

**Universität Leipzig  
Fakultät für Mathematik und Informatik  
Institut für Informatik**

**Virtualisierung und Cloning  
von Betriebssystemen unter z/VM**

**Diplomarbeit**

Leipzig, Februar 2009

vorgelegt von  
Georg Müller  
Studiengang Informatik

**Betreuender Hochschullehrer: Prof. Dr.-Ing. Wilhelm G. Spruth  
Fakultät für Mathematik und Informatik  
Institut für Informatik  
Abteilung Computersysteme**

## Kurzzusammenfassung

Diese Diplomarbeit beschäftigt sich mit der Virtualisierung und dem Cloning von Betriebssystemen auf *IBM System z* Großrechnern unter *z/VM*. Dieses Betriebssystem ist das älteste, technisch modernste und ausgereifteste für die Virtualisierung von Betriebssystemen (Plattform-Virtualisierung). Ziel der Arbeit ist es, *z/OS* und *Linux* unter *z/VM* zu betreiben und einfache Mechanismen bereitzustellen bzw. zu beschreiben, um weitere Systeme auf Basis eines vorhandenen Systems zu erstellen.

Die Arbeit beschäftigt sich mit einer komplexen und umfangreichen Installation von *z/VM* auf dem *IBM z900*-Großrechner der Universität Leipzig. Dabei wurde eine spezifische Konfiguration realisiert.

In einer bereits bestehenden *z/VM*-Installation, wurde ein zusätzliches *z/VM* als Gast-system in Betrieb genommen (Second Level *z/VM*), das als Plattform für weitere Gast-systeme und das Cloning dient. Dieses System wurde an das bereits vorhandene interne Netzwerk angeschlossen. Für die virtuellen Systeme, die innerhalb dieser Installation betrieben werden, wurde ein separates internes Netzwerk erstellt, in dem das Second Level *z/VM* als Router dient.

Innerhalb dieses Second Level *z/VM* wurden Gast-systeme in Betrieb genommen und anschließend für das Cloning neuer Instanzen vorbereitet. Als Gast-systeme wurden zum Einen *z/OS 1.8*, zum Anderen *SuSE Linux Enterprise Server (SLES) 10* eingerichtet.

Damit ist eine Konfiguration vorhanden, die für künftige Diplomarbeiten oder Praktika im Rahmen der akademischen Lehre einen Mehrwert gegenüber der bisherigen Installation bietet. Durch das neue System ist es möglich, einem Studenten die vollständige Kontrolle über ein eigenes Großrechner-Betriebssystem zu geben, ohne dass andere Installationen dadurch möglicherweise beeinträchtigt werden. Bisher war dies nicht möglich, da alle Studenten auf dem selben System gearbeitet haben und damit ein Fehler bei der Administration den Betrieb für alle Studenten stören konnte. Mit dem erstellten System können nun Übungen durchgeführt werden, um zum Beispiel einen Systemstart (IPL) von *z/OS* durchzuführen, eine *z/OS*-Installation zu konfigurieren oder einen *z/OS Master Catalog* anzulegen und zu bearbeiten.

## Danksagung

Ich danke Herrn Prof. Spruth für seine Betreuung und Unterstützung bei der Erstellung der Diplomarbeit.

Herrn Uwe Denneler und Frau Elisabeth Puritzscher vom IBM Entwicklungslabor in Böblingen danke ich für ihre Hilfe bei der Software-Installation. Vielen Dank auch an Herrn Lutz Kühner von der IBM System Technology Group für seine Hilfe bei spezifischen Problemen.

Ich danke Herrn Andreas Hermelink, Technical Sales Manager System z, für seine Unterstützung der Installation an der Universität Leipzig.

Herrn Dieter Begel von der Firma SYSeins GmbH sowie Herrn Uli Stormanns, Student der Fachhochschule Aachen, danke ich für ihre Arbeit bei der Erstellung der minimalen z/OS-Gäste.

Bei meiner Schwester Ulrike Müller und Herrn Jens Geier möchte ich mich herzlich für ihre wertvollen Hinweise und Tipps bedanken. Besonderer Dank gilt meinen Eltern und meiner Großmutter für ihre Unterstützung während des Studiums. Bei ihnen und meiner Freundin Beate Brysch möchte ich mich auch für ihre Geduld bedanken.

## Inhaltsverzeichnis

|       |  |    |
|-------|--|----|
| 1     | Einleitung.....                                      | 7  |
| 1.1   | Gegenstand.....                                      | 7  |
| 1.2   | Vorgehen.....  | 7  |
| 1.3   | Aufbau des Systems der Universität Leipzig.....      | 8  |
| 1.3.1 | Geschichte des Systems der Universität Leipzig.....  | 8  |
| 1.3.2 | Nutzung des Systems der Universität Leipzig.....     | 8  |
| 1.3.3 | Ausgangssituation.....                               | 9  |
| 2     | Grundlagen Großrechner und Virtualisierung.....      | 11 |
| 2.1   | Großrechner.....                                     | 11 |
| 2.1.1 | Heutige Bedeutung.....                               | 11 |
| 2.2   | Virtualisierung.....                                 | 12 |
| 2.2.1 | Vorteile der Virtualisierung.....                    | 12 |
| 2.2.2 | Cloning.....   | 13 |
| 2.2.3 | Plattform-Virtualisierung auf IBM z/Series.....      | 13 |
| 2.3   | Virtualisierungskonzepte.....                        | 14 |
| 2.3.1 | Virtualisierung von Betriebssystemen.....            | 14 |
| 2.3.2 | Virtualisierung von Ressourcen unter z/VM.....       | 16 |
| 3     | Das Betriebssystem z/VM.....                         | 18 |
| 3.1   | Beschreibung von z/VM.....                           | 18 |
| 3.2   | Konzepte von z/VM.....                               | 19 |
| 3.2.1 | CP.....  | 19 |
| 3.2.2 | Minidisks.....                                       | 20 |
| 3.2.3 | CMS.....   | 20 |
| 3.2.4 | Weitere Komponenten.....                             | 23 |
| 3.2.5 | z/VM User Directory.....                             | 23 |
| 3.3   | Administration von z/VM.....                         | 24 |
| 3.4   | Grundlegende Befehle zur Verwaltung des Systems..... | 24 |
| 3.4.1 | Formatieren einer DASD.....                          | 24 |
| 3.4.2 | Kopieren einer DASD.....                             | 27 |
| 4     | Einrichtung des Gast-VM-Systems.....                 | 31 |
| 4.1   | Zugriff auf das z/VM-System.....                     | 31 |

|  |    |
|--|----|
| Virtualisierung und Cloning von Betriebssystemen unter z/VM  | 5  |
| 4.1.1 Zugriff über Basis-z/VM-System.....                    | 32 |
| 4.1.2 Zugriff über OpenVPN.....                              | 34 |
| 4.2 Konfiguration des Gast-z/VM-Systems im Basis-System..... | 35 |
| 5 Basis-Konfiguration des Systems.....                       | 37 |
| 5.1 Das Netzwerk.....  | 37 |
| 5.1.1 Netzwerk-Konzepte unter z/VM.....                      | 37 |
| 5.1.2 Aufbau des Netzwerks.....                              | 39 |
| 5.1.3 Konfiguration des TCP/IP-Stacks.....                   | 40 |
| 5.2 DIRMAINT und DATAMOVE.....                               | 42 |
| 5.2.1 Konfiguration von DIRMAINT.....                        | 43 |
| 5.2.2 Die Festplatten.....                                   | 44 |
| 5.3 Gast-Betriebssysteme (Linux, z/OS) vorbereiten.....      | 45 |
| 5.3.1 Planungen für das Gastsystem.....                      | 45 |
| 5.3.2 Definition des Benutzers.....                          | 46 |
| 6 Linux als Gastsystem unter z/VM.....                       | 47 |
| 6.1 Linux installieren und einrichten.....                   | 47 |
| 6.1.1 Vorbetrachtungen.....                                  | 47 |
| 6.1.2 Definition des Systemverwalters LNXMAINT.....          | 47 |
| 6.1.3 Definition des Standard-Profiles für Linux-Gäste.....  | 48 |
| 6.1.4 Definition eines Prototypen.....                       | 48 |
| 6.1.5 Definition eines Linux-Gastsystem.....                 | 49 |
| 6.1.6 Installation des Systems.....                          | 51 |
| 6.1.7 Start des installierten Linux-Systems.....             | 61 |
| 6.2 Erstellung des Linux-Cloning-Systems.....                | 61 |
| 6.2.1 Anpassen des Linux-Master-Systems.....                 | 62 |
| 6.2.2 Anlegen eines neuen Gastes.....                        | 66 |
| 6.2.3 Kopieren des Systems.....                              | 67 |
| 6.2.4 Automatisierung des Starts.....                        | 67 |
| 7 z/OS als Gastsystem unter z/VM.....                        | 69 |
| 7.1 Voraussetzungen.....                                     | 69 |
| 7.1.1 Definition des Systems.....                            | 69 |
| 7.1.2 Einrichten der Netzwerkkarte.....                      | 71 |
| 7.1.3 Einrichtung des Netzwerks.....                         | 77 |
| 7.2 Weitere z/OS-Instanzen erstellen.....                    | 78 |

|  |          |
|--|----------|
| <u>Virtualisierung und Cloning von Betriebssystemen unter z/VM</u> | <u>6</u> |
| 7.2.1 Einfache z/OS-Kopie.....                                     | 78       |
| 7.2.2 Spezielle minimale z/OS-Instanzen.....                       | 79       |
| 8 Ausblick.....  | 82       |
| 8.1 Zusammenfassung.....   | 82       |
| 8.2 Bewertung.....   | 83       |
| 8.3 Weitere Themen.....  | 83       |
| 9 Abkürzungen und Glossar.....                                     | 85       |
| 10 Literaturverzeichnis.....                                       | 86       |
| 11 Abbildungsverzeichnis.....                                      | 88       |

# 1 Einleitung

## 1.1 Gegenstand

Virtualisierung ist eine Methodik, die in den letzten Jahren immer mehr an Bedeutung gewonnen hat. Sie verspricht eine Unabhängigkeit von Details der physischen Hardware und eine bessere Auslastung von Ressourcen, und hilft, Platz, Stromverbrauch und damit Kosten zu senken.

Diese Diplomarbeit beschäftigt sich mit *z/VM*, einem Betriebssystem für IBM-Großrechner der Serie *System z*, früher auch *z/Series* genannt. Die System *z*-Plattform stellt schon seit mehreren Jahrzehnten Virtualisierungsmöglichkeiten bereit, um effizient mehrere Betriebssysteme parallel auf derselben Hardware zu betreiben.

Diese Diplomarbeit ist eine praxisbezogene Arbeit, die sich mit dem an der Universität Leipzig eingesetzten System beschäftigt.

Ziel dieser Arbeit ist es, zu zeigen, wie Betriebssysteme unter *z/VM* eingerichtet und betrieben werden. Außerdem wird auf das so genannte Cloning von Betriebssystemen eingegangen, welches es auf schnelle und einfache Art und Weise ermöglicht, neue Betriebssystem-Instanzen anzulegen. Damit lassen sich beispielsweise mit wenig Aufwand neue Umgebungen für Praktika oder Diplomarbeiten erzeugen.

## 1.2 Vorgehen

In den Kapiteln 1 und 2 wird zunächst das Großrechner-System der Universität Leipzig und seine Geschichte vorgestellt, es folgen Grundlagen zu Großrechnern und Virtualisierung.

In Kapitel 3 wird das Betriebssystem *z/VM* vorgestellt, Kapitel 4 beschreibt die Einrichtung des *z/VM*-Gastsystems an der Universität Leipzig. In Kapitel 5 wird auf dessen Konfiguration eingegangen und es werden spezielle Anforderungen für den Betrieb des Cloning-Systems betrachtet.

Die Kapitel 6 und 7 beschäftigen sich mit dem Betrieb von Gast-Betriebssystemen unter *z/VM*. Dabei werden zunächst in Kapitel 6 Linux und anschließend in Kapitel 7 *z/OS* behandelt. In beiden Kapiteln wird jeweils zunächst auf den Betrieb eines Gastsystems eingegangen und anschließend das Cloning betrachtet, also die Anpassungen, die nötig

sind, um auf einfache Art und Weise weitere Gastsysteme zu erzeugen.

In Kapitel 8 findet eine Zusammenfassung statt sowie eine Betrachtung von Umfang und Grenzen dieser Diplomarbeit. Es wird ebenfalls ein Ausblick gegeben, womit sich zukünftige Diplomarbeiten beschäftigen können.

### **1.3 Aufbau des Systems der Universität Leipzig**

#### **1.3.1 Geschichte des Systems der Universität Leipzig**

Großrechner werden an der Universität Leipzig seit dem Jahr 2000 eingesetzt [1] [2]. Seit diesem Jahr hatte das Institut für Informatik der Universität Leipzig eine Multiprise 2000 mit 256MB Arbeitsspeicher im Einsatz. Auf diesem System lief als einziges Betriebssystem OS/390 Version 2.7.

Anfang 2003 bekam die Universität eine Multiprise 3000. Das System bestand aus zwei Prozessoren plus einem System Assist Processor (SAP) und 4GB Arbeitsspeicher. Weiterhin wurde an das System externer Festplattenspeicher mit ungefähr einem Terabyte Größe angeschlossen. Auf dieses System wurde die alte Installation der Multiprise 2000 kopiert. Zudem lief auf diesem System ein z/OS 1.5 sowie ein z/VM, welches auch Linux als Gastsysteme ermöglichte.

Der Schritt zu den 64bit-Rechnern wurde mit der Installation einer z/Series z900 im Jahr 2006 gemacht, die auch heute noch im Einsatz ist. Das System besitzt 9 Prozessoren (CPs) und 32GB Arbeitsspeicher. Diese Systemressourcen sind auf mehrere logische Partitionen (LPAR) verteilt. An die z900 ist ein Enterprise Storage Server (ESS oder auch Shark) angeschlossen, der eine Festplattenkapazität von 2,5 Terabyte bereitstellt. Auch hier wurden die vorherigen Betriebssystem-Instanzen (OS/390 2.7, z/VM und z/OS 1.5) übernommen. Auf einer weiteren logische Partition wurde z/OS 1.8 installiert.

#### **1.3.2 Nutzung des Systems der Universität Leipzig**

Das Großrechner-System der Universität Leipzig wird vielfältig genutzt. Es dient als Grundlage für praxisbezogene Übungen im Großrechner-Umfeld. Das System wird von mehr als 10 weiteren deutschen Universitäten und Fachhochschulen für die akademische Ausbildung im Großrechner-Umfeld eingesetzt.

Weiterhin wird das System für Diplomarbeiten genutzt. Gerade für Diplomarbeiten hat das in dieser Arbeit beschriebene Cloning-System eine praktische Relevanz. So können

Diplomanden an eigens für sie eingerichteten virtuellen Maschinen im vollen Maße Anpassungen vornehmen, was vorher beim Betrieb von nur einem Produktivsystem nicht möglich war oder zu größeren Problemen führen konnte.

Ein weiteres Anwendungsgebiet des Großrechner-Systems der Universität Leipzig ist die Durchführung eines Datenbankpraktikums. Dabei wird eine DB2-Datenbank auf einem z/OS-Betriebssystem als Grundlage verwendet, um Datenbankanwendungen zu entwickeln.

Ebenfalls auf dem Großrechner läuft ein WebSphere Web Application Server, der zum Einen die Homepage zum Großrechner-Projekt der Universität Leipzig bereitstellt, zum Anderen aber auch für Diplomarbeiten genutzt wird.

### **1.3.3 Ausgangssituation**

Auf dem z900-System der Universität Leipzig war bereits ein z/VM auf einer logischen Partition (LPAR) installiert. Dieses beherbergt diverse Testsysteme, die sowohl mit z/OS als auch mit Linux betrieben werden. Ein Linux-Gastsystem dient hierbei als Router der unter diesem z/VM installierten Gastsysteme.

In das vorhandene z/VM wurde ein weiteres z/VM als Gast hinzugefügt. Unter diesem Gast-z/VM fand die Entwicklung des Cloning-Systems statt. Dieser Aufbau hat Vor- und Nachteile. Ein wesentlicher Vorteil dieses Aufbaus ist es, dass das vorhandene z/VM-System nicht durch mögliche Fehler bei der Konfiguration in Mitleidenschaft gezogen werden kann. Bei einem Fehler wäre nur dieser Gast und keine weiteren Betriebssysteme betroffen. Der Nachteil des Gast-z/VM-Systems ist ein Einbruch in der Performance der Gäste unter diesem System. Ein- und Ausgabe (I/O) dauern im Vergleich zum Betrieb unter dem Basis-z/VM-System teils deutlich länger.

Die Variante mit der Verwendung des Gast-z/VM wird vorgezogen, weil die Geschwindigkeit in diesem Fall kein kritischer Faktor ist. Die Sicherheit, die durch die Isolation entsteht, ist hier ausschlaggebend.

|   |   |   |   |                  |                     |                              |                              |                               |     |  |  |
|---|---|---|---|------------------|---------------------|------------------------------|------------------------------|-------------------------------|-----|--|--|
| z/OS<br>1.8<br><br>binks<br>139.18.4.34 | z/OS<br>1.5<br><br>padme<br>139.18.4.35 | OS/390<br>2.7<br><br>lucas<br>139.18.4.36 | Linux<br>kenob<br>139.18.4.37<br>10.0.2.1 | MAINT<br><br>CMS | LNXMAINT<br><br>CMS | Linux<br>LIN001<br>10.0.10.1 | Linux<br>LIN002<br>10.0.10.2 | z/OS<br>ZOS001<br>10.0.10.100 | ... |  |  |
|   |   |   | 10.0.10.254<br>Cloning-z/VM               |                  |                     |                              |                              |                               |     |  |  |
|   |   |   | 10.0.2.253<br>Master-z/VM                 |                  |                     |                              |                              |                               |     |  |  |
| 139.18.4.49                             |   |   |   |                  |                     |                              |                              |                               |     |  |  |
| z/Series PR/SM                          |   |   |   |                  |                     |                              |                              |                               |     |  |  |

Abbildung 1: Großrechner-Struktur

Abbildung 1 zeigt die Konfiguration des Großrechners der Universität Leipzig. Die Systeme *binks*, *padme* und *lucas* sowie das Master-z/VM-System (auch First Level z/VM) laufen direkt in einer logischen Partition der z900 (LPAR oder Processor Resource/System Manager, kurz PR/SM).

Innerhalb des Master-z/VM-Systems laufen mehrere virtuelle Maschinen, unter anderem die Linux-Instanz *kenob*, die als Router für das interne Netzwerk dient. Zusätzlich läuft das Cloning-z/VM-System (auch Second Level z/VM), in dem das Cloning-System betrieben werden soll.

## 2 Grundlagen Großrechner und Virtualisierung

### 2.1 Großrechner

Die Geschichte der IBM z-Series Großrechner beginnt in den 1960er Jahren [3]. Angefangen hat diese Großrechnerarchitektur mit dem System/360 (S/360) im Jahr 1964. Ziel war es, sämtliche Datenverarbeitung mit einem Universal-Rechner zu ermöglichen, um nicht für spezielle Aufgaben jeweils neue Systeme konzipieren zu müssen. Das System/360 hatte einen Adressraum von 24bit, was eine Adressierung von 16 Megabyte Hauptspeicher ermöglichte.

Nachfolger dieses Systems wurde 1970 das System/370. Im Laufe der S/370-Evolution wurde der Adressraum auf 31bit erweitert, womit nun 2 Gigabyte Arbeitsspeicher adressiert werden konnten. Das 32. Bit wurde als Kontroll-Bit verwendet. Auf dem System/370 wurde auch erstmals der Vorläufer von z/VM, das Betriebssystem VM/370 eingesetzt.

1990 wurde als Nachfolger das System/390 vorgestellt, was auch ESA/390 (Enterprise Systems Architecture/390) genannt wurde. Dieses System unterstützte die logische Partitionierung eines Rechners und somit auch die Nutzung von mehr als 2 Gigabyte Arbeitsspeicher. Durch die logische Partitionierung war es auch ohne ein Virtualisierungs-Betriebssystem wie VM (unter S/390 VM/ESA genannt) möglich, mehrere Betriebssysteme parallel auf einem Rechner zu betreiben.

Im Jahr 2000 wurde der IBM eServer z-Series vorgestellt (z-Series, heute IBM System z). Mit diesem System wurde der Schritt zu einem 64bit-System gemacht. Das „Z“ in z-Series steht für „zero downtime“ („kein Ausfall“) und soll unterstreichen, wie sicher das System gegen Ausfälle geschützt ist.

Bei allen neuen Rechner-Generationen wurde immer Wert auf Abwärtskompatibilität gelegt. So sind Programme älterer Systeme auch auf aktuellen Rechnern lauffähig

#### 2.1.1 Heutige Bedeutung

IBM Großrechner haben auch heutzutage noch eine sehr große Verbreitung. 90% der weltweit größten Unternehmen setzen sie als ihre zentralen Server ein [1]. Banken, Versicherungen und Firmen mit sehr vielen Angestellten nutzen die Server in Bereichen, in denen jeder Ausfall erhebliche finanzielle Konsequenzen nach sich zieht. So werden

zum Beispiel über CICS, ein Transaktionsmonitor für z/OS, täglich ca. 30 Milliarden Transaktionen mit einem Volumen von ca. einer Billion Dollar getätigt [4]. Für unternehmenskritische Aufgaben sind die Ausfallsicherheit und Korrektheit die wichtigsten Kriterien. Ein weiterer Grund für die heutige Bedeutung ist der Umfang an vorhandenem Programmcode für Großrechner [1]. Die Anwendungen haben sich Jahre oder Jahrzehnte bewährt, teilweise besteht der Code zu 90% aus Recovery-Routinen, also Funktionen zur Behandlung von Fehlern jeglicher Art, was für eine hohe Zuverlässigkeit spricht. In die Anwendungsentwicklung allein wurden ca. 10 Millionen Personenjahre investiert. Eine Umstellung der Programme von COBOL, der wohl am häufigsten verwendeten Programmiersprache unter den IBM-Großrechnern, auf modernere Programmiersprachen ist damit sehr teuer.

## **2.2 Virtualisierung**

Virtualisierung ist ein Thema, das in den letzten Jahren immer mehr an Bedeutung gewonnen hat. Bei der Virtualisierung kann zwischen der Plattform-Virtualisierung und der Virtualisierung von Ressourcen unterschieden werden. Erstere beschäftigt sich mit der Virtualisierung der Systemarchitektur, um Betriebssysteme von den physischen Ressourcen zu entkoppeln. Bei der Virtualisierung von Ressourcen geht es darum, bestimmte Systemressourcen von der physischen Implementierung zu lösen, um so technische Details vor den Nutzern der Ressourcen zu verbergen.

### **2.2.1 Vorteile der Virtualisierung**

Die Virtualisierung von Systemen bietet zahlreiche Vorteile gegenüber dem Betrieb eines einzigen Betriebssystems pro Computer [5]. Ein entscheidender Vorteil der Virtualisierung ist die Konsolidierung von Rechnern. Das bedeutet, ein größerer Server ersetzt viele kleine, wodurch Platz und Strom gespart wird. Die Hardware für einen solchen Server ist jedoch meist teurer als die der kleineren Server, da besonders hoher Wert auf Ausfallsicherheit gelegt werden muss. Je mehr Systeme auf einem solchen System konsolidiert werden, desto besser ist das Kosten-Nutzen-Verhältnis. Durch die Konzentration auf einen Server wird auch die Wartung vereinfacht, was ebenfalls Kosten senkt. Darüber hinaus werden für den Betrieb der virtuellen Maschinen keinerlei zusätzliche Strom- oder Netzkabel benötigt.

Die Leistung des großen Systems muss nicht der Summe der Leistungen der einzelnen,

kleinen Systeme entsprechen, da viele Systeme ihre Ressourcen nicht ausschöpfen. Durch die Konsolidierung ergibt sich damit eine effizientere Nutzung der Ressourcen.

Durch die Plattform-Virtualisierung ist es schnell und leicht möglich, neue Systeme aufzusetzen und in Betrieb zu nehmen, da keine neue Hardware benötigt wird. So können bei Bedarf Systeme erzeugt, gestartet oder gestoppt werden, je nachdem, welche Aufgaben gerade zu erledigen sind.

Durch die Virtualisierung entsteht eine Entkopplung vom physischen System. Damit ist es ohne großen Aufwand möglich, ein System von einem physischen Server auf einen anderen zu verlagern. Weiterhin können Systeme „eingefroren“ werden und in diesem Zustand erneut gestartet werden, wenn zum Beispiel das virtualisierende System neu gestartet wird.

Weiterhin ist es möglich, komplett verschiedene Betriebssysteme gleichzeitig auf derselben Hardware zu betreiben. Manche Software ist nur auf einem bestimmten Betriebssystem lauffähig oder manche Betriebssysteme sind für spezifische Aufgaben besser geeignet. Durch die Virtualisierung von verschiedenen Betriebssystem auf derselben Hardware wird damit kein separates System benötigt.

## 2.2.2 Cloning

Viele Systeme haben den gleichen Grundaufbau des Betriebssystems gemeinsam. So benötigen verschiedene Anwendungssysteme wie ein Datenbank-Server oder ein Anwendungsserver, die das gleiche Betriebssystem verwenden (z.B. z/OS oder Linux), die gleiche Basis-Installation des Betriebssystems. Da in virtualisierten Umgebungen von der Hardware abstrahiert wird, reicht es aus, ein lauffähiges Basis-Betriebssystem zu erstellen und für jedes neue System eine exakte Kopie dieses Systems anzulegen. Dieses Vorgehen nennt sich Cloning. Es ermöglicht die schnelle Erzeugung und Inbetriebnahme neuer Betriebssystem-Instanzen. Für weitere Aufgaben (Datenbank, Anwendungsserver) wird dann jede Kopie separat angepasst.

## 2.2.3 Plattform-Virtualisierung auf IBM z/Series

Auf den z/Series-Großrechnern von IBM gibt es verschiedene Möglichkeiten der Virtualisierung, die im Folgenden erläutert werden [5].

Die Hardware-Plattform unterstützt eine logische Partitionierung (LPAR) des Großrech-

ners in mehrere virtuelle Systeme. Dafür zuständig ist der *Processor Resource/System Manager* (PRSM). Dieser erlaubt es, Prozessoren (CPU) und Arbeitsspeicher bestimmten Partitionen zuzuweisen. In jeder dieser Partitionen wird ein Betriebssystem betrieben. Eine CPU kann exklusiv von einer Partition oder von mehreren gemeinsam verwendet werden. Beim Arbeitsspeicher ist eine gemeinsame Nutzung durch mehrere LPARs nicht möglich, die Zuweisung entspricht einer Aufteilung des Speichers. Die Anzahl von logischen Partitionen auf einem Großrechner ist begrenzt, aktuelle Systeme der Serie IBM System z10 erlauben bis zu 60 logische Partitionen.

Eine weitere Möglichkeit der Virtualisierung stellt das Betriebssystem z/VM bereit. Das Betriebssystem dient als virtuelle Maschine für weitere Betriebssysteme. Unter z/VM sind die virtuellen Umgebungen nicht an die physischen Gegebenheiten gebunden. So ist es möglich, einem Gastsystem 10 virtuelle CPUs zur Verfügung zu stellen, auch wenn das reale System bzw. die logische Partition, in der z/VM betrieben wird, nur zwei besitzt. Unter z/VM ist eine gemeinsame Nutzung des vorhandenen Arbeitsspeichers möglich. Das Betriebssystem z/VM kümmert sich um die Verwaltung und gegebenenfalls die Auslagerung des Speichers auf Festplattenspeicher.

## **2.3 Virtualisierungskonzepte**

Virtualisierung dient dem Bereitstellen von Systemressourcen wie Festplatten, Speicher, Netzwerk und CPUs. Dadurch ist es möglich, viele verschiedene Systeme auf begrenzter Hardware auszuführen, was zu weniger Administrationsaufwand und zu einer Verringerung der Strom- und Hardwarekosten führt.

Für einen Gast in einem Virtualisierungssystem sollte es keinen Unterschied machen, ob er auf realen Ressourcen arbeitet oder auf virtuellen. Durch die Virtualisierung wird die darunter liegende Implementierung verdeckt.

### **2.3.1 Virtualisierung von Betriebssystemen**

Das System, das die Virtualisierung von Betriebssystemen ermöglicht, wird als *Virtual Machine Monitor* (VMM) bezeichnet, aber auch als *Hypervisor* oder *virtuelle Maschine* [5]. Es gibt verschiedene Arten von virtuellen Maschinen, die sich in die folgenden drei Arten unterteilen lassen.

### **Hardware-Emulation**

Bei der Emulation wird eine komplette Hardware-Plattform durch Software nachgebildet. Dadurch können nicht modifizierte Betriebssysteme in der virtuellen Maschine ausgeführt werden. Dieses Verfahren erlaubt es ebenfalls, Betriebssysteme, die für andere Prozessor-Architekturen geschrieben wurden, auszuführen.

Diese Art der Virtualisierung ist die langsamste, da sie die Befehle der virtualisierten Hardware interpretieren muss. Sie findet zum Beispiel Einsatz beim Emulator *Hercules*, mit dem es möglich ist, Großrechner-Betriebssysteme auf x86-Rechnern unter Linux oder Windows zu betreiben [6].

### **Paravirtualisierung**

Bei der Paravirtualisierung wird dem Gast-Betriebssystem ein Software-Interface bereitgestellt, das die darunter liegende Hardware abstrahiert. Die so dargestellte Hardware ist ähnlich, aber nicht identisch mit der eigentlichen Hardware. Um auf Ressourcen zuzugreifen, verwendet das virtuelle System einen Hypervisor-Aufruf. Dieser prüft die Befehle und führt sie aus

Dieses Verfahren ist sehr effizient. Allerdings muss das virtualisierte Betriebssystem für diese Art der Virtualisierung modifiziert werden. Ein Beispiel dieser Art der Virtualisierung ist der *XEN Hypervisor* mit Paravirtualisierung [7].

### **Hardware-Virtualisierung**

Die Hardware-Virtualisierung erlaubt wie die Emulation den Betrieb von unmodifizierten Betriebssystemen, jedoch müssen die Betriebssysteme für die jeweilige Plattform geeignet sein (gleiche Architektur). Es werden nur die Teile der physischen Hardware virtualisiert, bei deren Einsatz beim parallelen Betrieb mehrerer Betriebssysteme Konflikte verursacht werden können.

Wichtig ist hier der Umgang mit privilegierten Prozessor-Befehlen, die zum Beispiel für die Ein- und Ausgabe benötigt werden. Hierfür gibt es verschiedene Implementierungen. Zum einen können alle privilegierten Befehle vom Hypervisor abgefangen und dort emuliert werden. Dieses Verfahren verwendete zum Beispiel *VM/370*, einer der Vorgänger von z/VM. Bei der x86-Architektur müssen Befehle teilweise noch verändert werden, um ausgeführt zu werden. Dieses Verfahren verwendet beispielsweise *VMware* [8].

Auf z/Series-Großrechnern unterstützt die Hardware direkt das Ausführen von privile-

gierten Befehlen. Für eine Verbesserung der Performance sind aber Modifizierungen am Gast-Betriebssystem wie bei der Paravirtualisierung möglich. Dieses Verfahren ist hochgradig effizient, benötigt jedoch spezielle, kostenintensive Hardware.

### **2.3.2 Virtualisierung von Ressourcen unter z/VM**

Jenseits der System-Virtualisierung werden unter z/VM weitere Formen der Virtualisierung genutzt. Diese betreffen verschiedene Bereiche des Systems [5].

#### ***Resource Sharing***

Beim *Resource Sharing* werden mehrere virtuelle Ressourcen definiert, die sich die gleiche physische Ressource teilen. So können zum Beispiel Gastmaschinen CPUs zugewiesen werden, die sich dann den Zugriff auf diese teilen. Aus Sicht der Gastmaschine verwendet diese die CPU exklusiv. Dass andere Gäste die CPU ebenfalls nutzen, wird verborgen.

Dieses Verfahren wird von der LPAR-Virtualisierung und von z/VM selbst genutzt, um Ressourcen unter seinen Gästen zu teilen. Dieses Vorgehen findet sich aber auch bei virtuellen Disks, wobei eine Festplatte in verschiedene Teile zerlegt wird (Minidisks), die unterschiedlichen Gastsystemen zugewiesen werden können.

#### ***Resource Aggregation***

Bei der *Resource Aggregation* werden verschiedene physische Ressourcen zu einer Ressource zusammengefasst. Es wird beispielsweise bei Festplatten verwendet, um große Speicherpools, die aus mehreren Festplatten bestehen, zur Verfügung zu stellen. Das Gastsystem muss so zum Beispiel nicht mehr viele kleine Disks verwalten, sondern kann eine große, virtuelle Festplatte nutzen.

#### ***Emulation von Funktionen***

Durch die Emulation von Funktionen werden Funktionen für virtuelle Ressourcen bereitgestellt, welche die physischen Ressourcen nicht direkt unterstützen. Zum Beispiel wird mit iSCSI ein virtueller SCSI-Bus über ein IP-Netzwerk emuliert. Ein anderes Beispiel sind Lochkarten-Locher (Puncher) und Lochkarten-Lesegeräte (Reader). Diese werden in ihrer virtuellen Form noch heute benutzt, auch wenn natürlich keine echten Lochkarten mehr verwendet werden. So werden zum Beispiel für den Start der Installation von Suse Linux Enterprise Server 10 (SLES10) knapp 200.000 virtuelle Lochkarten

geloht und wieder gelesen.

Ein weiteres Beispiel für Emulation ist die Emulation von spezifischen Festplattenmodellen (Direct Access Storage Device, DASD), die heutzutage nicht mehr im Einsatz sind. So werden in *System z*-Umgebungen zum Beispiel DASDs vom Typ *3390 Model 3* (3390-3) verwendet, die eine Größe von 2,2GB haben. Physisch liegen diese jedoch auf größeren, heute gebräuchlichen Festplatten oder RAID-Systemen und werden dem System nur virtuell in dieser Form bereitgestellt.

### ***Isolation***

Durch Isolation lassen sich physische Ressourcen, die als Grundlage für virtuelle Ressourcen dienen, während des Betriebs des Systems austauschen, ohne dass die Systeme, die die virtuellen Ressourcen nutzen, dies registrieren.

So kann zum Beispiel ein defekter Prozessor oder eine defekte Festplatte in einem RAID-Verbund ausgetauscht werden, ohne dass das Gastsystem davon in Kenntnis gesetzt wird. Dieser Wechsel geschieht transparent. Genauso lässt sich ein System von einem Festplatten-System auf ein anderes migrieren, wenn dieses ausgetauscht werden soll.

### 3 Das Betriebssystem z/VM

z/VM geht auf das im Jahr 1972 entwickelte VM/370 zurück [5]. Die zentralen Komponenten von VM/370, das *Control Program* (CP) und das *Conversational Monitor System* (CMS), werden auch unter z/VM noch verwendet, wodurch es eine immer noch anhaltende Abwärtskompatibilität gewährleistet ist – eine der großen Stärken der IBM-Großrechner.

#### 3.1 Beschreibung von z/VM

z/VM wurde entwickelt, um mehrere Betriebssysteme parallel zu betreiben. Dafür wird jedem Gastsystem eine Laufzeitumgebung bereitgestellt, eine *virtuelle Maschine*. Diese simuliert ein komplettes, physisches System. Die Virtualisierung ist für die Gastsysteme transparent.

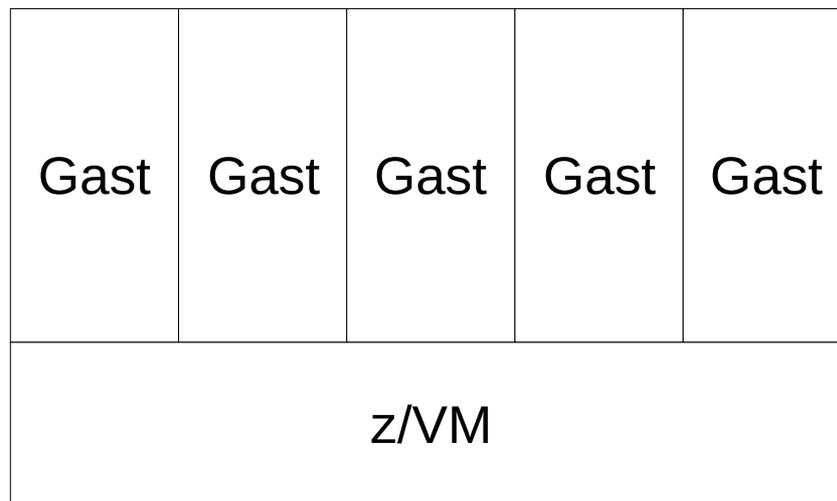


Abbildung 2: Gäste unter z/VM

Um diese Virtualisierung zu realisieren, bedient sich z/VM verschiedener Konzepte. Das *Control Program* (CP) ist der *Hypervisor*, der die Virtualisierung der Ressourcen zur Verfügung stellt.

Mittels des z/VM-Betriebssystems CMS werden höhere Funktionen für die Verwaltung von Systemen sowie Skriptsprachen zur Vereinfachung von Abläufen bereitgestellt.

## 3.2 Konzepte von z/VM

### 3.2.1 CP

z/VM wird über das Control Program (CP) verwaltet. CP stellt dafür jeder virtuellen Maschine – also jedem Gast – Befehle zur Verwaltung zur Verfügung [9].

Einige für das Cloning-System relevante Befehle sind:

|            |  |
|------------|--|
| DEFINE ... | Definition von Geräten   |
| QUERY ...  | Anzeige von System-Informationen, zum Beispiel ein Gerätestatus oder die derzeit laufenden Gäste   |
| DISCONNECT | Ausloggen aus dem System (Beenden einer 3270-Session), das Gastsystem läuft im Hintergrund weiter. |
| LOGOFF     | Ausloggen aus dem System, das Gastsystem wird beendet.   |
| SHUTDOWN   | Beenden von z/VM   |
| IPL ...    | Start eines Gast-Betriebssystems (Initial Program Load)  |

Nicht jeder Gast darf alle Befehle ausführen. Für die Kontrolle gibt es verschiedene Benutzerklassen (Privilege Class) [10]. Diese sind zum Teil System-definiert, es ist aber auch möglich, eigene Klassen anzulegen.

Die für das Cloning-System wichtigen System-definierten Klassen sind:

- Klasse A – System Operator
  - Benutzer der Klasse A verwalten das z/VM-System. Dazu gehört neben dem generellen Betrieb von z/VM das Anlegen von Gästen und die Verwaltung der Systemressourcen. Der Benutzer MAINT gehört zum Beispiel zu dieser Klasse.
- Klasse B – System Resource Operator
  - Benutzer der Klasse B verwalten die echten Ressourcen eines Systems (außer die durch Klasse-A-Nutzer verwalteten). Dazu gehört zum Beispiel der Benutzer TCPMAINT, der für die Verwaltung des TCP/IP-Stacks zuständig ist.
- Klasse G – General User
  - Benutzer der Klasse G verwalten jeweils nur eine spezielle virtuelle Maschine. Jedes Gastsystem wird im Cloning-System durch einen Nutzer dieser

Klasse repräsentiert, so auch einfache CMS-User.

### 3.2.2 Minidisks

z/VM erlaubt es, Festplatten zu partitionieren [11]. Damit ist es möglich, mehrere kleine, virtuelle Festplatten auf einer größeren abzulegen. Diese Partitionen heißen *Minidisks* (MDISK). Die kleinste Einheit für die Partitionierung ist ein Zylinder. Eine DASD vom Typ *3390 Model 3* hat 3339 Zylinder und pro Zylinder ca. 850kB Speicherkapazität. Bei der Definition einer Minidisk muss angegeben werden, auf welcher Festplatte (Gerät) sie sich befinden soll, an welcher Stelle auf der Festplatte sie beginnt (Start-Zylinder) und wie viele Zylinder (Größe der Partition) zu ihr gehören sollen. Für eine Minidisk, die nur Skripte aufnehmen soll, reichen 1 oder 2 Zylinder vollkommen aus. Aus der Sicht der Gastsysteme erscheint eine Minidisk, abgesehen von der Größe, wie eine normale DASD.

### 3.2.3 CMS

CP dient nur dazu, Ressourcen zu verwalten. Diese Ressourcen werden dem Betriebssystem zur Verfügung gestellt, welches in der virtuellen Maschine ausgeführt wird.

Für die Verwaltung von z/VM wird das *Conversational Monitor System* (CMS) verwendet [5]. CMS ist ein Betriebssystem und wird wie andere Betriebssysteme auch von CP gestartet. Es stellt dem Benutzer im Vergleich mit CP erweiterte Funktionen bereit, unter anderem ein Dateisystem für die Festplatten.

#### **CMS Dateisystem**

Einzelne Minidisks können mit dem CMS Dateisystem formatiert werden. Damit lassen sich diese Minidisks unter CMS als Speicherplatz für Dateien verwenden.

Dateien im CMS Dateisystem bestehen aus 3 Komponenten:

- dem Dateinamen (*file name, fn*),
- dem Typ (*file type, ft*) und
- dem Verzeichnisnamen (*file mode, fm*), der die entsprechende Minidisk identifiziert.

Wenn CMS gestartet wird, wird die Datei *PROFILE EXEC* ausgeführt. Als Standard wird die Minidisk mit der Adresse 0191 als A-Minidisk (*file mode = A*) verwendet.

### Wichtige CMS-Kommandos

Eine Liste aller CMS-Kommandos ist in der CMS-Befehlsreferenz [12] zu finden. Für eine Übersicht, welche Dateien auf den CMS-Minidisks liegen, gibt es das Kommando *filel*. Es zeigt Dateinamen, Dateityp, die Größe sowie die das Datum der letzten Änderung an.

| Cmd      | Filename    | Filetype | Fm  | Format    | Lrecl   | Records | Blocks | Date     | Time     |
|----------|-------------|----------|-----|-----------|---------|---------|--------|----------|----------|
| MAINT    | FILELIST A0 | V        | 169 | Trunc=169 | Size=72 | Line=1  | Col=1  | Alt=0    |          |
| LASTING  | GLOBALV     | A1       | V   |           | 38      | 3       | 1      | 7/16/07  | 23:55:26 |
| USER     | MDISKMAP    | A1       | V   |           | 78      | 388     | 7      | 7/09/07  | 23:54:58 |
| MAINT    | NETLOG      | A0       | V   |           | 106     | 30      | 1      | 7/09/07  | 23:54:04 |
| SPRUTH   | DIRECT      | A1       | F   |           | 80      | 2       | 1      | 7/09/07  | 23:36:39 |
| CMS      | PROTODIR    | A2       | V   |           | 34      | 3       | 1      | 7/09/07  | 23:24:35 |
| ALLVOLS  | FREEXT      | A1       | V   |           | 61      | 18      | 1      | 7/09/07  | 23:00:24 |
| DIAGNOSE | FREEXT      | A1       | V   |           | 82      | 1       | 1      | 7/09/07  | 23:00:24 |
| USER     | WITHPASS    | A0       | F   |           | 80      | 2161    | 43     | 7/05/07  | 13:03:47 |
| USER     | DIRECT      | A1       | F   |           | 80      | 135     | 3      | 7/03/07  | 12:24:34 |
| \$VMFSRV | \$MSGLOG    | A1       | V   |           | 80      | 3488    | 64     | 7/02/07  | 23:38:54 |
| SERVICE  | \$CONS      | A1       | V   |           | 132     | 80      | 3      | 7/02/07  | 23:38:54 |
| SERVICE  | \$PRODS     | A1       | V   |           | 12      | 2       | 1      | 7/02/07  | 23:38:54 |
| \$VMFBLD | \$MSGLOG    | A1       | V   |           | 80      | 2400    | 32     | 7/02/07  | 23:38:52 |
| SETUP    | \$LINKS     | A1       | V   |           | 26      | 72      | 1      | 7/02/07  | 23:38:52 |
| \$VMFINS | \$MSGLOG    | A1       | V   |           | 80      | 694     | 10     | 7/02/07  | 23:38:35 |
| USER     | BACKUP      | A1       | F   |           | 80      | 2284    | 45     | 7/02/07  | 23:33:22 |
| USER     | DISKMAP     | A1       | F   |           | 80      | 395     | 8      | 1/26/07  | 12:26:28 |
| PROFILE  | EXEC        | A1       | V   |           | 27      | 17      | 1      | 1/26/07  | 1:13:47  |
| ATTZ     | EXEC        | A1       | V   |           | 27      | 55      | 1      | 11/12/06 | 10:36:13 |
| \$VMFP2P | \$MSGLOG    | A1       | V   |           | 80      | 3233    | 60     | 1/07/04  | 15:34:29 |
| NLSHDR   | LISTING     | A1       | F   |           | 121     | 29      | 1      | 1/07/04  | 15:32:12 |
| NLSUCENG | DCSSMAP     | A5       | F   |           | 100     | 21      | 1      | 1/07/04  | 15:32:12 |
| \$NLS\$  | TEXT        | A2       | F   |           | 80      | 4405    | 87     | 1/07/04  | 15:32:11 |
| DMSUCENG | LANGMAP     | A5       | V   |           | 80      | 9       | 1      | 1/07/04  | 15:32:10 |
| NLSAMENG | DCSSMAP     | A5       | F   |           | 100     | 20      | 1      | 1/07/04  | 15:32:06 |

1= Help    2= Refresh    3= Quit    4= Sort(type)    5= Sort(date)    6= Sort(size)  
7= Backward    8= Forward    9= FL /n    10=    11= XEDIT/LIST    12= Cursor

====>

X E D I T 1 File

Zum Bearbeiten und Anzeigen von Dateien gibt es das Kommando *xedit*. Über das Kommando *filel* lässt sich *xedit* starten, indem ein „X“ in die erste Spalte der entsprechenden Zeile geschrieben oder F11 gedrückt wird. Die Bedienung ist vergleichbar mit der des ISPF-Editors von z/OS. Sollen die Änderungen an einer Datei gespeichert werden, so muss der Editor mit dem Befehl *file* geschlossen werden. Zum Verwerfen der Änderungen dient der Befehl *quit*.

```

PROFILE EXEC      A1 V 130 Trunc=130 Size=17 Line=0 Col=1 Alt=0

00000 * * * Top of File * * *
00001 /* */
00002 SET CMSTYPE HT
00003 SYNONYM SYN
00004 TERM MODE VM
00005 'CP SP CONS * START'
00006 SET PF12 RETRIEVE
00007 SET PF24 RETRIEVE
00008 cp link tcpip 592 592 rr
00009 cp link tcpmaint 198 198 rr
00010 ACCESS 5E5 B
00011 ACCESS 2CC C
00012 ACCESS 51D D
00013 ACCESS 592 T
00014 ACCESS 198 U
00015 'SET FILEPOOL VMSYS:'
00016 'SET LDRTBLS 25'
00017 SET CMSTYPE RT
00018 * * * End of File * * *

====>

```

Dateien mit dem Dateityp *EXEC* sind Skripte, die sich von der Konsole aus ausführen lassen oder in anderen Skripten verwendet werden. Um so ein Skript auszuführen, muss lediglich der Dateiname als Kommando verwendet werden.

Für eine Übersicht, welche Dateien auf einem Lochkarten-Leser (Reader) liegen, gibt es das Kommando *rdrlist*. Die Ausgabe ist mit der von *filel* vergleichbar.

```

MAINT  RDRLIST  A0 V 164 Trunc=164 Size=17 Line=1 Col=1 Alt=0
Cmd  Filename Filetype Class User  at Node  Hold Records  Date      Time
(none) (none)  CON T MAINT  GEORGVM1 NONE    157 6/30  10:21:07
(none) (none)  CON T MAINT  GEORGVM1 NONE    22 6/30  11:36:14
(none) (none)  CON T MAINT  GEORGVM1 NONE   116 6/30  12:52:35
(none) (none)  CON T MAINT  GEORGVM1 NONE    14 6/30  13:32:28
(none) (none)  CON T MAINT  GEORGVM1 NONE    25 7/03  20:49:30
(none) (none)  CON T MAINT  GEORGVM1 NONE    28 7/08   9:54:57
(none) (none)  CON T MAINT  GEORGVM1 NONE    31 8/03  13:09:58
(none) (none)  CON T MAINT  GEORGVM1 NONE     3 8/05  14:12:12
(none) (none)  CON T MAINT  GEORGVM1 NONE    39 8/05  14:12:31
(none) (none)  CON T MAINT  GEORGVM1 NONE     3 8/05  14:14:29
(none) (none)  CON T MAINT  GEORGVM1 NONE   217 8/05  14:59:01
(none) (none)  CON T MAINT  GEORGVM1 NONE   457 8/05  15:48:01
(none) (none)  CON T MAINT  GEORGVM1 NONE     3 8/07  17:03:51
(none) (none)  CON T MAINT  GEORGVM1 NONE    22 8/08  10:42:26
(none) (none)  CON T MAINT  GEORGVM1 NONE     8 8/11  11:02:19
(none) (none)  CON T MAINT  GEORGVM1 NONE   470 8/27  10:25:03
USER   WITHPASS PUN A DIRMAINT GEORGVM1 NONE   2237 8/31  16:28:40

1= Help      2= Refresh  3= Quit      4= Sort(type) 5= Sort(date) 6= Sort(user)
7= Backward 8= Forward  9= Receive  10=           11= Peek     12= Cursor

====>

X E D I T 1 File

```

Um sich den Inhalt dieser Dateien anzeigen zu lassen, kann mit F11 (Peek) in der entsprechenden Zeile der Anfang der Datei angezeigt werden. Um eine Datei auf die A-Minidisk zu übertragen, gibt es den Befehl *receive*, der durch Drücken von F9 ausgeführt wird.

### 3.2.4 Weitere Komponenten

Unter z/VM werden weitere Komponenten durch weitere Gäste realisiert. So arbeiten beispielsweise der TCP/IP-Stack von z/VM und der Verwaltungsdienst für das *z/VM User Directory (DIRMAINT)* als separate Benutzer.

Für die Automatisierung gibt es den Gast *AUTOLOGI*, der beim Systemstart immer automatisch gestartet wird.

### 3.2.5 z/VM User Directory

Die Konfiguration von Gästen und Profilen geschieht über das z/VM User Directory, eine Datei mit dem Namen *USER DIRECT* [13]. Darin werden alle Gäste definiert und spezifiziert, welche Ressourcen einem bestimmten Gast bereitgestellt werden.

Zur Verwaltung stehen zwei mögliche Ansätze zur Verfügung. Zum Einen lässt sich die Datei manuell pflegen. Dabei muss die System-Disk zunächst vom System (CP) losgelöst werden (*release*). Nun kann das User Directory mit einem Editor verändert werden. Zur Sicherheit sollte die geänderte Datei auf ihre Korrektheit überprüft werden, wofür z/VM geeignete Werkzeuge zur Verfügung stellt. Anschließend wird die System-Disk dem System wieder hinzugefügt (*attach*) und die neue Version des Verzeichnisses aktiviert.

Die zweite Methode ist die Verwendung von *DIRMAINT*. Dieser Dienst wird über das Kommando *DIRMAINT* (kurz: *DIRM*) administriert und arbeitet befehls-orientiert. Die Befehle bearbeiten damit das User Directory direkt und nehmen einem die oben genannten manuellen Schritte ab. Durch die Befehle werden auch Syntax-Fehler vermieden, da die Eingaben von *DIRMAINT* geprüft werden, bevor das User Directory verändert wird.

#### **Beispiel-Directory-Eintrag**

Ein Eintrag im User Directory sieht beispielsweise so aus:

```
USER TESTUSER GEHEIM 64M 2048M G
  INCLUDE IBMDFLT
  IPL CMS

  CONSOLE 0009 3215 T
  SPOOL 000C 2540 READER *
  SPOOL 000D 2540 PUNCH A
  SPOOL 000E 1403 A

MDISK 0191 3390 1774 20 440W02 MR
```

Die erste Zeile beschreibt den Benutzer *TESTUSER*, mit dem Passwort „GEHEIM“. Dem Benutzer werden standardmäßig 64MB Hauptspeicher zugeteilt, er kann ihn jedoch mit einem speziellen Befehl (DEFINE STORAGE) auf bis zu 2048MB erweitern. Er gehört der Benutzerklasse G an.

Die Eigenschaften des Profils IBMDFLT werden für diesen Benutzer übernommen. Mit Profilen können gemeinsame Eigenschaften verschiedener Gäste gesammelt werden. Dies verringert den Konfigurations- und Wartungsaufwand. Beim Login wird das Betriebssystem CMS gestartet (IPL, Initial Program Load).

In den übrigen Zeilen werden Geräte definiert, auf welche die Gastmaschine zugreifen kann. So wird eine Konsole definiert, dann ein virtueller Lochkarten-Leser (*Reader*, Adresse 000C), ein Lochkarten-Stanzer (*Puncher*, 000D) und ein Drucker (*Spooler*, 000E).

Zuletzt wird dem Benutzer als Festplattenspeicher eine Minidisk mit Schreibrechten zugeteilt, die 20 Zylinder groß ist und sich auf der 3390-DASD „440W02“ befindet.

### **3.3 Administration von z/VM**

Beim Cloning-System findet die Administration über den User *MAINT* statt, soweit nichts anderes angegeben wird. Die Administration kann auch durch jeden anderen User der Benutzerklasse A durchgeführt werden.

Nach dem Login von *MAINT* wird CMS gestartet und es kann mit der Administration begonnen werden.

### **3.4 Grundlegende Befehle zur Verwaltung des Systems**

Für die Verwaltung des Systems werden Befehle von CP [9] und CMS [12] verwendet. Im Folgenden werden Arbeiten erklärt, die beim Umgang mit Festplatten anfallen. Dies betrifft das Formatieren und Kopieren einer DASD.

#### **3.4.1 Formatieren einer DASD**

Ziel ist es, eine freie DASD zu formatieren und mit einem Label zu versehen. Dafür gibt es das CP Utility *CPFMTXA*.

Zunächst wird nach einer freien DASD gesucht:

```
q dasd all
DASD 0ACC CP OWNED 440RES 97
DASD 0ACD CP OWNED 440W01 43
DASD 0ACE CP OWNED 440W02 20
DASD 552B CP SYSTEM DEVNO 1
DASD 552C CP SYSTEM DEVNO 1
DASD 552D CP SYSTEM DEVNO 1
DASD 552E CP SYSTEM DEVNO 1
DASD 552F CP SYSTEM DEVNO 1
DASD 5530 CP SYSTEM LIN001 0
DASD 5531 CP SYSTEM Z0SC01 0
DASD 5532 CP SYSTEM DEVNO 1
DASD 0190 MNT190 , DASD 0191 ZVM191 , DASD 019D MNT19D , DASD 019E MNT19E
DASD 0401 MNT401 , DASD 0402 MNT402 , DASD 0405 MNT405 , DASD 0592 TCM592
DASD 5533 FREE , DASD 5534 FREE , DASD 5535 FREE , DASD 5536 FREE
DASD 5537 FREE , DASD 5538 FREE , DASD 5539 FREE
An offline DASD was not found.
Ready; T=0.01/0.01 15:02:15
```

Die freie DASD 5533 wird zunächst „angeschlossen“ (*attach*), um sie dann mittels des Kommandos *CPFMTXA* formatieren zu können.

```
attach 5533 *
DASD 5533 ATTACHED TO MAINT 5533 WITH DEVCTL
Ready; T=0.01/0.01 15:30:53
```

Nun soll die DASD 5533 formatiert und mit dem Label „LIN006“ versehen werden.

```
cpfmtxa 5533 LIN006 FORMAT
HCPCCF0024E CYLINDERS MISSING OR INVALID
Ready(00024); T=0.01/0.01 15:33:06
cpfmtxa
ENTER FORMAT, ALLOCATE, LABEL, OR QUIT:
format
ENTER THE VDEV TO BE PROCESSED OR QUIT:
5533
ENTER THE CYLINDER RANGE TO BE FORMATTED ON DISK 5533 OR QUIT:
000 end
ENTER THE VOLUME LABEL FOR DISK 5533:
LIN006
CPFMTXA:
FORMAT WILL ERASE CYLINDERS 00000-03338 ON DISK 5533
DO YOU WANT TO CONTINUE? (YES | NO)
yes
```

Das Formatieren der DASD startet jetzt.

```

HCPCCF6209I INVOKING ICKDSF.
ICK030E DEFINE INPUT DEVICE: FN FT FM, "CONSOLE", OR "READER"
CONSOLE
ICK031E DEFINE OUTPUT DEVICE: FN FT FM, "CONSOLE", OR "PRINTER"
CONSOLE
ICKDSF - CMS/XA/ESA DEVICE SUPPORT FACILITIES 17.0          TIME: 15:35:17
        05/27/08      PAGE 1

ENTER INPUT COMMAND:
  CPVOL FMT MODE(ESA) UNIT(5533) VOLID(LIN006) NOVFY -
ENTER INPUT COMMAND:
  RANGE(000,3338)
ICK00700I DEVICE INFORMATION FOR 5533 IS CURRENTLY AS FOLLOWS:
      PHYSICAL DEVICE = 3390
      STORAGE CONTROLLER = 3990
      STORAGE CONTROL DESCRIPTOR = E9
      DEVICE DESCRIPTOR = 0A
      ADDITIONAL DEVICE INFORMATION = 4A001B35
ICK04000I DEVICE IS IN SIMPLEX STATE
ICK00091I 5533 NED= 2105. .IBM.75.000000012653
ICK091I 5533 NED= 2105. .IBM.75.000000012653
ICK03020I CPVOL WILL PROCESS 5533 FOR VM/ESA MODE
ICK03090I VOLUME SERIAL = .....
ICK03022I FORMATTING THE DEVICE WITHOUT FILLER RECORDS
ICK03011I CYLINDER RANGE TO BE FORMATTED IS 0 - 3338
ICK003D REPLY U TO ALTER VOLUME 5533 CONTENTS, ELSE T
U
ICK03000I CPVOL REPORT FOR 5533 FOLLOWS:

      FORMATTING OF CYLINDER 0 STARTED AT: 15:35:53
      FORMATTING OF CYLINDER 100 ENDED AT: 15:36:05

...

      FORMATTING OF CYLINDER 3300 ENDED AT: 15:42:21
      FORMATTING OF CYLINDER 3338 ENDED AT: 15:42:26

      VOLUME SERIAL NUMBER IS NOW = LIN006

      CYLINDER ALLOCATION CURRENTLY IS AS FOLLOWS:
      TYPE      START      END      TOTAL
      ----      -
      PERM      0          3338    3339

ICK00001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
      15:43:12      05/27/08

ENTER INPUT COMMAND:
  END

ICK00002I ICKDSF PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0

```

Nun muss noch der Typ der Allokation angegeben werden

```

ENTER ALLOCATION DATA
TYPE CYLINDERS
.....
perm 0 end
end

```

Anschließend ist das Formatieren beendet.

```

ICK03000I CPVOL REPORT FOR 5533 FOLLOWS:

      CYLINDER ALLOCATION CURRENTLY IS AS FOLLOWS:
      TYPE      START      END      TOTAL
      ----      -
      PERM      0          3338    3339

ICK00001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
      15:47:06      05/27/08

ENTER INPUT COMMAND:
  END

ICK00002I ICKDSF PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0

```

Nach Abschluss der Formatierung kann das Gerät vom aktuellen User gelöst (*detach*) werden.

```
det 5533
DASD 5533 DETACHED
```

Die Übersicht der Festplatten zeigt, dass die DASD 5533 nun den eingegeben Namen erhalten hat.

```
q dasd all
DASD 0ACC CP OWNED 440RES 97
DASD 0ACD CP OWNED 440W01 43
DASD 0ACE CP OWNED 440W02 19
DASD 552B CP SYSTEM DEVNO 1
DASD 552C CP SYSTEM DEVNO 1
DASD 552D CP SYSTEM DEVNO 1
DASD 552E CP SYSTEM DEVNO 1
DASD 552F CP SYSTEM DEVNO 1
DASD 5530 CP SYSTEM LIN001 0
DASD 5531 CP SYSTEM ZOSC01 0
DASD 5532 CP SYSTEM DEVNO 1
DASD 0190 MNT190 , DASD 0191 ZVM191 , DASD 019D MNT19D , DASD 019E MNT19E
DASD 0401 MNT401 , DASD 0402 MNT402 , DASD 0405 MNT405 , DASD 0592 TCM592
DASD 5533 LIN006 , DASD 5534 FREE , DASD 5535 FREE , DASD 5536 FREE
DASD 5537 FREE , DASD 5538 FREE , DASD 5539 FREE
An offline DASD was not found.
```

### 3.4.2 Kopieren einer DASD

Wie auch beim Formatieren einer DASD muss zunächst ein freies Ausgabe-Gerät gesucht werden. In diesem Beispiel wird das Ausgabe-Gerät 5535 gewählt. Kopiert werden soll die DASD *LIN001*. Diese hat die Gerätenummer 5530.

```
q dasd all
DASD 0ACC CP OWNED 440RES 106
DASD 0ACD CP OWNED 440W01 42
DASD 0ACE CP OWNED 440W02 20
DASD 552B CP SYSTEM DEVNO 3
DASD 552C CP SYSTEM DEVNO 3
DASD 552D CP SYSTEM DEVNO 3
DASD 552E CP SYSTEM DEVNO 3
DASD 552F CP SYSTEM DEVNO 3
DASD 5530 CP SYSTEM LIN001 0
DASD 5531 CP SYSTEM ZOSC01 14
DASD 5532 CP SYSTEM LIN002 0
DASD 5533 CP SYSTEM LIN003 0
DASD 5534 CP SYSTEM LIN004 0
DASD 5539 ATTACHED TO ZOS001 5539 R/W FR5539
DASD 0190 MNT190 , DASD 0191 ZVM191 , DASD 019D MNT19D , DASD 019E MNT19E
DASD 0401 MNT401 , DASD 0402 MNT402 , DASD 0405 MNT405 , DASD 0592 TCM592
DASD 5535 FREE , DASD 5536 FREE , DASD 5537 FREE , DASD 5538 FREE
An offline DASD was not found.
Ready; T=0.01/0.01 10:27:18
```

Um die Kopier-Aktionen durchzuführen, müssen die Geräte an den eigenen Benutzer (MAINT) gebunden werden. Dafür muss das Volume 5530 zunächst vom System entkoppelt werden. Dies geschieht mit dem Kommando *detach*.

```
det 5530 from system
DASD 5530 DETACHED SYSTEM
Ready; T=0.01/0.01 10:37:02
```

Nun werden Quelle und Ziel mit dem Kommando *attach* an den eigenen Benutzer gebunden:

```
att 5530 *
DASD 5530 ATTACHED TO MAINT 5530 WITH DEVCTL
Ready; T=0.01/0.01 10:39:02
att 5535 *
DASD 5535 ATTACHED TO MAINT 5535 WITH DEVCTL
Ready; T=0.01/0.01 10:39:11
```

Zum Kopieren von DASDs ist das CP Utility *DDR* zuständig. Nach dem Aufruf von *DDR* erfolgt die Beschreibung der zu erledigenden Aktion. Durch eine Leerzeile werden die Arbeitsschritte erledigt.

```
ddr
z/VM DASD DUMP/RESTORE PROGRAM
ENTER:
```

Zunächst werden Ein- und Ausgabegerät bekannt gemacht.

```
in 5530 3390
ENTER:
out 5535 3390
ENTER:
```

Der Typ der Geräte muss jeweils mit angegeben werden. Bei den DASDs handelt es sich um Geräte vom Typ 3390.

Anschließend wird der *Copy*-Befehl eingegeben. Zum Kopieren des kompletten DASD-Volumens reicht der Befehl *COPY ALL*. Zur Sicherheit werden noch zwei Abfragen gemacht, damit nicht versehentlich eine falsche DASD überschrieben wird.

```
copy all
HCPDDR711D VOLID READ IS LIN001
DO YOU WISH TO CONTINUE? RESPOND YES, NO OR REREAD:
yes
HCPDDR716D NO VOL1 LABEL FOUND
DO YOU WISH TO CONTINUE? RESPOND YES, NO OR REREAD:
yes
COPYING LIN001
```

Die letzte Zeile zeigt an, dass das Kopieren begonnen hat. Dieser Vorgang kann mehrere Minuten dauern. Wenn das Kopieren beendet ist, erscheint eine Erfolgsmeldung. Eine abschließende Leer-Eingabe beendet das Programm.

```
END OF COPY
END OF JOB
```

Damit ist der Kopiervorgang abgeschlossen. Um Verwechslungen zu vermeiden, wird das kopierte Volume anschließend noch umbenannt, das heißt, das Label wird geändert. Dafür wird wie beim Formatieren das Utility *CPFMTXA* verwendet.

```
cpfmtxa 5535 CPY001 LABEL
HCPCCF6209I INVOKING ICKDSF.
ICK030E DEFINE INPUT  DEVICE: FN FT FM, "CONSOLE", OR "READER"
CONSOLE
ICK031E DEFINE OUTPUT DEVICE: FN FT FM, "CONSOLE", OR "PRINTER"
CONSOLE
ICKDSF - CMS/XA/ESA DEVICE SUPPORT FACILITIES 17.0          TIME: 11:44:03
          08/27/08      PAGE 1

ENTER INPUT COMMAND:
  CPVOL LABEL UNIT(5535) VOLID(CPY001) NOVFY
ICK00700I DEVICE INFORMATION FOR 5535 IS CURRENTLY AS FOLLOWS:
          PHYSICAL DEVICE = 3390
          STORAGE CONTROLLER = 3990
          STORAGE CONTROL DESCRIPTOR = E9
          DEVICE DESCRIPTOR = 0A
          ADDITIONAL DEVICE INFORMATION = 4A001B35
ICK04000I DEVICE IS IN SIMPLEX STATE
ICK00091I 5535 NED= 2105. .IBM.75.000000012653
ICK091I 5535 NED= 2105. .IBM.75.000000012653
ICK03090I VOLUME SERIAL = LIN001
ICK003D REPLY U TO ALTER VOLUME 5535 CONTENTS, ELSE T
U
ICK03000I CPVOL REPORT FOR 5535 FOLLOWS:

          VOLUME SERIAL NUMBER IS NOW = CPY001

ICK00001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
          11:44:03 08/27/08

ENTER INPUT COMMAND:
  END
ICK00002I ICKDSF PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

Nun können die Volumes mittels des Kommandos *detach* vom eigenen Benutzer entkoppelt werden und zurück an das System gekoppelt werden.

```
det 5530
DASD 5530 DETACHED
Ready; T=0.01/0.01 11:47:23
det 5535
DASD 5535 DETACHED
Ready; T=0.01/0.01 11:47:26
attach 5530 to system
DASD 5530 ATTACHED TO SYSTEM LIN001
Ready; T=0.01/0.01 11:47:31
attach 5535 to system
DASD 5535 ATTACHED TO SYSTEM CPY001
Ready; T=0.01/0.01 11:47:44
```

Die Übersicht der DASDs zeigt nun das Volume mit der Gerätenummer 5535 mit dem neuen Label CPY001.

```
q dasd
DASD 0ACC CP OWNED 440RES 106
DASD 0ACD CP OWNED 440W01 42
DASD 0ACE CP OWNED 440W02 20
DASD 552B CP SYSTEM DEVNO 3
DASD 552C CP SYSTEM DEVNO 3
DASD 552D CP SYSTEM DEVNO 3
DASD 552E CP SYSTEM DEVNO 3
DASD 552F CP SYSTEM DEVNO 3
DASD 5530 CP SYSTEM LIN001 0
DASD 5531 CP SYSTEM ZOSC01 14
DASD 5532 CP SYSTEM LIN002 0
DASD 5533 CP SYSTEM LIN003 0
DASD 5534 CP SYSTEM LIN004 0
DASD 5535 CP SYSTEM CPY001 0
DASD 5539 ATTACHED TO ZOS001 5539 R/W FR5539
Ready; T=0.01/0.01 11:48:49
```

Das Kopieren von Minidisks ist auf die gleiche Art und Weise möglich. Es ist wichtig, dass Ein- und Ausgabegeräte vom gleichen Typ sind und die gleiche Größe haben.

Für das Kopieren von Minidisks gibt es noch einen anderen, einfacheren Mechanismus, der im Kapitel 6.2.3 beschrieben wird.

## 4 Einrichtung des Gast-VM-Systems

Das als Cloning-System verwendete z/VM wird wie in Kapitel 1.3.3 beschrieben als Gast eines anderen z/VM (Basis-z/VM) betrieben. Die Definition des Cloning-Systems findet im Basis-z/VM statt.

Bevor nun das Cloning stattfinden kann, muss das z/VM-System eingerichtet werden. Dieses Kapitel beschreibt diese Konfiguration. Informationen zu aktuellen und vorherigen Versionen von z/VM lassen sich in der z/VM Internet Library [14] finden.

Wenn bereits ein z/VM vorhanden ist, werden lediglich die Informationen über die lokalen Hardware-Ressourcen benötigt, die in diesem Kapitel zu finden sind.

### 4.1 Zugriff auf das z/VM-System

Zur Konfiguration und Wartung des Cloning-Systems wird eine Verbindung mit dem Basis-z/VM-System (*UNIZVM52*), in dem das Cloning-z/VM-System als virtuelle Maschine (Gast) betrieben werden soll, hergestellt. Dafür wird ein 3270-Terminal-Emulator (kurz: 3270-Client) verwendet. Das Basis-z/VM-System der Universität Leipzig ist über die öffentliche IP-Adresse 139.18.4.49 erreichbar. Der 3270-Client zeigt den Login-Screen des Basis-z/VM-Systems:

```
z/VM ONLINE

          / VV          VVV MM          MM
         / VV          VVV MMM         MMM
        / VV          VVV MMMM        MMMM
       / VV          VVV MM MM MM MM
      / VV          VVV MM MM MM
     / VVVV         MM M MM
    / VVV          MM MM
   / V            MM MM

built on IBM Virtualization Technology

Fill in your USERID and PASSWORD and press ENTER
(Your password will not appear when you type it)
USERID  ==>
PASSWORD ==>
COMMAND ==>

RUNNING UNIZVM52
```

Das eine Verbindung mit dem Basis-System besteht, ist am Namen des Systems in der

unteren rechten Ecke erkennbar (*UNIZVM52*).

Das Gastsystem hat den Namen *ZVMND*. Zur Konfiguration und Wartung, aber zum Beispiel auch zum Start des Gastsystems, loggt man sich als dieser Benutzer an der Login-Maske mit der User-ID *ZVMND* und zugehörigem Passwort ein.

Um auf eine laufende virtuelle Maschine als Benutzer zuzugreifen, stehen verschiedene Möglichkeiten zur Verfügung. Wenn die virtuelle Maschine es unterstützt, kann mit dem Kommando *DIAL* die 3270-Session direkt an die Gast-Maschine weitergeleitet werden. Wenn die Gast-Maschine über Netzwerk erreichbar ist, ist auch eine direkte Verbindung mit möglich. Dabei sind auch andere Zugriffsmethoden wie *FTP* oder *SSH* denkbar, wenn sie von der Gastmaschine bereitgestellt werden.

Das z/VM-System für das Cloning-System, mit dem sich diese Diplomarbeit beschäftigt, hat keine öffentliche IP-Adresse. Der Zugriff auf das lokale Netz ist jedoch über eine OpenVPN-Verbindung möglich, die Zugriff auf das private Netzwerk ermöglicht, an welches das Gast-z/VM-System angeschlossen ist.

Im Folgenden sollen diese zwei Zugriffsmethoden genauer beleuchtet werden.

#### **4.1.1 Zugriff über Basis-z/VM-System**

Das Basis-z/VM-System hat die öffentliche IP-Adresse 139.18.4.49. Zu dieser IP-Adresse wird eine Verbindung mit einem 3270-Client hergestellt. Nach dem Verbindungsaufbau erscheint der Login-Screen. Mit dem Kommando *DIAL* wird die 3270-Session zu dem Gast *ZVMND* weitergeleitet:

```

z/VM ONLINE

          / VV          VVV MM          MM
          / VV          VVV MMM         MMM
ZZZZZZ / VV          VVV MMMM        MMMM
          / VV          VVV MM MM MM MM
          / VV          VVV MM MMM MM
          / VV          VVV MM M MM
ZZZZZZ / VVV          MM          MM
          / V          MM          MM

built on IBM Virtualization Technology

Fill in your USERID and PASSWORD and press ENTER
(Your password will not appear when you type it)
USERID  ==>
PASSWORD ==>

COMMAND ==> DIAL ZVMND

RUNNING UNIZVM52

```

Nach dem *DIAL*-Kommando erscheint der Login-Screen des Gastsystems:

```

z/VM ONLINE

Georg - Testsystem

          / VV          VVV MM          MM
          / VV          VVV MMM         MMM
ZZZZZZ / VV          VVV MMMM        MMMM
          / VV          VVV MM MM MM MM
          / VV          VVV MM MMM MM
          / VV          VVV MM M MM
ZZZZZZ / VVV          MM          MM
          / V          MM          MM

built on IBM Virtualization Technology

Fill in your USERID and PASSWORD and press ENTER
(Your password will not appear when you type it)
USERID  ==>
PASSWORD ==>

COMMAND ==>

RUNNING GEORGV1

```

Dass nun eine Verbindung mit dem Gastsystem besteht, zeigt der System-Name unten rechts in der Ecke. Das Gastsystem nennt sich selbst „GEORGV1“. Der User-Name, den das System im Basis-z/VM-System hat (ZVMND), muss also nicht mit dem internen Namen des Gastes korrelieren. Weiterhin besitzt das Gastsystem einen leicht ange-

passten Login-Screen. Von hier aus kann nun ein Login zum Gast-z/VM-System stattfinden.

### 4.1.2 Zugriff über OpenVPN

Das Gast-z/VM-System hat die interne IP-Adresse 10.0.2.253. Der OpenVPN-Tunnel stellt eine Verbindung mit einem Linux-Gastsystem her, der als Gateway für die lokalen Netzwerke dient, die im Basis-z/VM-System oder im Gast-z/VM-System – dem Cloning-System – definiert sind. Dieser Linux-Gast hat eine öffentliche IP-Adresse, die über das Internet erreichbar ist. Auf der Linux-Maschine läuft ein OpenVPN-Server.

Nun wird ein OpenVPN-Client benötigt, der dort betrieben wird, von wo aus auf das private Netz zugegriffen werden soll. Zur Authentifizierung dienen Zertifikate für Client und Server. Die Konfigurationsdatei für den Client sieht wie folgt aus:

```
client
dev tap
proto udp
remote 139.18.4.37 1194
resolv-retry infinite
ca unileip-ca.crt
cert client.crt
key client.key
ns-cert-type server
nobind
comp-lzo
verb 4
```

Nach dem Start des Clients wird ein OpenVPN-Tunnel mit dem Rechner 139.18.4.37 hergestellt. Für den Server ist das Zertifikat „unileipzig-ca.crt“ angegeben, der Client besitzt ein Zertifikat (client.crt) sowie einen privaten Schlüssel (client.key).

In der Server-Konfiguration sind folgende Zeilen wichtig:

```
server 10.0.3.0 255.255.255.0
push "route 10.0.1.0 255.255.255.0"
push "route 10.0.2.0 255.255.255.0"
push "route 10.0.10.0 255.255.255.0"
```

Die erste Zeile stellt den Adress-Pool für die OpenVPN-Clients bereit. Die Push-Einträge teilen dem Client mit, welche Netze über den VPN-Tunnel erreichbar sind. Der zweite Push-Eintrag ermöglicht somit den Zugriff auf das Netzwerk 10.0.2.0/24.

Nachdem der OpenVPN-Tunnel aufgebaut wurde, ist der direkte Zugriff auf das Gast-z/VM-System möglich. Dafür wird eine Verbindung mit einem 3270-Client zur IP-Adresse 10.0.2.253 aufgebaut und es erscheint der z/VM-Login-Screen des Cloning-Systems (GEORGVM1).

## 4.2 Konfiguration des Gast-z/VM-Systems im Basis-System

Im Basis-z/VM wird durch einen User-Directory-Eintrag das Gastsystem definiert, der alle benötigten Ressourcen bereitstellt. Der folgende Ausschnitt zeigt diesen Eintrag.

```
USER ZVMND SECRET 4G 7G G
CPU 01 CPUID 111111
CPU 02 CPUID 111222
IPL CMS PARM AUTOCR
MACH ESA 2
OPTION QUICKDSP MAINTCCW
POSIXINFO UID 12823
*-----*
* Z/VM 2ND LEVEL
*-----*
SHARE RELATIVE 2000
DEDICATE 0ACC 0ACC
DEDICATE 0ACD 0ACD
DEDICATE 0ACE 0ACE
CONSOLE 0100 3270 A
SPECIAL 0101 3270
SPECIAL 0102 3270
SPECIAL 0103 3270
SPECIAL 0104 3270
SPECIAL 0105 3270
SPECIAL 0106 3270
SPECIAL 0107 3270
SPECIAL 0108 3270
SPECIAL 0109 3270
SPECIAL 010A 3270
SPECIAL 010B 3270
SPECIAL 010C 3270
SPECIAL 010D 3270
SPECIAL 010E 3270
SPECIAL 010F 3270
SPOOL 000C 2540 READER *
SPOOL 000D 2540 PUNCH A
SPOOL 000E 3203 A
LINK MAINT 0190 0190 RR
LINK MAINT 019E 019E RR
LINK MAINT 019D 019D RR
LINK LINUX390 191 191 RR
```

Die erste Zeile zeigt die Definition des Benutzers. Der Name der virtuellen Maschine ist „ZVMND“, das Passwort ist „SECRET“ und es werden vier Gigabyte Arbeitsspeicher zugeteilt, der Benutzer kann ihn aber auf maximal sieben Gigabyte erhöhen. Der Benutzer gehört der Benutzerklasse G an.

In den weiteren Zeilen werden die Systemeigenschaften definiert. Im System sind zwei CPUs definiert. Beim Starten der Maschine (IPL, Initial Program Load) wird zunächst CMS gestartet. Der Start von z/VM erfolgt dann über einen weiteren IPL in der CMS-Datei *ZVMND EXEC*. Mittels DEDICATE werden dem System die 3390-DASDs zugewiesen, auf denen sich z/VM befindet. Diese haben die Adressen 0ACC, 0ACD und 0ACE. Um auch ohne TCP/IP auf die virtuelle Maschine zuzugreifen, werden eine Konsole (0100) und 3270-Terminals (0101-010F) definiert. Letztere ermöglichen einen Zugriff auf die virtuelle Maschine mittels *DIAL*.

Zusätzliche Geräte werden in der Datei *ZVMND EXEC* erstellt. Diese wird über die CMS-Datei *PROFILE EXEC* gestartet, welche nach dem Login beim Start von CMS

ausgeführt wird.

Der Inhalt der Datei *ZVMND EXEC* sieht wie folgt aus:

```
/* REXX PROFILE EXEC FOR VM GUEST          */
/*                                          */
SET CMSTYPE HT
'CP SET MSG ON'
'CP SET EMSG ON'
'CP SET RUN ON'
'CP SET PF12 RETRIEVE'
'CP SET PF24 RETRIEVE'
'CP term charde1 off'
'CP LINK TCPIP 592 592 RR'
'CP DEF NIC FF00 TYPE HIPER'
'CP COUPLE FF00 TO SYSTEM LINHIPER'
'ACC 592 T'
SYNONYM SYN
'CP SP CONS * START'
SET RUN ON
SET CMSTYPE RT
/* SAY ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' */
/* SAY ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' */
'CP DEFINE STORAGE 4G' '15'X 'IPL ACC'
```

Diese Anweisungen enthalten unter anderem die Definition des Netzwerks (grau hervorgehoben). Die Konfiguration bindet die virtuelle Maschine an das Gast-Netzwerk des First-Level-z/VM-Systems. Alternativ könnte dies auch über einen Eintrag im z/VM User Directory definiert werden.

Die letzte Definition erhöht den Hauptspeicher für die virtuelle Maschine auf 4GB und startet anschließend das System. Der Start des Betriebssystems (IPL) findet über den Befehl *IPL ACC* statt. *ACC* steht hierbei für die Adresse des Geräts (DASD), von dem gestartet werden soll.

Im Folgenden wird nur noch auf das Gast-z/VM-System Bezug genommen. Das Basis-z/VM *UNIZVM52* wird nicht weiter betrachtet.

## 5 Basis-Konfiguration des Systems

Nach dem erstmaligen Start findet eine Basis-Konfiguration statt. Diese umfasst die Installation des Netzwerks sowie die Einrichtung von DIRMAINT.

### 5.1 Das Netzwerk

Das Betriebssystem z/VM stellt den TCP/IP-Stack über einen separaten Gast bereit, der im Normalfall *TCPIP* heißt [15]. Es ist auch möglich, mehrere TCP/IP-Stacks parallel zu verwenden, was aber in diesem System keine Verwendung findet.

Zur Administration von TCP/IP wird der Benutzer *TCPMAINT* verwendet. Dieser hat Zugriff auf Minidisks, die spezielle Netzwerk-Programme enthalten. *TCPMAINT* gehört der Benutzerklasse B an – so kann eine separate Person mit der TCP/IP-Konfiguration betraut werden, ohne ihr kompletten Zugriff auf das System zu gewähren.

Zunächst folgt eine kurze Einführung in die Netzwerk-Konzepte von z/VM.

#### 5.1.1 Netzwerk-Konzepte unter z/VM

Die virtuellen Netze unter z/VM lassen sich gut mit realen Netzwerken vergleichen. Für die Konzepte in der realen Welt gibt es jeweils ein Pendant. Im Folgenden sollen diese Parallelen aufgezeigt und an einem Beispiel verdeutlicht werden.

Ein Netzwerk besteht aus:

- Netzwerkkarten, die einem Rechner die Anbindung an das Netzwerk erlauben,
- einem oder mehreren Switches, über welche die Rechner miteinander kommunizieren können, sowie
- Netzwerk-Kabeln, die die Verkabelung zwischen Rechnern und Switch physisch realisieren.

Unter z/VM gibt es für diese 3 Komponenten ein virtuelles Gegenstück.

Eine Netzwerkkarte wird für einen Gast als Gerät definiert:

- DEFINE NIC FF00 TYPE HIPER

Dieser Befehl erzeugt die virtuelle Netzwerkkarte für einen bestimmten Netzwerktyp (Hipersockets) unter der Adresse FF00. Das entspricht dem Einbauen einer Netzwerkkarte in einen Rechner.

Als Grundlage für die Kommunikation wird ein lokales Netzwerk definiert:

- `DEFINE LAN LINHIPER TYPE HIPER`

Dieses Kommando erzeugt ein virtuelles Netzwerk eines bestimmten Typs (Hipersockets) unter einem bestimmten Namen, an das Gastsysteme angeschlossen werden können. Das entspricht dem Bereitstellen eines Netzwerk-Switches. In der realen Welt können theoretisch mehrere Switches miteinander verbunden werden, um die Anzahl der Anschlussmöglichkeiten zu erhöhen. Das ist unter z/VM nicht der Fall. Das Netzwerk entspricht der Gesamtheit aller zusammengeschlossenen Switches.

Nach Definition des Netzwerkes ist zunächst kein Gastsystem angeschlossen. Um dies zu ändern, findet die virtuelle Verkabelung statt. Dabei wird definiert, welche Netzwerkkarte mit welchem Netzwerk (Switch) verbunden werden soll.

- `COUPLE FF00 TO SYSTEM LINHIPER`

Dieses Kommando verbindet die Netzwerkkarte, die unter der Adresse FF00 angelegt wurde, mit dem Netzwerk mit dem Namen LINHIPER. Netzwerkkarte und Netzwerk müssen dafür vom gleichen Typ sein, in diesem Beispiel vom Typ Hipersocket. Nun ist die Netzwerkkarte mit dem Netzwerk verbunden und kann mit anderen Geräten, die ebenfalls an das gleiche Netzwerk angeschlossen sind, kommunizieren.

Um das Netzwerk mit anderen Netzen zu verbinden, wird ein Router benötigt. Diese Funktion übernimmt in diesem Fall der TCP/IP-Stack von z/VM. Es ist jedoch auch möglich, ein anderes am Netzwerk beteiligtes Gastsystem zu verwenden.

Die Anbindung des routenden Gasts beginnt genauso wie bei einem normalen Gast, für ihn wird eine Netzwerkkarte definiert und diese wird zunächst mit dem internen Netzwerk verbunden, welchem er die Kommunikation mit externen Netzwerken (Internet) ermöglichen soll. Zusätzlich bekommt er eine zweite Netzwerkkarte, die mit einem externen Netz verbunden ist.

Im Fall des Cloning-Systems ist die zweite Netzwerkkarte eine externe Netzwerkkarte, also ein aus Sicht des Gast-z/VM-Systems reales Gerät, das nicht über *DEFINE NIC* erstellt werden muss. Dass dieses wiederum nur virtuell ist, da das Betriebssystem in einem anderen z/VM betrieben wird, spielt für das Cloning-z/VM keine Rolle.

## 5.1.2 Aufbau des Netzwerks

Abbildung 3 zeigt die Netzwerk-Konfiguration für das Cloning-System.

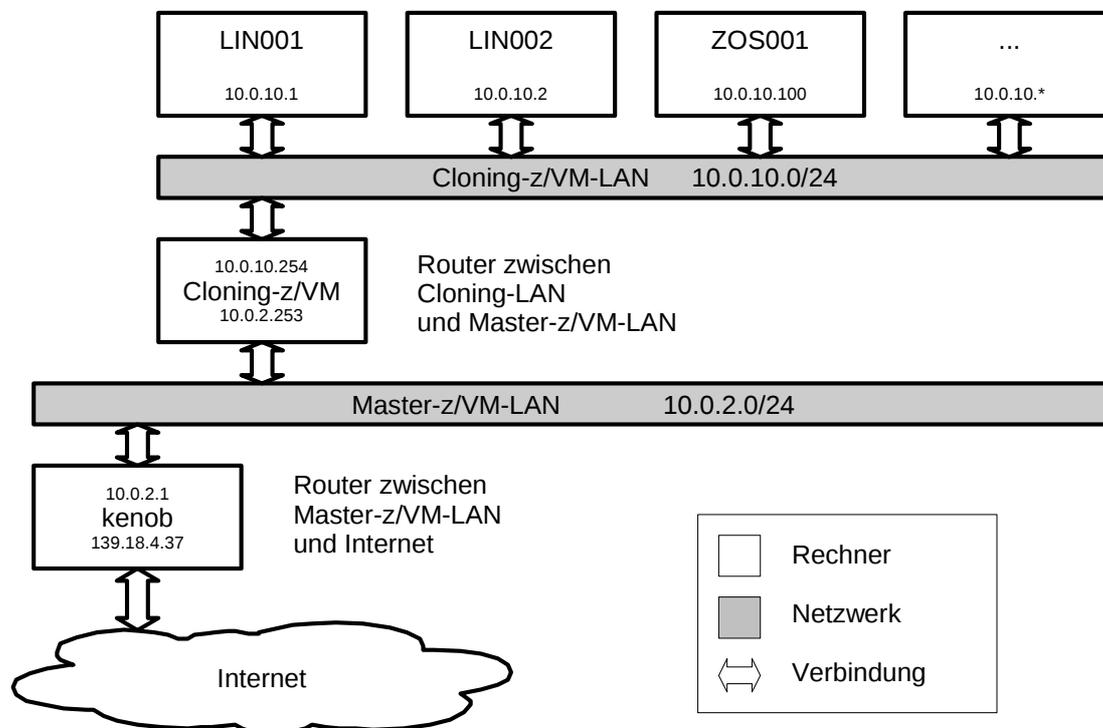


Abbildung 3: Netzwerk-Konfiguration

Das Master-z/VM-System verwendet das Netzwerk 10.0.2.0/24. Die CIDR-Notation „/24“ bedeutet, dass die ersten 24 Bit der Netzwerkwerk-Maske gesetzt sind und die verbleibenden 8 Bit für die verschiedenen Rechner innerhalb des Netzes verwendet werden können (entspricht der Netzwerk-Maske 255.255.255.0). Damit stehen in diesem Netzwerk die IP-Adressen 10.0.2.1 bis 10.0.2.254 zur Verfügung. Als Gateway zum Internet dient der Rechner *kenob*, der in diesem Netzwerk die private IP-Adresse 10.0.2.1 und für den Zugriff auf das Internet die öffentliche IP-Adresse 139.18.4.37 hat.

Mit diesem Netzwerk soll die externe Netzwerkkarte des Cloning-z/VM-Systems verbunden werden. Dafür ist die IP-Adresse 10.0.2.253 vorgesehen.

Für die Gäste des Cloning-z/VM-Systems soll ein weiteres Netzwerk 10.0.10.0/24 verwendet werden. Damit stehen in diesem Netzwerk die IP-Adressen 10.0.10.1 bis 10.0.10.254 zur Verfügung. Als Gateway wird der TCP/IP-Stack des Cloning-z/VM-Systems verwendet. Dieser wird ebenfalls mit diesem Netzwerk verbunden – mit der IP-Adresse 10.0.10.254. Diese IP-Adresse wird für alle weiteren Gäste dieses Netzes als

Gateway verwendet.

### 5.1.3 Konfiguration des TCP/IP-Stacks

Der TCP/IP-Stack wird über die beiden Dateien *PROFILE TCPIP* und *TCPIP DATA* konfiguriert [15].

Dabei werden zum einen die Geräte bekanntgemacht, die das Netzwerk physisch bereitstellen (wenn auch virtuell) und zum anderen die Netze beschrieben, die im System vorhanden sind.

Für die Konfiguration existiert das (aus Sicht des Cloning-z/VM) reale Gerät FF00 (bereitgestellt durch das Basis-z/VM) sowie das virtuelle Gerät FF10.

Das virtuelle Gerät FF10 wird für den User TCPIP im *z/VM User Directory* definiert:

```
NICDEF FF10 TYPE HIPERSOCKETS LAN SYSTEM LINHIPER
```

Durch diese Zeile wird zugleich das systemweite lokale Netzwerk „LINHIPER“ definiert und es findet eine Verkabelung (COUPLE ...) zwischen der Netzwerkkarte und dem Netzwerk statt.

#### **PROFILE TCPIP**

Die Datei *PROFILE TCPIP* heißt auf dem Cloning-System wie das System selbst *GEORGVM1 TCPIP*.

In dieser Datei erfolgt zunächst die Definition von einigen Eigenschaften des TCP/IP-Stacks, damit dieser zum Beispiel als Router fungiert. Details dazu befinden sich in den Handbüchern für TCP/IP unter z/VM [15] [16].

Danach werden die Netzwerk-Geräte definiert und konfiguriert:

```
DEVICE $NETWORK-FF00 HIPERS FF00 PORTNAME LINHIPER  
LINK $NETWORK-FF00 QDIOIP $NETWORK-FF00  
DEVICE $NETWORK-FF10 HIPERS FF10 PORTNAME LINHIPER  
LINK $NETWORK-FF10 QDIOIP $NETWORK-FF10
```

Hier werden unter den Adressen FF00 und FF10 zwei Netzwerk-Geräte definiert. Dabei ist zu beachten, dass bei Netzwerkadaptern immer drei Geräte am Stück definiert werden. Dies sind je ein Kanal für Lese- und Schreibkontrolle sowie ein Kanal für den Datenaustausch. Am Beispiel von FF00 werden implizit die Geräte FF00, FF01 sowie FF02 definiert.

Es folgt die Definition der IP-Adressen:

```
HOME
10.0.2.253  $NETWORK-FF00
10.0.10.254 $NETWORK-FF10
```

Die Netzwerkkarte FF00 bekommt die IP-Adresse 10.0.2.253 zugewiesen, die andere Karte die 10.0.10.254.

Nun werden die Routen-Einträge konfiguriert:

```
GATEWAY
10      = $NETWORK-FF00 1492 0.255.255.0 0.0.2.0
10      = $NETWORK-FF10 1492 0.255.255.0 0.0.10.0
DEFAULTNET 10.0.2.1 $NETWORK-FF00 1492 0
```

Hierbei wird definiert, dass das Netzwerk 10.0.2.0/24 über das Gerät FF00 zu erreichen ist und das Netzwerk 10.0.10.0/24 über FF10.

Als Default-Route wird die IP-Adresse 10.0.2.1 definiert, die über das Interface FF00 angesteuert wird. Die Maximum Transmission Unit (MTU) wird für alle Links auf 1492 Bytes gesetzt.

Zuletzt werden die Netze noch gestartet.

```
START $NETWORK-FF00
START $NETWORK-FF10
```

Beim Starten des Benutzers TCPIP wird das Netzwerk bereitgestellt und kann von anderen Gästen benutzt werden.

Einen Überblick über die definierten Netze zeigt der Befehl *QUERY LAN*. Eine detailliertere Ausgabe liefert der Zusatz *DET*.

```
q lan det
LAN SYSTEM LINHIPER  Type: HIPERS  Active: 4  MAXCONN: INFINITE
PERSISTENT UNRESTRICTED MFS: 16384  ACCOUNTING: OFF
Adapter Owner: SYS1Z01  NIC: FF00  Name: UNASSIGNED
Adapter Owner: SYS1Z02  NIC: FF00  Name: UNASSIGNED
Adapter Owner: TCPIP  NIC: FF10  Name: LINHIPER
Device: FF12  Unit: 002  Role: DATA
VLAN: ANY  Assigned by user
Unicast IP Addresses:
 10.0.2.253  Mask: 255.0.0.0
 10.0.10.254  Mask: 255.0.0.0
Multicast IP Addresses:
224.0.0.1  MAC: 01-00-5E-00-00-01
Adapter Owner: ZOS001  NIC: FF00  Name: UNASSIGNED
LAN SYSTEM LINQDIO  Type: QDIO  Active: 1  MAXCONN: INFINITE
PERSISTENT UNRESTRICTED MFS: 8192  ACCOUNTING: OFF
Adapter Owner: TCPIP  NIC: EE00  Name: UNASSIGNED
Ready; T=0.01/0.01 16:34:06
```

Ähnlich wie unter Linux lässt sich unter z/VM die Konfiguration der Netzwerkkarten

mit dem Befehl *ifconfig* anzeigen.

```
ifconfig
$NETWORK-FF00 inet addr: 10.0.2.253 mask: 255.255.255.0
  UP BROADCAST MULTICAST MTU: 1492
  vdev: FF00 rdev: FF00 type: HIPERS
  vlan: ANY
  RX bytes: 1639 TX bytes: 3587

$NETWORK-FF10 inet addr: 10.0.10.254 mask: 255.255.255.0
  UP BROADCAST MULTICAST MTU: 1492
  vdev: FF10 type: HIPERS
  LAN owner: SYSTEM name: LINHIPER
  vlan: ANY
  RX bytes: 0 TX bytes: 116
Ready; T=0.04/0.05 16:34:45
```

Damit ist die Einrichtung des TCP/IP-Stacks von z/VM abgeschlossen.

### **Anbindung von Gastsystemen an das Netzwerk**

Damit ein anderer Gast (zum Beispiel ein Linux-System) das Netzwerk benutzen kann, benötigt er eine virtuelle Netzwerkkarte, die an das System-LAN *LINHIPER* angeschlossen wird. Dies kann zum Beispiel durch eine Definition im *z/VM User Directory* geschehen:

```
SPECIAL FF00 HIPER 3 SYSTEM LINHIPER
```

Der Gast bekommt eine virtuelle Netzwerkkarte FF00 zugewiesen, die gleich mit dem System-LAN *LINHIPER* verbunden wird. Da es sich um ein virtuelles Gerät handelt, überschneidet es sich nicht mit dem realen Gerät FF00 des Systems.

Ohne das User Directory zu verändern, könnte ein Gast die gleiche Konfiguration durch mehrere CP-Kommandos erreichen (zum Beispiel als Einträge in der Datei *PROFILE EXEC*):

```
DEFINE NIC FF00 TYPE HIPER
COUPLE FF00 TO SYSTEM LINHIPER
```

## **5.2 DIRMAINT und DATAMOVE**

Mit dem Dienst DIRMAINT wird das *z/VM User Directory* verwaltet. Zunächst muss der Dienst konfiguriert werden, was in diesem Abschnitt beschrieben wird. Im Anschluss folgt eine Übersicht für typische Aufgaben im Zusammenhang mit DIRMAINT.

Der Dienst ist, genauso wie laufende virtuelle Maschinen, auch nur ein Gastsystem unter z/VM [17]. Diese Maschine empfängt Befehle über den *DIRMAINT*-Befehl (kurz:

*DIRM*, [18]) und verändert das User Directory entsprechend. Dabei verifiziert die Maschine die eingegeben Befehle und gibt bei fehlerhaften Eingaben eine entsprechende Rückmeldung. Für Änderungen am User Directory verwendet DIRMAINT das Kommando *DIRECTXA*. Ohne DIRMAINT müssen das User Directory manuell gepflegt und Änderungen mit dem Befehl *DIRECTXA* aktivieren werden.

Ein weiterer Dienst ist DATAMOVE. Diese Maschine ist für die Verwaltung von Festplattenspeicher zuständig. Dazu gehört das Formatieren von Festplatten, das Kopieren von Daten sowie die Definition von Volume Labels. Der Dienst übernimmt auf Wunsch auch das Formatieren von Festplattenspeicher, der von einer Maschine entfernt wird, um Daten sicher zu löschen.

### 5.2.1 Konfiguration von DIRMAINT

Der Nutzer DIRMAINT ist standardmäßig so konfiguriert, dass er sich nicht einloggen kann. Um dies zu ändern, muss das z/VM User Directory angepasst werden.

Um zu verhindern, dass sich ein Benutzer einloggen kann, steht an der Stelle des Passworts das Schlüsselwort „NOLOG“.

```
USER DIRMAINT NOLOG 32M 64M BDG
```

An diese Stelle wird ein Passwort für diesen Benutzer gesetzt. Die gleiche Änderung muss auch beim Benutzer DATAMOVE vorgenommen werden.

Nachdem dort ein Passwort eingetragen wurde, kann das User Directory wieder geschlossen werden. Aktiviert wird das veränderte User Directory mit *DIRECTXA* (User Directory befindet sich auf der C-Minidisk):

```
DIRECTXA USER DIRECT C
```

Nun können die Benutzer DIRMAINT und DATAMOVE gestartet werden. Dies erfolgt als Benutzer MAINT mit dem Befehl *XAUTOLOG*.

```
XAUTOLOG DIRMAINT  
XAUTOLOG DATAMOVE
```

Um eine Kopie des aktuell laufenden User Directory zu erhalten, gibt es den *DIRMAINT*-Befehl *USER*.

```
dirm user w
DVHXT1191I Your USER request has been sent for processing.
Ready; T=0.05/0.06 17:03:42
DVHREQ2288I Your USER request for MAINT at * has been accepted.
RDR FILE 0569 SENT FROM DIRMAINT PUN WAS 0063 RECS 2171 CPY 001 A NOHOLD NOKEEP
DVHREQ2289I Your USER request for MAINT at * has completed; with RC = 0.
```

Der Parameter „W“ (*WITHPASS*) sorgt dafür, dass in der Kopie auch die Passwörter der einzelnen Gäste enthalten sind. Sollen die Passwörter nicht angezeigt werden, lassen sich diese mit dem Parameter „N“ (*NOPASS*) verbergen.

Die Datei befindet sich nun auf dem virtuellen Kartenleser (Reader) und muss noch eingelesen werden. Um eine Übersicht über die Dateien auf dem Reader zu bekommen, wird das Kommando *RDRLIST* (kurz: *RDRL*) ausgeführt.

```
MAINT  RDRLIST  A0  V 164  Trunc=164  Size=1  Line=1  Col=1  Alt=0
Cmd  Filename  Filetype  Class  User  at  Node  Hold  Records  Date  Time
USER  WITHPASS  PUN  A  DIRMAINT  GEORGM1  NONE  2171  11/22  17:08:22
```

Es können sich mehrere Dateien auf dem Reader befinden. Es wird die Zeile „USER WITHPASS“ ausgewählt. Mit F9 (Receive) wird die Datei empfangen und auf die A-Minidisk geschrieben. Dort befindet sich nun die Datei *USER WITHPASS* (bzw. *USER NOPASS*), die sich in einem Editor betrachten lässt.

## 5.2.2 Die Festplatten

Für die Verwaltung der Festplatten durch DIRMAINT müssen Pools von Festplattenspeicher angelegt werden, aus denen DIRMAINT Speicher verteilt. Dabei werden Minidisks erstellt, deren Größe durch eine beliebige Anzahl an Zylindern angegeben wird.

Um auch kleinere Einheiten als eine komplette Festplatte verfügbar zu machen, führt z/VM wie in Kapitel 3.2.2 beschrieben das Konzept der Minidisks (MDISK) ein. Damit lässt sich eine DASD an Zylinder-Grenzen beliebig aufteilen.

Die Festplatten-Allokationen von DIRMAINT werden in der Datei *EXTENT CONTROL* auf der Minidisk mit der Adresse 1DF verwaltet.

Hierbei werden zunächst Regionen auf DASDs definiert, die DIRMAINT verwaltet.

```

:REGIONS.
 *RegionId   VolSer   RegStart   RegEnd     Type
440RES      440RES   START      3338       3390-03
440W01      440W01   START      3338       3390-03
440W02      440W02   START      3338       3390-03
440W03      440W03   START      3338       3390-03
440W04      440W04   START      3338       3390-03
LIN001      LIN001   START      3338       3390-03
LIN002      LIN002   START      3338       3390-03
LIN003      LIN003   START      3338       3390-03
LIN004      LIN004   START      3338       3390-03
:END.

```

Für die Verwaltung der Regionen werden Gruppen definiert.

```

:GROUPS.
 *GroupName RegionList
USERGRP 440W02
LINUX   LIN001 LIN002 LIN003 LIN004
:END.

```

Beim Erstellen einer Minidisk einer bestimmten Größe für einen Benutzer kann nun die Gruppe „LINUX“ angegeben werden. DIRMAINT wird nun einen Speicherbereich auf einer der DASDs LIN001, LIN002, LIN003 oder LIN004 suchen, die diese Minidisk aufnehmen kann. Der Befehl zur Erstellung einer Minidisk ist *DIRM AMDISK*.

```
DIRM FOR LIN001 AMDISK 200 X AUTOG LINUX
```

Dieser Befehl erstellt eine Minidisk für den Benutzer LIN001 mit 200 Zylindern in einer freien Region aus dem Filepool „LINUX“.

### 5.3 Gast-Betriebssysteme (Linux, z/OS) vorbereiten

Bevor das Gast-Betriebssystem installiert bzw. gestartet wird, müssen noch verschiedene Schritte erledigt werden. Dies betrifft zunächst die Planung, das heißt welche Ressourcen ein Gastsystem benötigt. Anschließend müssen diese Überlegungen durch Definitionen im z/VM User Directory umgesetzt werden. Informationen zum Betrieb von Gästen befinden sich unter [19].

#### 5.3.1 Planungen für das Gastsystem

Bei der Planung für das Gastsystem gilt es, eine vernünftige Auswahl an Systemressourcen für dieses System zu machen. Dies entspricht der Auswahl an Komponenten beim Kauf eines neuen Systems, also Hauptspeicher, Anzahl der CPUs, Festplatten und Peripherie.

Je nach Einsatzzweck des Gastsystems können sich die Anforderungen an die Größe des

Hauptspeichers oder die Anzahl der CPUs unterscheiden. Ein weiterer Punkt ist die Größe des Festplattenspeichers. Ein DNS-Server benötigt zum Beispiel deutlich weniger Festplattenkapazität als ein Datei- oder Datenbank-Server.

Diese Auswahl lässt sich später immer noch ändern bzw. erweitern. Für die Testsysteme sollte eine Konfiguration mit minimalem Ressourcenbedarf gewählt werden, je nach Einsatzzweck können später noch zusätzliche Festplatten oder Hauptspeicher zu einem System hinzugefügt werden.

Als weitere Geräte enthalten die Gastsysteme zunächst nur eine Netzwerkkarte, denkbar wären jedoch auch mehrere Netzwerkkarten, beispielsweise für einen Router, oder Band-Laufwerke oder -Libraries für ein Backup-System.

### 5.3.2 Definition des Benutzers

Jedes Gastsystem benötigt einen Benutzer (User) im z/VM User Directory. Hierfür gibt es zwei verschiedene Möglichkeiten. Zum Einen lassen sich Benutzer direkt anlegen. Alternativ ist eine vereinfachte Definition neuer Benutzer mit Hilfe von Vorlagen möglich.

Für das direkte Anlegen eines Benutzers wird zunächst eine Datei auf einer CMS-Festplatte angelegt, die die User-Definition enthält. Für den Benutzer LIN001 wird zum Beispiel die Datei LIN001 DIRECT erstellt und mit der User-Definition gefüllt. Anschließend wird der Benutzer mittels *DIRMAINT* installiert. Dies geschieht mit dem folgenden Befehl:

```
DIRM ADD USER LIN001
```

Für das Anlegen eines Nutzers mit einer Vorlage wird bei der Definition auf diese Vorlage verwiesen, was die manuelle Vorbereitung der User-Definition unnötig macht. Ein Benutzer wird mit dem folgenden Kommando aus einer Vorlage erstellt:

```
DIRM ADD USER LIN001 LIKE LINPROTO
```

Ein Beispiel zum Erstellen eines Benutzers mittels Vorlagen wird im Kapitel 6.1.5 beschrieben.

## 6 Linux als Gastssystem unter z/VM

### 6.1 Linux installieren und einrichten

Für das Cloning soll zunächst ein Linux-System installiert werden, welches dann als Grundlage für weitere Clones dienen soll.

Dieses Kapitel beschäftigt sich mit den Schritten, die notwendig sind, um einfache Linux-Clones anzulegen. Es wird ebenfalls die Installation von Linux unter z/VM beschrieben.

#### 6.1.1 Vorbetrachtungen

Vor dem Beginn der Installation erfolgt eine Analyse der benötigten Hardware-Ressourcen. 128MB Arbeitsspeicher sowie zwei CPUs sind für ein Linux-System ausreichend. Für eine minimale Linux-Installation reicht die Plattenkapazität einer DASD vom Typ 3390 Model 3 aus. Darauf stehen ca. 2,2GB Speicherplatz zur Verfügung.

Um die Festplatten an ein System zu binden, stehen verschiedene Möglichkeiten zur Verfügung [13]. Eine DASD lässt sich direkt einem Gast zuweisen (DEDICATE) oder der Plattenspeicher wird über eine Minidisk bereitgestellt (MDISK). Handelt es sich hauptsächlich um Lese-Operationen, ist die Minidisk-Variante zu bevorzugen, da hier der Minidisk-Cache von z/VM für eine bessere Performance sorgen kann [20].

#### 6.1.2 Definition des Systemverwalters LNXMAINT

Das Linux-System soll auf einer DASD vom Typ 3390-3 Platz finden. Diese Festplatte wird das Root-Dateisystem beinhalten. Es ist allerdings auch möglich, mehrere DASDs oder Minidisks zu definieren und so z.B. eine separate */home*-Partition anzulegen.

Der Benutzer LNXMAINT ist für die Verwaltung der Linux-Gäste zuständig. Im folgenden Screenshot ist die Definition dieses Users zu sehen.

```
USER LNXMAINT SECRET 2M 16M BG
INCLUDE IBMDFLT
IPL CMS PARM AUTOOCR
IUCV ALLOW
IUCV ANY PRIORITY
MACH XA
OPTION V=R
XAUTOLOG AUTOLOG1 OP1 MAINT
MDISK 0191 3390 1466 100 440W02 MR
```

Die CMS-Minidisk 191 wird für alle weiteren Gäste zum Lesen freigegeben. Das Zu-

griffsrecht *MR* bedeutet, dass andere Gäste lesend auf die Festplatte zugreifen können, der schreibende Zugriff jedoch nur einem System gestattet ist. Wenn bereits ein anderes System das Schreibrecht besitzt (z.B. der User *MAINT*), so erhält der Gast nur Lesezugriff. Die Linux-Gäste werden so konfiguriert, dass sie nur lesenden Zugriff auf diese Minidisk bekommen.

### 6.1.3 Definition des Standard-Profiles für Linux-Gäste

Für die eigentlichen Linux-Gäste wird ein z/VM-Profil angelegt. Dieses Profil wird später bei der Definition der Linux-Gäste verwendet. Die definierten Eigenschaften werden von jedem Gast geerbt, der dieses Profil einbindet.

```
PROFILE LINDFLT
CLASS G
STORAGE 128M
MAXSTORAGE 2047M
CPU 00
CPU 01
IPL CMS PARM AUTOCR
IUCV ALLOW
MACHINE ESA
OPTION QUICKDSP
CONSOLE 0009 3215 T
SPECIAL FF00 HIPER 3 SYSTEM LINHIPER
SPOOL 000C 2540 READER *
SPOOL 000D 2540 PUNCH A
SPOOL 000E 1403 A
LINK MAINT 0190 0190 RR
LINK MAINT 019D 019D RR
LINK MAINT 019E 019E RR
LINK TCPMAINT 0592 0592 RR
LINK LNXMAINT 0191 0191 RR
```

Das Standard-Profil definiert für jeden Gast zwei CPUs (*CPU ..*) und 128MB Hauptspeicher (*STORAGE*). Der Hauptspeicher kann auf bis zu 2GB erweitert werden (*MAXSTORAGE*).

Zusätzlich werden die Standardgeräte Spool, Reader, Puncher und Console sowie Links zu fremden Minidisks definiert, unter anderem zur Minidisk 191 des Benutzers *LNX-MAINT*.

Auch die Netzwerk-Definition befindet sich bereits in diesem Profil. Dabei wird eine Hipersocket-Netzwerkkarte mit den Adressen *FF00*, *FF01* und *FF02* definiert und mit dem Switch *LINHIPER* verbunden.

### 6.1.4 Definition eines Prototypen

Um später auf einfache Art und Weise Gäste zum System hinzuzufügen, wird ein Prototyp definiert. Dieser dient *DIRMAINT* als Vorlage für alle Linux-Instanzen. Zur Definiti-

on wird die Datei *LINPROTO PROTODIR* mit folgendem Inhalt angelegt:

```
USER LINPROTO SECRET * * *  
INCLUDE LINDFLT
```

Nun wird der Prototyp dem System bekannt gemacht:

```
dirm file linproto protodir a  
PUN FILE 0365 SENT TO DIRMAINT RDR AS 0266 RECS 0010 CPY 001 0 NOHOLD NOKEEP  
DVHXMT1191I Your FILE request has been sent for processing.  
Ready; T=0.05/0.05 14:16:43  
DVHREQ2288I Your FILE request for MAINT at * has been accepted.  
DVHRCV3821I File LINPROTO PROTODIR A has been received; RC = 0.  
DVHREQ2289I Your FILE request for MAINT at * has completed; with RC = 0.
```

## 6.1.5 Definition eines Linux-Gastsystem

Um nun ein Linux-Gastsystem anzulegen, wird mit *DIRMAINT* mit Bezug auf den soeben erstellten Prototypen ein neuer Benutzer erstellt und ihm anschließend eine System-Disk bereitgestellt [18].

Für das Cloning-System soll eine Master-Installation durchgeführt werden. Das Erstellen neuer Clones wird durch das Kopieren dieser Installation erreicht.

### **Erstellen des neuen Benutzers**

Das Master-System soll den Namen „LINMSTR“ erhalten. Dafür wird zunächst der Benutzer angelegt:

```
DIRM ADD LINMSTR LIKE LINPROTO PW SECRET  
DVHXMT1191I Your ADD request has been sent for processing.  
Ready; T=0.05/0.05 14:16:59  
DVHREQ2288I Your ADD request for LINMSTR at * has been accepted.  
DVHBIU3450I The source for directory entry LINMSTR has been updated.  
DVHBIU3425I The next ONLINE will take place as scheduled.  
DVHREQ2289I Your ADD request for LINMSTR at * has completed; with RC = 0.  
DVHDRC3428I Changes made to directory entry LINMSTR have just been placed  
DVHDRC3428I online.
```

Damit ist der Benutzer erstellt. Es folgt die Definition der System-Festplatte.

### **Erstellen der System-Festplatte**

Dem Benutzer LINMSTR muss nun noch eine Minidisk zugewiesen werden, die das Root-Dateisystem aufnehmen soll.

Dafür gibt es zwei verschiedene Ansätze, die manuelle Definition einer Minidisk oder deren automatische Erstellung durch die Plattengruppen von DIRMAINT.

## Manuelle Auswahl der Minidisk

Bei der manuellen Auswahl muss zunächst ein freies Volume ausgewählt werden, welches die Minidisk aufnehmen soll. Der Befehl *QUERY DASD ALL* zeigt eine Übersicht über die vorhandenen DASDs des Systems.

```
Q DASD ALL
DASD 0ACC CP OWNED 440RES 94
DASD 0ACD CP OWNED 440W01 42
DASD 0ACE CP OWNED 440W02 18
DASD 552B CP SYSTEM DEVNO 1
DASD 552C CP SYSTEM DEVNO 1
DASD 552D CP SYSTEM DEVNO 1
DASD 552E CP SYSTEM DEVNO 1
DASD 552F CP SYSTEM DEVNO 1
DASD 5530 CP SYSTEM LIN001 0
DASD 5531 CP SYSTEM ZOSC01 0
DASD 5532 CP SYSTEM LIN002 0
DASD 5533 CP SYSTEM LIN003 0
DASD 5534 CP SYSTEM LIN004 0
DASD 5539 ATTACHED TO ZOS001 5539 R/W FR5539
DASD 0190 MNT190 , DASD 0191 ZVM191 , DASD 019D MNT19D , DASD 019E MNT19E
DASD 0401 MNT401 , DASD 0402 MNT402 , DASD 0405 MNT405 , DASD 0592 TCM592
DASD 5535 FREE , DASD 5536 FREE , DASD 5537 FREE , DASD 5538 FREE
```

Auf der DASD LIN001 soll nun eine Minidisk angelegt werden, die als System-Disk für den Gast LINMSTR dient.

```
dirm for LINMSTR amdisk 200 3390 1 3337 LIN001
DVHXT1191I Your AMDISK request has been sent for processing.
Ready; T=0.05/0.05 13:09:02
DVHREQ2288I Your AMDISK request for LINMSTR at * has been accepted.
DVHSCU3541I Work unit 30130903 has been built and queued for processing.
DVHSHN3541I Processing work unit 30130903 as MAINT from GEORGV1,
DVHSHN3541I notifying MAINT at GEORGV1, request 723 for LINMSTR
DVHSHN3541I sysaffin *; to: AMDISK 0200 3390 1 3337 LIN001
DVHBIU3450I The source for directory entry LINMSTR has been updated.
DVHDRC3428I Changes made to directory entry LINMSTR have just been
DVHDRC3428I placed online.
DVHSHN3430I AMDISK operation for LINMSTR address 0200 has finished (WUCF
DVHSHN3430I 30130903).
DVHREQ2289I Your AMDISK request for LINMSTR at * has completed; with
DVHREQ2289I RC = 0.
```

Damit wurde nun eine Minidisk auf der DASD LIN001 erstellt. Im z/VM User Directory besitzt der Benutzer LINMSTR nun einen zusätzlichen Eintrag:

```
MDISK 0200 3390 1 3337 LIN001
```

Bei der manuellen Auswahl einer Minidisk ist Vorsicht geboten. Da keine Checks durchgeführt werden, ob der Bereich auf der DASD bereits von anderen Minidisks verwendet wird, muss darauf geachtet werden, dass die ausgewählten Zylinder der DASD wirklich frei zur Verfügung stehen.

## Automatische Auswahl der Minidisk

DIRMAINT ermöglicht es mittels DATAMOVE, die Auswahl der Minidisks zu automa-

tisieren. Dazu werden eine Reihe von Festplatten in einem Extent Control File als Pool für die Verwendung durch DATAMOVE definiert [17]. Auf dem z/VM-System existiert der in Kapitel 5.2.2 beschriebene Pool LINUX.

Nun kann die System-Festplatte aus diesem Pool angefordert werden:

```
dirm for LINMSTR amdisk 200 x autog linux
```

DIRMAINT wählt jetzt aus dem Pool der Gruppe LINUX eine DASD aus und erstellt darauf eine Minidisk mit einer Größe von 200 Zylindern. Wie bei der manuellen Definition besitzt der Benutzer nun einen zusätzlichen Eintrag im z/VM User Directory.

### Eintrag im User Directory

Im Directory befindet sich nun folgender Eintrag, der das System LINMSTR vollständig definiert:

```
USER LINMSTR SECRET * * *  
INCLUDE LINDFLT  
MDISK 0200 3390 1 3337 LIN001
```

Damit ist die Definition des Benutzers abgeschlossen und es kann mit der Installation des Betriebssystems begonnen werden.

## 6.1.6 Installation des Systems

### *Neuinstallation*

Auf der Minidisk 191 des Benutzers LNXMAINT befinden sich alle für die Installation benötigten Dateien [21]:

- PARMFILE SLES10
  - Text, enthält kernel-Parameter
- INITRD SLES10
  - enthält die initiale Ramdisk (inkl. Installations-Programm)
- KERNEL SLES10
  - enthält den Kernel

Die nötigen Befehle, um diese Dateien auf die Minidisk zu laden, befinden sich in der Dokumentation der jeweiligen Distribution.

Die gleichen Dateien liegen auch für andere Distributionen vor (SLES9, DEBIAN). Installiert werden soll SUSE Linux Enterprise Server 10 (SLES10) [21].

Zunächst erfolgt ein Login als Benutzer LINMSTR. Das Kommando *FILEL \* SLES10* zeigt, dass die benötigten Dateien existieren:

| Cmd | Filename | Filetype | Fm | Format | Lrecl | Records | Blocks | Date    | Time     |
|-----|----------|----------|----|--------|-------|---------|--------|---------|----------|
|     | PARMFILE | SLES10   | A1 | F      | 80    | 2       | 1      | 7/13/07 | 23:31:08 |
|     | KERNEL   | SLES10   | A1 | F      | 80    | 75579   | 1122   | 7/13/07 | 23:22:47 |
|     | INITRD   | SLES10   | A1 | F      | 80    | 103269  | 2017   | 7/13/07 | 23:20:26 |

Für die Installation des Betriebssystems wird mehr Speicher als die standardmäßig zugewiesenen 128MB benötigt, empfohlen werden mindestens 512MB. Mit dem Befehl *DEFINE STORAGE* wird der Hauptspeicher der virtuellen Maschine vergrößert, in diesem Fall auf 512MB:

```
define storage 512M
00: STORAGE = 512M
00: Storage cleared - system reset.
```

Nach dem Kommando *DEFINE STORAGE* wird das System zurückgesetzt, so dass kein Betriebssystem mehr ausgeführt wird. Somit muss CMS mit einem *IPL*-Befehl neu gestartet werden.

Auf der A-Minidisk (Adresse 191) existiert ein Skript *BOOTLIN*, dem der Distributionsname übergeben wird.

```
/* */
ARG DIST_NAME .
say dist_name
'cl rdr'
'purge rdr all'
'spool punch * rdr'
'PUNCH KERNEL ' dist_name ' A (NOH'
'PUNCH PARMFILE ' dist_name ' A (NOH'
'PUNCH INITRD ' dist_name ' A (NOH'
'PUNCH INITRD2 ' dist_name ' A (NOH'
'ch rdr all keep nohold'
'i 00c'
```

Dieses Skript wird mit dem Befehl *BOOTLIN SLES10* aufgerufen und erzeugt die folgende Ausgabe:

```
bootlin sles10
SLES10
00: NO FILES PURGED
00: RDR FILE 0001 SENT FROM LINMSTR PUN WAS 0001 RECS 076K CPY 001 A NOHOLD NO
KEEP
00: RDR FILE 0002 SENT FROM LINMSTR PUN WAS 0002 RECS 0002 CPY 001 A NOHOLD NO
KEEP
DMSOPN002E File INITRD2 SLES10 A not found
00: 0000003 FILES CHANGED
00: 0000003 FILES CHANGED
Linux version 2.6.16.21-0.8-default (geeko@buildhost) (gcc version 4.1.0 (SUSE L
inux)) #1 SMP Mon Jul 3 18:25:39 UTC 2006
We are running under VM (64 bit mode)
Detected 2 CPU's
Boot cpu address 0
Built 1 zonelists
Kernel command line: ramdisk_size=65536 root=/dev/ram1 ro init=/linuxrc TERM=dum
b
PID hash table entries: 1024 (order: 10, 32768 bytes)
Dentry cache hash table entries: 32768 (order: 6, 262144 bytes)
Inode-cache hash table entries: 16384 (order: 5, 131072 bytes)
Memory: 111104k/131072k available (4298k kernel code, 0k reserved, 1401k data, 1
96k init)
Security Framework v1.0.0 initialized
Mount-cache hash table entries: 256
checking if image is initramfs... it is
Freeing initrd memory: 8067k freed
cpu 0 phys_idx=0 vers=FF ident=051537 machine=2064 unused=0000
cpu 1 phys_idx=1 vers=FF ident=051537 machine=2064 unused=0000
Brought up 2 CPUs
migration_cost=1000
NET: Registered protocol family 16
debug: Initialization complete
TC classifier action (bugs to netdev@vger.kernel.org cc hadi@cyberus.ca)
audit: initializing netlink socket (disabled)
audit(1196952739.175:1): initialized
VFS: Disk quotas dquot_6.5.1
Dquot-cache hash table entries: 512 (order 0, 4096 bytes)
Initializing Cryptographic API
```

Nun erscheint das Hauptmenü für den Installationsprozess:

```
Activating manual setup program.
>>> Linuxrc v2.0.46 (Kernel 2.6.16.21-0.8-default) <<<
Main Menu
1) Settings
2) System Information
3) Kernel Modules (Hardware Drivers)
4) Start Installation or System
5) Verify Installation CD-ROM/DVD
6) Eject CD
7) Exit or Reboot
8) Power off
>
```

Es wird Option 4, *Start Installation or System*, ausgewählt.

```
1) Start Installation or Update
2) Boot Installed System
3) Start Rescue System
>
```

Die Installation beginnt mit Option 1.

```
Choose the source medium.
```

- 1) CD-ROM
- 2) Network
- 3) Hard Disk

```
>
```

Die Installation erfolgt über das Netzwerk (Option 2) ...

```
Choose the network protocol.
```

- 1) FTP
- 2) HTTP
- 3) NFS
- 4) SMB / CIFS (Windows Share)
- 5) TFTP

```
>
```

... via FTP.

Nun erfolgt zunächst die Konfiguration des Netzwerks:

```
Please select the type of your network device.
```

- 1) OSA-2 or OSA Express
- 2) Hipersockets
- 
- 3) Channel To Channel (CTC)
- 4) ESCON
- 5) Inter-User Communication Vehicle (IUCV)

```
>
```

Im System befindet sich eine Hipersocket-Netzwerkkarte mit den Adressen FF00 bis FF02. Nach Auswahl von Option 2 werden diese Adressen angezeigt.

```
NET: Registered protocol family 10  
lo: Disabled Privacy Extensions  
IPv6 over IPv4 tunneling driver  
qdio: loading QDIO base support version 2  
qeth: loading qeth S/390 OSA-Express driver
```

```
0.0.ff00 1731/05  
0.0.ff01 1731/05  
0.0.ff02 1731/05
```

Nun werden die drei Adressen für Lese-, Schreib- und Datenkanal abgefragt. Es muss immer auch die *0.0* vor der Geräte-Adresse mit angegeben werden.

```
Device address for read channel > 0.0.ff00
```

```
Device address for write channel > 0.0.ff01
```

```
Device address for data channel > 0.0.ff02
```

Nun folgt eine Frage nach OSI Layer 2 Support. Die Hipersockets arbeiten auf Layer 3 (IP), weshalb der Support nicht aktiviert wird.

```
Enable OSI Layer 2 support?
```

```
1) Yes
2) No
> 2
```

Anschließend erscheint eine Status-Meldung für die Aktivierung des Netzwerk-Geräts.

```
qeth: Device 0.0.ff00/0.0.ff01/0.0.ff02 is a Guest LAN Hiper card (level: V440)
with link type GuestLAN Hiper.
qeth: set adapter parameters not supported on device 0.0.ff00.
qeth: Hardware IP fragmentation not supported on hsi0
qeth: VLAN enabled
qeth: Multicast enabled
qeth: IPv6 not supported on hsi0
qeth: Broadcast enabled
qeth: Using SW checksumming on hsi0.
qeth: Outbound TSO not supported on hsi0
```

Da in diesem Netzwerk kein DHCP-Server betrieben wird, erfolgt eine manuelle Konfiguration der IP-Adresse.

```
Automatic configuration via DHCP?
```

```
1) Yes
2) No
> 2
```

Für das System werden die folgenden Parameter verwendet:

IP-Adresse: 10.0.10.250/24 (/24 = Netzmaske 255.255.255.0)

Gateway: 10.0.10.254 (IP-Adresse von z/VM, das als Router konfiguriert ist)

Nameserver: 139.18.25.3 (Nameserver der Universität Leipzig)

```
Enter your IP address> 10.0.10.250
```

```
Enter your netmask. For a normal class C network, this is usually
255.255.255.0 Ý255.255.255.0" > 255.255.255.0
If you need a gateway to reach the server, enter the IP address of the
gateway. If you do not need a gateway, enter your own IP address
Ý10.0.10.13" > 10.0.10.254
Enter the IP address of your name server. If you do not use a name server,
press ESC Ý10.0.10.13" > 139.18.25.3
```

Als FTP-Server dient der Rechner 10.0.2.1, dazu werden Username und Login angegeben. Das Verzeichnis auf dem FTP-Server ist */pub/suse/s390x/sles10/DVD/*.

```
Enter the IP address of the FTP server "10.0.10.13"> 10.0.2.1
Usually an FTP installation is performed via anonymous FTP. It is possible to
specify a user name and password for the FTP server access.
Use a user name and password?
1) Yes
2) No
> 1
Enter the user name with which to access the FTP server> ftpuser
Enter the password for the FTP server> XXXXX
Use a HTTP proxy?
1) Yes
2) No
> 2
Trying to connect to the FTP server...
Enter the directory on the server> /pub/suse/s390x/sles10/DVD
```

Nun wird nach der Installationsmethode gefragt. Das System wird über SSH installiert (Option 3). Für die Installation wird ein Passwort gesetzt, das beim Login über SSH abgefragt wird.

```
Select the display type.
1) X11
2) VNC
3) SSH
> 3
Enter your temporary SSH password> SECRET
```

Nun wird das System für die Installation vorbereitet:

```

Loading Installation System (86220 kB) -          1%      2%      3%      4%
 5%      6%      7%      8%      9%     10%     11%     12%     13%     14%
15%     16%     17%     18%     19%     20%     21%     22%     23%     24%
25%     26%     27%     28%     29%     30%     31%     32%     33%     34%
35%     36%     37%     38%     39%     40%     41%     42%     43%     44%
45%     46%     47%     48%     49%     50%     51%     52%     53%     54%
55%     56%     57%     58%     59%     60%     61%     62%     63%     64%
65%     66%     67%     68%     69%     70%     71%     72%     73%     74%
75%     76%     77%     78%     79%     80%     81%     82%     83%     84%
85%     86%     87%     88%     89%     90%     91%     92%     93%     94%
95%     96%     97%     98%     99%     100%

starting hald... ok
starting syslogd (logging to /dev/tty4)... ok
starting klogd... ok
sshd found, prepare remote login
generating SSH keys ...
Generating /etc/ssh/ssh_host_key.
Generating public/private rsa1 key pair.
Your identification has been saved in /etc/ssh/ssh_host_key.
Your public key has been saved in /etc/ssh/ssh_host_key.pub.
The key fingerprint is:
ca:a6:19:da:60:7f:7f:e1:1a:12:86:18:dd:26:9c:7d root@10.0.10.13
Generating /etc/ssh/ssh_host_dsa_key.
Generating public/private dsa key pair.
Your identification has been saved in /etc/ssh/ssh_host_dsa_key.
Your public key has been saved in /etc/ssh/ssh_host_dsa_key.pub.
The key fingerprint is:
15:e3:f3:9d:15:85:ef:bc:ec:fd:b8:9f:d1:7b:0e:13 root@10.0.10.13
Generating /etc/ssh/ssh_host_rsa_key.
Generating public/private rsa key pair.
Your identification has been saved in /etc/ssh/ssh_host_rsa_key.
Your public key has been saved in /etc/ssh/ssh_host_rsa_key.pub.
The key fingerprint is:
26:09:2c:ca:4a:f7:44:97:41:e3:c9:72:6a:97:b8:dc root@10.0.10.13
Starting SSH daemon ...

hsi0: <BROADCAST,MULTICAST,NOARP,UP> mtu 8192 qdisc pfifo_fast qlen 1000
link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
inet 10.0.10.13/24 brd 10.0.10.255 scope global hsi0
inet6 fe80::200:ff:fe00:0/64 scope link
    valid_lft forever preferred_lft forever

***  sshd has been started  ***

***  login using 'ssh -X root@10.0.10.250'  ***
***  run 'yast' to start the installation  ***

```

Ab sofort ist das System über SSH erreichbar.

Der Benutzer LINMSTR kann nun erst einmal mit dem Befehl *#CP DISCONNECT* abgemeldet werden, da an der System-Konsole keine weiteren Einstellungen getätigt werden müssen:

```

00: CP DISC
00: DISCONNECT AT 15:30:10 UTC THURSDAY 12/06/07
00:
00: Press enter or clear key to continue

```

Mit einem SSH-Client wird eine Verbindung mit der eben eingerichteten IP-Adresse 10.0.10.250 und dem gewählten Passwort hergestellt.

```
kenob:~ # ssh root@10.0.10.13
The authenticity of host '10.0.10.13 (10.0.10.13)' can't be established.
DSA key fingerprint is 15:e3:f3:9d:15:85:ef:bc:ec:fd:b8:9f:d1:7b:0e:13.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.10.13' (DSA) to the list of known hosts.
root@10.0.10.13's password:

SUSE Linux Enterprise Server 10 Installation

Welcome to the inst-sys...
Linux 10.0.10.13 2.6.16.21-0.8-default #1 SMP Mon Jul 3 18:25:39 UTC 2006 s390x s390x s390x
GNU/Linux
/root

run yast to start the installation

inst-sys:~ #
```

Um Probleme durch Verbindungsabbrüche zu vermeiden, wird die Installation in einem *Screen* (Linux-Programm *screen*, [22]) durchgeführt. Ein neuer *Screen* wird durch den Befehl *screen* gestartet. Innerhalb dieser Sitzung läuft eine Linux-Shell.

Durch das Drücken der Tasten-Kombination *CTRL+A*, *CTRL+D* wird die Sitzung in den Hintergrund geschoben und kann mit Hilfe des Befehls *screen -r* wieder reaktiviert werden. Ein Programm, was innerhalb des Screens läuft, wird die ganze Zeit weiter ausgeführt, auch wenn keine Verbindung mit dem System besteht. Weitere Hilfe liefert die Man-Page von *screen*, die mit dem Programm mitgeliefert wird.

Im *Screen* wird *yast* gestartet – das Installationsprogramm von SuSE. Dies kann eine Weile dauern, da die Installationsvorbereitungen sehr rechen- und I/O-intensiv sind. Es erscheint ein Dialog zur Auswahl der Sprache.

```
YaST @ 10.0.10.13                                     Press F1 for Help

Language

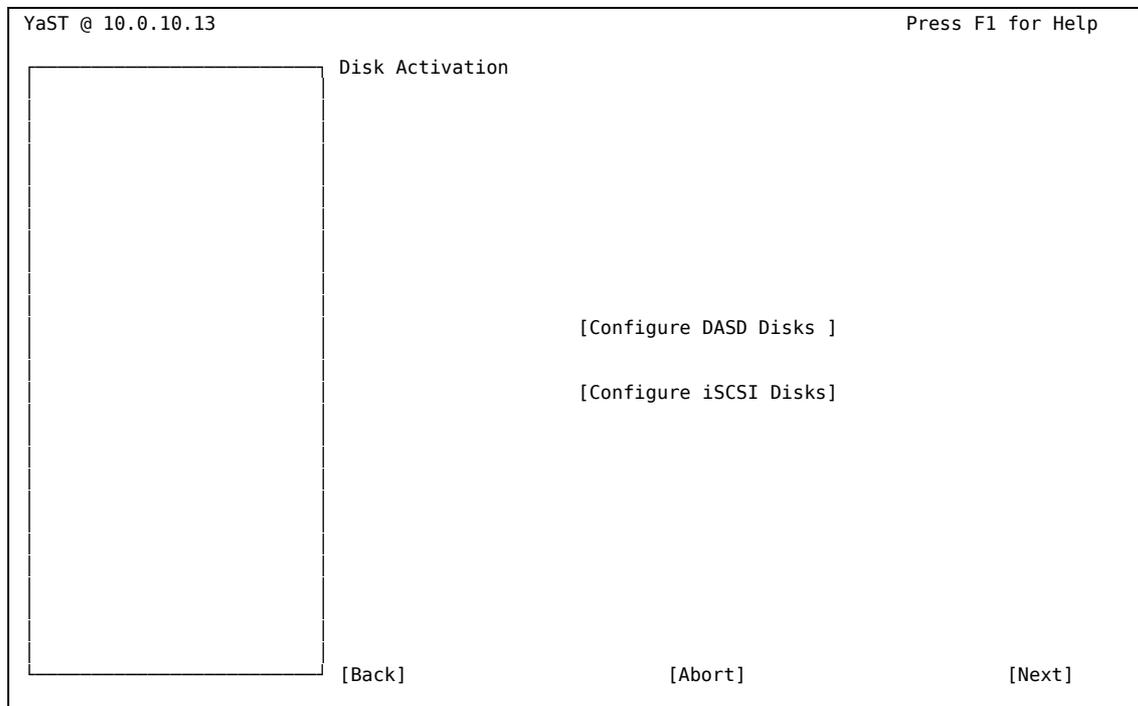
Choose the Language to
use during installation
and for the installed
system.
Click Accept to
proceed to the next
dialog.
Nothing will happen to
your computer until you
confirm all your settings
in the last installation
dialog.
You can select Abort
Installation at any time
to abort the installation
process.

Cymraeg
Dansk
Deutsch
English (UK)
English (US)
Espanol
Francais
Greek
Hindi
Hrvatski

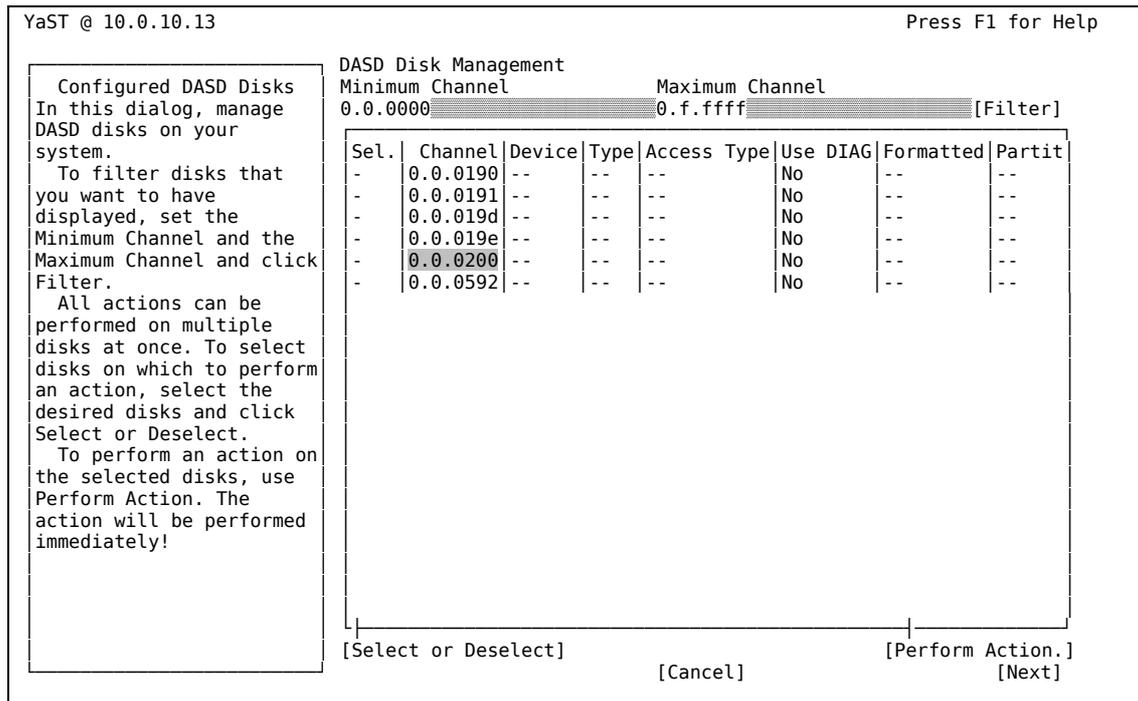
[Back]                                     [Abort]                                     [Next]
```

Als Sprache wird Deutsch ausgewählt und mit „Next“ (ALT+N) geht es weiter, nun müssen noch die Allgemeinen Geschäftsbedingungen bestätigt werden.

Anschließend erscheint ein Dialog zur Aktivierung der Festplatten:



Hier wird die Konfiguration von DASDs ausgewählt (ALT+D):



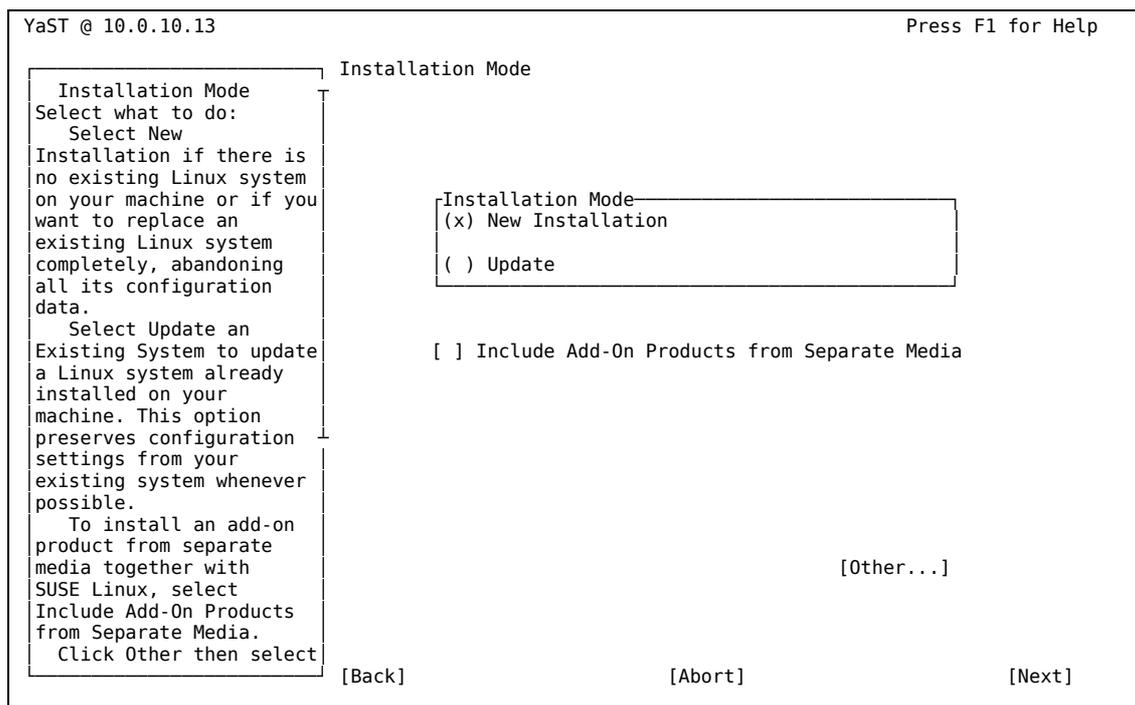
Als Festplatte wurde das Gerät 200 bei der Benutzer-Konfiguration vorgesehen. Mit ALT+S für SELECT wird diese ausgewählt. Es erscheint ein „Yes“ in der ersten Spalte:

|     |          |    |  |    |  |    |  |    |  |    |  |    |  |
|-----|----------|----|--|----|--|----|--|----|--|----|--|----|--|
| Yes | 0.0.0200 | -- |  | -- |  | -- |  | No |  | -- |  | -- |  |
|-----|----------|----|--|----|--|----|--|----|--|----|--|----|--|

Nun wird als Aktion *Activate* (ALT+A) ausgewählt, um die Festplatte über einen Gerätenamen erreichbar zu machen. Damit wird die Disk als /dev/dasda konfiguriert:

|     |          |            |                  |    |  |    |  |  |
|-----|----------|------------|------------------|----|--|----|--|--|
| Yes | 0.0.0200 | /dev/dasda | 3990/E9, 3390/0A | RW |  | No |  |  |
|-----|----------|------------|------------------|----|--|----|--|--|

Anschließend kann die Installation fortgesetzt werden, bis die Auswahl des Installationsmodus erscheint. Dort wird die Neuinstallation des Betriebssystems selektiert.



Damit sind die Vorbereitungen für die Installation abgeschlossen. Die weitere Installation unterscheidet sich nicht von der Linux-Installation auf einer anderen Architektur und wird hier nicht näher betrachtet.

### 6.1.7 Start des installierten Linux-Systems

Der Start des installierten Systems erfolgt durch einen IPL von der System-Festplatte des Benutzers, welche die Geräte-Adresse 200 hat.

Nach kurzer Zeit steht das System komplett zur Verfügung.

## 6.2 Erstellung des Linux-Cloning-Systems

Das soeben installierte System LINMSTR soll nun so angepasst werden, dass ohne großen Aufwand neue Clones erstellt werden können.

Der Clone wird durch das Anlegen eines neuen Gasts mit dem Befehl *DIRMAINT* unter Verwendung des Profils LINPROTO und dem anschließenden Cloning der System-Disk von LINMSTR erstellt.

Weiterhin muss sichergestellt werden, dass nicht zwei Linux-Clones die gleiche IP-Adresse bekommen, um Kommunikationsprobleme zu vermeiden.

In diesem Kapitel werden die Anpassungen am System LINMSTR beschrieben, die das

reibungslose Cloning von Linux-Gästen ermöglichen sollen.

## 6.2.1 Anpassen des Linux-Master-Systems

### **Identifizieren der zu ändernden Dateien**

Um das Cloning zu vereinfachen, werden zunächst Änderungen am Gast LINMSTR vorgenommen. Ziel ist es, den Hostnamen des Systems zu setzen sowie eine individuelle IP-Adresse pro Gast zu vergeben. Dafür sind zunächst Kenntnisse darüber nötig, an welchen Stellen das System Hostnamen und IP-Adresse speichert.

Der Hostname wird in der Datei `/etc/HOSTNAME` hinterlegt. Dieser Wert wird beim Start des Systems ausgelesen und als Hostname des Systems gesetzt. Im installierten System sieht diese Datei wie folgt aus:

```
linmstr.uni-leipzig
```

Die IP-Adresse wird in der Datei `/etc/sysconfig/network/ifcfg-qeth-bus-ccw-0.0.ff00` gesetzt. Diese sieht für den Gast LINMSTR so aus:

```
BOOTPROTO="static"
UNIQUE=""
STARTMODE="onboot"
IPADDR="10.0.10.250"
NETMASK="255.255.255.0"
NETWORK="10.0.10.0"
BROADCAST="10.0.10.255"
```

Für eine automatische Konfiguration bestehen zwei Aufgaben. Es muss

1. der Name des z/VM-Gasts ausgelesen werden und als Hostname gesetzt werden,
2. die IP-Adresse gesetzt werden, die für diesen Gast vorgesehen wurde.

### **Kommunikation mit dem Host-System**

Für die Kommunikation mit dem Host-System gibt es für Linux das Paket `cpint`. Dieses besteht aus einem Kernel-Modul sowie Anwendungsprogrammen. Dadurch ist es möglich, aus einem Linux-Gast heraus `CP`-Kommandos für z/VM abzusetzen und die Ausgaben dieser Kommandos zu erhalten.

Mit dem `CP`-Kommando `QUERY USERID` lässt sich die eigene User-ID auslesen. Für das Absetzen der Kommandos ist das Programm `hcp` zuständig. Dafür muss das Kernel-Modul `cpint` zunächst mittels `modprobe` geladen werden. Um von einer Linux-Shell die

eigene z/VM-User-ID auszulesen, werden die folgenden Befehle ausgeführt:

```
linmstr:~ # modprobe cpint
linmstr:~ # hcp q userid
LINMSTR AT GEORGVM1
```

Damit das Modul *cpint* beim Starten des System automatisch geladen wird, muss es in der Konfigurationsdatei */etc/sysconfig/kernel* eingetragen werden.

```
MODULES_LOADED_ON_BOOT="cpint"
```

Damit steht für die erste Aufgabe eine Lösung bereit.

### **Lesen von CMS Minidisks**

Die zweite Aufgabe – das Zuordnen einer IP-Adresse – bedarf eines zweistufigen Ansatzes.

Auf der Minidisk 191, die vom Benutzer LNXMAINT gelinkt ist, soll eine Konfigurationsdatei abgelegt werden, die einer User-ID eine IP-Adresse zuordnen soll. Dafür vorgesehen ist die Datei *LINUX IPCONF*. Da nur der Benutzer LNXMAINT Schreibrechte für diese Minidisk besitzt, muss zuvor eine 3270-Session mit dieser User-ID gestartet werden. Dort wird mit *XEDIT* die Datei auf der Minidisk (Minidisk 191 = file mode A) angelegt.

```
x LINUX IPCONF A
```

Im Editor wird eine Username-IP-Zuordnung angelegt, wobei die erste Spalte die User-ID und die zweite Spalte die IP-Adresse für diesen Gast enthalten soll. Theoretisch könnten hier noch weitere Konfigurationsparameter hinterlegt werden, für das aktuelle Problem reichen diese zwei Spalten.

```
==== * * * Top of File * * *
|...+...1...+...2...+...3...+...4...+...5...+...6...+...7...
==== # <HOST> <IPADDR>
==== LINMSTR 10.0.10.250
==== LIN001 10.0.10.1
==== LIN002 10.0.10.2
==== LIN003 10.0.10.3
==== LIN004 10.0.10.4
==== LIN005 10.0.10.5
==== * * * End of File * * *
```

Mit dem Kommando *file* wird die Datei gespeichert.

Das Linux-System soll diese Datei nun beim Systemstart auslesen und die für sich ange-

dachte IP-Adresse auslesen und setzen.

Um die Minidisk lesen zu können, muss sie zunächst vom Linux-System aus erreichbar sein. Diese Minidisk wird im SUSE Linux Enterprise Server 10 unter `yast` → Hardware → DASD aktiviert. Die Aktivierung erfolgt genau wie bei der Aktivierung der System-Disk während der Installation (Auswählen, Aktivieren). Nach der Aktivierung steht die Minidisk 191 als Gerät `/dev/dasdb` bereit.

Für das Lesen von CMS-Minidisks gibt es unter Linux das Paket `cmsfs`. In diesem Paket befindet sich das Kommando `cmsfscat`, was das Pendant zum Kommando `cat` für das Ausgeben von Dateien auf Minidisks mit CMS-Dateisystem ist.

Mit dem folgenden Befehl wird der Inhalt der Datei `LINUX.IPCONF` auf der Minidisk 191 (konfiguriert als `/dev/dasdb`) ausgegeben:

```
linmstr:~ # cmsfscat -d /dev/dasdb -a LINUX.IPCONF
# <HOST> <IPADDR>
LINMSTR 10.0.10.250
LIN001 10.0.10.1
LIN002 10.0.10.2
LIN003 10.0.10.3
LIN004 10.0.10.4
LIN005 10.0.10.5
```

Mittels `grep` lässt sich die Zeile mit der eigenen User-ID auswählen und die IP-Adresse durch Abschneiden der User-ID einlesen.

Damit steht auch eine Lösung für die zweite Aufgabe bereit.

### **Erstellen des Cloneconfig-Skripts**

Da jetzt Möglichkeiten gefunden wurden, Hostnamen und IP-Adresse zu setzen, müssen diese automatisiert beim Start des Systems konfiguriert werden.

Dafür wird ein Init-Skript `/etc/init.d/boot.cloneconfig` erstellt, das die User-ID und die IP-Adresse auslesen und setzen soll.

```

#!/bin/sh
#
# /etc/init.d/boot.cloneconfig
#
### BEGIN INIT INFO
# Provides:          boot.cloneconfig
# Required-Start:    boot.cleanup
# Required-Stop:
# Default-Start:     B
# Default-Stop:
# Description:       setup ip and hostname for a clone
### END INIT INFO

. /etc/rc.status

rc_reset

case "$1" in
start)
    echo "Setting clone configuration"
    #
    # get hostname and domainname
    #
    XHOSTSTRING=`hcp q userid`

    XHOST=${XHOSTSTRING%%\ *}

    XHOST_SMALL=`echo $XHOST | tr 'A-Z' 'a-z'`

    XHOSTDOMAIN=${XHOST_SMALL}.uni-leipzig

    test -n "$XHOSTDOMAIN" && {
        echo Setting /etc/HOSTNAME to \`${XHOSTDOMAIN}\`
        echo ${XHOSTDOMAIN} > /etc/HOSTNAME
        rc_status -v -r
    }
    #
    # get IP config from cms mdisk
    #
    XHOST_IP_LINE=`cmsfscat -d /dev/dasdb -a LINUX.IPCONF | grep ^$XHOST`

    test -n "$XHOST_IP_LINE" && {
        IP=`echo "$XHOST_IP_LINE" | cut -f 2 -d " "`
        echo "Setting IP to $IP"
        sed -i -e "s/^IPADDR\(.*\)/IPADDR="\${IP}\"/" /etc/sysconfig/network/ifcfg-qeth-bus-
ccw-0.0.ff00
        rc_status -v -r
    }

    ;;
stop|restart)
    # skip / nothing to do
    ;;
status)
    rc_failed 4
    rc_status -v
    ;;
*)
    echo "Usage: $0 {start|stop|status|restart}"
    exit 1
    ;;
esac

rc_exit

```

Die Abschnitte, die die Informationen auslesen und an den entsprechenden Stellen in den Konfigurations-Dateien setzen, sind grau markiert. Der Hostname unter Linux wird in Kleinbuchstaben gesetzt (Variable XHOST\_SMALL).

Das Skript wird nun noch so in das Init-System eingepflegt, dass es an der richtigen Stelle beim Systemstart geladen wird. Es ist notwendig, die Einstellungen vor dem Setzen von Hostname und IP-Adresse vorzunehmen, aber nachdem das benötigte Modul

*cpint* geladen worden ist. Die Einstellungen müssen gesetzt werden, nachdem das Skript *boot.loadmodules*, aber bevor das Skript *boot.localnet* ausgeführt wird, welches für das Setzen des Hostnamen zuständig ist. Die IP-Adressen werden erst später konfiguriert. Im Ordner */etc/init.d/boot.d* werden symbolische Verknüpfungen (Symlinks) mit dem Skript für Start und Stop angelegt. Der Name eines Symlinks für das Init-System hat die Form *TXXskriptname*, wobei der Typ T angibt, ob es sich um einen Startskript (S) oder ein Stopskript (K) handelt und die Zahl XX für eine Sortierung der Skripte während Start und Stop sorgt. Das Skript *S01skript1* wird beim Start vor dem Skript *S99skript2* ausgeführt.

Das Skript *boot.localnet* ist beim Start des boot-Runlevels mit S11 verlinkt, das Laden der Module mit S03. Eine Verlinkung muss also dazwischen erfolgen (z.B. als *S10*). Der Stop des Skripts ist irrelevant, da das Zurücksetzen der Einstellungen nicht notwendig ist.

```
linmstr:/etc/init.d/boot.d # ln -s ../boot.cloneconfig S10boot.cloneconfig
```

Um eine sequentielle Abarbeitung zu gewährleisten, wird das parallele Starten der Skripte ausgeschaltet. Dies geschieht in der Datei */etc/sysconfig/boot* durch das Setzen der Variable „RUN\_PARALLEL“ auf „no“.

```
RUN_PARALLEL="no"
```

Damit sind die Modifikationen für das automatische Setzen von Hostname und IP-Adresse abgeschlossen.

Das System LINMSTR wird nun heruntergefahren. Um Kopien eines Betriebssystems anzulegen, darf dieses nicht in Betrieb sein. Da während des Betriebs die Systempartition mit Schreibrechten eingebunden ist, kann in diesem Zustand eine exakte Kopie nicht gewährleistet werden.

## 6.2.2 Anlegen eines neuen Gastes

Zunächst wird, wie auch beim User LINMSTR, ein neuer z/VM-Gast angelegt. Die Namen sollen zur leichten Erkennung mit „LIN“ beginnen gefolgt von einer dreistelligen Nummer. Als erster Clone wird also die User-ID LIN001 angelegt.

```
dirm add LIN001 LIKE LINPROTO PW SECRET
```

### 6.2.3 Kopieren des Systems

Für das Kopieren einer Minidisk bietet *DIRMAINT* das Kommando *CLONEDISK*.

Geklonet werden soll die Minidisk 200 des Benutzers LINMSTR. Damit das Betriebssystem ohne Anpassungen funktioniert, muss die virtuelle Adresse 200 erhalten bleiben.

```
dirm for lin001 clonedisk 200 linmstr 200 g LINUX
```

Dieser Vorgang dauert eine Weile. Es erscheint mehrfach die System-Meldung, dass der User Directory-Eintrag von LIN001 angepasst wurde. Am Ende erfolgt eine Meldung, dass der Vorgang abgeschlossen ist („finished“). Nun ist das System nutzbar und kann gestartet werden.

Der Cloning-Master LINMSTR sollte nicht mehr gestartet werden, es sei denn es sollen Änderungen an der Installation vorgenommen werden, die allen danach erstellten Clones zur Verfügung stehen sollen.

### 6.2.4 Automatisierung des Starts

Bevor ein Linux-Gast startet, wird zunächst das integrierte Betriebssystem CMS gestartet. Beim Login eines CMS-Gasts wird standardmäßig die Datei *PROFILE EXEC* ausgeführt. Diese Datei befindet sich auf der Minidisk 191 von Benutzer LNXMAINT, auf die jedes Gastssystem mit Leserechten zugreifen kann.

Über diese Datei wird der Systemstart gesteuert. Dabei werden bestimmte CP-Kommandos abgesetzt, um Systemressourcen für das Gastssystem bereitzustellen. Zum Start des Systems wird der IPL-Befehl zum Starten von der System-Festplatte des Linux-Gasts abgesetzt.

Es ist möglich, für jeden Gast eine gesonderte Start-Datei auszuführen. Um dies zu ermöglichen gibt es unter der Skriptsprache REXX die Funktion *USERID()*, deren Rückgabewert (die User-ID) für einfache Verzweigungen oder zum Ausführen von Kommandos verwendet werden kann. Für ein Cloning-System empfiehlt sich jedoch ein einfacher Einsatz. Dafür ist es von Vorteil, wenn die Hardware-Definitionen aller Gäste identisch sind, sodass keine Anpassungen für neue Gäste notwendig sind.

Die Datei *PROFILE EXEC*, die beim Start der Linux-Benutzer ausgeführt wird, sieht wie folgt aus:

```
/* REXX PROFILE EXEC FOR LINUX 390 VM GUEST */
/*
'CP SET MSG ON'
'CP SET EMSG ON'
'CP SET RUN ON'
'CP SET PF12 RETRIEVE'
'CP term chardel off'
/* VMFCLEAR */
/* SAY ' ' */
/* SAY ' ' */
/* SAY ' ' */
IF USERID() = LNXMAINT THEN LINBOOT
EXIT
```

Für alle Benutzer außer LNXMAINT wird nun noch der Script LINBOOT EXEC ausgeführt. Dieser ist sehr kurz gehalten, kann aber bei Bedarf noch erweitert werden.

```
/* REXX */
'CP IPL 200'
```

Damit wird ein IPL von Gerät 200 durchgeführt, das Betriebssystem startet.

## 7 z/OS als Gastsystem unter z/VM

### 7.1 Voraussetzungen

Als Ausgangssituation dient ein fertiges, lauffähiges z/OS-System, was in Form von fünf DASDs zur Verfügung gestellt wurde. Dieses System basiert auf der „Application Developers Controlled Distribution“ (ADCD) von IBM [23] [24]. Weiterführende Dokumente über z/OS befinden sich in der z/OS Internet Library [25].

Um dieses System unter z/VM zu betreiben, muss für das z/OS-Gastsystem ein Benutzer mit Zugriff auf diese Festplatten erstellt werden. Neben den DASDs werden weitere spezielle Geräte angelegt, welche z/OS für den ordnungsgemäßen Betrieb benötigt.

#### 7.1.1 Definition des Systems

Das Basis-System besteht aus folgenden fünf DASDs mit der entsprechenden Geräte-Adresse:

- Z8SYS1 – 0A80
- Z8RES1 – 0A81
- Z8RES2 – 0A82
- Z8RES3 – 0A83
- Z8USS1 – 0A84

Die DASDs werden bei der Definition des Gasts im *z/VM User Directory* als Full-Volumen-Minidisks angelegt. Diese Platten werden so konfiguriert, dass sie parallel von weiteren Systemen lesbar sind (Zugriffsrecht *MR*).

Für die Verwaltung der z/OS-Gäste wird analog zu Kapitel 6.1.2 der Systemverwalter ZOSMAINT eingerichtet. Den Gastsystemen wird dessen A-Minidisk (191) mit Lese-rechten zugeordnet.

Dem z/OS-Gast werden zwei CPUs sowie 512MB Hauptspeicher zugeordnet. Weiterhin werden für das System Displays definiert. Diese virtuellen 3270-Displays sind nötig, um sich ohne eine lauffähige Netzwerkkonfiguration mit einem 3270-Emulator am System anzumelden.

Für die Kommunikation wird eine Netzwerkkarte mit der Adresse FF00 definiert. Im Zusammenhang mit z/OS muss die „Channel Path ID“ (CHPID) explizit angegeben werden, die dann auch im Betriebssystem so konfiguriert werden muss.

Es folgt der komplette Eintrag für den Gast ZOS001 im z/VM User Directory.

```
USER ZOS001 SECRET 512M 1024M G
  INCLUDE IBMDFLT
  CPU 00 CPUID 111000
  CPU 01 CPUID 111111
  IPL CMS PARM AUTOCR
  IUCV ANY
  IUCV ALLOW
  MACHINE ESA
  OPTION QUICKDSP MAINTCCW RMCHINFO
  CONSOLE 0700 3270 A
  NICDEF FF00 TYPE HIPER LAN SYSTEM LINHIPER CHPID FF
  SPECIAL 0701 3270
  SPECIAL 0702 3270
  SPECIAL 0703 3270
  SPECIAL 0704 3270
  SPECIAL 0705 3270
  LINK ZOSMAINT 0191 0191 RR
  MDISK 0A80 3390 DEVNO 552B MR
  MDISK 0A81 3390 DEVNO 552C MR
  MDISK 0A82 3390 DEVNO 552D MR
  MDISK 0A83 3390 DEVNO 552E MR
  MDISK 0A84 3390 DEVNO 552F MR
```

Der Gast ZOS001 startet beim Login zunächst CMS (IPL CMS). Dort wiederum ist ein REXX-Skript dafür zuständig, das System ordnungsgemäß zu starten. In diesem Skript befindet sich der Befehl zum Start (IPL) von z/OS:

```
IPL A81 LOADP 0A80CS
```

Nach dem IPL ist das System erreichbar. Um auf das System zuzugreifen, muss der Benutzer sich per *DIAL* mit dem System verbinden. Dabei wird die 3270-Session wie in Kapitel 4.1.1 beschrieben an den Gast weitergeleitet. Als Parameter wird der Name des Gasts – ZOS001 – angegeben.

```

z/VM ONLINE

Georg - Testsystem

          / VV          VVV MM          MM
         / VV          VVV MMM         MMM
        / VV          VVV MMMM        MMMM
       / VV          VVV MM MM MM MM
      / VV          VVV MM MMM MM
     / VVVV         MM M MM
    / VVV          MM MM
   / V            MM MM
  /

built on IBM Virtualization Technology

Fill in your USERID and PASSWORD and press ENTER
(Your password will not appear when you type it)
USERID  ==>
PASSWORD ==>

COMMAND ==> dial zos001

RUNNING GEORGVM1

```

Nach dem Kommando erscheint der Login-Screen von z/OS:

```

z/OS Z18 Level 0609                               IP Address =
                                                    VTAM Terminal = LCL701

Application Developer System

          // 0000000 SSSS
         // 00 00 SS
        zzzzzz // 00 00 SS
       zz // 00 00 SSSS
      zz // 00 00 SS
     zz // 00 00 SS
    zzzzzz // 0000000 SSSS

System Customization - ADCD.Z18.*

==> Enter "LOGON" followed by the TSO userid. Example "LOGON IBMUSER" or
==> Enter L followed by the APPLID
==> Examples: "L TSO", "L CICS", "L IMS3270"

```

Von hier aus kann eine Anmeldung an z/OS erfolgen [26].

## 7.1.2 Einrichten der Netzwerkkarte

Für das Anbinden von z/OS an das z/VM Hipersocket-Netzwerk muss zunächst die Hardware-Definition von z/OS angepasst werden [27]. In der vorliegenden Hardware-Definition fehlte der Support für Hipersockets und damit auch die Möglichkeit, eine Hipersocket-Netzwerkkarte zu definieren.

Um diesen Support hinzuzufügen, wird zunächst der *Hardware Configuration Dialog (HCD)* über das Menü *Additional IBM Products* gestartet (Option *M.4* unter *ZOS001*).

Auf der Übersichtsseite des HCD ist erkennbar, welches I/O Definition File zurzeit bearbeitet wird. Zur Erstellung der nötigen Definitionen wird Option 1 ausgewählt.

```
z/OS V1.7 HCD
Command ==>-----
(C) Copyright IBM Corp. 1990, 2005. All rights reserved.
      Hardware Configuration

Select one of the following.

1 1. Define, modify, or view configuration data
  2. Activate or process configuration data
  3. Print or compare configuration data
  4. Create or view graphical configuration report
  5. Migrate configuration data
  6. Maintain I/O definition files
  7. Query supported hardware and installed UIMs
  8. Getting started with this dialog
  9. What's new in this release

For options 1 to 5, specify the name of the IODF to be used.
I/O definition file . . . 'SYS1.IODF99'          +

F1=Help   F2=Split   F3=Exit   F4=Prompt   F9=Swap   F12=Cancel
F22=Command
```

Die benötigte Hipersocket-Unterstützung ist ein Prozessor-Feature, welches im anschließenden Dialog unter Option 3 zu finden ist.

```
Select type of objects to define, modify, or view data.

3 1. Operating system configurations
   consoles
   system-defined generics
   EDTs
   esoterics
   user-modified generics
  2. Switches
   ports
   switch configurations
   port matrix
  3. Processors
   channel subsystems
   partitions
   channel paths
  4. Control units
  5. I/O devices
F1=Help   F2=Split   F3=Exit   F9=Swap   F12=Cancel
```

Der einzige aktive Prozessor, der definiert ist, ist vom Typ 7060 Model H30. Von diesem Prozessor-Modell werden Hipersockets nicht unterstützt.

```

Goto Filter Backup Query Help
-----
Processor List          Row 1 of 1 More:  >
Command ==>> _____ Scroll ==>> PAGE

Select one or more processors, then press Enter. To add, use F11.

/ Proc. ID Type +  Model +  Mode+ Serial-# + Description
- M3000H30 7060   H30    BASIC
***** Bottom of data *****

F1=Help      F2=Split    F3=Exit     F4=Prompt   F5=Reset    F7=Backward
F8=Forward   F9=Swap     F10=Actions F11=Add     F12=Cancel  F13=Instruct
F20=Right   F22=Command

```

Für die Hipersocket-Unterstützung wird ein Prozessor vom Typ 2064 oder 2066 benötigt. Dieser wird nun mittels F11 (Add) definiert.

```

Specify or revise the following values.

Processor ID . . . . . Z900VM__
Processor type . . . . . 2064  +
Processor model . . . . . 2C2  +
Configuration mode . . . . . BASIC +
Number of channel subsystems . . - +

Serial number . . . . . _____
Description . . . . . _____

Specify SNA address only if part of an S/390 microprocessor cluster:

Network name . . . . . _____ +
CPC name . . . . . _____ +

F1=Help      F2=Split    F3=Exit     F4=Prompt   F5=Reset    F9=Swap
F12=Cancel

```

Nach der Bestätigung folgt eine Nachfrage zum Support Level des Prozessors.

```

Support Level
Basic 2064 support, IQD, FCP, CF Duplex
#
Basic 2064 support, IQD, FCP, CF Duplex, CAS
#

```

Das für die Hipersockets benötigte Feature *IQD* ist in beiden Support Levels vorhanden, durch nochmaliges Bestätigen (RETURN) wird das erste ausgewählt.

Nun erscheint der zweite Prozessor in der Prozessor-Liste.

```

/ Proc. ID Type +  Model +  Mode+ Serial-# + Description
- M3000H30 7060   H30    BASIC
S Z900VM   2064   2C2    BASIC

```

Durch ein „S“ in der Zeile des neu definierten Prozessors wird dieser für die weitere Konfiguration ausgewählt.

```

Goto Filter Backup Query Help
-----
                          Channel Path List
Command ==> _____ Scroll ==> PAGE

Select one or more channel paths, then press Enter. To add use F11.

Processor ID . . . . : Z900VM
Configuration mode . : BASIC
Channel Subsystem ID :

                          DynEntry Entry +
/ CHPID Type+ Mode+ Switch + Sw Port Con Mngd Description
***** Bottom of data *****

F1=Help      F2=Split    F3=Exit      F4=Prompt    F5=Reset     F7=Backward
F8=Forward   F9=Swap      F10=Actions  F11=Add     F12=Cancel   F13=Instruct
F20=Right    F22=Command

```

Für diesen Prozessor sind noch keine Channel Paths definiert. Durch Drücken von F11 (Add) wird der Dialog zum Erstellen eines neuen Channel Paths gestartet.

```

Specify or revise the following values.

Processor ID . . . . : Z900VM
Configuration mode . : BASIC
Channel Subsystem ID :

Channel path ID . . . . FF +          PCHID . . . . ___
Number of CHPIDs . . . . 1
Channel path type . . . . IQD +
Operation mode . . . . DED +
Managed . . . . . No (Yes or No) I/O Cluster _____ +
Description . . . . . _____

Specify the following values only if connected to a switch:
Dynamic entry switch ID __ + (00 - FF)
Entry switch ID . . . . __ +
Entry port . . . . . __ +
F1=Help      F2=Split    F3=Exit      F4=Prompt    F5=Reset     F9=Swap
F12=Cancel

```

Als Channel Path ID wird *FF* ausgewählt und als Channel Path Type *IQD*. Nach der Bestätigung folgt die Nachfrage nach der „Maximum Frame Size“, die bei 16 KB gelassen werden kann. Anschließend erscheint der neu definierte Eintrag in der *Channel Path*-Liste.

```

/ CHPID Type+ Mode+ Switch + Sw Port Con Mngd Description
_ FF   IQD   DED   __   __   __   No   _____

```

Der neu definierte Channel Path wird durch ein „S“ in der ersten Spalte zur weiteren Konfiguration ausgewählt. Es folgt eine Übersicht über die definierten Control Units, die noch keinen Eintrag enthält. Mittels F11 wird eine neue Control Unit angelegt.

```

Specify or revise the following values.

Control unit number . . . . . FF00 +
Control unit type . . . . . IQD ----- +

Serial number . . . . . -----
Description . . . . . -----

Connected to switches . . . -- -- -- -- -- -- -- -- +
Ports . . . . . -- -- -- -- -- -- -- -- -- +

If connected to a switch:

Define more than eight ports . . 2 1. Yes
                                         2. No

Propose CHPID/link addresses and
unit addresses . . . . . 2 1. Yes
                                         2. No

F1=Help  F2=Split  F3=Exit  F4=Prompt  F5=Reset  F9=Swap
F12=Cancel

```

Im Dialog zum Hinzufügen der neuen Control Unit wird *FF00* als *Control Unit number* eingetragen und der Typ auf *IQD* gesetzt.

Als nächster Schritt muss diese Control Unit über die Channel Path ID FF erreichbar gemacht werden. Dies geschieht durch das Eintragen von FF in die erste dafür vorgesehene Spalte.

```

Select Processor / CU      Row 1 of 2 More: >
Command ==> ----- Scroll ==> PAGE
Select processors to change CU/processor parameters, then press Enter.

Control unit number . . : FF00      Control unit type . . . : IQD

-----Channel Path ID . Link Address + -----
/ Proc.CSSID 1----- 2----- 3----- 4----- 5----- 6----- 7----- 8-----
- M3000H30
- Z900VM      FF-----

```

Durch Drücken von F20 (Shift+F8) wird die Übersicht erreicht, in der die *Control Unit*-Adressen eingetragen werden können. Es wird „00.256“ eingetragen (Start und Anzahl).

```

Select Processor / CU      Row 1 of 2 More: < >
Command ==> ----- Scroll ==> PAGE
Select processors to change CU/processor parameters, then press Enter.

Control unit number . . : FF00      Control unit type . . . : IQD

CU -----Unit Address . Unit Range + -----
/ Proc.CSSID Att ADD+ 1----- 2----- 3----- 4----- 5----- 6----- 7----- 8-----
- M3000H30
- Z900VM      -- 00.256 -----

```

Nach zweimaliger Bestätigung ist diese Konfiguration der Control Unit abgeschlossen und sie erscheint in der Übersicht. Diese wird, wie schon in den anderen Dialogen davor, mittels „S“ zur Bearbeitung ausgewählt.

|   |      |        |      |     |          |               |
|---|------|--------|------|-----|----------|---------------|
| / | CU   | Type + | #CSS | #MC | Serial-# | + Description |
| S | FF00 | IQD    |      | 1   |          |               |

Nun erfolgt die eigentliche Definition der Netzwerkkarte. In der leeren I/O Device List wird mit F11 ein neues Gerät hinzugefügt.

```
Specify or revise the following values.
Device number . . . . . FF00 + (0000 - FFFF)
Number of devices . . . . . 32
Device type . . . . . IQD +
Serial number . . . . .
Description . . . . .
Volume serial number . . . . . (for DASD)
Connected to CUs . . FF00 +
F1=Help F2=Split F3=Exit F4=Prompt F5=Reset F9=Swap
F12=Cancel
```

Als *Device number* wird FF00 eingetragen. Mit dieser Adresse wurde die Netzwerkkarte unter z/VM definiert. Der Typ ist wie bei Channel Path und Control Unit IQD. Es werden 32 Geräte angelegt, auch wenn zur Zeit nur 3 für die Verbindung benötigt werden. Die Control Unit FF00 ist bereits voreingestellt.

Nach der Bestätigung muss die Unit-Adresse für den Prozessor auf 00 gestellt werden.

|   |        |       |     |     |          |        |       |            |      |
|---|--------|-------|-----|-----|----------|--------|-------|------------|------|
| / | Proc.  | CSSID | SS+ | UA+ | Time-Out | STADET | CHPID | + Explicit | Null |
| _ | Z900VM |       | -   | 00  | No       | No     | --    | No         | ---  |

Anschließend wird die Betriebssystem-Konfiguration durch ein „S“ ausgewählt.

|               |            |      |                   |                       |         |
|---------------|------------|------|-------------------|-----------------------|---------|
| Device number | :          | FF00 | Number of devices | :                     | 32      |
| Device type   | :          | IQD  |                   |                       |         |
| /             | Config. ID | Type | SS                | Description           | Defined |
| S             | 05390      | MVS  |                   | M3000H30 Davinci IODF |         |

Nach Bestätigen der nächsten Dialoge ohne Änderungen erscheint der gleiche Dialog wieder, diesmal jedoch mit dem Eintrag „Yes“ in der Spalte „Defined“. Durch nochmaliges Bestätigen erscheint das Gerät in der „I/O Device List“.

|   |         |        |     |    |      |      |      |      |      |      |      |      |
|---|---------|--------|-----|----|------|------|------|------|------|------|------|------|
| / | Number  | Type + | CSS | OS | 1--- | 2--- | 3--- | 4--- | 5--- | 6--- | 7--- | 8--- |
| _ | FF00,32 | IQD    | 1   | 1  | FF00 |      |      |      |      |      |      |      |

Damit ist die Hardware-Definition abgeschlossen. Anschließend muss das System noch darauf vorbereitet werden, die geänderte Hardware-Definition zu verwenden. Da die aktuelle Konfiguration nicht verändert werden kann, muss hierfür einen Umweg gewählt

werden.

Die erste Möglichkeit ist, dass für den nächsten IPL eine andere Hardware-Definition verwendet wird. Alternativ kann auf dem System eine temporäre Kopie der aktuellen Definition aktiviert werden und anschließend das Original aktualisiert werden. Nach dem nächsten Start des Betriebssystems sind die Änderungen aktiv.

### 7.1.3 Einrichtung des Netzwerks

Damit die neu definierte Netzwerkkarte auch verwendet wird, muss die TCP/IP-Konfiguration von z/OS angepasst werden [27]. Unter ZOS001 ist die aktive TCP/IP-Konfiguration im Dataset *ADCD.Z18.TCPPARMS(PROFILE)* zu finden.

Die wichtigen Zeilen für die Konfiguration des Netzwerks werden im Folgenden gezeigt. Zunächst erfolgt die Definition des Geräts:

```
DEVICE IUTIQDFF MPCIPA      NONR
LINK IUTIQDFFL IPAQIDIO IUTIQDFF
```

Die Buchstaben *FF* am Ende des Gerätenamens *IUTIQDFF* entsprechen der Channel Path ID, die in der Hardware-Definition unter z/OS und bei der Definition der Netzwerkkarte unter z/VM verwendet wurde. Der Link in der zweiten Zeile kennzeichnet durch den Typ *IPAQIDIO*, dass es sich um ein Hipersocket-Netzwerk handelt.

In der Sektion „Home“ wird die IP-Adresse des Systems angelegt:

```
HOME
 10.0.10.110      IUTIQDFFL
```

Die IP-Adresse 10.0.10.110 wird dabei dem Link *IUTIQDFFL* zugeordnet, welcher die Verbindung mit der Netzwerkkarte herstellt. Diese Zeile muss für jede z/OS-Kopie angepasst werden, damit nicht zwei Rechner mit der gleichen IP-Adresse aktiv sind.

Anschließend wird noch die lokale Netzmaske sowie die Default-Route (Gateway) eingerichtet:

```
GATEWAY
10      =      IUTIQDFFL 1492    0.255.255.0  0.0.10.0
defaultnet 10.0.10.254 IUTIQDFFL 1492 0
...
START IUTIQDFF
```

Die letzte Zeile sorgt für den automatischen Start des Links beim Start des TCP/IP-Stacks.

Damit ist die Netzwerk-Konfiguration von z/OS abgeschlossen.

## 7.2 Weitere z/OS-Instanzen erstellen

Es gibt verschiedene Möglichkeiten, ein weiteres z/OS-System aus einem bestehenden zu erzeugen. Die einfachste ist die direkte Kopie eines vorhandenen Systems. Dabei werden alle Festplatten (DASD), die zu einem System gehören, kopiert und dem neuen System zugewiesen. Im Nachhinein muss lediglich die IP-Adresse des geklonten Systems angepasst werden.

Eine weitere Möglichkeit ist das Erzeugen eines minimalen, speziell konfigurierten z/OS-Systems, welches zum großen Teil die DASDs einer vollständigen z/OS-Installation nutzt. Lediglich Bereiche, die für jedes System einzigartig sein müssen, werden dabei für ein neues System speziell angelegt. Andere Bereiche, die sich alle Systeme teilen können, wie zum Beispiel Programme und Bibliotheken, werden dem System mit Lese-rechten bereitgestellt.

### 7.2.1 Einfache z/OS-Kopie

Dieser Abschnitt beschreibt die einfache Kopie eines kompletten z/OS-Systems.

Zunächst ist ein Benutzer nötig, der die neue z/OS-Instanz aufnimmt. Die Definition ist der des ursprünglichen Systems ähnlich, jedoch müssen andere DASDs verwendet werden.

```
USER ZOS002 SECRET 512M 1024M G
INCLUDE IBMDFLT
CPU 00 CPUID 111000
CPU 01 CPUID 111111
IPL CMS PARM AUTOOCR
IUCV ANY
IUCV ALLOW
MACHINE ESA
OPTION QUICKDSP MAINTCCW RMCHINFO
CONSOLE 0700 3270 A
NICDEF FF00 TYPE HIPER LAN SYSTEM LINHIPER CHPID FF
SPECIAL 0701 3270
SPECIAL 0702 3270
SPECIAL 0703 3270
SPECIAL 0704 3270
SPECIAL 0705 3270
LINK ZOSMAINT 0191 0191 RR
MDISK 0A80 3390 DEVNO 5530 MR
MDISK 0A81 3390 DEVNO 5531 MR
MDISK 0A82 3390 DEVNO 5532 MR
MDISK 0A83 3390 DEVNO 5533 MR
MDISK 0A84 3390 DEVNO 5534 MR
```

Wenn das System ZOS001 als Grundlage genommen wird, müssen nun die DASDs mit den Adressen 552B bis 552F auf die DASDs mit den Adressen 5530 bis 5534 kopiert werden. Aus Sicht von z/OS ändert sich nichts, da die virtuellen Adressen 0A80 bis 0A84 gleich geblieben sind. Das Kopieren der DASDs findet wie in Kapitel 3.4.2 beschrieben mit dem Programm *DDR* statt.

Durch die generische Konfiguration des Start-Skripts für den Benutzer ZOSMAINT führt ein Login des Gasts direkt den IPL aus. Nach dem IPL sollte die IP-Adresse innerhalb der Installation angepasst werden, sodass keine Kollision mit der IP-Adresse von ZOS001 entsteht.

## 7.2.2 Spezielle minimale z/OS-Instanzen

Dieser Abschnitt beschreibt das Cloning von z/OS-Instanzen anhand eines minimalen Master-Clones. Zunächst muss dieser Master-Clone erzeugt werden, was eine einmalige Aufgabe ist. Im Nachhinein lässt sich dieser Master-Clone als Grundlage für das einfache Erzeugen weiterer z/OS-Gastsysteme verwenden.

Das System der minimalen z/OS-Gäste wird in der Diplomarbeit von Herrn Uli Stormanns von der Fachhochschule Aachen beschrieben [28].

Für weitere z/OS-Instanzen wird zunächst der Master-Clone erstellt. Dieses System ist ein stark reduziertes z/OS-System, das große Teile der Installation des Basis-z/OS nutzt. Hierfür wird dem System Lese-Zugriff auf die 3390-DASDs der Basis-Installation gegeben.

### **Vorbereitungen unter z/VM**

Zunächst muss wieder ein Benutzer angelegt werden und es muss genügend freier DASD-Speicher bereitstehen. Für das System werden 7 Minidisks verwendet. Auf den Disks liegen Daten, die für einen eigenständigen Start benötigt werden, die restlichen Daten, für die ein lesender Zugriff ausreicht, werden von den DASDs der Basis-Installation gelesen.

Auf den einzelnen Minidisks befinden sich für den Start des Systems nötige Dateien wie der z/OS Nucleus (Kern des Betriebssystem) und die Konfigurationsdateien (Parmlib), der Master Catalog und Benutzerkataloge, die RACF-Datenbank sowie Auslagerungs- und Spooling-Dateien. Weiterhin wird Festplattenspeicher für MVS-Datensätze und die *Unix System Services* bereitgestellt.

Die vollständige Definition des Benutzers sieht wie folgt aus:

```
USER ZOSMSTR SECRET 192M 512M G
  INCLUDE IBMDFLT
  CPU 00 CPUID 111000
  CPU 01 CPUID 111111
  IPL CMS PARM AUTOOCR
  IUCV ANY
  IUCV ALLOW
  MACHINE ESA
  OPTION QUICKDSP MAINTCCW RMCHINFO CFUSER TODENABLE
  CONSOLE 0700 3270 A
  NICDEF FF00 TYPE HIPER LAN SYSTEM LINHIPER CHPID FF
  SPECIAL 0701 3270
  SPECIAL 0702 3270
  SPECIAL 0703 3270
  SPECIAL 0704 3270
  SPECIAL 0705 3270
  SPECIAL FF20 MSGPROC SYS1CF01
  SPECIAL FF30 MSGPROC SYS1CF02
  LINK ZOS001 0A80 0A80 RR
  LINK ZOS001 0A81 0A81 RR
  LINK ZOS001 0A82 0A82 RR
  LINK ZOS001 0A83 0A83 RR
  LINK ZOS001 0A84 0A84 RR
  LINK BEGEL 0191 0191 RR
  MDISK 0300 3390 1 100 ZOSC01 MW READ WRITE MULTI
  MDISK 0301 3390 101 100 ZOSC01 MW READ WRITE MULTI
  MDISK 0302 3390 201 100 ZOSC01 MW READ WRITE MULTI
  MDISK 0303 3390 301 1000 ZOSC01 MW READ WRITE MULTI
  MDISK 0306 3390 1501 1000 ZOSC01 MW READ WRITE MULTI
  MDISK 0305 3390 1401 100 ZOSC01 MW READ WRITE MULTI
  MDISK 0304 3390 1301 100 ZOSC01 MW READ WRITE MULTI
```

Die DASDs A80 bis A84 werden per Link mit Leserechten eingebunden. Die Minidisks 300 bis 306 werden für die Konfiguration der minimalen Installation verwendet.

### ***Erstellen von Kopien des Master-Klons***

Für Kopien des Master-Klons wird wie in Kapitel 6.1.4 beschrieben ein Prototyp erzeugt. In dieses Profil werden sämtliche Definitionen außer die DASDs aufgenommen, die vom Benutzer mit Schreibrechten verwendet werden (Adressen 300 bis 306).

Nach der Erstellung eines neuen Nutzers unter Verwendung des Prototypen müssen die Minidisks nun vom Master-Clone wie in Kapitel 6.2.3 beschrieben kopiert werden.

```
dirm for syslz11 clonedisk 300 zosmstr 300 g LINUX
dirm for syslz11 clonedisk 301 zosmstr 301 g LINUX
dirm for syslz11 clonedisk 302 zosmstr 302 g LINUX
dirm for syslz11 clonedisk 303 zosmstr 303 g LINUX
dirm for syslz11 clonedisk 304 zosmstr 304 g LINUX
dirm for syslz11 clonedisk 305 zosmstr 305 g LINUX
dirm for syslz11 clonedisk 306 zosmstr 306 g LINUX
```

Alternativ kann auch die komplette DASD kopiert werden (Kapitel 3.4.2), auf welcher sich die Minidisks 300 bis 306 des Master-Clones befinden. Anschließend müssen die Minidisks mit den gleichen Offsets wie auf der Quell-DASD definiert werden. Wenn als Ziel die Festplatte mit dem Label ZOSC02 als Ziel ausgewählt wurde, sieht das Einrichten der Minidisks wie folgt aus:

```
dirm for sys1z11 amdisk 300 3390 1 100 ZOSC02 MW READ WRITE MULTI
dirm for sys1z11 amdisk 301 3390 101 100 ZOSC02 MW READ WRITE MULTI
dirm for sys1z11 amdisk 302 3390 201 100 ZOSC02 MW READ WRITE MULTI
dirm for sys1z11 amdisk 303 3390 301 1000 ZOSC02 MW READ WRITE MULTI
dirm for sys1z11 amdisk 304 3390 1301 100 ZOSC02 MW READ WRITE MULTI
dirm for sys1z11 amdisk 305 3390 1401 100 ZOSC02 MW READ WRITE MULTI
dirm for sys1z11 amdisk 306 3390 1501 1000 ZOSC02 MW READ WRITE MULTI
```

Nachdem die Definition komplett ist, kann System gestartet werden. Hierfür kann das gleiche Start-Skript verwendet werden, das auch bei der Basis-Installation des minimalen Systems zum Einsatz kommt.

Es ist wichtig, dass nach dem Start die IP-Adresse des neuen Systems angepasst wird, um doppelte IP-Adressen zu vermeiden. Das kopierte System ist jetzt voll funktionsfähig und kann verwendet werden.

## 8 Ausblick

### 8.1 Zusammenfassung

In der Arbeit wurde der Aufbau einer komplexen und umfangreichen Installation beschrieben, die auf dem z900-Großrechner der Universität Leipzig umgesetzt wurde. In einer bereits vorhandenen z/VM-Installation (First Level z/VM) wurde ein weiteres z/VM als Gast-Betriebssystem in Betrieb genommen (Second Level z/VM). Dieses System wurde verwendet, um z/OS und Linux zu betreiben sowie Mechanismen zur einfachen Erzeugung neuer Betriebssystem-Instanzen bereitzustellen.

Zunächst wurde das Second Level z/VM eingerichtet und das Netzwerk konfiguriert. Es wurde an das bereits vorhandene interne Netzwerk des First Level z/VM angeschlossen und stellt für seine eigenen Gäste ein separates internes Netzwerk zur Verfügung, für das es als Netzwerk-Router dient.

Innerhalb des Second Level z/VM wurden nun Gastsysteme in Betrieb genommen und für das Cloning neuer Instanzen vorbereitet. Als Gastsysteme wurden z/OS 1.8 und SuSE Linux Enterprise Server (SLES) 10 eingerichtet. Anschließend wurde beschrieben, was notwendig ist, neue Instanzen dieser Betriebssysteme zu erzeugen.

Als eine Voraussetzung für schnelle und einfache Kopien von z/OS wurde in der Diplomarbeit von Herrn Uli Stormanns an der Fachhochschule Aachen eine minimale z/OS-Installation erzeugt und in Betrieb genommen, die auf nur einer System-Festplatte untergebracht werden kann.

Die im Rahmen der Diplomarbeit erstellte Konfiguration erlaubt es, für Praktika im Rahmen der akademischen Lehre eine Umgebung zur Verfügung zu stellen, in der auch administrative Aufgaben gelehrt werden können. Dabei wird einem Studenten die vollständige Kontrolle über eine eigene Instanz eines Großrechner-Betriebssystems gegeben. Andere Systeme sind vor Fehlern, die in diesem Zusammenhang entstehen können, geschützt. Bisher war dies nicht möglich, da alle Studenten auf dem selben System gearbeitet haben und so ein Fehler bei der Administration ebenfalls andere Studenten beeinträchtigen konnte.

## **8.2 Bewertung**

Ziele dieser Arbeit waren es, eine Konfiguration von z/VM in Betrieb zu nehmen, mit der z/OS und Linux als Gastsysteme betrieben und mit geringen Aufwand weitere Instanzen erstellt werden können. Das Betriebssystem z/VM für das Anlegen des Cloning-Systems wurde erfolgreich konfiguriert. Als Gäste wurden z/OS 1.8 und SuSE Linux Enterprise Server 10 in Betrieb genommen.

Der Aufwand für das Cloning eines Linux-Systems umfasst das Anlegen eines neuen Benutzers, das Kopieren der System-Festplatte sowie das Aktualisieren einer Datei für die Zuordnung der IP-Adresse. Damit wurde das Ziel einer einfachen Inbetriebnahme neuer Linux-Gäste erfüllt.

Um eine Kopie von z/OS in Betrieb zu nehmen, müssen ein neuer Benutzer angelegt, mehrere Minidisks kopiert und die Konfiguration der IP-Adresse innerhalb der neuen Kopie vorgenommen werden. Das Einrichten der IP-Adresse vor dem Systemstart, wie es bei Linux der Fall ist, ist nicht möglich. Das Ziel, auf einfache Art und Weise neue z/OS-Instanzen zu erzeugen, wurde erreicht.

Bei beiden Cloning-Varianten sind Vereinfachungen denkbar, welche mit erhöhtem Zeitaufwand erreichbar sind. Das Setzen der IP-Adresse unter z/OS könnte sich verbessern lassen, um eine ähnliche Konfiguration wie bei den Linux-Gästen zu erlauben. Auch wäre das Erstellen neuer Instanzen durch nur einen Befehl denkbar.

## **8.3 Weitere Themen**

Auf Basis des in dieser Diplomarbeit erstellten Systems können die bestehenden Übungsaufgaben für die akademische Lehre um solche ergänzt werden, bei denen der Student Administrator-Rechte benötigt. So kann zum Beispiel ein Systemstart (IPL) von z/OS durchgeführt werden. Weiterhin wären Aufgaben wie die Konfiguration von z/OS denkbar, beispielsweise das Starten oder Einrichten von Diensten. Beim Betrieb eines Parallel Sysplex können spezielle Anwendungen genauer betrachtet werden, wie zum Beispiel der Sysplex-Betrieb der Datenbank DB2 oder des WebSphere Application Servers für z/OS.

Das einfache Erstellen von Betriebssystem-Instanzen ist für künftige Diplomarbeiten interessant, da somit Produktiv-Systeme durch Tests nicht in Mitleidenschaft gezogen werden. Erste Installationen für diesen Zweck sind bereits in Betrieb.

Ein anderes Thema stellt die Spezialisierung von Betriebssystem-Instanzen für bestimmte Einsatz-Gebiete dar. So könnten für ein Datenbank-Praktikum spezielle Linux- und z/OS-Installationen erstellt und betrieben werden.

Für Linux wäre es interessant, von mehreren Instanzen gemeinsam genutzte Partitionen zu verwenden, so wie dies bei z/OS bereits der Fall ist. Hier sind sowohl Netzwerk-Dateisysteme wie *NFS*, aber auch Cluster-Dateisysteme wie das *Global File System (GFS)* oder das *Oracle Cluster File System 2 (OCFS2)* zu betrachten.

## 9 Abkürzungen und Glossar

|                       |   |
|-----------------------|---|
| Channel Path<br>CHPID | Verbindung von Control Units mit dem Channel-Subsystem, die Channel Path ID (CHPID) dient der eindeutigen Identifizierung |
| CMS                   | Conversational Monitor System   |
| CP                    | Control Program, Hypervisor von z/VM  |
| CU                    | Control Unit, Steuereinheit für den Anschluß von Ein-/Ausgabe-Einheiten   |
| DASD                  | Direct Access Storage Device, Festplatte  |
| DIRMAINT              | Directory Maintenance Facility, Dienst zur Verwaltung des z/VM User Directory   |
| HCD                   | Hardware Configuration Definition   |
| Hipersocket           | Technologie für die Hochgeschwindigkeitskommunikation zwischen Servern innerhalb von z/Series-Rechnern                    |
| IPL                   | Initial Program Load, Start eines Betriebssystems   |
| IODF                  | I/O Definition File, Datei zur Definition der Ein-Ausgabe-Geräte, die an einen z/Series-Rechner angeschlossen sind        |
| IQDIO                 | Internal Queued Direct Input/Output, virtuelles Gerät als Grundlage für Hipersockets                                      |
| ISPF                  | Interactive System Productivity Facility, eine Benutzeroberfläche von z/OS für 3270-Terminals                             |
| JCL                   | Job Control Language, Beschreibungssprache für Stapelverarbeitung   |
| LAN                   | Local Area Network, lokales Netzwerk  |
| LPAR                  | Logical Partition, Hardware-Virtualisierung eines z/Series-Rechners   |
| MDISK                 | Minidisk, Konzept zum Partitionieren einer DASD   |
| MVS                   | Multiple Virtual Storage, Vorgänger von OS/390 und z/OS   |
| NIC                   | Network Interface Card, Netzwerkkarte   |
| QDIO                  | Queued Direct I/O, Architektur für Datentransfer (z.B: Netzwerk)  |
| RACF                  | Ressource Access Control Facility, Implementierung der Sicherheitsschnittstelle von z/OS oder z/VM                        |
| REXX                  | Restructured Extended Executor Language, Skriptsprache  |

## 10 Literaturverzeichnis

- [1] Herrmann, P.; Kebschull, U; Spruth, W.G.: Einführung in z/OS und OS/390. Oldenbourg Verlag, 2004, ISBN 3-486-27393-0
- [2] Universität Leipzig: Uni Leipzig z/OS Web Application Server. <http://jedi.informatik.uni-leipzig.de/de/>, 2008 (abgerufen: 13.01.2009)
- [3] IBM: IBM Mainframes. [http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe\\_intro.html](http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_intro.html), 2008 (abgerufen: 13.01.2009)
- [4] IBM: CICS - An Introduction. [ftp://service.boulder.ibm.com/software/http/cics/PDF/cics\\_introduction.pdf](ftp://service.boulder.ibm.com/software/http/cics/PDF/cics_introduction.pdf), 2004 (abgerufen: 13.01.2009)
- [5] IBM Redbooks: Introduction to the New Mainframe: z/VM Basics. IBM, 2007, IBM Form Nr. SG24-7316-00
- [6] Jay Maynard: The Hercules System/370, ESA/390, and z/Architecture Emulator. <http://www.hercules-390.org/>, 2009 (abgerufen: 13.01.2009)
- [7] Citrix: Xen Paravirtualization. <http://www.xen.org/about/paravirtualization.html>, 2009 (abgerufen: 13.01.2009)
- [8] VMware, Inc.: VMware. <http://www.vmware.com/de/>, 2009 (abgerufen: 13.01.2009)
- [9] IBM: z/VM CP Commands and Utilities Reference. IBM, 2006, IBM Form Nr. SC24-6081-03
- [10] IBM Redbooks: Linux on IBM zSeries and S/390: Large Scale Linux Deployment. IBM, 2002, IBM Form Nr. SG24-6824-00
- [11] IBM: z/VM Virtual Machine Operation. IBM, 2006, IBM Form Nr. SC24-6128-02
- [12] IBM: z/VM CMS Commands and Utilities Reference. IBM, 2005, IBM Form Nr. SC24-6073-01
- [13] IBM: z/VM CP Planning and Administration. IBM, 2006, IBM Form Nr. SC24-6083-03
- [14] IBM: z/VM Internet Library. <http://www.vm.ibm.com/library/>, 2009 (abgerufen: 13.01.2009)
- [15] IBM: z/VM TCP/IP Planning and Custimization. IBM, 2006, IBM Form Nr. SC24-6125-02
- [16] IBM: z/VM TCP/IP User's Guide. IBM, 2006, IBM Form Nr. SC24-6127-02

- [17] IBM: z/VM Directory Maintenance Facility Tailoring and Administration Guide. IBM, 2005, IBM Form Nr. SC24-6135-01
- [18] IBM: z/VM Directory Maintenance Facility Command Reference. IBM, 2006, IBM Form Nr. SC24-6133-02
- [19] IBM: z/VM Running Guest Operating Systems. IBM, 2005, IBM Form Nr. SC24-6115-01
- [20] IBM: z/VM Getting Started with Linux on System z9 and zSeries. IBM, 2005, IBM Form Nr. SC24-6096-01
- [21] Novell: Suse Linux Enterprise Server 10: Start-Up Guide. <http://www.novell.com/documentation/sles10/pdfdoc/startup/startup.pdf>, 2008 (abgerufen: 13.01.2009)
- [22] GNU: GNU Screen. <http://www.gnu.org/software/screen/screen.html>, 2008 (abgerufen: 13.01.2009)
- [23] IBM: Application Developers Controlled Distribution (ADCD). <http://dtsc.dfw.ibm.com/adcd.html>, 2008 (abgerufen: 13.01.2009)
- [24] IBM Redbooks: OS/390 (and z/OS) New Users Cookbook. IBM, 2001, IBM Form Nr. SG24-6204-00
- [25] IBM: z/OS Internet Library. <http://www-1.ibm.com/servers/eserver/zseries/zos/bkserv/>, 2009 (abgerufen: 13.01.2009)
- [26] IBM Redbooks: Introduction to the New Mainframe: z/OS Basics. IBM, 2006, IBM Form Nr. SG24-6366-00
- [27] IBM Redbooks: Hipersockets Implementation Guide. IBM, 2007, IBM Form Nr. SG24-6816-01
- [28] Uli Stormanns: Erstellung eines Parallel Sysplex in einer virtuellen Umgebung. Fachhochschule Aachen, 2009

## 11 **Abbildungsverzeichnis**

|  |    |
|--|----|
| Abbildung 1: Großrechner-Struktur.....   | 10 |
| Abbildung 2: Gäste unter z/VM.....       | 18 |
| Abbildung 3: Netzwerk-Konfiguration..... | 39 |

## Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe, insbesondere sind wörtliche oder sinngemäße Zitate als solche gekennzeichnet.

Mir ist bekannt, dass Zuwiderhandlung auch nachträglich zur Aberkennung des Abschlusses führen kann.

Leipzig, 17.02.2009