virtualisiert und virtualisiert mit bestehender Hardware. In der Grafik wird von 20 bestehenden und netzwerkbootfähigen aber alten Geräten ausgegangen.

Der zweite Aspekt lässt sich vorwiegend in Testumgebungen finden. Hierbei wird oft mehr als ein Betriebssystem beziehungsweise „Computer“ benötigt. Diese Hardware ist dann oftmals für die Dauer des Projektes für andere Dinge nicht zu gebrauchen.

4.4.2 Operative Kosten

Im Folgenden sollen die beiden Ansätze hinsichtlich der entstehenden operativen Kosten analysiert werden. Auch wenn durch die einfache Handhabung beziehungsweise dem Einsatz von wartungsarmen ThinClients geringere Wartungskosten zu erwarten sind, werden diese im Folgenden nicht berücksichtigt. Dies rührt daher, dass kaum eine seriöse Aussage über den tatsächlichen Wartungsbedarf beziehungsweise der dafür nötigen Zeit ermittelt werden kann. Die folgende Kalkulation beinhaltet nun exemplarisch den Stromverbrauch in Kilowatt Stunden definiert durch die folgenden Rahmenbedingungen:

Tabelle 4-4 zeigt nun die maximale Leistungsaufnahme der Virtualisierungskomponenten (laut Hersteller).

Gerät	Max. Stromverbrauch	Kommentar
HP t5145 Thin Client	9 kWh[1]	ThinClient
HP ProLiant DL380	437 kWh[2]	VM-Server
HP ProLiant DL120	235 kWh[3]	Dispatch/PXE
HP StorageWorks 20	232 kWh[4]	Datenspeicher

Tabelle 4-4: Stromverbrauch pro Virtualisierungskomponente

[1] Vgl. [HP09a]
[2] Vgl. [HP09b]
[3] Vgl. [HP09c]
[4] Vgl. [HP09d]

Für die nicht virtualisierte Lösung wird diese Tabelle etwas einfacher und kürzer

Gerät	Max. Stromverbrauch	Kommentar
HP Compaq dc5700 SFF	44 kWh[5]	Arbeitsplatz-PC

Tabelle 4-5: Stromverbrauch pro nicht virtualisiert

Der gesamte Leistungsverbrauch setzt sich aus unterschiedlichen Parametern zusammen. Dies sind der maximale Verbrauch jeder Komponente, die Laufzeit jeder Komponente und auch die Anzahl aller Komponenten. Am besten lassen sich die beiden Lösungen (virtualisiert und nicht virtualisiert) auch hier wieder an der Anzahl der Arbeitsplätze vergleichen.

Der Verbrauch laut Hersteller ist schon ermittelt und de facto vorgegeben und um auch die letzten Parameter zu definieren ist die Laufzeit beziehungsweise die vorrausichtlichen Betriebsstunden jeder Komponente in Tabelle 4-6 ersichtlich. Um nicht unnötig Komplexität in die Kalkulation zu bringen wird dies nun nur auf zwei Komponenten geteilt, Arbeitsplatzcomputer und die Serverkomponenten.

Verbraucher	Laufzeit	Kommentar
Arbeitsplatz	5 Tage/Woche; 8 Stunden 1 Tag/Woche; 6 Stunden	Montag – Freitag Samstag
Server	7 Tage/Woche; 24 Stunden	

Tabelle 4-6: Laufzeit der Komponenten

Ansonsten gelten wieder die gleichen Annahmen wie auch schon bei den Anschaffungskosten. Ein VM-Server kann 20 virtuelle Instanzen parallel betrieben und jeder Schüler bekommt eine virtueller Maschine mit 10GB Festplattenspeicher.

[5] Vgl. [HP09e]

Abb. 4-10 zeigt diese Rahmenbedingungen in Abhängigkeit mit der Anzahl an Arbeitsplätzen.

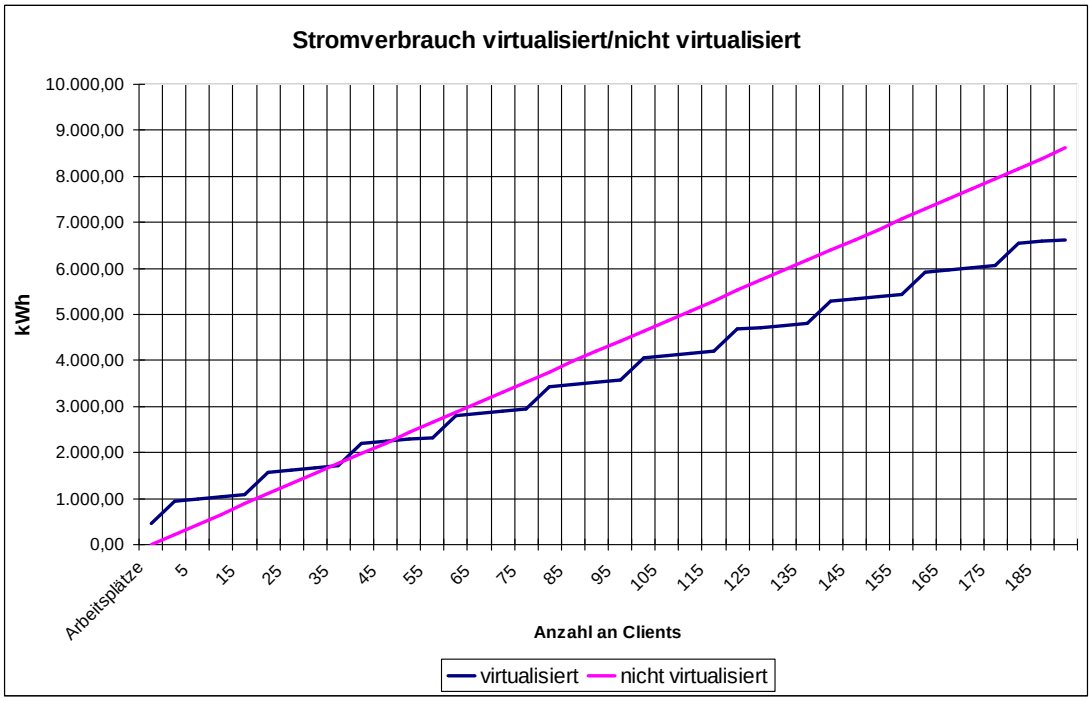

Abb. 4-10: Kosten Stromverbrauch in Abhängigkeit der Computeranzahl

Es wurde darauf verzichtet das Resultat in Euro umzurechnen, da dies eine starke Abhängigkeit zum jeweiligen Stromanbieter und dessen Konditionen hat. Als Richtwerte wurde immer der maximale Verbrauch des Herstellers in Watt verwendet.

Wie zu erwarten steiget der Stromverbrauch einer nicht virtualisierten Lösung linear an. Das virtuelle System startet zwar mit einem verhältnismäßig hohen Stromverbrauch kann diesen allerdings so lange nahezu konstant halten wie virtuelle Maschinen auf einem VM-Server betrieben werden können. Auch hier ergibt sich somit wieder ein sukzessiver Anstieg. Die Linien haben dadurch auch mehr als nur einen Schnittpunkt. Diese Punkte bewegen sich zwischen einer Arbeitsplatzanzahl von zirka 30 bis 70. Abb. 4-11 bringt dieses Wechselspiel besser in den Blickpunkt:

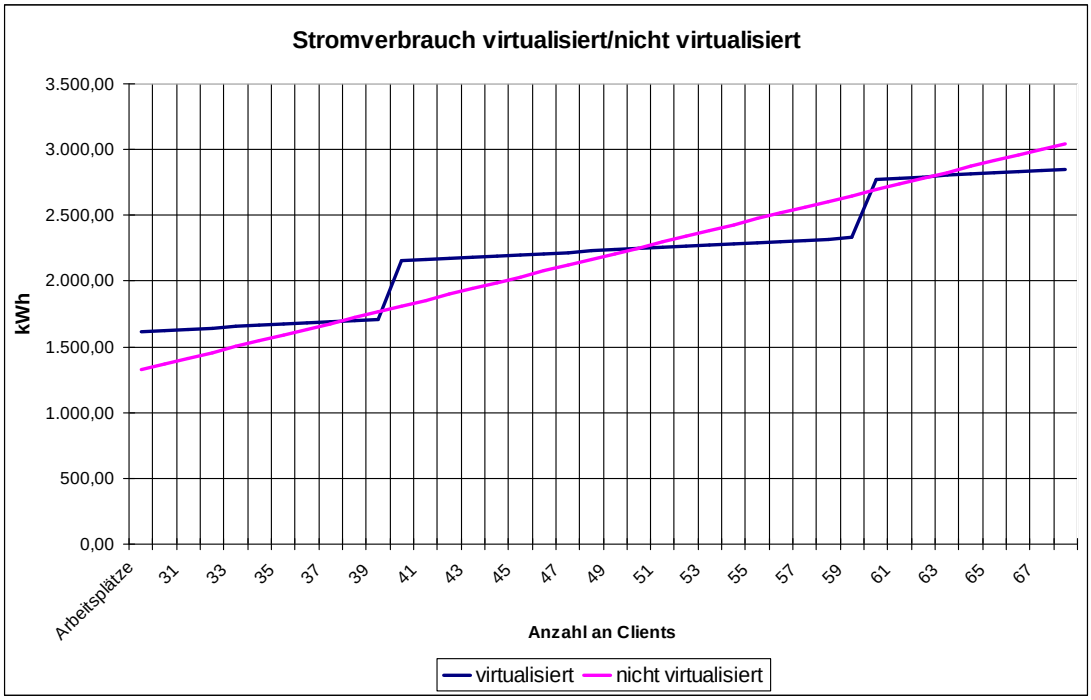

Abb. 4-11: Kosten Stromverbrauch in Abhängigkeit der Computeranzahl (ZOOM)

In dem Moment wo ein dritter VM-Server in Betrieb genommen werden muss (Anzahl der Arbeitsplätze größer als 40) steiget der Verbrauch schlagartig an. Die Linien kreuzen sich wiederum zwischen dem 50 und 51 Arbeitsplatz wobei der nächste VM-Server (mehr als 60 Arbeitsplätze) die nicht virtualisierte Lösung zum letzten Mal weniger Strom verbrauchen lässt bis der 64 Arbeitsplatz in Betrieb genommen wird. Somit lässt sich beweisen, dass der Stromverbrauch ab einer Anzahl von 64 Arbeitsplätzen durch den Einsatz von Virtualisierung drastisch reduziert werden kann.

Als kleine Anmerkung sei noch angebracht, dass auch diese Kalkulation noch etwas Optimierung beim Einsatz der virtuellen Variante erlauben würde. So könnten zum Beispiel ein Teil der Virtualisierungsserver über die Nacht und das Wochenende ausgeschaltet werden. Diese würde den Stromverbrauch noch weiter drosseln und keinen Nachteil mit sich bringen. Auch könnte eine zusätzliche Intelligenz auf der Dispatch Komponente dafür sorgen, dass ein Virtualisierungsserver wenn nötig eingeschaltet wird und nachdem alle virtuellen Instanzen heruntergefahren wird auch wieder

ausgeschalten wird. Diese Maßnahmen wären technisch definitiv machbar und würden die Leistungsaufnahme erheblich senken.

4.5 Integration in bestehende Schulnetzwerke

Frei nach dem Motte „never touch a running System“ ist die Resonanz auf Änderungen in dem Ausmaß vermutlich mit etwas Vorsicht behaftet. Auch die Verwendung von virtuellen anstatt physikalischen Maschinen weckt anfangs vor allem Skepsis.

Allerdings ist die Umsetzung beziehungsweise der Umstieg auf eine virtualisierte Plattform wie hier beschrieben wesentlich einfacher als es am ersten Blick erscheinen mag. Das System ist in sich sehr transparent, homogen und vor allem modular. Speziell durch die Separierung in einzelne Komponenten lassen sich auch erste Ergebnisse sichtbar machen ohne gleich intensiv in Hardware investieren zu müssen. Sollte sich das Konzept bewähren, lässt es sich beliebig skalieren.

Eine Migration könnte nun im Versuchsstadium beginnen. Dies könnte zum Beispiel anfangs nur einen Informatiksaal betreffen, und dieser ist im Idealfall nicht sonderlich beliebt da ohnehin hauptsächlich alte Hardware vorhanden ist. Sofern diese alte Hardware noch über eine Netzwerkschnittstelle starten kann, ist dies für einen ersten Test völlig ausreichend.

Auch ist es gut möglich, dass einige der benötigten Komponenten schon vorhanden sind. In erster Linie könnte ein zentraler LDAP-Server, mit allen Schülern erfasst, vermutet werden. Dieser könnte nun auch gleich zur PXE-Authentifizierung beziehungsweise zur Verteilung der virtuellen Betriebssysteme dienen.

Alle anderen Komponenten können so wie in der exemplarischen technischen Implementierung auf einem physikalischen Server

zusammengefasst werden. Nachdem dieser Server dann auch die Virtualisierung übernimmt, wäre es hier nicht ratsam die Dienste virtualisiert zu betreiben.

Somit würden sich die Anschaffungskosten für einen initialen Testbetrieb im Wesentlichen auf einen Server beschränken. Dieser sollte so gewählt werden, dass er auch in einem weiteren Ausbau einen Dienst ausführen kann.

Wie weit sich nun eine solche Testinfrastruktur netzwerktechnisch in das bestehende Schulnetz integrieren lässt, lässt sich hauptsächlich durch den Aufbau dieses definieren. Dies umfasst aber prinzipiell nur Aspekte wie den Zugriff von zuhause oder generell extern. Allerdings würden sich diese Anforderungen in den meisten Fällen auch irgendwie erfüllen lassen.

5. TECHNISCHE IMPLEMENTIERUNG

Die folgende technische Implementierung ist als ein Prototyp beziehungsweise eine Art Machbarkeitsstudie zu verstehen. Durch die Verwendung von OpenSource Software, sowohl beim Betriebssystem als auch bei der Virtualisierungsapplikation, lässt sich bei dieser Komplexität kaum eine generische Anleitung produzieren die dann auch wirklich unter allen Umständen funktioniert. Umstände sind hierbei allein schon sehr viele durch die Hardware gegeben. Hardware Hersteller entwickeln Treiber oftmals nur für Windows Systeme. Dies ist speziell bei so genannter Consumer Hardware der Fall. Nachdem aber auch ein Ziel dieses Konzeptes die Kostenersparnis ist, kommt hier teure Serverhardware nicht in Frage, was das Treiber Problem natürlich brisant macht. Somit bietet dieses Konzept einen ungefähren Leitfaden wie es unter den hier gegeben Umständen funktioniert. Vieles davon wird vermutlich ohne weiteres unter anderen Gegebenheiten funktionieren, jedoch können vereinzelt Probleme auftreten.

5.1 Umsetzung des Konzeptes

Der modulare Aufbau des Konzeptes wird in der Umsetzung auch gleich verwendet. Nachdem es möglich ist die Komponenten zusammenzufassen kann relativ rasch und auch mit limitierten Hardwareressourcen ein Ergebnis erzielt werden.

In diesem Fall wird lediglich die Virtualisierungskomponente als einzelne physikalische Maschine betrieben. Alle anderen Komponenten werden zusammengefasst auf einem zweiten Computer. Dabei wird darauf geachtet, dass die Beispielimplementierung auch durchaus skaliert werden kann und einzelne Komponenten migriert werden können.

Abbildung (Abb. 5-1) zeigt schematisch den Versuchsaufbau

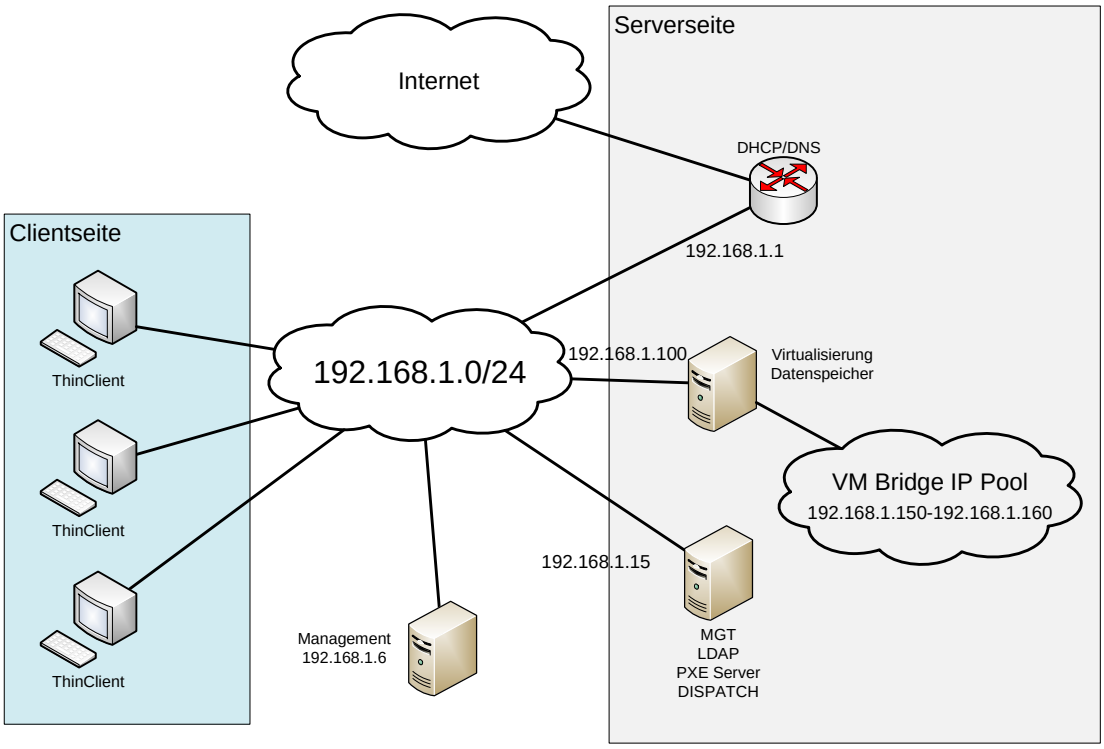

Abb. 5-1: Übersicht der Beispielimplementierung

Trotz Zusammenfassung braucht jede Komponente letzten Endes eine zugrunde liegende Applikation. Tabelle 5-1 zeigt die für die Komponenten ausgewählte Software. Komponenten die nicht als „fertige“ Applikation verfügbar sind werden exemplarisch entwickelt. Die Virtualisierungsapplikationen sind im Zuge von Kapitel 3 diskutiert worden.

Komponente	Applikation	Kommentar
Virtualisierung	KVM	
ThinClients	OpenThinClient	
LDAP	OpenThinClient-LDAP	
DISPATCH	Selbstentwickeltes Skript	Perl Skript
Management	Selbstentwickelte Skripte	HTML und PHP

Tabelle 5-1: verwendete Software

5.2 Installation von CentOS

In den folgenden Unterkapiteln wird in groben Zügen erklärt wie die einzelnen Applikationen beziehungsweise das Betriebssystem installiert werden. Wie in der Linuxwelt üblich, gibt es meist mehr als nur eine Möglichkeit um ans Ziel zu kommen, somit sind die folgenden Ausführungen lediglich als „gangbarer Weg“ anzusehen.

Die Installation von CentOS verläuft relativ einfach. Dies kann sowohl mittels CD/DVD als auch über ein Netzwerk gesehen. Im Fall von mehreren Installationen bietet sich sicherlich die Netzwerkvariante an. Hierbei wird ein kleines CD Image aus dem Internet geladen, das einen TCP/IP-Stack zur Verfügung stellt und somit die Verbindung zu dem im Netzwerk verfügbaren Medium herstellt (HTTP, FTP, NFS).

Sollte kein CD/DVD-Laufwerk verfügbar sein, da es ja an sich im Betrieb nicht nötig ist, kann das Netzwerkinstallationsmedium auch mittels eines bootfähigen USB-Geräts gestartet werden. Ein nützliches Werkzeug ist hier der „Liveusb-Creator“ (https://fedorahosted.org/liveusb-creator/). Dieses Tool funktioniert sowohl unter Windows als auch unter Linux. Damit ist es möglich ein ISO-Image auf einen USB-Gerät zu spielen und dieses Gerät bootfähig zu machen. Natürlich ist aber auch ein externes USB-CD/DVD Laufwerk eine Option.

Ein großer Teil des Installationsprozesses ist selbsterklärend, lediglich die folgenden Punkte sollten speziell beachtet werden:

Partitionsschema

Es empfiehlt sich immer die Betriebssystem eigenen Daten von den Nutzdaten (in dem Fall die virtuellen Image Dateien) zu trennen. Im Fall, dass das Betriebssystem nicht mehr funktionsfähig ist und eine

Neuinstallation benötigt, lässt sich dies durchführen ohne die Nutzdaten zu verlieren.

Paketauswahl

Nachdem das Hostsystem lediglich die Basis für die einzelnen Komponenten dienen soll, ist es hier unnötig diverse Office Applikationen zu installieren. Dies hilft nicht nur den benötigten Speicherplatz zu reduzieren, sondern auch die Sicherheit und Stabilität des Systems zu erhöhen. Sicherheit in dem Sinne, dass Applikationen zwangsläufig Schwachstellen haben die von Angreifern ausgenutzt werden können. Stabilität ergibt sich aus der Tatsache, dass Prozesse die zwar laufen aber nicht benötigt werden, im Fehlerfall dennoch das gesamte System beeinflussen können.

Eine grafische Oberfläche ist prinzipiell nicht notwendig, bringt aber vermutlich mehr Nutzen als Schaden. Im Wesentlichen gibt es zwei so genannte Fenstermanager (Window-Manager) die im Moment von den meisten Distributionen verwendet werden, GNOME und KDE. Nachdem das Hostsystem aber mehr als Server fungieren soll, ist das etwas schlankere GNOME die vermutlich bessere Wahl. KDE mag zwar im Desktop/Anwenderbereich durch die große Anzahl an grafischen Besonderheiten glänzen, benötigt aber dafür jede Menge Hardware Ressourcen.

IP-Konfiguration

CentOS aktiviert standardmäßig IPv6. Dies führt dazu, dass der Computer regelmäßig versucht mittels IPv6 Kontakt zu seiner Umwelt aufzunehmen. Dies beeinflusst das System zwar nicht wesentlich, ist aber in den meisten Fällen unnötig, führt nur zu Verwirrungen und bereitet teilweise Probleme mit der Implementierung von so genannten Network-Bridges.

Modifikationen am System

Nachdem es sich hierbei um ein Konzept, einen Prototypen handelt, liegt kein wirklicher Fokus auf Sicherheit. Somit können Funktionen wie die Linuxfirewall und SELinux deaktiviert werden. SELinux verhindert zum Beispiel das Prozesse unautorisiert in Verzeichnisse schreiben beziehungsweise von ihnen lesen.

VNC Server

Um auch ohne angeschlossenen Monitor das Hostsystem mittels grafischer Oberfläche warten zu können empfiehlt es sich den VNC Server Support zu aktivieren. VNC an sich ist eine relativ alte Technologie die sich häufig der negativen Kritiken von Sicherheitsexperten beugen muss. Sämtlicher VNC Verkehr ist standardmäßig nicht verschlüsselt und somit für jeden lesbar. Dies soll in dem Konzept aber nicht weiter von Bedeutung sein und somit kann VNC aktiviert werden.

```
bash# vncserver
```

Beim ersten Aufruf konfiguriert sich der VNC Server de facto selbst. Dazu wird ein Passwort generiert das dann von den VNC Clients beim Aufruf eingeben wird. Damit wird eine Art einfache Authentifizierung möglich.

Netzwerkkonfiguration

In der Beispiel Realisierung wurde nur eine Netzwerkschnittstelle benutzt, sprich es wurde auf eine dedizierte Managementanbindung verzichtet. Auch wird die nötige DHCP Funktionalität von einem Router und nicht von einem CentOS Server zur Verfügung gestellt.

Um den virtuellen Maschinen nun Zugriff auf das Netzwerk zu geben beziehungsweise diese vom Netzwerk (wie zum Beispiel den ThinClients)

erreichbar zu machen wird eine Network-Bridge implementiert. Dies ist im Wesentlichen als eine virtuelle Bridge zu verstehen. Die Netzwerkschnittstelle eth0 wird dieser Bridge als physikalisches Gerät zugeordnet und darauf können dann mehrere unterschiedliche virtuelle IP Adressen, inklusive virtueller MAC Adresse, betrieben werden.

```
[root@kvm ~]# cat /etc/sysconfig/network-scripts/ifcfg-bridge0
DEVICE=br0
TYPE=Bridge
BOOTPROTO=static
BROADCAST=192.168.1.255
IPADDR=192.168.1.100
NETMASK=255.255.255.0
NETWORK=192.168.1.0
ONBOOT=yes
```

```
[root@kvm ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
HWADDR=00:1C:C0:AD:87:5B
ONBOOT=yes
BRIDGE=br0
```

SSH Konfiguration und Schlüsselaustausch

Die SecureSHell (SSH) bietet unter anderem die Funktion gleich beim Verbindungsaufbau Befehle für das entfernte System als Parameter zu übergeben. Die Ausgabe wird direkt lokal angezeigt. Nachdem die Anforderung besteht die Kommunikation zwischen der Dispatch Komponente und den VM-Servern mit Hilfe eines standardisierten Protokolls zu realisieren bietet sich diese Methode gut an.

SSH bietet im Wesentlichen zwei Arten um den Zugriff zu gewähren, dies wäre zum einen die Authentifizierung mittels Benutzernamen und dem dazugehörigen Passwort oder mit Hilfe von vorher ausgetauschten Schlüsseln. Nachdem in diesem Fall ein automatisiertes Login nötig ist, müssen somit die SSH Schlüssel ausgetauscht werden. Die SSH-Schlüssel basieren auf RSA, einem asymmetrischen Verschlüsselungsverfahren, das ein Schlüsselpaar, bestehend aus einem öffentlichen und einem privaten Schlüssel, generiert. Wird nun der öffentliche Schlüssel von Computer A in

Computer B als autorisiert konfiguriert, kann Computer A eine SSH Verbindung ohne Passwortabfrage zu Computer B aufbauen.

Um dies zu ermöglichen muss folgende Konfiguration vorgenommen werden:
Computer A:

```
computerA# ssh-keygen
computerA # cat .ssh/idkey_pub
```

Der öffentliche Schlüssel kann nun in der Datei authorized_keys von Computer B eingefügt werden. Sollte diese Datei noch nicht existieren, kann sie einfach angelegt werden. Der folgende Befehl öffnet die authorized_keys Datei für den Benutzer root am Computer B falls vorhanden, wenn nicht wird diese angelegt.

```
computerB# vi  ~root/.ssh/authorized_keys
```

Im Anschluss kann von Computer A wie folgt getestet werden (vorausgesetzt der Name „computerB“ wird auf die richtige IP Adresse aufgelöst):

```
computerA# ssh root@computerB hostname
```

Ist die Ausgabe der Hostname von Computer B (also „computerB“) hat alles geklappt.

5.3 Virtualisierung KVM

KVM lässt sich unter CentOS 5.3 zwar nicht im Installationsprozess mitinstallieren, allerdings sind die nötigen KVM Kernelmodule schon vorkompiliert in den CentOS Repositories verfügbar. Daher beschränkt sich die Installation auf das Absetzen des folgenden Kommandos:

```
Bash# yum install kvm kmod-kvm qemu libvirt phyton-virtinstall
virt-manager
```

Yum überprüft vor der eigentlichen Installation ob alle Abhängigkeiten erfüllt sind. Sollte dies nicht der Fall sein, werden die fehlenden Packete herausgesucht und ebenfalls zur Installation angeboten. Dies sollte für eine fehlerfreie Installation beziehungsweise für den anschließenden Betrieb auch getan werden. Danach ist ein Neustart des Systems erforderlich.

Nachdem das System neu gestartet hat, sollte überprüft werden ob das nötige KVM-Kernelmodul auch tatsächlich gestartet wurde. Hierfür gibt das das Linux Werkzeug lsmod, welches alle geladenen Module anzeigt. Um hier nicht länger nach KVM suchen zu müssen kann die Ausgabe auf die relevante Information gekürzt werden (grep). In Summe ergibt sich folgendes Kommando:

```
bash# lsmod | grep –i kvm
```

Ist das Modul geladen sollte noch der Status des KVM-QEMU-Hypervisors überprüft werden. KVM greift hier auf die virsh (Virtual Shell) zurück. Das folgende Kommando zeigt alle aktuell laufenden virtuellen Maschinen.

```
Bash# virsh -c qemu:///System list
```

der –c Parameter bedeutet „connect“ und übergibt der virsh als qemu Server das lokale System (qemu:///System). Die list Operation listet dann alle Kommandos auf.

Nachdem bis jetzt noch keine Virtuelle Maschine gestartet wurde, ist die Liste natürlich leer. Nachdem das System aber in der Lage ist den aktuellen Status auszulesen, kann davon ausgegangen werden, dass es voll funktionsfähig ist.

5.4 OpenThinClient Server

Der OpenThinClient Server basiert auf Java, und daher ist auch die Installationsroutine in Java gehalten. Die folgenden Kommandos laden die Javainstallationsdatei und führen diese aus.

Nachdem die Java JRE in der Version 1.6 im CentOS Repository verfügbar ist lässt sich diese sehr einfach mit folgendem Befehl installieren.

```
Bash# yum install java
```

Nach der Installation von Java muss die OpenThinClient Installationsdatei vom Internet geladen werden. Es empfiehlt sich diese Datei in ein temporäres Verzeichnis zu laden.

```
Bash# mkdir /temp/otcs
Bash# cd /temp/otcs
Bash# wget http://downloads.openthinclient.org/openthinclient-0.4.1.jar
```

Sobald der Download beendet ist kann der OpenThinClient Server installiert werden. Die Installationsroutine ist grafisch gehalten und intuitiv fertig zu stellen.

```
Bash# java –jar *.jar
```

Das temporäre Verzeichnis und die Installationsdatei können nun wieder gelöscht werden.

```
Bash# cd ..
Bash# rm –rf /temp/otcs
```

5.5 LDAP

Nachdem der OpenThinClient Server schon einen LDAP-Verzeichnis-Server mitbringt und dieser an sich auch streng nach LDAP-Spezifikation arbeitet, wurde im Zuge dieser exemplarischen Implementierung auf die Installation und Konfiguration eines eigenen LDAP-Servers verzichtet. Auch sind in vielen IT-Infrastrukturen schon LDAP-Server anzufinden, welche dann auch durchaus weiterverwendet werden können, was den beispielhaften Aufbau hier nicht zwingend nötig macht.

Um den LDAP-Server für den Einsatz im Gesamtkonzept vorzubereiten sind folgende Modifikationen nötig:
Für jede virtuelle Instanz die einem Benutzer zugeordnet wird werden drei Parameter dem benutzerspezifischen Teilbaum im LDAP-Verzeichnis zugeordnet. Diese Parameter sind:

- Name
- Dateiname der das virtuelle Betriebssystem am Datenspeicher reflektiert
- Dedizierte MAC Adresse des Benutzers

Zum Modifizieren, Löschen und Anlegen von Benutzern sowie das Hinzufügen zusätzlicher Parameter können sämtliche verfügbare LDAP-Clients verwendet werden. Einziger Unterschied zur vielen anderen LDAP-Verzeichnisservern (wie zum Beispiel OpenLDAP) ist, dass hier standardmäßig der TCP Port 10389 anstatt 389 verwendet wird. Als Clientsoftware empfiehlt es sich die Linux Programme ldapadd, ldapdel und ldapmodify zu installieren. Unter CentOS 5.3 kann dies mit folgendem Befehl geschehen:

```
Bash# yum install ldap-clients
```

5.6 Dispatch

Die Dispatch-Komponente wurde durch ein Perl Skript realisiert. Um diese möglichst modular und skalierbar zu gestalten wird die Konfigurationsdatei kvm_server_list in den Standardlinuxkonfigurationsordner /etc erstellt. Diese Datei soll eine Liste von allen KVM-Servern enthalten. Die Referenz zwischen KVM-Servername und IP-Adresse wird durch Standardlinux Methoden, der /etc/hosts Datei, hergestellt.

Die Initialkonfiguration könnte nun wie folgt aussehen:

```
Bash# touch /etc/kvm_server_list
Bash# echo “kvm01” >> /etc/kvm_server_list
Bash# echo “kvm01    192.168.1.100“ >> /etc/hosts
```

Die obigen Befehle erstellen die KVM-Server-Liste, fügen als ersten Wert kvm01 hinzu und erstellen die nötige IP-Adressreferenz in der /etc/hosts Datei.

Abb. 5-2 zeigt den prinzipiellen Programmablauf des Dispatch-Skriptes als Struktogramm. Im Wesentlichen werden alle registrierten KVM-Server ausgelesen, der Zustand beziehungsweise die aktuelle Anzahl an laufenden virtuellen Maschinen ausgelesen und der Server mit der geringsten Anzahl wird ausgewählt.

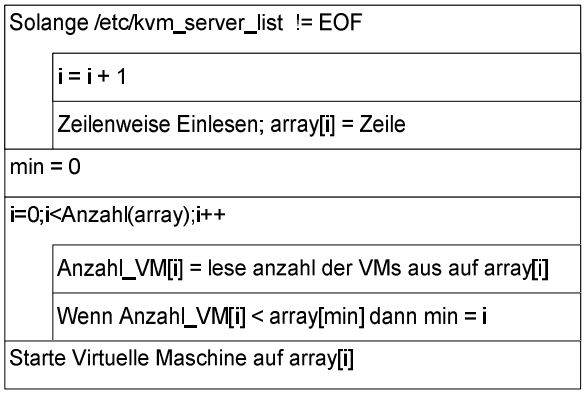

Abb. 5-2: Struktogramm Dispatch

Nachdem jede virtuelle Maschine durch die Dispatch Komponente gestartet wird wäre es prinzipiell auch denkbar, dass der aktuelle Zustand beziehungsweise die aktuelle Anzahl an gestarteten virtuellen Maschinen nicht durch Abfragen der KVM-Server sondern durch Mitloggen der Aktivitäten ermittelt wird. Der Nachteil dabei ist nur, dass KVM standardmäßig keine Rückmeldung liefern wird, wenn eine virtuelle Maschine heruntergefahren wird, abstützt oder der KVM-Prozess am KVM-Server einfach gestoppt wird. Sprich ohne KVM-Server-Rückmeldung würden die Listen in der Dispatch Komponente nur wachsen, und dies könnte das Entscheidungskriterium negativ beeinflussen, da laut Dispatch-Liste zum Beispiel 10 virtuelle Maschinen auf KVM-Server 1 laufen. In Wahrheit sind davon aber schon längst 8 wieder heruntergefahren.

5.7 Management Interface

Um das Management Interface auf möglichst vielen „Gastsystemen“ zur Verfügung zu stellen wurde es webbasierend realisiert. Somit kann es auf nahezu jedem Client und ohne den vorherigen Austausch einer Clientsoftware verwendet werden.

Als Webserver wurde Apache gewählt, welcher wie folgt unter CentOS installiert und gestartet werden kann.

```
Bash# yum install apache
Bash# service httpd status
Bash# service httpd start
```

Zur Realisierung wurde für jede Schnittstelle in HTML eingebetteter PHP Code verwendet. Sprich die grafische Umsetzung (was der Benutzer sieht) ist HTML und die Intelligenz dahinter PHP. Innerhalb des PHP-Codes werden auch Systembefehle verwendet, wie zum Beispiel die LDAP-Suche.

Starten von virtuellen Maschinen

Das Starten von virtuellen Maschinen folgt im Wesentlichen folgendem Ablauf:

1) Eingabe des Benutzernamen
2) LDAP-Abfrage um die Parameter herauszulesen
3) Auswahl der virtuellen Maschine(n) die gestartet werden soll(en)
4) Erstellen des/der KVM-Konfigurationsdateien
5) Aufruf des Dispatchers und Übergabe der Konfigurationsdatei(en)

Eine auf HTML und PHP basierende Beispielimplementierung befindet sich im Anhang.

Neues Betriebssystem für Benutzer

Sollte ein Benutzer ein neues oder zweites virtuelles Betriebssystem benötigen so kann dies auch komfortabel über eine Webseite erstellt werden.

Im Wesentlichen lässt sich dies auf die folgenden Schritte aufteilen:

- Anmeldung mit LDAP-Verzeichnis-Benutzernamen
- Auslesen der verfügbaren Betriebssysteme
- Auswahl des gewünschten Betriebssystems und Eingabe einer freien MAC Adresse
- Duplizieren der Betriebssystem-Datei für den Benutzer
- Reflexion der Änderung im LDAP-Verzeichnisbaum

Eine auf HTML und PHP basierende Beispielimplementierung befindet sich im Anhang.

Konfiguration des PXE-Servers

Der OpenThinClient-Server bringt in der Standardinstallation ein Webportal mit sich. Dieses Portal basiert auf Java und erlaubt umfangreiche Einstellungen hinsichtlich Clients, Benutzer und Applikationen und ist für

diesen Zweck mehr als ausreichend. Auch im schulischen Einsatz kann durchaus auf das schon Bestehende zurückgegriffen werden. Modifikationen an dieser Stelle sind ohnehin nur von den Administratoren zu tätigen und es ist davon auszugehen, dass die Bedienung für einen Experten weitgehend intuitiv ist.

Sollte sich in der Praxis herausstellen, dass es sinnvoll ist wenn auch Lehrer bestimmte Änderungen vornehmen können, kann angedacht werden spezielle Benutzerrollen zu definieren und ihnen somit Zugriff auf bestimmte Teile der Konfiguration geben. Ist dies nicht ausreichend kann ein Portal entwickelt werden, das die nötigen Änderungen im LDAP-Verzeichnis „kundenfreundlich“ erlaubt.

6. ANWENDUNG/FALLBEISPIEL DIDAKTISCHE ASPEKTE

Computer beziehungsweise IT fällt didaktisch betrachtet unter den Begriff Medien. Medien sind im Wesentlichen all jene „Objekte“ die im Unterricht eingesetzt werden können. Tabelle 6-1 zeigt nun eine solche Klassifizierung inklusive Beispiele pro Klasse. [KRON04]:

Medienklasse	**Beispiele**
Selbstinszinierende Medien	Sprache, Spiel, Theater, …
Bewegte Bildmedien	Film, Fernsehen, …
Mechanische Medien	Dias, Mikroskop, Kassette/CD, ..
Printmedien	Buch, Comic, Foto, …
Auditive Medien	Hörfunk, Tonband, ..
Neue Medien	Video, Computer, Internet, Handy, Palm, …

Tabelle 6-1: Medienklassifizierung [KRON04]

Die klassischen Zusammenhänge der Didaktik können grafisch sehr anschaulich durch das 1997 von Rudolf Cohn entwickelte didaktische Dreieck [Link02] dargestellt werden. Jede Spitze des Dreiecks (Abb. 6-1: Didaktisches Dreieck (vgl. [KRON04]Abb. 6-1) repräsentiert die grundlegenden Strukturelemente, Lehrer, Schüler und Sache, der Didaktik. [KRON04]

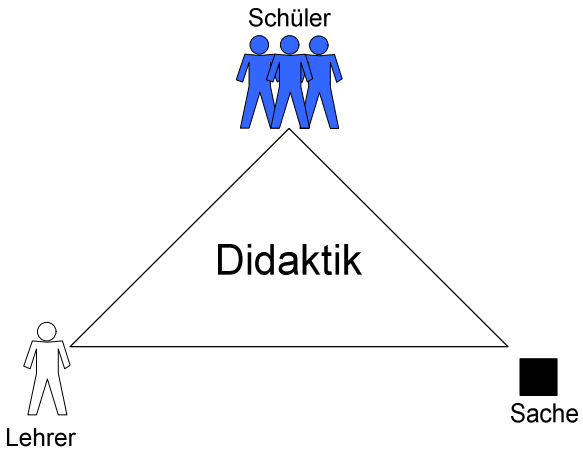

Abb. 6-1: Didaktisches Dreieck (vgl. [KRON04])

Dieser traditionelle Zusammenhang ist heutzutage allerdings oftmals nicht mehr ausreichend, da mehr und mehr Medien den Unterricht prägen. Dies lässt sich durch das Hinzufügen einer weiteren Ecke grafisch darstellen. Somit wird aus dem didaktischen Dreieck das didaktische Viereck (Abb. 6-2). [KRON04]

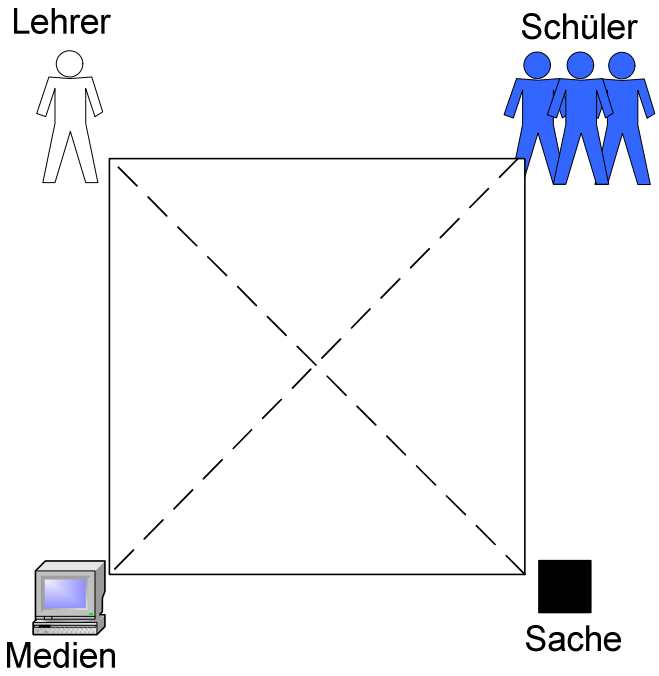

Abb. 6-2: Didaktisches Viereck (vgl. [KRON04])

In weiterer Folge hat dies zu bedeuten, dass Medien eine zentrale Komponente in der Unterrichtsplanung sind. Dies gilt nicht nur für jene Unterrichtsgegenstände die den Umgang mit Medien als Ziel haben, sondern auch all jene bei denen Medien als sinnvolle Ergänzung eingesetzt werden können.

6.1 Notebook Klassen

Der Einsatz von Notebooks in Schulen beziehungsweise im Unterricht wurde in den letzten Jahren ausgiebig diskutiert. Dabei teilen sich allerdings oftmals die Meinungen hinsichtlich der didaktischer Aspekte, Vorteile und Nachteile sowie der generellen Machbarkeit. Ziel dieser Arbeit ist es nun nicht diese Diskussion wieder aufzuwerfen oder gar eine Antwort auf die offenen Fragen zu finden, allerdings ist dieser Trend durchaus steigend und die schulische IT-Infrastruktur muss dafür gerüstet sein.

Das größte Problem bei Notebook Klassen ist, dass Schüler zwangsläufig Administratorrechte auf ihren Geräten brauchen. Nachdem diese Geräte auch meist (zumindest zu einem Teil) von den Schülern respektive deren Eltern gekauft werden lässt sich auch schwer argumentieren warum die Rechte dann beschnitten werden sollen. Dies führt in weiterer Folge zu einer großen Anzahl an unkontrollierten Computern, die aber auch irgendwie mit dem Schulnetzwerk interagieren müssen.

Dieses Problem wird häufig dadurch relativiert indem den Schülernotebooks lediglich der Zugang zum Internet gewährt wird. Dies führt allerdings das Prinzip der vernetzen Klasse in gewisser Weise ad absurdum, da viele Möglichkeiten einfach nicht benutzt werden können.

Weiters werden diese Notebooks privat verwendet, was nicht ausschließt, dass diese sowohl Hardware Defekte als auch Software Probleme bekommen können.
Durch den virtuellen Ansatz, beziehungsweise der Tatsache, dass jedem Schüler eine eigene virtuelle Umgebung zur Verfügung gestellt wird, lassen sich viele der Probleme weitgehend relativieren. Die Schüler können sowohl von zuhause als auch in der Schule auf ihr eigenes Betriebssystem zugreifen und werden dadurch unabhängig von dem Betriebssystem beziehungsweise der Softwarekonfiguration ihres Notebooks.

Möchten die Schüler von zuhause aus arbeiten (Hausübung, Fernlehre, privates Interesse, etc.) können sie das „normale“ Betriebssystem starten. Welches dies nun sein könnte ist an sich unerheblich, wobei sich wirtschaftlich gesehen vermutlich ein freies/kostenloses Linuxsystem anbieten würde. Die Schule muss einen Fernzugriffspunkt zur Verfügung stellen und schon kann auf die virtuelle Umgebung zugegriffen werden. Dabei ist es technisch überhaupt kein Problem alle Funktionen und Werkzeuge (starten von virtuellen Maschinen, etc.) auch außerhalb der Schule zu nutzen.

Wird das Notebook im EDV Saal verwendet sollte dieses in einen ThinClient „umfunktioniert“ werden. Wie früher schon erwähnt ist die einzige Anforderung die Fähigkeit mittels PXE über eine Netzwerkschnittstelle zu booten. Dies muss lediglich vor dem Kauf in den Anforderungen aufscheinen, und sollte kein weiteres Problem darstellen, da dies ohnehin von fast jedem gängigen Notebook Modell unterstützt wird. Viele BIOS bieten auch die Option beim Booten das so genannte Boot Device auszuwählen, und somit ist es nicht nötig, den Netzwerkboot standardmäßig zu aktivieren.

Gleiches gilt natürlich auch für den Einsatz im Klassenraum. Sofern PXE Server vorhanden und netzwerktechnisch erreichbar sind, kann das Notebook immer als ThinClient verwendet werden.

Dadurch ist jegliche „Gefahr“ von unkontrollierten Notebooks gebannt. Was auch immer am Schülergerät passiert ist, ob das Betriebssystem noch funktioniert oder nicht, ist vollkommen unerheblich, da auf die eigentliche Festplatte nicht mehr zugegriffen wird. Ein weiterer Vorteil ist, dass es vollkommen unerheblich ist wo der Schüler sich mit dem Netzwerk verbindet. Der Desktop wird auf Benutzerebene generiert und bietet das normale Arbeitsumfeld.

Neben der Netzwerksicherheit und der höheren Verfügbarkeit der Unterrichtsmedien (in diesem Fall der Notebooks) ergibt sich noch ein

weiterer positiver Aspekt. Nachdem die Notebooks im schulischen Einsatz nur als „Schnittstelle“ zur virtuellen Infrastruktur dienen, sei es nun als ThinClient oder durch natives OS, sind die Hardware Anforderungen sehr gering was in weiterer Folge weniger Anschaffungskostenbedeutet.

6.2 unterschiedliche Betriebssysteme im Unterricht

In technisch orientierten Bildungseinrichtungen ist die Verwendung von unterschiedlichen Betriebssystemen oftmals unumgänglich. Die Motivation dahinter ist dabei durchaus unterschiedlicher Natur allerdings oftmals von Lehrplänen und Unterrichtszielen getrieben.

So sind im Lehrplan von Informationstechnologie HTLs zwei Unterrichtsstunden „Betriebssysteme und Computerarchitektur“ im zweiten Jahrgang sowie vier Unterrichtseinheiten im dritten Jahrgang im Lehrplan vorgeschrieben. Der Lehrplan sieht nun vor, dass Schüler nach Abschluss der Lehrveranstaltung über folgende Kenntnisse und Fähigkeiten verfügen: [BMU09]

1. Aufbau und Wirkungsweise von Computersystemen und Peripheriegeräten kennen [BMU09]
2. Systembusse innerhalb eines Computersystems sowie die Schnittstellen zum Datenaustausch zwischen Computern und Peripheriegeräten kennen [BMU09]
3. Konzepte moderner Betriebssysteme verstehen können [BMU09]
4. Unterschiedliche Betriebssysteme kennen, diese installieren, warten und bedienen können [BMU09]
5. Häufig wiederkehrende Administrationsaufgaben automatisieren können [BMU09]
6. Imstande sein, die Tauglichkeit von Betriebssystemen für verschiedene Einsatzgebiete zu beurteilen [BMU09]

Diese sechs Bildungs- beziehungsweise Lehraufgaben stellen nun eine direkte Anforderung an die schulische IT-Infrastruktur. Zwar können Punkt 1,

2 und 3 auch eher theoretisch abgehandelt werden, dies lässt sich aber bei den Punkten 4-6 nicht mehr glaubhaft argumentieren.

Speziell Punkt 4 verlangt unterschiedliche Betriebssysteme und einen versierten Umgang mit diesen. Dies lässt sich durch rein theoretische Abhandlungen nicht mehr verwirklichen. Und genau dieser Punkt verlangt nun von den Informatiksälen die Verfügbarkeit der unterschiedlichen Betriebssysteme.

Aber auch Punkt 5 und 6, die Automatisierung wiederkehrender Aufgaben beziehungsweise die Evaluierung von verfügbaren und gängigen Betriebssystemen, setzen die Verfügbarkeit dieser voraus.

All diese Anforderungen haben gemeinsam, dass das Betriebssystem innerhalb einer Lehrveranstaltungsdauer, wenn auch auf über ein ganzes Semester verteilt, wechseln kann und muss.

Auch bietet sich bei den vorgegeben Lehrzielen an, den praktischen Teil des Unterrichtes mit Form von kleinen Projekten zu gestalten. Daraus würden sich unter Anderem folgende Vorteile ergeben:

- interdisziplinären Aspekt in Kombination mit Projektmanagement (ebenfalls vorgesehen im dritten Jahrgang)
- praktischeren Bezug zum Berufsalltag
- Gruppenarbeiten fördern die Klassendynamik und die soziale Kompetenz der Schüler
- Schüler müssen eigenständig arbeiten und lernen Verantwortung zu übernehmen
- Durch Gruppenarbeiten können wesentlich umfangreichere Themen behandelt werden

Soweit würde sich dieses Unterrichtskonzept auch ohne Virtualisierung planen lassen. Der Knackpunkt ist allerdings, dass Projektarbeit per Definition nicht in zwei Unterrichtsstunden abgeschlossen sein kann.

Abhängig von der Aufgabe ist es nun aber unter Umständen nicht möglich, dass die von den Projektteams benutzte Hardware sowie die darauf laufende Softwarekonfiguration von anderen Schülern benutzt werden kann. Somit würde dies eine enorme Anzahl an verfügbarerer Hardware verlangen, die in der Woche für zirka zwei Stunden benutzt wird. Auch ist die Option Teile des Projektes als Hausaufgabe zu geben nur bedingt gegeben. Bauen die Schüler ihr Projekt in der Schule, wird sich auch die Hardware dort befinden. Ist somit ein Weiterarbeiten am Projekt außerhalb der dafür vorgesehen Schulstunden gewünscht, muss den Schülern auch außerhalb des Unterrichts der Zugang ermöglicht werden.

Genau dieses Dilemma bringt nun das in Kapitel 4 entworfene und in Kapitel 5 realisierte Konzept ins Spiel. Denn durch den Einsatz einer virtuellen Umgebung lässt sich diese Anforderung sehr dynamisch bewältigen. Das Generieren von virtuellen Instanzen kann weitgehend automatisch erfolgen, und durch die Verwendung des ThinClients können auch mehrere Verbindungen zu unterschiedlichen Virtuellen Instanzen parallel aufgebaut werden. Diese können dann auch noch untereinander virtuell vernetzt werden. Der Vorteil liegt hier klar auf der Hand. Durch die sinnvolle Verwendung von Virtualisierung lassen sich selbst komplexe Szenarien einfach und ohne zusätzliche Hardware realisieren. Das System wächst sozusagen mit der Anforderung.

Auch ist es durchaus möglich, dass mehr als ein System zur gleichen Zeit verwendet werden muss. Abseits von der Betriebssystemauswahl, werden auch immer wieder unterschiedliche Applikationen verwendet, welche unter Umständen auch andere Betriebssysteme verlangen.

Das folgende Beispiel soll die nun erkannten Zusammenhänge anschaulich zusammenfassen. Die folgende Aufgabenstellung könnte an Schüler gegeben werden:

Es soll ein Webserver aufgebaut werden welcher die Benutzer durch ein LDAP-Verzeichnis authentisiert. Zwischen den Komponenten (LDAP, Webserver, Client) soll eine Firewall den Verkehr regeln.

Lösungsansatz

Zwar könnte dies auch auf einer Maschine gebaut werden, allerdings lässt sich der Verkehr dann schwer durch eine Firewall regeln. Auch kann das zugrunde liegende Netzwerk sowie die nötigen Kommunikationsströme nicht erklärt und erfasst werden. Dies ist allerdings wichtig um den Schülern das Zusammenspiel mehrer Komponenten zu verdeutlichen.

Um diese Aufgabenstellung nun praxisnah zu bauen werden de facto drei Computer benötigt. Zwei Server (LDAP und Webserver) und ein Client. Dem traditionellen Ansatz folgend würde jeder Schüler somit drei physikalische Computer benötigen. Zwar sind die Anforderungen an die Hardware hier nicht besonders hoch, aber dennoch muss sie verfügbar sein. Weiters ist es nötig diese Laborumgebung zu pflegen beziehungsweise unterschiedlichen Schülern und auch Klassen zur Verfügung zu stellen. Auch lässt sich so ein Projekt nicht in einem Schultag lösen und daher müssen die Zwischenergebnisse bis zur nächsten Schulstunde aufgehoben werden. Ein weiteres Problem ist, dass die Schüler eventuell auch Teile der Implementierung als Hausübung machen könnten oder sollten. Um dies zu schaffen, müssten nun alle drei Computer auch irgendwie über das Internet erreichbar gemacht werden.

Angenommen es wären zwei Klassen mit jeweils 20 Schülern, so würden alleine 40 Schüler drei Computer brauchen, was letztlich 120 Computer zumindest für die Dauer des Projektes „blockieren“ würde.

Durch den Einsatz einer Virtuellen Umgebung lässt sich dieses Problem auf einem Schlag lösen. Jeder Schüler bekommt drei virtuelle Instanzen, mit der schon bestehenden kann auf die zusätzlichen zugegriffen werden. Nachdem

sich das Projekt auf die Implementierung eines Webservers und eines LDAP-Verzeichnisses, sowie der Interaktion dazwischen beschränkt ist das Aufsetzen des Betriebssystems nicht nötig und kann daher ausgelassen werden. Somit bekommt jeder Schüler ein fertiges virtuelles Betriebssystem und kann sofort mit der eigentlichen Arbeit loslegen. Die Betriebssysteme sind vorgefertigt und die Dateien werden einfach vervielfältigt. Für den Lehrer oder IT-Verantwortlichen ist hier kein zusätzlicher Aufwand nötig, also keine Installation des Betriebssystems, keine Treiber und Hardware Konfiguration.
Sollte ein Schüler sein Betriebssystem derart beschädigen, dass es nicht mehr funktionsfähig ist, kann dies einfach ersetzt werden. Eine zeitintensive Neuinstallation ist nicht nötig!

Arbeiten in der Schule

Hierbei kommt der ThinClient beziehungsweise ein privates Notebook als ThinClient verwendet zum Einsatz. Dieser stellt wiederum die Verbindung zu den Virtuellen Instanzen zur Verfügung.

Arbeiten zu Hause

Schüler können sich über das Internet mit der Schule verbinden und ihre virtuellen Maschinen starten. Die Arbeit erfolgt de facto wie in der Schule.

Vorteile

Da das „Grundsystem“ jederzeit sehr einfach und schnell wiederhergestellt werden kann und dafür keine besonderen IT-Kenntnisse vorhanden sein müssen kann experimentiert werden, da selbst wenn das ganze Betriebssystem unbrauchbar wird dies keinen wesentlichen Einfluss auf den weiteren Betrieb hat. Auch ist es nicht nötig, den Schülern bestimmte Zugriffsrechte zu nehmen oder bestimmte Aktionen von ihnen zu verhindern.

Durch das Wegfallen von zeitintensiven Wartungsarbeiten beziehungsweise von Versuchen die Arbeitsumgebung aufzubauen können Schüler sowie Lehrer sich wesentlich besser auf die eigentlichen Aufgaben und Lehrziele konzentrieren.

Jeder Schüler kann seine virtuelle Maschine auch von zuhause aus erreichen und hat somit auch außerhalb des Schulgebäudes die gleichen Vorraussetzungen und Gegebenheiten. Somit können Teile der Gesamtaufgabe auch als Hausübungen oder Fernlehre gegeben werden.

vor allem die Bewertung von komplexeren Projekten führt Lehrer oft vor eine schwierige Aufgabe, da diese oft nur im „Ganzen“ begutachtet werden können. Die Ausarbeitung und Dokumentation der Schüler mag zwar oft alle Elemente behalten, doch diese kann natürlich auch von Schulkollegen stammen. Durch die Verwendung von virtuellen Instanzen können auch Lehrer das Gesamtresultat in Aktion sehen und zwar ohne dabei in der Schule sein zu müssen. Dies kann dann ähnlich wie schriftliche Hausübungen betrachtet werden, welche auch an beliebigen Orten ausgewertet werden können.

Werden Teile einer Aufgabenstellung als Hausübung oder Fernlehre behandelt, stellt sich die Frage wie Lehrer hierbei sinnvoll helfen können. Beim Einsatz von virtuellen Maschinen ist dies sehr einfach, da auch Lehrer Zugriff darauf haben. Dies führt soweit, dass Lehrer und Schüler den gleichen Bildschirm sehen können und gemeinsam an einem Problem arbeiten. Dies ist speziell im Bereich der Distance Education interessant.

Oftmals gibt es mehr als nur einen Weg eine Aufgabe zu lösen und abhängig von der gestellten Aufgabe hat dies mehr oder weniger Auswirkungen auf das Betriebssystem beziehungsweise die Applikationen. So kann es durchaus sein, dass zwei unterschiedliche Lösungsszenarien jeweils ein „frisches“ Betriebssystem erfordern. Durch den virtuellen Ansatz stellt dies

kein Problem dar, da die einzige Abhängigkeit die Anzahl beziehungsweise Kapazitäten der KVM-Server ist.

Wie schon erwähnt fordert praktischer, abwechslungsreicher und experimentierfreudiger Unterricht oft mehr als nur einen physikalischen Computer. Durch mangelnde Ressourcen ist dieser leider deshalb dann nicht durchführbar. Als Probleme könnte hier erheblicher Wartungsaufwand und natürlich auch der finanzielle Aspekt genannt werden. Durch den Einsatz von Virtualisierung lassen sich beide Probleme elegant und einfach lösen und durch den modularen Aufbau des Konzeptes kann die gesamte „virtuelle Welt“ der Schule mit den Anforderungen wachsen.

7. FAZIT/SCHLUSSWORT

Die Vorteile, wie Kostenersparnis, Flexibilität, Transparenz, Skalierbarkeit und homogene Systeme, die sich durch Virtualisierung ergeben lassen sich nicht leugnen. Die vergleichbar geringen Nachteile, welche letztlich immer auf anfangs fehlendes Know-How sowie mangelnde Erfahrung zurückzuführen sind, sind dabei vergleichsweise gering und vernachlässigbar. Zumal speziell diese Nachteile durch den Aufbau virtueller Systeme sehr schnell relativiert werden können da dann die Lücken zwangsläufig geschlossen werden.

Es hat sich herausgestellt, dass Schulen durchaus Bedürfnisse haben die durch den Einsatz von traditionellen Computersystemen nur teilweise zu erfüllen sind, aber durch die Verwendung von Virtualisierung befriedigt werden können. Sei dies nun die einfache Wartung, die Flexibilität oder auch das zur Verfügung stellen von unterschiedlichen Betriebssystemen.

7.1 Reflexion der Ziele

Kostengünstig

Kostenersparnisse sind weitläufig das beste und am häufigsten genannte Argument der Virtualisierungsvertreter. Dies geht auch aus der Grafik in Kapitel 2.5 hervor. Auch das exemplarisches Beispiel in Kapitel 4.4 hat dies bestätigt.

Flexibilität

Auch dieser Punkt wurde in Kapitel 2.5 schon als Vorteil der Virtualisierung genannt. Auch sieht das Konzept die Bereitstellung unterschiedlicher

Basiskonfigurationen vor, um möglichst viele verschiedene Anwendungsfälle abzudecken. Zum Beispiel, Programmieren, Office Anwendungen, Betriebssysteme, Netzwerktechnik, und so weiter. Sollte eine solche Musterkonfiguration fehlen, braucht diese nur einmal erstellt werden und kann beliebig oft dupliziert werden. Auch es ist es völlig unerheblich ob spezielle Applikationen nun Microsoft Windows oder Linux voraussetzten, da beides schnell verfügbar sein kann.

Aktiver, kreativer und experimenteller Umgang mit IT

Experimentieren ist in traditionellen IT-Systemen zwangsläufig mit einem sehr hohen Wartungsaufwand verbunden. In einer virtuellen Umgebung ist dies allerdings stark relativiert, da der alte Zustand jederzeit sehr leicht wiederhergestellt werden kann.

Einfache Wartung

Das Konzept ist homogen aufgebaut und beinhaltet klar definierte Managementschnittstellen. Sämtliche Operative Aufgaben, wie hinzufügen oder löschen von Benutzern/virtuellen Betriebssystemen, werden nur einmal getätigt und gelten für die gesamte Infrastruktur.

7.2 Ausblick

Im Zuge der Arbeit wurden unterschiedliche Methoden und Ansätze getestet um letztlich ein Konzept vorzulegen das es ermöglicht relativ einfach zu einer strukturierten und zukunftsorientierten virtuellen Infrastruktur zu gelangen. Die Ansätze die im Moment noch nicht zum Erfolg führen werden dies womöglich mit weiterentwickelter Technologie schaffen und räumen dann womöglich noch die letzten Hürden aus dem Weg.

Anhang/Ergänzende Informationen

ANHANG

Dispatch Skript

```
#!/usr/bin/perl
#
#
#       KVM Dispatch Unit, Version 1 Subversion 0
#
#

# read in all provisioned KVM Servers
open ( kvm_server_list, "</etc/kvm_server_list ");
while (<kvm_server_list>)
        {
        $_ =~ s/\012//sg;
        push ( @kvm_server_list_array,$_);
        }
close(kvm_server_list);
$kvm_servers = @kvm_server_list_array;



# check all KVM servers and push amount of virtual instances to another
array
for ($i=0;$i<@kvm_server_list_array;$i++)
        {
        $kvm_list    =    `ssh    @kvm_server_list_array[$i]    virsh    -c
qemu:\/\/\/system list`;
        push         (         @kvm_server_list_array_VMcount,         `ssh
@kvm_server_list_array[$i] virsh -c qemu:\/\/\/system list | wc -l`);
        }



# find Array position with the least amount of started virtual maschines
$min_pos = 0;

for ($i=0; $i<@kvm_server_list_array_VMcount; $i++)
        {
        if               (@kvm_server_list_array_VMcount[$i]               <
@kvm_server_list_array_VMcount[$min_pos])
                {
                $min_pos = $i;
                }
        }



# copy VM configuration file and start VM
system("scp                                                    /@ARGV[0]/@ARGV[1]
@kvm_server_list_array[$min_pos]:/opt/startup/");

system(" ssh @kvm_server_list_array[$min_pos] virsh -c qemu:///system create
/opt/startup/@ARGV[1] ");
```

Webschnittstelle zum Starten virtueller Maschinen

index.html

```
<html>
   <head>
    <title>VM Starten</title>
   </head>
   <body>

<form action="script.php" method="post">
    <fieldset>
        <legend>Logindaten eingeben</legend>
        <label>Benutzername: <input type="text" name="Username" /></label>
        <label>Password: <input type="text" name="Pass" /></label>
        <input type="submit" name="formaction" value="Einloggen" />
    </fieldset>
</form>

   </body>
 </html>
```

script.php

```
<?
        $daten = $_POST['Username'];
        $datei = fopen("/opt/admin/temp","w");


        $cmdi    =    "ldapsearch    -h    192.168.1.15    -p    10389    -b
dc=openthinclient,dc=org -x cn=$daten | grep -i description";
        $output = shell_exec($cmdi);
        $VMinfo = $output;
        $array = explode ('##', $VMinfo);
//        echo "<pre>$output</pre>";

        fwrite($datei,$output);

        fclose($datei);

echo "<form action=\"script1.php\" method=\"post\">";
echo "    <fieldset> ";
echo "        <fieldset> ";
echo "            <legend>Welche VM soll gestartet werden?</legend> ";
for ($i=0;$i<=count($array)-2;$i++) {
echo  "                        <label><input  type=\"checkbox\"  name=\"VM[]\"
value=\"$daten $array[$i]\" /> $array[$i]</label> <br> ";
}
echo "        </fieldset> ";
echo "         <input type=\"submit\" name=\"formaction\" value=\"Starten\"
/> ";
echo "    </fieldset> ";
echo "</form> ";

?>
```

script1.php

```
<?php
$VM_daten = $_POST['VM'];
$VM_daten_count = count ( $VM_daten);
```

```
for ($i=0;$i<$VM_daten_count;$i++)
{
$line = explode ( ' ',$VM_daten[$i]);

$datei = fopen ( "/opt/admin/startKVM_$line[2]", "w");
echo " generiere KVM Kickstart File /opt/admin/startKVM_$line[2]";

fwrite ($datei,  "<domain type='kvm'>\n");
fwrite ($datei,  "  <name>$line[0]_$line[2]</name>\n");
fwrite ($datei,  "  <memory>524288</memory>\n");
fwrite ($datei,  "  <currentMemory>524288</currentMemory>\n");
fwrite ($datei,  "  <vcpu>1</vcpu>\n");
fwrite ($datei,  "  <os>\n");
fwrite ($datei,  "    <type>hvm</type>\n");
fwrite ($datei,  "    <boot dev='hd'/>\n");
fwrite ($datei,  "  </os>\n");
fwrite ($datei,  "  <features>\n");
fwrite ($datei,  "    <acpi/>\n");
fwrite ($datei,  "  </features>\n");
fwrite ($datei,  "  <clock offset='utc'/>\n");
fwrite ($datei,  "  <on_poweroff>destroy</on_poweroff>\n");
fwrite ($datei,  "  <on_reboot>restart</on_reboot>\n");
fwrite ($datei,  "  <on_crash>destroy</on_crash>\n");
fwrite ($datei,  "  <devices>\n");
fwrite ($datei,  "    <emulator>/usr/bin/qemu-kvm</emulator>\n");
fwrite ($datei,  "    <disk type='file' device='disk'>\n");
fwrite ($datei,  "      <source file='$line[4]'/>\n");
fwrite ($datei,  "      <target dev='hda'/>\n");
fwrite ($datei,  "    </disk>\n");
fwrite ($datei,  "    <interface type='bridge'>\n");
fwrite ($datei,  "      <mac address='$line[6]'/>\n");
fwrite ($datei,  "      <source bridge='br0'/>\n");
fwrite ($datei,  "    </interface>\n");
fwrite ($datei,  "    <input type='mouse' bus='ps2'/>\n");
fwrite   ($datei,       "              <graphics    type='vnc'    port='-1'
listen='127.0.0.1'/>\n");
fwrite ($datei,  "  </devices>\n");
fwrite ($datei,  "</domain>\n");
fclose ( $datei );
shell_exec ("dispatch /opt/admin startKVM_$line[2]");
}

echo " Dateien generiert \n";

?>
```

Duplizieren von Musterbetriebssystemen

index.html

```
<html>
   <head>
    <title>Hallo-Welt-Beispiel</title>
   </head>
   <body>

<form action="script.php" method="post">
    <fieldset>
        <legend>Logindaten eingeben</legend>
        <label>Benutzername: <input type="text" name="Username" /></label>
        <input type="submit" name="formaction" value="Einloggen" />
    </fieldset>
</form>

   </body>
 </html>
```

script.php

```
<?
        $daten = $_POST['Username'];
        $data_os = "/etc/OS_templates";

        $myarg = "/bin/update_OS_templates";
        shell_exec($myarg);
        $datei = fopen($data_os,"r");

//      while (!feof($datei))
//      {
//      $line = fgets ($datei, 100);
//      echo "$line";
//      }

echo "<form action=\"script1.php\" method=\"post\">";
echo " <h1> ACHTUNG: Das reproduzieren von image Dateien kann bis zu
mehreren Minuten dauern, Browser nicht bee
nden </h1>";
echo "    <fieldset> ";
echo "        <fieldset> ";
echo "            <legend>Welches image soll kopiert werden</legend> ";
while (!feof($datei))
        {
        $line = fgets ($datei, 100);
        if (strlen($line)>0) {
        echo "                  <label><input type=\"radio\" name=\"VM[]\"
value=\"$daten $line\" /> $line</label> <br> ";
        }
        }
echo "  <label>MAC Adresse: <input type=\"text\" name=\"mac\" /></label>";
echo "        </fieldset> ";
echo "         <input type=\"submit\" name=\"formaction\" value=\"Starten\"
/> ";
echo "    </fieldset> ";
echo "</form> ";

?>
```

script1.php

```
<?php

$VM_daten = $_POST['VM'];
$MACa = $_POST['mac'];
$datei = fopen ( "/tmp/addlinetoldap", "w");


$store_dir = "/vm/OS_templates";
$stud_img_dir = "/vm/si";

$line = explode (' ',  $VM_daten[0]);

// line[0] ist LDAP NAME
// line[1] ist zu kopierendes image

$arg       =       "ssh       root@data01       cp       $store_dir/$line[1]
$stud_img_dir/$line[1]_$line[0]";
shell_exec ( $arg );

fwrite                    ($datei,                                          "dn:
cn=$line[0],ou=users,ou=openthinclient,dc=openthinclient,dc=org\n");
fwrite ($datei,  "changetype: modify\n");
fwrite ($datei,  "add: description\n");
fwrite ($datei,  "description: $line[0] # $stud_img_dir/$line[1]_$line[0] #
$MACa ##\n");

$arg  =  "ldapmodify   -h  192.168.1.15  -p  10389  -f  /tmp/addlinetoldap  -D
\"uid=admin,ou=System\" -w secret -x";
shell_exec ( $arg );
//shell_exec (ldapmodify  -h 192.168.1.15 -p 10389 -f /tmp/addlinetoldap -D
"uid=admin,ou=System" -w secret -x);

echo "\n\nkopievorgang beendet!!\n\n";

?>
```

ABBILDUNGSVERZEICHNIS

Abb. 2-1: Virtualisierung in Java [PICH09] 8
Abb. 2-2: Emulation [PICH09] 9
Abb. 2-3: Systemvirtualisierung [PICH09] 10
Abb. 2-4: Hypervisor Typ-1 [PICH09] 11
Abb. 2-5: Hypervisor Typ-2 [PICH09] 11
Abb. 2-6: XEN Architektur [CHAG01] 15
Abb. 2-7: Schematischer Aufbau von KVM [KVM06] 17
Abb. 2-8: Nachteile der Virtualisierung (Quelle: RAAD) 18
Abb. 2-9: Vorteile der Virtualisierung (Quelle: RAAD) 20
Abb. 3-1: PXE Startprozess [INTEL99] 37
Abb. 4-1: ThinClient starten 45
Abb. 4-2: zu startendes virtuelles Betriebssystem auswählen 46
Abb. 4-3: Virtuelles Betriebssystem starten 47
Abb. 4-4: Verbindung zu virtuellen Betriebssystem 48
Abb. 4-5: Interaktion der Komponenten 49
Abb. 4-6: Virtuelles Betriebssystem hinzufügen 50
Abb. 4-7: Beispiel Architektur bei maximalem Ausbau 52
Abb. 4-8: Kosten in Abhängigkeit der Arbeitsplätze 55
Abb. 4-9: Virtualisiert/nicht virtualisiert/mit bestehender HW in Abhängigkeit der Anzahl 56
Abb. 4-10: Kosten Stromverbrauch in Abhängigkeit der Computeranzahl 59
Abb. 4-11: Kosten Stromverbrauch in Abhängigkeit der Computeranzahl (ZOOM) 60
Abb. 5-1: Übersicht der Beispielimplementierung 64
Abb. 5-2: Struktogramm Dispatch 73
Abb. 6-1: Didaktisches Dreieck (vgl. [KRON04]) 77
Abb. 6-2: Didaktisches Viereck (vgl. [KRON04]) 78

TABELLENVERZEICHNIS

Tabelle 4-1: Stückliste Thinclients 54
Tabelle 4-2: Stückliste Server Infrastruktur 54
Tabelle 4-3: Stückliste Arbeitsplatzcomputer 55
Tabelle 4-4: Stromverbrauch pro Virtualisierungskomponente 57
Tabelle 4-5: Stromverbrauch pro nicht virtualisiert 58
Tabelle 4-6: Laufzeit der Komponenten 58
Tabelle 5-1: verwendete Software 64
Tabelle 6-1: Medienklassifizierung [KRON04] 77

LITERATURVERZEICHNIS

Bücher:

[PICH09] H. J. Picht, XEN Kochbuch, 1. Auflage, O'Reilly Verlag GmbH &CO. KG, Köln, 2009

[CHAG07] P. Chaganti, Xen Virtualization, 1. Auflage, Packt Publishing Ltd., Olton, 2007

[SMIT05] J. E. Smith, Virtual Machines – Versatile Platforms for Systems and Processes, Elsevier Ltd, Oxford, 2005

[KRON04] F. W. Kron, Grundwissen Didaktik, 4. Auflage, Ernst Reinhardt Verlag München Basel, 2004

[VIRT09] Virtuatopia.com, Xen Virtualization Essentials, First Edition, Virtuatopia

Studien:

[SCHW06] G. Schwed, Linux an Schulen, Donauuniversität Krems, Krems, 2006

[KNER08] C. Knermann, Ökologischer Vergleich der Klimarelevanz von PC und Thin Client ARbeitsplatzgeräten 2008, Frauenhofer Institut für Umwelt-, Sicherheits- und Energietechnik UMSICHT, Oberhausen, 2008

Proceedings:

[KIVI07] A. Kivity, kvm: the Kernel-based Virtual Machine for Linux, Proceedings of the Linux Symposium, Ottawa, 2007

[TIROL02] IT-Konzept für Tiroler Schulen, 2002

Article:

[BURG08] T.W. Burger, Intel® Virtualization Technology for Directed I/O (VT-d): Enhancing Intel platforms for efficient virtualization of I/O devices, Intel, 2008

[ABRA06] D. Abramson, Intel® Virtualization Technology for Directed I/O, Intel® Technology Journal, Volume 10 2008

Lehrplan:

[BMU09]
Lehrplan der höheren Lehranstalt für Informationstechnologie
http://www.htl.at/fileadmin/content/Lehrplan/HTL_Informationstechnologie.pdf
letzter Aufruf, 15.07.2009

Weblinks:

[Link01]
http://rmconsult.biz/news/20090423/vorteile_und_nachteile_bei_der_servervirtualisierung/
letzter Aufruf: 24.06.2009

[Link02]
http://gonline.univie.ac.at/htdocs/site/browse.php?artiid=2819&arttyp=k
letzter Aufruf, 28.06.2009

[QEMU09]
http://www.qemu.org/qemu-doc.html#SEC1
letzer Aufruf: 01.07.2009

[INTEL99]
http://www.pix.net/software/pxeboot/archive/pxespec.pdf
letzter Aufruf: 28.06.2009

[VMWARE09a]
https://www.vmware.com/products/product_index.html
letzter Aufruf: 01.07.2009

[VMWARE09b]
http://www.vmware.com/files/pdf/vmware_esxi_datasheet.pdf
letzter Aufruf: 01.07.2009

[VMWARE09c]
https://www.vmware.com/files/pdf/server_datasheet.pdf
letzter Aufruf: 01.07.2009

[VMWARE09d]
VMWare Server User Manual
https://www.vmware.com/pdf/vmserver2.pdf
letzter Aufruf: 01.07.2009

[XEN08]
http://bits.xensource.com/Xen/docs/user.pdf
letzter Aufruf: 01.07.2009

[KVM06]
http://www.qumranet.com/files/white_papers/KVM_Whitepaper.pdf
letzter Aufruf: 01.07.2009

[CentOS09]
http://centos.org/modules/tinycontent/index.php?id=3
letzter Aufruf: 01.07.2009

[INTEL09a]
Intel® 4 Series Chipset Family Datasheet
http://www.intel.com/Assets/PDF/datasheet/319970.pdf
letzter Aufruf: 01.07.2009

[INTEL09b]
Intel® Q35 and Q33 Express Chipsets Product Brief
http://www.intel.com/Assets/PDF/prodbrief/317312.pdf
letzter Aufruf: 01.07.2009

[INTEL09c]
Intel® Core™2 Duo Desktop Processor Family
http://ark.intel.com/ProductCollection.aspx?familyID=26547
letzter Aufruf: 01.07.2009

[INTEL08a]
Intel® Desktop Board DQ45CB Product Brief
http://www.intel.com/Assets/PDF/prodbrief/320268.pdf
letzter Aufruf: 01.07.2009

[HP09a]
HP ThinClient t5145 Datasheet
http://h20195.www2.hp.com/v2/GetPDF.aspx/4AA2-1037DEE.pdf
letzer Aufruf: 05.07.2009

[HP09b]
HP ProLiant DL380 G5 Serverserie - Spezifikationen
http://h10010.www1.hp.com/wwpc/de/de/sm/WF06a/15351-15351-3328412-241475-241475-1121516.html

letzer Aufruf: 05.07.2009

[HP09c]
HP ProLiant DL120 G5 Serverserie Spezifikationen
http://h10010.www1.hp.com/wwpc/de/de/sm/WF06b/15351-15351-3328412-3328421-3328421-3683232-3940957.html?jumpid=reg_R1002_DEDE
letzer Aufruf: 05.07.2009

[HP09d]
HP StorageWorks 600 Modular Disk System
http://h20195.www2.hp.com/v2/GetPDF.aspx/c01724593.pdf
letzer Aufruf: 05.07.2009

[HP09e]
HP Compaq dc5700 Small Form Factor PC - Spezifikationen
http://h10010.www1.hp.com/wwpc/de/de/sm/WF06a/12454-12454-64287-3328896-3328896-3249646.html
letzer Aufruf: 05.07.2009

[IETF97]
Dynamic Host Configuration Protocol
http://tools.ietf.org/html/rfc2131
letzter Aufruf: 05.07.2009