INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Fortgeschrittenenpraktikum

Evaluierung des Ressourcen Bedarfs für die Live Migration virtueller Maschinen

Bastian Asam

INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Fortgeschrittenenpraktikum

Evaluierung des Ressourcen Bedarfs für die Live Migration virtueller Maschinen

Bastian Asam

Aufgabensteller:	Prof. Dr. Heinz-Gerd Hegering
Betreuer:	Tobias Lindinger Feng Liu
Abgabetermin:	12. März 2010

Hiermit versichere ich, dass ich die vorliegende Diplomarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 12. März 2010

. . . . . .
(Unterschrift des Kandidaten)

Abstract

Die Live Migration ermöglicht der Virtualisierungstechnologie einige wichtige Besonderheiten. Genauere Untersuchungen von Live Migrationen hinsichtlich deren Ressourcenbedarfs gibt es bis jetzt noch nicht. Die derzeit veröffentlichten Downtimes solcher Live Migrationen wirken ambitioniert und sollen vor dem Nutzer transparente Migrationen ermöglichen, d.h. der Nutzer soll die Migration nicht bemerken.

In dieser Arbeit werden zwei Testreihen durchgeführt, die das Verhalten zum einen während stetiger Ping Anfragen und zum anderen während eines Video Streams untersuchen. Besonderes Augenmerk wird dabei auf die dem Nutzer gegenüber auftretende Dienstausfallzeit gelegt. Wie sehr wird beispielsweise das Video durch eine Live Migration gestört. In jeder Testreihe werden nacheinander vier unterschiedliche Ressourcen belastet, zum Vergleich wird zusätzlich einmal ohne Last getestet. Gemessen wird der Einfluss von Speicher, Netzwerk, CPU und I/O Last auf das Verhalten einer Live Migration.

Damit soll untersucht werden, wie sehr der Nutzer bestimmter Dienste von der Live Migration gestört wird, wie sehr die Störung von bestimmten unter Last gesetzten Ressourcen abhängt und welche Ressourcen eine Live Migration für sich selbst beansprucht.

Inhaltsverzeichnis

1 Einleitung

Virtualisierung erlebt seit den letzten Jahren einen regelrechten Boom. Auch im Jahr 2010 gehört die Virtualisierung nach einer Umfrage des BITKOM zu den Top IT- und Telekommunikations-Trends [BI10]. Sie ist eine ausgereifte und kostengünstige Möglichkeit aktuelle Herausforderungen in der IT Branche zu meistern. Zu diesen zählen vor allem Kostensenkungen aufgrund der Wirtschaftskrise, Leistungssteigerung sowie der bewusstere Umgang mit Energie aufgrund der Klimaerwärmung, bekannt als Green-IT.

„Virtualisierung bezeichnet im übergreifenden Sinne Software- oder Hardware-Techniken, welche eine Abstraktionsschicht zwischen dem Benutzer (oder Applikationen oder Schnittstellen) einerseits und physischen Ressourcen, wie z. B. Hardwarekomponenten eines Rechners andererseits, implementieren“ [Lip]. Die Virtualisierung erlaubt weitere Techniken, die in Kombination dem Produkt Virtualisierung so viele Vorteile ermöglichen. Eine sehr wichtige dabei ist die Migration und insbesondere die Live Migration. Diese soll es ermöglichen, ganze Betriebssysteme von einem physischen Host auf einen anderen zu transferieren, wobei die sogenannte Downtime, also die Zeit in der das Betriebssystem deaktiviert wird, so kurz sein soll, dass es dem Benutzer dieses Betriebssystems nicht auffällt. Konkret wird dies z.B. in dem häufig zitiertem Paper 'Live Migration of Virtual Machines' [CFH$^+$05] beschrieben, deren Tests unter der Virtualisierungssoftware Xen bei realistischen Server Workloads Downtime Werte zwischen 60 und 210 Millisekunden ergeben. Nur im künstlichen worst-case Szenario wird eine Downtime von 3,5 Sekunden erreicht.

In dieser Arbeit wird genauer untersucht, welche Ressourcen bei unterschiedlichen Workloads beansprucht werden und wie sich dies auf die Migrationszeit und die Downtime auswirkt.

2 Grundlagen

Um die Testresultate interpretieren zu können, ist ein wenig Grundwissen über die virtuelle Maschine Xen im allgemeinen und über deren Technik der Live Migration im besonderen von Nöten.

2.1 Xen

Die Software Xen ist ein Virtuelle-Maschinen-Monitor (VMM), der unter der GNU General Public License veröffentlicht wird [oCCL08]. Diese ermöglichen das gleichzeitige Ausführen mehrerer, auch unterschiedlicher Betriebssysteme, sogenannte Gastsysteme, auf einem physischen Host. Xen selbst läuft dabei in der sogenannten Domain 0, während jedem Gastsystem eine Domain U zugewiesen wird. Xen gehört zu der Klasse der Paravirtualisierungen. Diese erlauben den Gastsystemen mit sehr kleinem Overhead direkt auf die vorhandene Hardware zuzugreifen und gelten daher als performanter als Vollvirtualisierungslösungen [BDF$^+$]. Der Hauptnachteil der Paravirtualisierung war lange Zeit die Notwendigkeit, das Gastbetriebssystem anpassen zu müssen. Erst mit Version 3.0 und dem Einsatz neuer Prozessoren von Intel (Virtualization Technology VT bzw. Vanderpool) oder von AMD (Pacifica) sind Anpassungen im Kernel des Gastbetriebssystems nicht mehr notwendig. Nun sind auch proprietäre Betriebssysteme wie z.B. Microsoft Windows einsetzbar [Rad06].

2.2 Live Migration

Wie anfangs schon erwähnt, ist die Migration eine Schlüsseltechnologie für Virtualisierungslösungen und ermöglicht erst eine Reihe von virtualisierungsspezifischen Vorteilen zu realisieren. Der Begriff Migration kommt ursprünglich aus dem Lateinischen. Das Verb migrare bedeutet soviel wie wandern oder auswandern. Im politischen Kontext bezeichnet die Migration alle Formen räumlicher Mobilität von Individuen, (religiösen, ethnischen etc.) Gruppen, Minderheiten und Volksteilen [SK06]. In dieser Arbeit ist das Umziehen von virtuellen Umgebungen gemeint. Eine solche virtuelle Umgebung wird virtuelle Maschine (VM) genannt und beherbergt je nach Art der Virtualisierung nur einen Prozess oder ein ganzes Betriebssystem. Die Idee und Notwendigkeit der Migration von Prozessen wurde schon Ende der 70er Jahre von Solomon und Fenkel diskutiert. Bereits 1981 folgen dann erste erfolgreiche Implementierungen von Lazowska u. a., sowie Rashid und Robertson [HFV96]. Die dort beschriebene Art der Migration bringt jedoch eine Menge Schwierigkeiten mit sich, wonach unter anderem aufgrund von verbleibenden Abhängigkeiten der Prozesse, das alte ehemalige Hostsystem verfügbar bleiben muss. Daher spielt es heute in der Praxis eine untergeordnete Rolle.

Diese Arbeit beschäftigt sich nur mit der Live Migration von Betriebssystemen. Eine Live Migration ist eine spezielle Form der Migration mit dem Fokus auf eine möglichst geringe

Downtime. Die Downtime bezeichnet die Zeit, in der Dienste, die auf dem virtualisierten und zu migrierenden Betriebssystem ausgeführt werden, nicht verfügbar sind.

Die Live Migration ganzer Betriebssysteme bietet weitreichende Möglichkeiten. Sie ermöglicht es zum Beispiel den urspünglichen Host nach der erfolgreichen Migration abzuschalten um Wartungsarbeiten zu erleichtern. Ebenso wird hierbei der komplette Speicher mitsamt interner Kernel- und Anwendungsstatus konsistent und effizient transferiert, so dass z.B. Netzwerkverbindungen erhalten bleiben [CFH+05]. Virtualisierte Betriebssysteme trennen ausserdem den physischen Host vollständig von den Nutzern, denen innerhalb ihrer virtuellen Maschine keine Restriktionen auferlegt werden müssen und vereinfachen dadurch die Administration immens. Komplette virtuelle Maschinen lassen sich duplizieren, wodurch beispielsweise eine Standard Konfiguration leicht verbreitet werden kann. Und zur Last Verteilung lassen sich die virtuellen Maschinen beliebig im Server Cluster verschieben. Interessant ist hierbei vor allem, inwiefern der Nutzer von Diensten, die auf solchen virtuellen Maschinen laufen, von der Migration Notiz nimmt und inwieweit sich eine Migration auf den physischen Server und damit auch auf die übrigen virtuellen Maschinen auswirkt.

Um die Auswirkungen so gering wie möglich zu halten, wird in der Live Migration versucht, die virtuelle Maschine nur so kurz wie möglich anzuhalten. Ein wichtiger Punkt dabei ist, den Hintergrundspeicher über ein Network-attached Storage (NAS) zu realisieren, um diese Daten nicht auch migrieren zu müssen. Damit ergeben sich leider auch zwei Nachteile, da dieses NAS natürlich von beiden in der Migration involvierten physischen Servern aus erreichbar sein muss und I/O Operationen über das Netzwerk langsamer sind als lokale Festplattenzugriffe. Als vorteilhaft erweist sich aber, dass es nun ausreicht, den Inhalt des Arbeitspeichers und den Status des Kernels und aller Anwendungen zu übertragen, selbst wenn dieser von unterschiedlichen Prozessen zusammen verwendet wird. So wird es sogar möglich, den Status eines verwendeten Netzwerk Protokolls zu übertragen, ohne bestehende Verbindungen trennen zu müssen. Dabei behält jede virtuelle Maschine ihre IP und MAC Adresse, wodurch kein Weiterleitungsmechanismus die weitere Verfügbarkeit des alten Systems diktiert. Um dem Netz den Umzug bekannt zu geben, wird eine unaufgeforderte ARP reply Nachricht von dem migriertem Host entweder gebroadcasted oder, wenn Broadcasts nicht erlaubt werden, direkt an alle bekannten Interfaces aus dem ARP Cache gesendet. Alternativ reicht es in einem geswitchtem Netzwerk aus, die original Ethernet MAC Adresse zu behalten, wodurch der Switch den Umzug zu einem neuen Port erkennen kann. Diese Funktionalität ist zwingende Voraussetzung für das Ziel, Migrationen möglichst transparent den Nutzern gegenüber durchzuführen.

Naiver Standard Ansatz bei der normalen Migration ist einfach das Gastbetriebssystem zu stoppen, alle Daten zu transferieren und auf dem neuen Host wieder zu starten (pure stop-and-copy). Dies würde zwar zu einer kurzen totalen Migrationszeit (total migration time) führen, allerdings auch zu einer inakzeptabel langen Downtime der laufenden Dienste. Zur möglichst optimalen Reduzierung der Downtime wird bei der Live Migration der Ablauf in insgesamt 6 Phasen unterteilt:

- Phase 0: Pre-Migration
  Eine aktive VM läuft auf einem physischem Host A. Um zukünftige Migrationen zu beschleunigen, wird ein Zielhost, auf dem die für eine Migration nötigen Ressourcen garantiert werden können, vorgewählt.

- Phase 1: Reservation
  Eine Anfrage auf Migration eines Betriebssystems von Host A zu Host B wird initialisiert. Host B bestätigt, dass die notwendigen Ressourcen zur Verfügung stehen und reserviert einen VM Container in dieser Größe. Falls hier ein Fehler auftreten sollte, läuft die VM davon unbeeinflusst auf Host A weiter.

- Phase 2: Iterative Pre-Copy
  Während der ersten Iteration werden alle Seiten aus dem Arbeitsspeicher der VM von Host A zu Host B transferiert. Die folgenden Iterationen transferieren nur noch die Seiten, die inzwischen wieder verändert (dirtied) worden sind.

- Phase 3: Stop-and-Copy
  Jetzt wird das in der VM laufende Betriebssystem auf Host A gestoppt, der Netzwerkverkehr wie oben beschrieben zu B umgeleitet und alle restlichen inkonsistenten Seiten, sowie der CPU Status werden transferiert. Am Ende existieren zwei gleiche Kopien der VM auf Host A und B. A bleibt die primäre Kopie und wird im Fehlerfall wieder aktiviert.

- Phase 4: Commitment
  Host B signalisiert A, dass er erfolgreich ein konsistentes Betriebssystem Image erhalten hat. Nachdem dies von B bestätigt wurde, kann A die original VM verwerfen und B wird primärer Host.

- Phase 5: Activation
  Nachdem die migrierte VM auf B wieder aktiviert ist, werden Gerätetreiber mit der neuen Maschine neu verbunden, sowie die umgezogenen IP Adressen bekannt gemacht.

Dieser Ansatz stellt sicher, dass zu jeder Zeit mindestens ein konsistentes Betriebssystem Image vorhanden ist, welches im Fehlerfall problemlos gestartet werden kann. Abbildung 2.1 stellt die Phasen noch einmal graphisch dar.

Die Live Migration ist damit die einzige Lösung, mit dem Potenzial derart kurze Downtimes zu ermöglichen, dass der Nutzer diese nicht bemerkt. Diese Arbeit überprüft dies im folgenden unter verschiedenen Bedingungen.

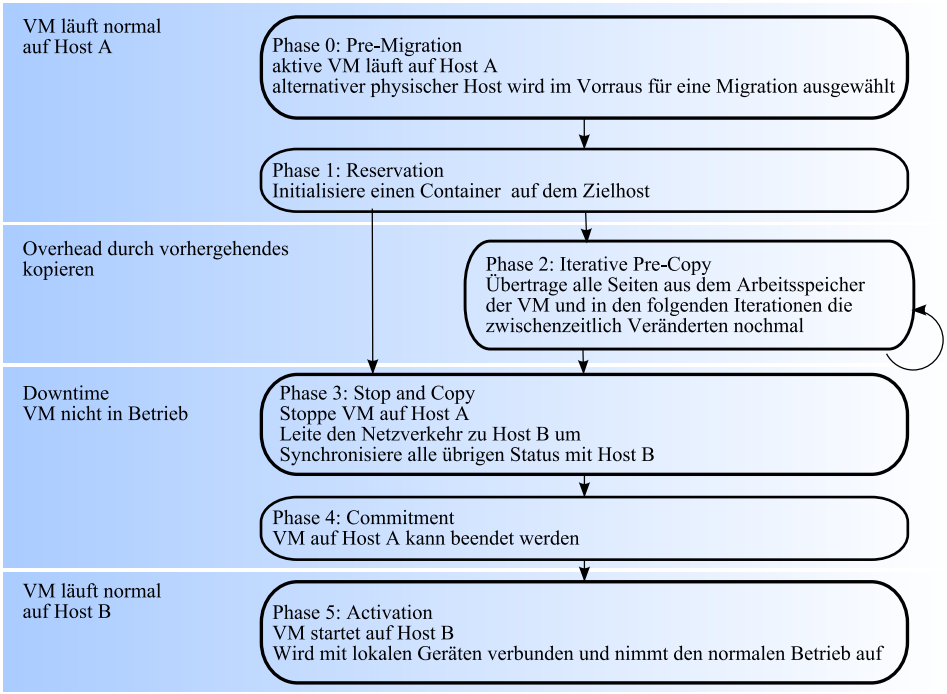

Abbildung 2.1: Phasen einer Live Migration [CFH+05]

3 Testsystem

Zur Durchführung der Tests stehen 3 physische Maschinen mit den Namen Projekt3, Kirsche und Apfel zur Verfügung. Projekt3 dient dabei zur Steuerung und als unbeteiligter Beobachter der Migrationen, sowie als Diensteempfänger. Kirsche und Apfel sind weitgehend ähnliche Systeme. Um eine Live Migration überhaupt zu ermöglichen, wird als physischer Speicher der virtuellen Maschinen ein NAS verwendet, das bei Kirsche und Apfel unter '/mnt/storage3/' gemounted ist. Kirsche und Apfel synchronisieren ihre Systemzeiten per NTP mit der Projekt3 Systemzeit, welche wiederum mit dem universitätsinternen NTP Server (ntp.in.nm.ifi.lmu.de) synchronisiert wird, um einheitliche Zeitstempel zu erhalten und Messungen auf den unterschiedlichen Systemen zeitlich in Beziehung setzen zu können. Auch wenn der universitäre NTP Server ausfällt, ist so gewährleistet, dass die 3 Testsysteme immer synchronisierte Zeiten besitzen.

3.1 Hardware der Testsysteme

	Projekt3	**Kirsche**	**Apfel**
Prozessor	AMD Athlon 64 3200+	Intel Xeon CPU 2.00GHz	Intel Xeon CPU 2.40GHz
Takt	2200 MHz	2004 MHz	2405 MHz
Anzahl CPUs	1	1	2
Speicher	1 GByte	3.5 GByte	3.5 GByte
Netzwerk	100 MBit	1 GBit	1 GBit
Betriebsystem	Debian 2.6.26-1.amd64	Debian 2.6.18.8-xen	Debian 2.6.18.8-xen

Abbildung 3.1: Testsysteme

3.2 Aufbau der Testumgebung

Um störende Einflüsse auf die Migration auf ein Minimum zu begrenzen, wurden die Rechner über 3 getrennte Netze miteinander verbunden. Wie die Grafik 3.2 zeigt, gibt es einmal das Netz 192.168.0.0, über welches die Migration abläuft und im folgenden als Migrationsnetz bezeichnet wird, sowie das Netz 192.168.2.0, über welches der Projekt3 Rechner den Kirsche und Apfel Rechner steuert und im weiteren als Steuerungsnetz bezeichnet wird. Das dritte Netz wird dhcp gesteuert und dient der Verbindung zum gemeinsamen Speicher im Netz. Über das Interface eth1 auf Projekt3 ist zudem über das Netz 192.168.218.0 der Zugang ins Universitätsnetz und damit auch zum Internet gegeben. Projekt3 routet aus dem Steuerungsnetz Anfragen von Kirsche und Apfel ins Internet weiter. Sämtliche Netze sind geswitched, unterscheiden sich allerdings in den möglichen Übertragungsraten, was später zu leichten Modifikationen in den Tests führt.

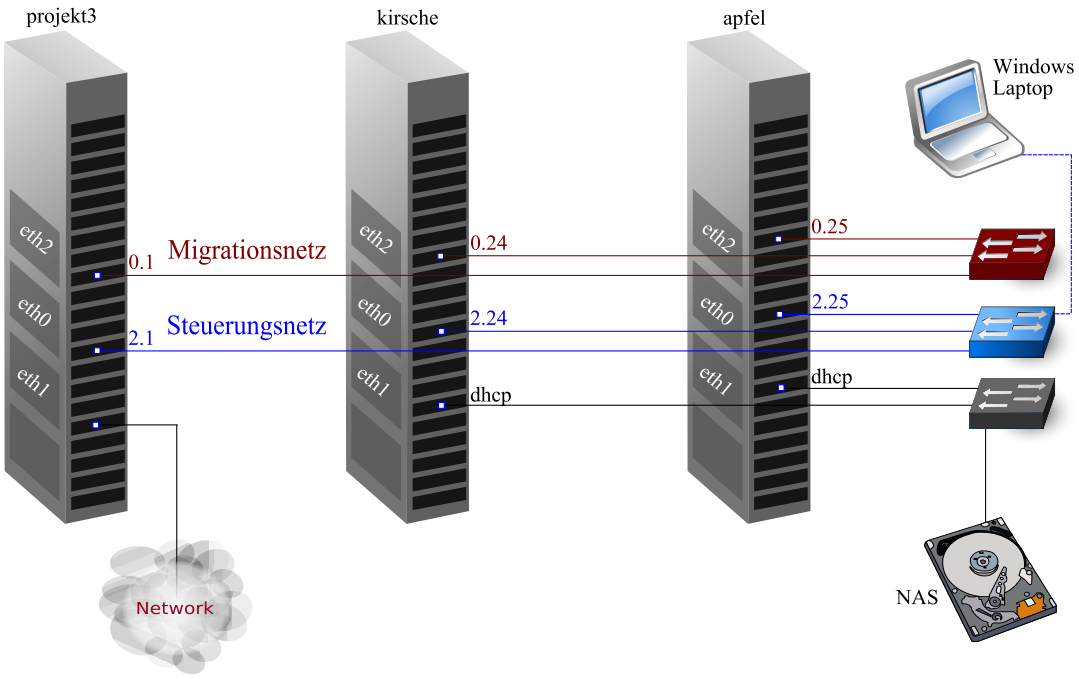

Abbildung 3.2: Testumgebung

3.3 Konfiguration der virtuellen Maschine

Die VM asam wird über die Datei asam.cfg (Listing 3.1) konfiguriert. Anhand dieser wird als Kernel vmlinuz-2.6.18.8-xen (Zeile 9) gesetzt und der VM 128 MB RAM zugewiesen (Zeile 11). Als nächstes werden zwei Image Dateien verlinkt, die auf dem Netzlaufwerk gespeichert sind und als swap Disk (Zeile 18) und normale Festplatte (Zeile 19) dienen. Die swap Datei ist 128 MB und die disk Datei 1 GB groß. Das virtuelle Netzwerk Interface der VM wird auf die IP Adresse 192.168.2.70 und eine Xen MAC Adresse in Zeile 31 eingestellt. Des Weiteren wird noch der Hostname auf 'asam' gesetzt und ein paar Verhaltensweisen definiert. Der Zusatz extra='xencons=tty1' am Ende der Datei ist nötig, um sich per Konsole auf die VM einloggen zu können.

Listing 3.1: Konfigurationsdatei für die Xen Instanz asam

```
1  #
2  # Configuration file for the Xen instance asam, created
3  # by xen-tools 3.9 on Thu Jun  4 20:25:24 2009.
4  #
5
6  #
7  #  Kernel + memory size
8  #
9  kernel      = '/boot/vmlinuz-2.6.18.8-xen'
10 #ramdisk     = '/boot/initrd.img-2.6.18.8-xen'
```

```
11 memory        = '128'
12
13 #
14 #  Disk device(s).
15 #
16 root          = '/dev/sda2 ro'
17 disk          = [
18                       'file:/mnt/storage3/xen/domains/asam/swap.img,sda1,w',
19                       'file:/mnt/storage3/xen/domains/asam/disk.img,sda2,w',
20                   ]
21
22
23 #
24 #  Hostname
25 #
26 name          = 'asam'
27
28 #
29 #  Networking
30 #
31 vif           = [ 'ip=192.168.2.70,mac=00:16:3E:14:96:FC' ]
32
33 #
34 #  Behaviour
35 #
36 on_poweroff = 'destroy'
37 on_reboot   = 'restart'
38 on_crash    = 'restart'
39
40 extra='xencons=tty1'
```

3.4 Verwendete Tools

Gemessen werden soll die Auswirkung einer Live Migration auf die CPU, den Speicher und das Netzwerk. Dazu werden unterschiedliche Tools verwendet, die während der Migration die jeweiligen Auslastungen protokollieren. Natürlich benötigen diese Tools ebenfalls selbst Ressourcen und verfälschen somit die Messergebnisse etwas. Da aber immer dieselben Tools bei allen Tests eingesetzt werden, ist ein Vergleich unterschiedlicher Tests repräsentativ. Als erstes und wichtigstes Tool kommt dstat [Wie08] zum Einsatz. Es kombiniert und erweitert die Funktionalität bekannter Statistik Tools wie vmstat, iostat, netstat, nfsstat und ifstat. Anhand dieses Tools können detaillierte Aussagen über die CPU Nutzung, die Speicherauslastung und die Netzwerk Geschwindigkeiten getroffen werden. Dabei wird bei der CPU Nutzung unterschieden, ob der Prozessor gerade vom User oder vom System genutzt wird, sich im Leerlauf befindet oder auf I/O Geräte wartet. Ausserdem erweist sich noch die Anzahl an Software Interrupts, die nach Start einer Migration auftreten, als durchaus interessant.

Dstat liefert ebenso differenzierte Werte für die Speichernutzung, unterteilt diese in be-

nutzten und freien Speicher, sowie Puffer und Cache Speicher. Zur Netzwerkauslastung liefert es einfach die erhaltenen und gesendeten Bits, wobei es hier aufgrund der internen Verarbeitung mit 32 Bit zu Zählerüberläufen kommen kann, die aber manuell bereinigt werden können [Wie]. Allerdings beziehen sich die Werte von dstat nur auf die Domäne, in der es ausgeführt wird. Um die Auswirkungen auf das System zu untersuchen und die VM nicht zusätzlich zu belasten, wird hier in der Domain 0 gemessen.

Um für die CPU Auslastung nun auch Werte im Bezug auf die Nutzung durch die virtuelle Maschine zu erhalten, wird zusätzlich noch das bei Xen enthaltene Tool xentop eingesetzt. Dadurch ist es ausserhalb der VM möglich, festzustellen, wie stark das Gastsystem den Prozessor beansprucht. Xentop misst die CPU Auslastung der Domain 0, sowie aller vorhandenen Domain U's separat. Zu beachten ist hierbei, dass xentop die Prozessorauslastung mehrerer Prozessoren einfach aufaddiert und dadurch ein vielfaches von 100% anzeigen kann [Hol07]. Um dies zu umgehen, werden die Werte manuell durch die Anzahl der aktiven CPUs geteilt. Misst xentop eine Domain 0 CPU Auslastung von 50% und dstat eine vom Benutzer verursachte CPU Auslastung von 100%, bedeutet das, dass der Benutzer der Domain 0 50% der verfügbaren Prozessorleistung auf dem System auslastet. Gesteuert werden die Tools durch Skripte, die von einem Hauptskript auf dem Projekt3 Rechner aus gestartet werden. Für die Durchführung der Tests kommen noch weitere Tools als Benchmarks zum Einsatz, deren Funktion später erläutert wird.

3.5 Eingesetzte Skripte

Um die Tests möglichst einfach auch wiederholt durchführen zu können, werden drei Skripte eingesetzt, die die jeweiligen Tools und Benchmarks starten, sowie die Migration anstoßen. Zusätzlich wird noch ein viertes Skript zur Messung der Downtime innerhalb der virtuellen Maschine eingesetzt. Prinzipiell sind die Skripte einfach gehalten und können durchaus falsch aufgerufen werden, da die Tests aber nur von einer Person durchgeführt werden, ist die korrekte Nutzung der Skripte jederzeit gewährleistet.

3.5.1 Das Shellskript 'main'

Ein Test wird gestartet, indem auf dem Projekt3 Rechner das Skript 'main' (Listing 3.2) mit entsprechenden Parametern aufgerufen wird. Erwartet werden drei Parameter, deren Reihenfolge relevant ist. Der erste Parameter gilt nur der Namensgebung der während des Testlaufs erzeugten Dateien. Dadurch erhalten alle Dateien eines Testdurchlaufs einen einheitlichen Namen und können so später einander zugeordnet werden. Der zweite Parameter spezifiziert die Art des Testlaufs bzw. den Namen des Benchmarks, der für Last auf dem Hostsystem sorgen soll. Mögliche Eingaben lauten:

- ohne - für einen Testdurchlauf ohne Last
- memtester - für einen Testdurchlauf mit Speicherlast
- netperf - für einen Testdurchlauf mit Netzwerklast
- cpu - für einen Testdurchlauf mit CPU Last
- iometer - für einen Testdurchlauf mit I/O Last

Als dritten und letzten Parameter schließlich wird die Dauer des Tests in Sekunden definiert. Dieser Wert beeinflusst, wie lange die oben genannten Tools mitprotokollieren. Ein Test dauert dabei immer etwas länger als die hier definierte Zeit. Je nach Grad der Migrationsbehinderung durch einen Benchmark muss ein Wert gewählt werden, der groß genug ist, um den vollständigen Migrationsvorgang von kurz vor Begin bis zum vollständigen Abschluss der Migration protokolliert zu bekommen.

Nach kurzer Überprüfung, ob dem Skript korrekte Parameter übergeben wurden, wird getestet, ob auf dem Rechner Kirsche die virtuelle Maschine mit dem Namen asam gestartet ist und sich im Zustand 'blocked' befindet (Zeile 12). Ist dies nicht der Fall, wird die VM asam gestartet und 30 Sekunden gewartet, bis das System hochgefahren ist. Dies wird erneut überprüft. Sollte die VM nach den 30 Sekunden nicht den Zustand 'blocked' melden, wird der Benutzer darauf hingewiesen und das Skript beendet (Zeile 31). Getestet wird dies jeweils durch einen ssh-remote Aufruf von xentop mit den Optionen '`-d1`', für sekundenweise Updates, '`-b`' für die Ausgabe im batch Modus und '`-i1`' für das automatische Ende nach der ersten Ausgabe. Die damit erhaltene Ausgabe wird nun zeilenweise per awk nach dem Schlüsselwort 'asam' durchsucht. Gibt es eine solche Zeile, wird die zweite Zeichenkette, separiert durch Leerzeichen, ausgegeben und später mit "`--b---`"auf Gleichheit verglichen.

Anschließend wird in den Zeilen 38 bis 67 dem Wert des zweiten Parameters entsprechend ein oder kein Benchmark gestartet. Die Benchmarks werden im jeweiligen Abschnitt im Kapitel 4 genauer erklärt.

In den Zeilen 71 und 74 werden dann die im Folgenden beschriebenen Skripte auf den Rechnern Apfel und Kirsche gestartet und zwei Sekunden gewartet, bis die Skripte ihre Tools starten konnten. Im Folgenden wird der Dienst gestartet, der während der Migration einen Service der virtuellen Maschine asam benutzt. Je nach Dienst, der getestet werden soll, muss das Skript an dieser Stelle angepasst werden. In diesem Beispiel wird in Zeile 80 eine ping Abfrage gestartet und deren Ausgabe durch gawk modifiziert in eine Datei geschrieben. Der direkt darunter auskommentierte Teil startet VLC sowohl in der VM als auch lokal auf dem Projekt3 Rechner und wird für den Streaming Test in Kapitel 4.2 verwendet, für den im Gegenzug der ping Befehl von oben auskommentiert wird.

Zum Schluss wird die Protokoll Datei der ping Ausgabe im Falle eines durchgeführten Pingtestes auf den externen Rechner Pcheger12 kopiert, um dort alle Protokolle zu sammeln (Zeile 98). Falls der Benchmark Netperf verwendet wurde, wird auch dessen Protokoll kopiert (Zeile 104).

Um alle SSH Verbindungen innerhalb der Skripte automatisiert ablaufen lassen zu können, ist es erforderlich, Shared Keys einzusetzen. Dabei wird ein Authentifizierungsschlüsselpaar (private und public Key) auf dem Zielrechner erzeugt und gespeichert. Der public key wird dann auf den zugreifenden Rechner übertragen und gespeichert [bem04]. Somit wird eine Kennworteingabe unnötig und das Skript kann ohne Unterbrechungen durchlaufen.

Listing 3.2: Skript 'main' auf Projekt3

```
1 # Script fuer Fopra
2 # von Bastian Asam
3 # 26.01.2010
4
5 if test $# -eq 0
6 then echo "Bitte geben sie als Agument den Namen des Testlaufs, den
      Benchmark und die Dauer des Tests ein!"
7 echo "z.B. asam.script memtester1 memtester 100"
```

```
8  echo "Moegliche Benchmarks sind ohne, netperf, memtester, cpu und
       iometer."
9  exit 0
10 fi
11
12 var=`ssh root@192.168.2.24 xentop -d1 -b -i1 | awk -F " " '/asam/ {print
       $2}'`
13 if test ! $var
14         then var="false"
15 fi
16 if test $var = "--b---"
17         then echo "VM asam bereits vorhanden"
18         else echo "VM asam wird gestartet"
19         ssh root@192.168.2.24 "xm create /mnt/storage3/xen/asam.cfg"
20         echo "VM asam auf Kirsche gestartet"
21         echo "Warte bis VM asam hochgefahren ist"
22         i=30
23         while [ $i -gt 0 ]
24         do
25                 echo -n "."
26                 ((i--))
27                 sleep 1
28         done
29 fi
30
31 var=`ssh root@192.168.2.24 xentop -d1 -b -i1 | awk -F " " '/asam/ {print
       $2}'`
32 if test $var = "--b---"
33         then echo "VM asam ist bereit"
34         else echo "VM asam noch nicht bereit - Skript wird beendet!"
35        exit 0
36 fi
37
38 echo "Testlauf $1 wird "
39 if test $2 = "ohne"
40 then echo "ohne Benchmark "
41 fi
42 if test $2 = "netperf"
43 then echo "mit netperf $3"
44 ssh root@192.168.2.25 "netperf -H 192.168.0.24 -l$3 -t UDP_STREAM >/mnt/
       storage3/xen/extras/test1/$1-netperf.test" &
45 fi
46 if test $2 = "memtester"
47 then echo "mit memtester "
48 ####
49 ssh root@192.168.2.24 "memtester 110 >/dev/null &"
50 ####
51 # bei Streaming Test oben auskommentieren und unten aktivieren!
52 ####
53 #ssh root@192.168.2.24 "memtester 100 >/dev/null &"
54 ####
55
56 sleep 10
```

```
57 fi
58 if test $2 = "iometer"
59 then echo "mit iometer, dynamo starten... "
60 ssh root@192.168.2.70 ~/dynamo -i 192.168.2.71 -m 192.168.2.70 >~/dynamo
       .log &
61 read -p "Iometer gestartet?"
62 fi
63 if test $2 = "cpu"
64 then echo "mit cpuburn-in "
65 ssh root@192.168.2.24 "/fopra/cpuburn-in 2 >/dev/null &"
66 fi
67 echo "gestartet..."
68
69
70 echo "Starten des Apfel Skripts..."
71 ssh root@192.168.2.25 "/mnt/storage3/xen/extras/apfel.script $1 $3" &
72
73 echo "Starten des Kirsche Scripts..."
74 ssh root@192.168.2.24 "/mnt/storage3/xen/extras/kirsche.script $1 $3" &
75
76 sleep 2
77 ping -c1 192.168.2.70
78 ####
79 echo "pinging..."
80 ping -c500 -i0.2 192.168.2.70 | gawk -F " |=" '/PING/ {print "#" $0}
       /---/ {print "#" $0} /rtt/ {print "#" $0} /transmitted/ {print "#" $0
       } /64 bytes/ {print systime() "." i++%100 " " $6 " " $10}' > /fopra/
       test1/$1-ping.test
81 ####
82 # bei Streaming Test oben auskommentieren und unten aktivieren!
83 ####
84 #echo "Starte VLC local mit timestamp..."
85 #/fopra/test2/timestamp $3 &
86 #vlc --quiet --verbose 1 --extraintf logger udp://@ &
87 #
88 #echo "Starte VLC Server auf VM asam"
89 #ssh root@192.168.2.70 "vlc ~/vlc/Planet_51.ts vlc://quit --sout '#
       standard{access=udp,mux=ts,url=192.168.2.255,sap}' --intf dummy"
90
91 #sleep 5
92 ####
93
94 echo " fertig! "
95
96 echo "Kopiere Ergebniss auf pcheger12"
97 ####
98 scp /fopra/test1/$1-ping.test asamb@pcheger12.mm.ifi.lmu.de:/users/gast/
       asamb/fopra/
99 ####
100 # bei Streaming Test oben auskommentieren!
101 ####
102
103 if test $2 = "netperf"
```

```
104 then scp /fopra/test1/$1-netperf.test asamb@pcheger12.nm.ifi.lmu.de:/
        users/gast/asamb/fopra/
105 fi
106
107 sleep 20
108
109 echo "Test beendet"
110
111 # Ende
```

3.5.2 Das Shellskript 'kirsche'

Das kirsche Skript benötigt nur noch zwei Parameter, den Namen und die Länge des Testlaufs. Auch hier wird kurz überprüft, ob Parameter übergeben wurden. Dies dient im Allgemeinen vor allem dazu, die korrekte Benutzung des Skripts angezeigt zu bekommen. In den Zeilen 15 und 17 werden die beiden oben erwähnten Tools zur Protokollierung der Ressourcennutzung gestartet. Dstat bietet bereits eine Option, die Ausgabe in eine Datei umzuleiten, während xentop erst durch eine relativ aufwendige Nachbearbeitung der Ausgabe durch gawk eine brauchbare Protokolldatei erzeugt. Die gesetzten Parameter für den Aufruf von dstat erklären sich folgendermaßen: '`-T`' setzt jeweils einen epoch Zeitstempel, '`-c`' aktiviert die CPU Statistiken, '`-m`' die Speicher Statistiken, '`-n`' die Statistiken für das Netzwerk und '`--output`' schreibt die Ausgabe in die anschließend definierte Datei. Die zwei anschließenden Argumente definieren die Abstände der Updates auf eine Sekunde und die Anzahl der Updates auf den im Skriptaufruf definierten Wert für die Dauer des Tests. Da dstat trotz der '`--output`' Option seine Ausgabe weiterhin auch auf stdout schickt, wird diese Ausgabe noch nach '/dev/null' umgeleitet. Für xentop sorgen die Optionen '`-d1`' und '`-i`' für die gleiche Anzahl an sekundengenauen Updates, wie die beiden oben erwähnten Argumente von dstat. Zusätzlich sorgt hier noch die Option '`-b`' für eine bash Ausgabe, die in die am Ende definierte Datei geschrieben wird. Da xentop keine Zeitstempel anbietet, muss dies, neben weiteren Umstrukturierungen zur besseren Importierbarkeit in OpenOffice Calc mit gawk modifiziert werden.

Die anschließenden fünf Sekunden Wartezeit dienen dazu, in den Protokolldateien ein paar Einträge im Normalzustand vor dem Beginn der Migration zu erhalten. Die Migration wird danach in Zeile 23 gestartet, wobei die Option '`-l`' dafür sorgt, dass eine Live Migration durchgeführt wird, während das time Tool die Ausführungsdauer des Migrationsprozesses misst. Zusammen mit der Ausgabe von time wird zuvor noch die Epoch Zeit in eine Datei geschrieben, um später in den Protokollen sehen zu können, wann die Migration angestoßen wurde, wie lange der Prozess dauerte und welche Ressourcen er beanspruchte. In einer Schleife wird nun für die eingestellte Dauer des Tests sekundenweise ein '.' ausgegeben, um den Fortschritt zu visualisieren. Nachdem die Zeit abgelaufen ist, werden alle während des Testlaufs erzeugten und nun fertiggestellten Protokolle sowohl von Kirsche als auch von Apfel auf den Pcheger12 und Projekt3 übertragen (Zeilen 37 und 39). Dies ist von Kirsche aus möglich, da die Protokolle auf das gemeinsame NAS Laufwerk gespeichert werden. Auch hier ist ein vorausgehender Keyaustausch nötig.

Listing 3.3: Skript 'kirsche' auf Kirsche

```
1  # Script fuer Fopra
2  # von Bastian Asam
3  # 14.12.2009
4
5  if test $# -eq 0
6  then echo "Bitte geben sie als Agument den Namen des Testlaufs, gefolgt
       von der Laenge ein"
7  exit 0
8  fi
9
10
11 echo "Kirsche Skript gestartet!"
12
13 echo "Starte dstat und xentop auf Kirsche"
14
15 dstat -T -c -m -n --output /mnt/storage3/xen/extras/test1/$1-dstat-
       kirsche.test 1 $2 >/dev/null &
16
17 xentop -d1 -b -i$2 | gawk -F " " 'BEGIN {test = "false"; time = systime
       ()} /asam/ {test = "true"; printf time++ " asam: " $4 " " $5 " " $6}
       /Domain/ {if(test == "false") printf time++ " asam: - - -"; print " |
        Dom0: " $4 " " $5 " " $6; test ="false"}' >/mnt/storage3/xen/extras/
       test1/$1-xentop-kirsche.test &
18
19 sleep 5
20
21 echo "Starte Migration von Kirsche nach Apfel..."
22 date +%s > /mnt/storage3/xen/extras/test1/$1-migration-time.test
23 /usr/bin/time -a -o /mnt/storage3/xen/extras/test1/$1-migration-time.
       test -f "%E real, %U user, %S sys, %D KB unshared Data, %K Average
       Total memory use, %P CPU Percentage " xm migrate asam 192.168.0.25 -l
24
25 echo "Warte bis Migration beendet ist"
26 i=$2
27 while [ $i -gt 0 ]
28 do
29         echo -n "."
30         ((i--))
31         sleep 1
32 done
33 echo " fertig! "
34
35 echo "Kopiere Ergebnisse auf pcheger12 und projekt3"
36
37 scp /mnt/storage3/xen/extras/test1/$1-dstat-kirsche.test /mnt/storage3/
       xen/extras/test1/$1-dstat-apfel.test /mnt/storage3/xen/extras/test1/
       $1-xentop-kirsche.test /mnt/storage3/xen/extras/test1/$1-xentop-apfel
       .test /mnt/storage3/xen/extras/test1/$1-migration-time.test
       asamb@pcheger12.nm.ifi.lmu.de:/users/gast/asamb/fopra/
38
```

```
39 scp /mnt/storage3/xen/extras/test1/$1-dstat-kirsche.test /mnt/storage3/
       xen/extras/test1/$1-dstat-apfel.test /mnt/storage3/xen/extras/test1/
       $1-xentop-kirsche.test /mnt/storage3/xen/extras/test1/$1-xentop-apfel
       .test /mnt/storage3/xen/extras/test1/$1-migration-time.test root@192
       .168.2.1:/fopra/test1/
40
41
42
43 echo "Ende Kirsche Skript"
44
45
46 #ende
```

3.5.3 Das Shellskript 'apfel'

Das Skript für den Apfel Rechner ist das einfachste, da die VM dorthin migriert wird, also kein Startbefehl benötigt wird und zum Schluss die erzeugten Protokolle bereits durch das kirsche Skript übertragen werden. Nach der gleichen Überprüfung auf übergebene Argumente wird in Zeile 13 und folgende überprüft, ob eine virtuelle Maschine asam vorhanden ist. Ist dies der Fall, wird die virtuelle Maschine und das Skript beendet, da so die Migration scheitern würde. Ist aber keine VM mit dem Namen asam vorhanden, kann der Test weiterlaufen und es werden die Tools genauso wie im kirsche Skript gestartet (Zeilen 28 und 30).

Listing 3.4: Skript 'apfel' auf Apfel

```
1  # Script fuer Fopra
2  # von Bastian Asam
3  # 16.12.2009
4
5  if test $# -eq 0
6  then echo "Bitte geben sie als Argument den Namen des Testlaufs, gefolgt
       von der Laenge ein"
7  exit 0
8  fi
9
10
11 echo "Apfel Skript gestartet!"
12
13 var=`xentop -d1 -b -i1 | awk -F " " '/asam/ {print $2}'`
14 if test ! $var
15         then var="false"
16 fi
17 if test $var = "--b---"
18         then echo "VM asam vorhanden"
19         echo "VM asam wird beendet"
20         xm shutdown asam
21         echo "VM asam auf Apfel wird beendet"
22         echo "Warte bis VM asam beendet ist und starte Skript neu!"
23         exit 0
24 fi
25
```

```
26 echo "Starte dstat und xentop auf Apfel"
27
28 dstat -T -c -m -n --output /mnt/storage3/xen/extras/test1/$1-dstat-apfel
       .test 1 $2 >/dev/null &
29
30 xentop -d1 -b -i$2 | gawk -F " " 'BEGIN {test = "false"; time = systime
       ()} /asam/ {test = "true"; printf time++ " asam: " $4 " " $5 " " $6}
       /Domain/ {if(test == "false") printf time++ " asam: - - -"; print " |
        Dom0: " $4 " " $5 " " $6; test ="false"}' >/mnt/storage3/xen/extras/
       test1/$1-xentop-apfel.test
31
32
33 echo "Ende Apfel Skript"
34
35
36 #ende
```

3.5.4 Das Shellskript 'zeit' in der VM

Da die Migrationszeiten in den ersten Tests bis zu 100fach über den erwarteten Zeiten aus dem Paper 'Live Migration of Virtual Machines' [CFH+05] lagen, wurden unterschiedliche Versuche unternommen, die Zeiten zu senken. Fehler wurden keine protokolliert, so dass eine fehlerhafte Xen Installation ausgeschlossen werden konnte. Auch ein Austausch des Switches im Steuerungsnetz durch ein Hub brachte keine Veränderung. Um zu testen, ob die Downtime innerhalb der VM in etwa der durch den Ping ermittelten Downtime entspricht, protokolliert ein sehr einfaches Skript alle 0,01 Sekunden einen Zeitstempel (Listing 3.5). Dieses kleine Skript allein veranlasst die VM nun mit weniger als einer Sekunde Downtime zu migrieren, statt der vorher eher 8 Sekunden. Dabei stimmen die beiden Downtimes, ermittelt einerseits durch das zeit Skript und andererseits durch das Pingen, ziemlich genau überein. Da der Abstand von 0,01 Sekunde für das System zu gering ist, schafft es das Skript nur, etwa jeden zweiten Eintrag zu erzeugen. Damit ist sichergestellt, dass so genau wie nur möglich gemessen wird. Merkwürdigerweise darf das Skript nicht modifiziert werden. Schon eine Änderung des sleep Befehls auf beispielsweise nur 0,1 Sekunden, lässt die VM wieder langsam migrieren. Eine Schleife zum selbständigen Beenden des Skriptes ließ die VM nach der Migration gar nicht mehr starten. Dies deutet auf einen Fehler in der hier benutzten Xen Version hin.

Listing 3.5: Skript 'zeit' in der VM

```
1 date > zeit.log
2 while [ 1 ]
3 do
4 date +%H:%M:%S,%N >> zeit.log
5 sleep 0.01
6 done
```

Deshalb werden im Folgenden die Tests mit einer offenen ssh Verbindung durchgeführt, worüber das zeit Skript manuell aufgerufen und beendet wird. Die Protokolldaten daraus bieten eine weitere Datenquelle zur Downtime Bestimmung, wenn auch beachtet werden muss, dass die Zeitmessung innerhalb von virtuellen Maschinen mit Vorsicht zu genießen ist und keinesfalls exakte Werte liefert [Sti07]. Die Genauigkeit dürfte für diese Zwecke allerdings ausreichen.

4 Testdurchführung

Ziel dieser Arbeit ist vor allem die Auswirkungen einer Live Migration auf den Dienstnutzer zu evaluieren. Kritische Anwendungen sind hier vor allem Echtzeitanwendungen, deren Funktionalität im wesentlichen von dauerhaftem, ununterbrochenem Informationsaustausch abhängig ist. Klassisches Beispiel sind Streaming Dienste, bei denen dem Nutzer auch nur eine kurze Unterbrechung im Datenstrom sofort negativ auffällt. Die Downtime ist dabei die alles entscheidende Größe. Dem Paper 'Live Migration of Virtual Machines' [CFH+05] folgend, sind selbst bei hoher Last Downtimes von unter 250ms möglich. Im Speziellen wurde eine virtuelle Maschine mit einem darauf laufendem Webserver migriert, welcher durch den SPECweb99 Benchmark zu 90% ausgelastet wurde. Es gelang den Autoren, diese Migration mit einer totalen Downtime von nur 210ms durchzuführen. Ebenfalls wurde ein Quake 3 Server mit nur 60ms Downtime migriert, so dass die Migration angeblich von den Spielern nicht bemerkt wird. Im folgenden wird versucht zu klären, ob diese Ergebnisse nachvollziehbar sind.

Die Virtualisierung ermöglicht in der Praxis vor allem eine effizientere Ausnutzung der Server und die Live Migration bietet dabei die Möglichkeit zum Load Balancing. Somit darf man davon ausgehen, dass Migrationen selten von Systemen mit geringer Last durchgeführt werden. Der weitaus realistischere Fall ist die Migration von einem stark belastetem System auf ein eher unbelastetes System. Dennoch werden hier alle Tests auf unbelasteten Systemen durchgeführt, um gute und unbeeinflusste Messergebnisse zu erhalten. Der Rechner Kirsche ist dabei immer der Ausgangspunkt. Je nach Test wird die zu migrierende VM mit Hilfe von Benchmarks unterschiedlichen Lasten ausgesetzt, um so die Auswirkung auf die Migration und deren Downtime zu messen. Migriert wird immer auf den Rechner Apfel. Zum Vergleich werden alle Tests auch einmal komplett ohne Last durchgeführt. Um möglichst aussagekräftige Ergebnisse zu erzielen, werden alle Tests fünf mal durchgeführt und deren Mittelwerte gebildet. Dabei sollte keine zu starke Streuung bei den Ergebnissen auftreten.

4.1 Ping Test

Der erste Test soll die Einflüsse einer Live Migration auf eine VM evaluieren, die nur Antworten auf Ping Anfragen generiert. Dabei benötigt die VM so gut wie keine Ressourcen für sich selbst, somit wird kaum Speicher verändert, der die pre-copy Phase verlängert. Folglich sollten hier sehr geringe Downtimes erreicht werden.

Der Aufruf von ping erfolgt generell mit den Optionen '`-c500`' für 500 zu sendende Ping Requests, '`-i0,2`' für ein Request alle 0,2 Sekunden und natürlich der Zieladresse der virtuellen Maschine asam (Listing 4.1). Um auch hier Zeitstempel und eine brauchbare Anordnung in der erzeugten Datei zu erhalten, wird die Ausgabe wiederum mittels gawk modifiziert.

Listing 4.1: Ping Aufruf

```
ping -c500 -i0.2 192.168.2.70 | gawk -F " |=" '/PING/ {print "#" $0}
    /---/ {print "#" $0} /rtt/ {print "#" $0} /transmitted/ {print "#" $0
    } /64 bytes/ {print systime() "." i++%100 " " $6 " " $10}' > /fopra/
    test1/$1-ping.test
```

4.1.1 Ping ohne Last

Der Ping Test wird wie alle Tests einmal ohne Last durchgeführt. Die hier gemessene Downtime markiert die untere Grenze. Alle weiteren Tests lassen eine höhere Downtime erwarten.

Im Mittel dauerte hier der Prozess, der die Live Migration auslöst 8,87 Sekunden. Nach dieser Zeit ist die Migration auf dem Host System abgeschlossen, d.h. Phase 0 bis Phase 4 des Migrations Ablaufs, wie anfangs erklärt, sind abgeschlossen und die VM muss nur noch auf dem Zielsystem aktiviert werden. Interessant bei diesen Ping Tests ist vor allem die Anzahl an nicht beantworteten Requests und der damit verbundenen Zeit, in der der Ping Dienst nicht verfügbar ist. Die Zeiten lassen sich mit der durch das zeit Skript innerhalb der VM ermittelten Downtime vergleichen. Eine Aufstellung darüber gibt Tabelle 4.1.

	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**
Anzahl nicht beantworteter Requests	4	4	4	5	4
Ping Ausfallzeit (in Sekunden)	0,8	0,8	0,8	1,00	0,8
zeit Skript Ausfallzeit (in Sekunden)	0,78	0,86	0,93	0,95	0,96

Abbildung 4.1: Ausfallzeiten ohne Last

Hierbei ist anzumerken, dass die Ping Ausfallzeit einfach aus der Anzahl nicht beantworteter Requests berechnet wird. Da die Ping Requests nur alle 0,2 Sekunden gesendet werden, ist die Zeitmessung auch nur auf 0,2 Sekunden genau möglich. Im Mittel liegt hier also die Ausfallzeit des Pingdienstes bei 0,84 Sekunden und dürfte daher nicht unbemerkt bleiben. Wohlgemerkt verlief dieser Test ohne jegliche Last, was wohl eher selten in der Realität vorzufinden ist. Wird mit Standardeinstellungen gepingt, also nur jede Sekunde ein Request, kann die Migration tatsächlich unbemerkt bleiben.

Im folgenden wird die Belastung auf die Ressourcen Prozessor, Speicher und Netzwerk auf den beiden Xen Systemen genauer untersucht.

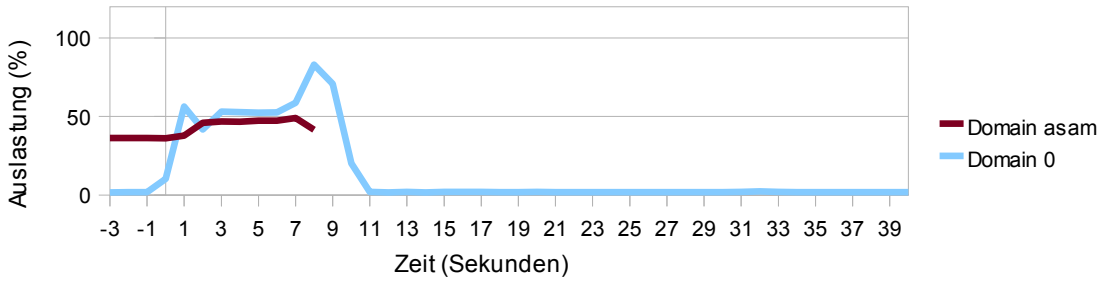

Abbildung 4.2: Pingtest ohne Last: Prozessor Auslastung auf Kirsche (xentop)

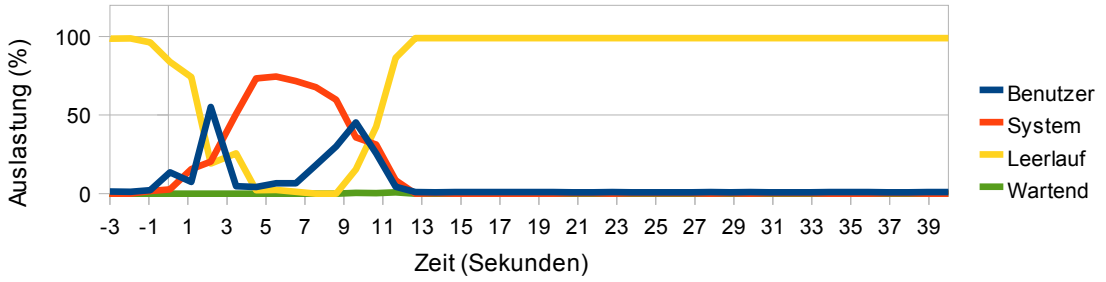

Abbildung 4.3: Pingtest ohne Last: Prozessor Auslastung auf Kirsche (dstat)

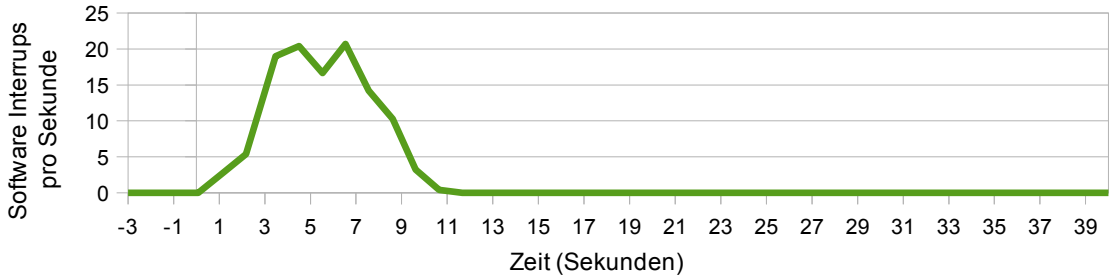

Abbildung 4.4: Pingtest ohne Last: Software Interrupts auf Kirsche

Abbildung 4.2 zeigt die globale Prozessor Auslastung auf dem Host Migrationsrechner Kirsche. Hier ist zu erkennen, dass die Domain asam bis 8 Sekunden nach Start der Live Migration zum Zeitpunkt 0 vorhanden ist und aufgrund des zeit Skriptes auch bis zu 50% Prozessorlast erzeugt. Ansonsten wird die Migration ausschließlich von Domain 0 durchgeführt und beansprucht zusammen mit der VM über etwa 9 Sekunden fast die gesamte CPU Leistung. In dieser Zeit finden entsprechend auch einige Software Interrupts statt (Abbildung 4.4). Dabei wird der Prozessor innerhalb der Domain 0 überwiegend vom System selbst beansprucht, nur am Anfang und am Ende der Migration durch den Benutzer selbst (Abbildung 4.3). Erst nach ca. 12 Sekunden nach dem Start der Migration befindet sich das System im Leerlauf, da nun keine VM mehr vorhanden ist.

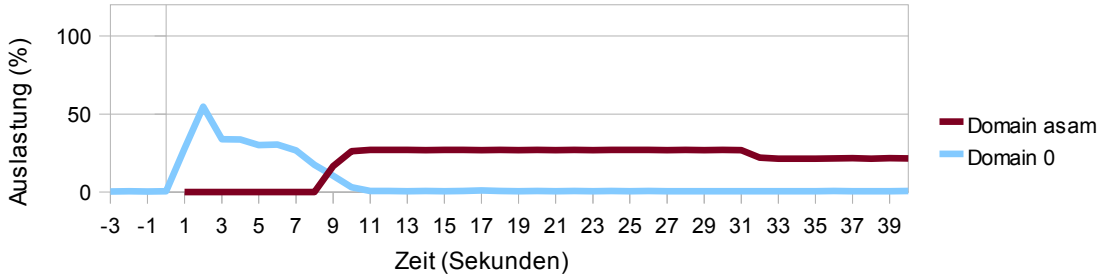

Abbildung 4.5: Pingtest ohne Last: Prozessor Auslastung auf Apfel (xentop)

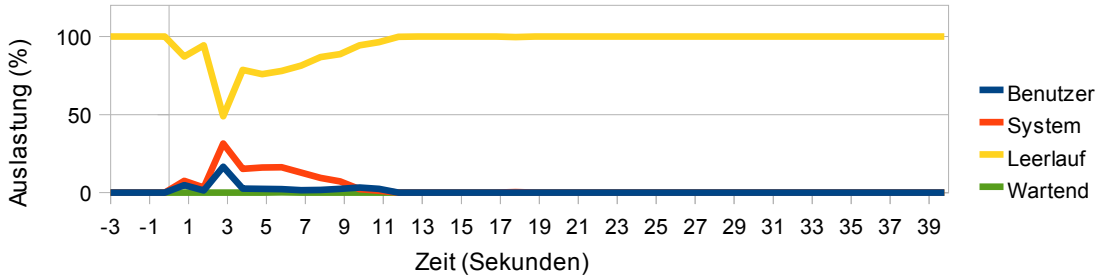

Abbildung 4.6: Pingtest ohne Last: Prozessor Auslastung auf Apfel (dstat)

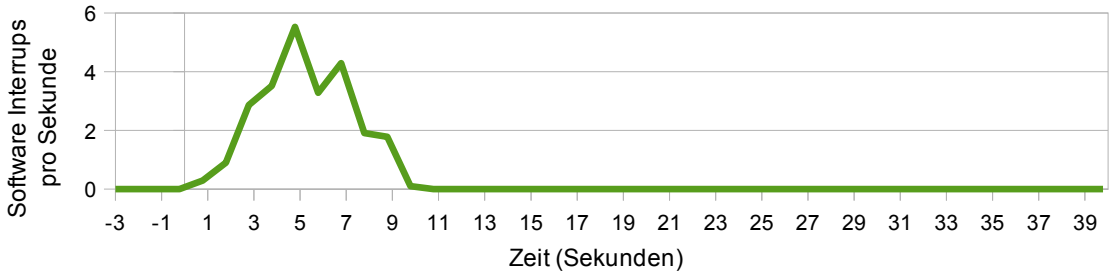

Abbildung 4.7: Pingtest ohne Last: Software Interrupts auf Apfel

Auf den ersten Blick verhält sich die Prozessor Auslastung auf dem Ziel Migrationsrechner Apfel sehr unterschiedlich, wie Abbildung 4.5 zeigt. Trotzdem korrelieren die Werte recht gut. Zu allererst zu beachten ist, dass im Apfel Rechner ein Dual Core Prozessor zum Einsatz kommt. Da die Auslastung so gut wie nicht über 50% steigt, ist der Migrationsprozess anscheinend nicht multi-thread fähig und Xen kann kaum Vorteile aus den zwei Prozessorkernen ziehen. Der zweite Kern steht dadurch auch während einer Migration anderen Prozessen zur Verfügung.

Hier sieht man auch deutlich, dass die Migration ausschließlich von Domain 0 bearbeitet wird. Dabei belegt wiederum das System selbst die CPU stärker als der Nutzer (Abbildung 4.6), die sowohl im Apfel als auch im Kirsche Rechner zu keiner Zeit auf I/O Geräte warten muss. Bereits eine Sekunde nach Start der Migration existiert die Domain asam auf dem Zielsystem. Die anschließende Pre-Copy Phase dauert etwa sieben Sekunden, in dieser Zeit

existiert die VM auf beiden Systemen parallel, bevor die VM auf Apfel gestartet wird und das zeit Skript Prozessorlast in der Domain asam erzeugt. Zwei Sekunden eher als auf dem Kirsche Rechner ist hier der Migrationsprozess schon nach etwa 10 Sekunden beendet. Die Software Interrupts befinden sich dabei aber auf deutlich niedrigerem Niveau als auf dem Kirsche Rechner (Abbildung 4.7).

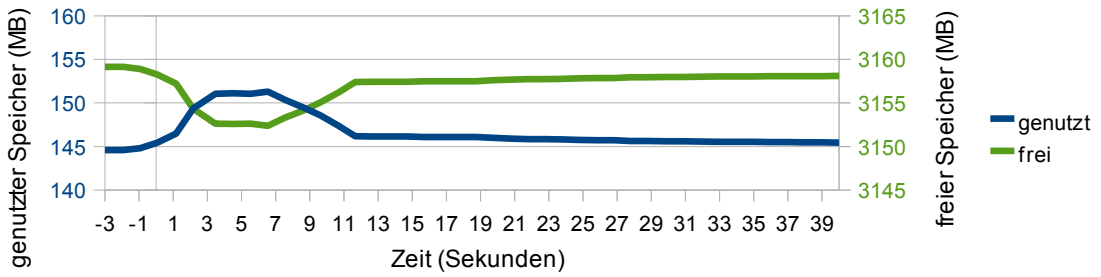

Abbildung 4.8: Pingtest ohne Last: Speichernutzung auf Kirsche

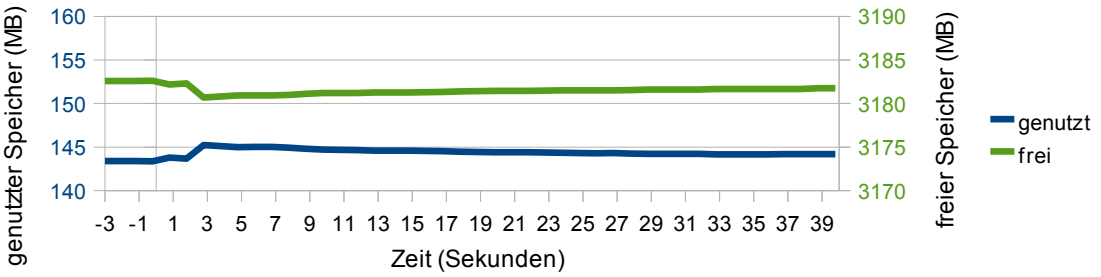

Abbildung 4.9: Pingtest ohne Last: Speichernutzung auf Apfel

Bei der Speichernutzung zeigt sich dagegen ein überraschender Unterschied zwischen Host und Ziel Migrationssystem. Da die VM bis auf das zeit Skript keine Prozesse ausführt, belegt sie praktisch keinen Speicher. Kirsche benötigt bis zu 7 MB zusätzlichen Speicher für die Durchführung der Migration (Abbildung 4.8), während Apfel mit weniger als 2 MB auskommt (Abbildung 4.9). Der Speichergewinn auf Kirsche bzw. der Speicherverlust auf Apfel nach der Migration ist in dieser Größenordnung bei 3,5 GByte RAM allerdings zu vernachlässigen.

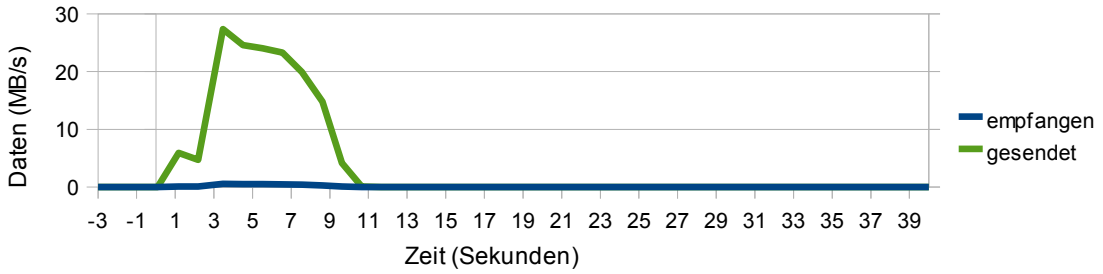

Abbildung 4.10: Pingtest ohne Last: Netzwerkauslastung auf Kirsche

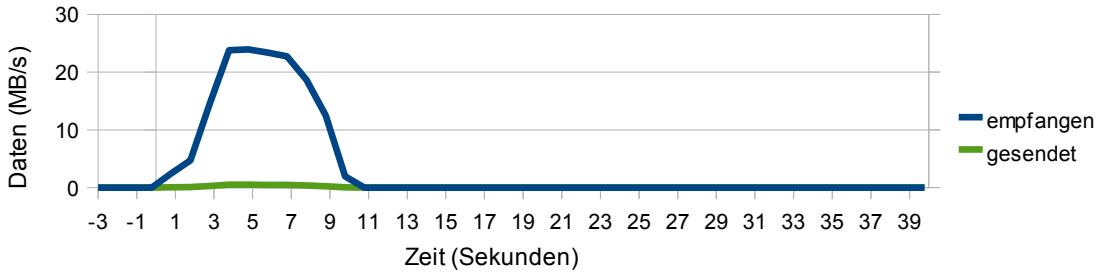

Abbildung 4.11: Pingtest ohne Last: Netzwerkauslastung auf Apfel

Die Netzwerkauslastung ist weitgehend invers äquivalent, da kein zusätzlicher Netzverkehr vorhanden ist. Was von Kirsche gesendet wird, wird von Apfel empfangen und umgekehrt (Abbildungen 4.10 und 4.11). Mit bis zu 25 MB/s ist die Datenmenge aber auch hier schon durchaus beachtlich.

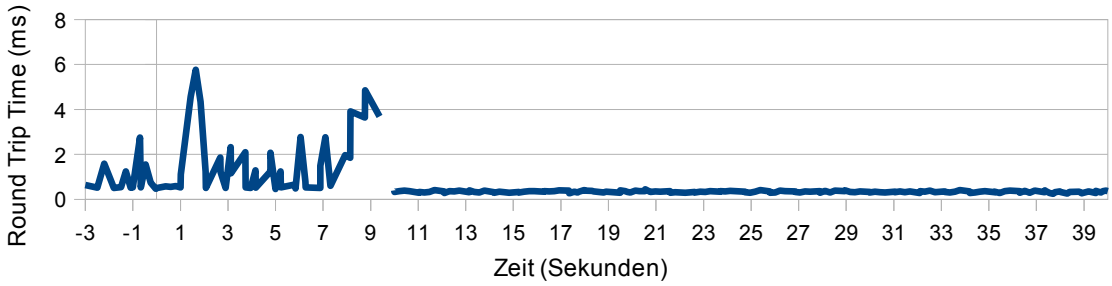

Abbildung 4.12: Pingtest ohne Last: Ping Round Trip Time

Die Messung der Round Trip Time wird in Abbildung 4.12 wiedergegeben. Nach anfänglichen Ausschlägen von bis zu 6 Millisekunden, vor allem während der Pre-Copy Phase, kommen wie oben bereits beschrieben für weniger als einer Sekunde gar keine Ping Antworten mehr an. Anschließend bewegt sich die Round Trip Time sofort wieder auf normal niedrigem Niveau, was darauf schliessen lässt, dass zu dem Zeitpunkt, zu dem die Ping Anfragen wieder ihr Ziel erreichen und beantwortet werden, die Migration bereits vollständig beendet ist und keine Einschränkungen mehr auftreten. Die Ausschläge noch vor Beginn der Migration sind durch die Starts der Skripte und den damit einhergehenden Aufrufen der Protokollierungstools zu erklären.

4.1.2 Ping mit Speicherlast

Um die VM asam unter Speicherlast zu setzen, wird hier das Tool Memtester [Caz09] eingesetzt. Memtester prüft den Speicher durch mehrere Tests auf Fehler. Aufgerufen wird es durch das main Skript über ssh in der VM mit in Listing 4.2 gezeigtem Befehl.

Listing 4.2: Memtester Aufruf während Pingtest

```
memtester 110 1 >~/memtester.log &
```

Durch diesen Aufruf versucht Memtester 110 MB Speicher mittels mlock für sich zu reservieren. Ein höherer Wert ist nicht möglich, da nur 128 MB Speicher der VM zur Verfügung stehen. Normalerweise ermittelt memtester bei zu hohen Werten automatisch den maximal möglichen Wert. Interessanterweise funktioniert dies nicht innerhalb der VM. Hier meint Memtester sogar 200 MB zu bekommen und scheitert dann am mlock. In Tests erwies sich 110 MB als ein sicherer, stets reservierbarer und möglichst hoher Wert. Damit steht dieser Speicher keinen anderen Prozessen mehr zur Verfügung und wird zusätzlich durch verschiedene Schreib- und Lesevorgänge immer wieder verändert.

	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**
Anzahl nicht beantworteter Requests	82	83	43	45	40
Ping Ausfallzeit (in Sekunden)	16,4	16,6	8,6	9,0	8,0
zeit Skript Ausfallzeit (in Sekunden)	17,8	16,34	8,58	9,11	8,54

Abbildung 4.13: Ausfallzeiten mit Memtester

Dies entspricht während der Pre-Copy Phase dem Worst Case Szenario, da hier nach jeder Iteration sehr viele Seiten „schmutzig“ (dirty) geworden sind und nochmals übertragen werden müssen. Entsprechend dauert alles deutlich länger. So liegt hier die durchschnittliche Dauer des Migrationsprozesses bei 26,49 Sekunden mit einer Standardabweichung von nur 0,5 Sekunden. Wie Tabelle 4.13 zeigt, variiert die Downzeit jedoch teilweise ziemlich stark, was wohl an dem Algorithmus liegt, welcher entscheidet, wann das Pre-Copying nutzlos wird und die VM stoppt, um die restlichen Seiten aus dem Speicher zu übertragen, ohne dass diese wieder verändert werden können. Trotzdem ist eindeutig eine erhebliche Steigerung der Downtime festzustellen, die durchaus zu Problemen im Betrieb führen kann.

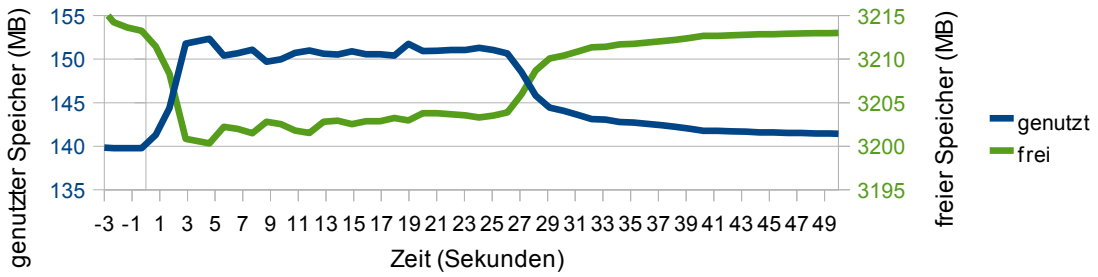

Abbildung 4.14: Pingtest mit Memtester: Speichernutzung auf Kirsche

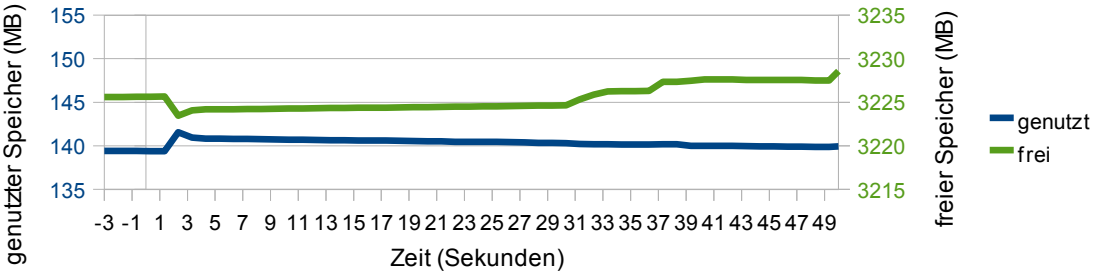

Abbildung 4.15: Pingtest mit Memtester: Speichernutzung auf Apfel

Da der Speicher für die VM sowieso reserviert wird, ist der verfügbare Speicher für die Domain 0 unabhängig von dem tatsächlich belegtem Speicher innerhalb der VM. Wie in Abbildung 4.14 zu sehen, wird anfangs auf Kirsche sogar fast 5 MB weniger Speicher belegt und fast 55 MB mehr bleiben frei als im Test ohne Last. Allerdings benötigt die Domain 0 dann bis zu 12,5 MB Speicher für die Durchführung der Migration und das natürlich über einen wesentlich längeren Zeitraum von über 28 Sekunden. Auf Apfel bietet sich analog zum Test ohne Last ein anderes Bild. Hier werden lediglich knapp über 2 MB zusätzlich belegt, die sich auch schon während der Pre-Copy Phase kontinuierlich reduzieren (Abbildung 4.15). Allerdings bleibt auch hier die Speicherbelegung insgesamt niedriger als im Test ohne Last.

Leider erzeugt ein solcher Speichertest, wie ihn Memtester durchführt, eine erhebliche Prozessorlast, weshalb sich ein autarker Speichertest nicht realisieren lässt. In Abbildung 4.16 und 4.17 ist deshalb auch deutlich zu erkennen, wie der Prozessor auf Kirsche vor dem Start der Migration zu 100% ausgelastet ist und sich während der Pre-Copy Phase den Prozessor mit der Domain 0 teilt. Zwischen der 23. und 30. Sekunde ist die VM dann inaktiv, sobald sie aber auf Apfel wieder gestartet ist, erzeugt sie sofort wieder die volle Prozessorlast. Der abgebildete stufenartige Anstieg erklärt sich durch die unterschiedlich langen Zeiten für den Migrationsvorgang in den Testläufen. Innerhalb eines Tests steigt die Prozessorlast sofort auf 100% an.

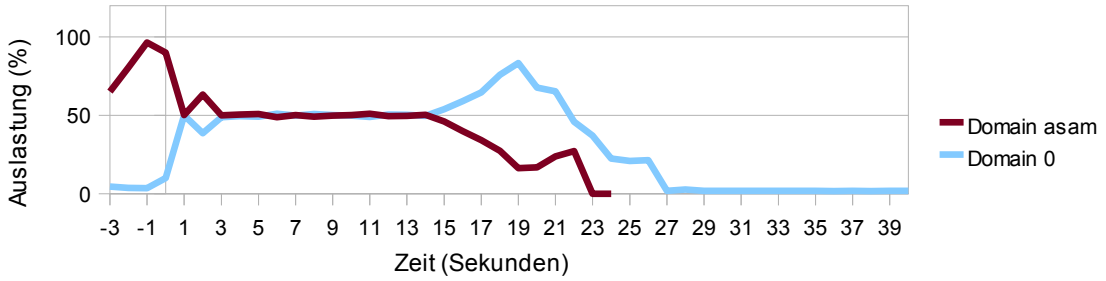

Abbildung 4.16: Pingtest mit Memtester: Prozessor Auslastung auf Kirsche (xentop)

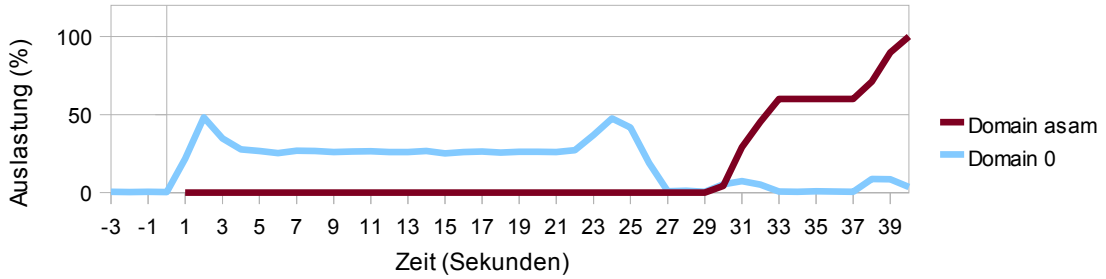

Abbildung 4.17: Pingtest mit Memtester: Prozessor Auslastung auf Apfel (xentop)

Das Diagramm in Abbildung 4.18 verdeutlicht noch einmal schön den Ablauf einer Migration. Der Benutzer stößt den Migrationsprozess zum Zeitpunkt 0 an und erzeugt dadurch kurz Last auf Seiten des Benutzers, welche anschließend vom System übernommen wird, welches die eigentliche Migration innerhalb von 27 Sekunden durchführt. Der Vorgang verläuft genauso auch auf dem Apfel Rechner mit denselben niedrigen Werten wie aus dem Test ohne Last schon bekannt. Ebenso zeigen die Software Interrupts keine nennenswerten Veränderungen.

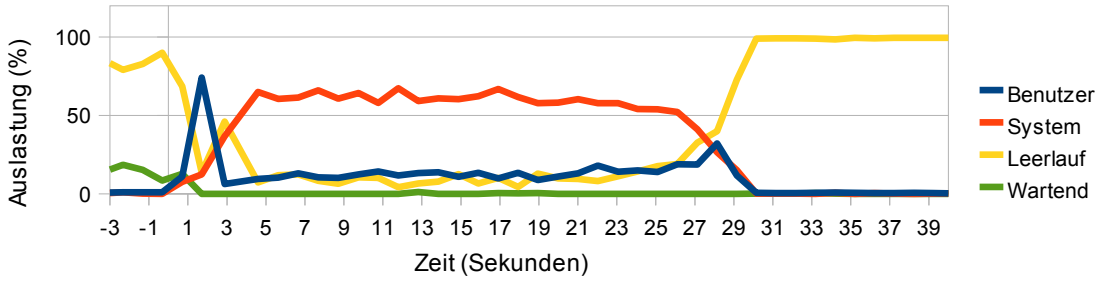

Abbildung 4.18: Pingtest mit Memtester: Prozessor Auslastung auf Kirsche (dstat)

Geradezu ein Paradebeispiel einer Netzlast während einer Live Migration zeigt Abbildung 4.19 für Apfel. Von Sekunde 0 bis 23 verläuft die Pre-Copy Phase mit wiederholtem Empfangen „verschmutzter“ Speicherdaten. Zwischen Sekunde 23 und 27 ist dann ein deutlicher Anstieg der Empfangsdaten auszumachen. In dieser Zeit ist die VM gestoppt und alle noch übrigen „schmutzigen“ Daten werden so schnell wie möglich übertragen. Nach diesen 27 Sekunden ist nun ein vollständiges und konsistentes Abbild der VM asam auf dem Apfel Rechner und Kirsche kann die VM asam beenden.

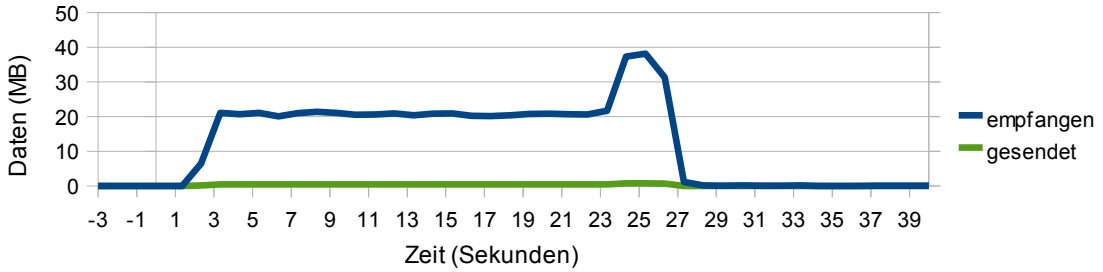

Abbildung 4.19: Pingtest mit Memtester: Netzwerk Auslastung auf Apfel

Den wichtigsten und zugleich auch deutlichsten Unterschied zeigt die Betrachtung der Ping Round Trip Zeiten in Abbildung 4.20. Die starke Last innerhalb der VM bewirkt sofort nach Start der Migration einen deutlichen Anstieg der Round Trip Time auf teilweise über 100ms und liegt damit ein zehnfaches über den Werten des vorangegangenen Tests (Abbildung 4.12). Bei besonders empfindlichen Echtzeitanwendungen kann dies hier bereits schon während der Pre-Copy Phase zu Problemen führen, welche spätestens während der hohen Downtime bei den meisten Anwendungen auftreten dürften. Hier zeigen sich deutlich die Grenzen einer Live Migration. Trotz einer relativ langen Pre-Copy Phase, die bereits zu bemerkenswerten Einschränkungen in der Dienstgüte führt und damit für den Dienstnutzer bemerkbar wird, lässt sich die Downtime nicht auf eine unproblematische Länge verkürzen. Folglich ist über den gesamten Zeitraum der Live Migration die Nutzung des angebotenen Dienstes bestenfalls eingeschränkt möglich.

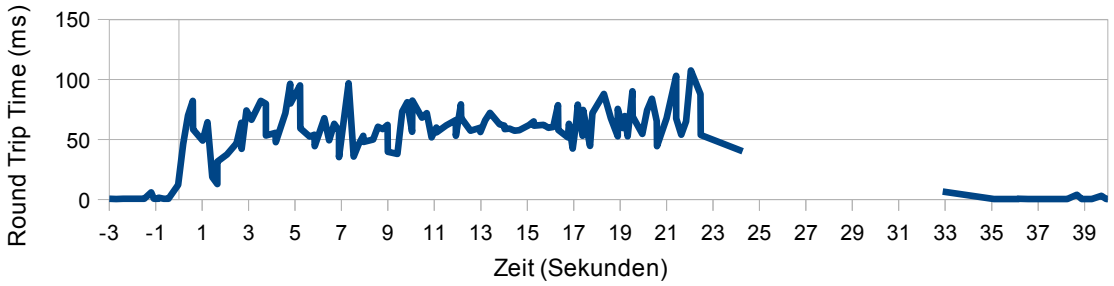

Abbildung 4.20: Pingtest mit Memtester: Ping Round Trip Time

4.1.3 Ping mit Netzwerklast

In diesem Versuch wird die Migrationszeit unter starkem Netzwerkverkehr untersucht. Hierzu wird über den Benchmark Netperf [Jon09] ein UDP Stream Test durchgeführt, der eigentlich dazu dient, die maximale Geschwindigkeit einer Netzwerkverbindung zu testen. Dabei wird die maximale Bandbreite genutzt, um zu messen, wieviel realer Durchsatz am Ende zwischen zwei Endsystemen erreicht werden kann. Da der Projekt3 Rechner kein Gigabit Interface hat, wird der Stream Test zwischen Kirsche und Apfel innerhalb des Migrationsnetzes durchgeführt. Dies ist leider nicht optimal, da der Benchmark selbst natürlich Last erzeugt und eventuell nicht das Netzwerk, sondern das Interface der Flaschenhals ist. Dennoch kann so gezeigt werden, wie sehr ein Server unter starker Netzlast bei einer Live Migration beeinflusst wird.

Listing 4.3: netperf Aufruf

```
ssh root@192.168.2.24 "netperf -H 192.168.0.25 -l$3 -t UDP_STREAM >/mnt/
    storage3/xen/extras/test1/$1-netperf.test" &
```

Wie gehabt wird der Benchmark aus dem main Skript über einen ssh Remote Aufruf gestartet (Listing 4.3). Damit wird für die übergebene Dauer der UDP Stream Test zwischen Kirsche und Apfel durchgeführt und dessen Ausgabe in eine Datei geschrieben. Davor muss noch auf Apfel der Netserver von Netperf gestartet sein.

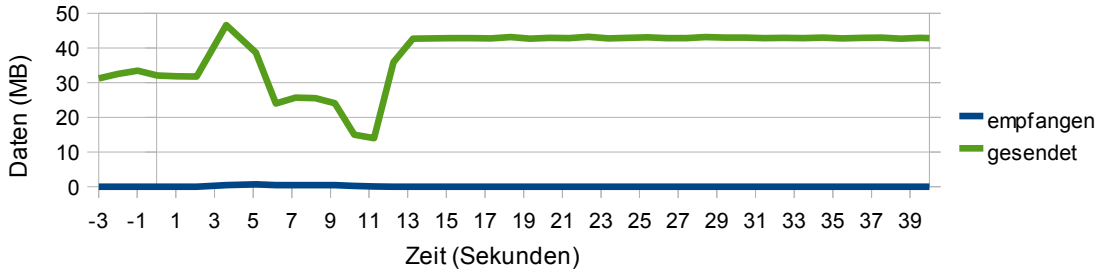

Abbildung 4.21: Pingtest mit Netperf: Netzwerk Auslastung auf Kirsche

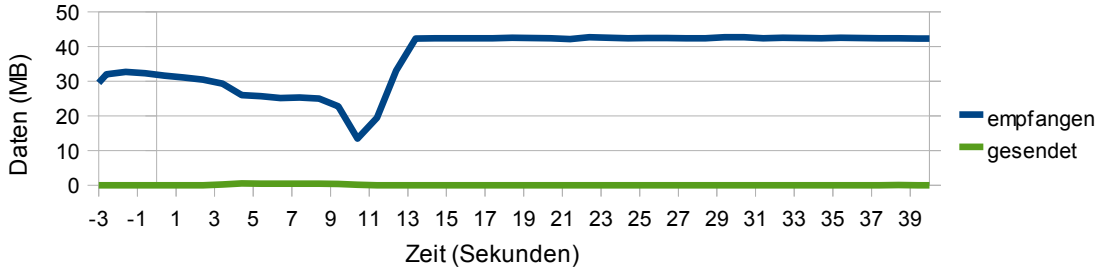

Abbildung 4.22: Pingtest mit Netperf: Netzwerk Auslastung auf Apfel

In diesem Test sind daher die Werte zur Netzauslastung besonders interessant. Abbildung 4.21 und Abbildung 4.22 zeigen auch schön, wie nun das Netzwerk überwiegend bei 42 bis 43 MB/s (ca. 340 Mbit/s) ausgelastet ist. Deutlich ist auf beiden Rechnern zu erkennen, dass während der Migration die Datenraten einbrechen und sich erst nach Abschluss konstant

bei etwa 42,5 MB/s bewegen. Anscheinend bremst der Migrationsprozess die Netzwerkgeschwindigkeit.

Die starke Netzlast wirkt sich auch auf die übrigen entscheidenden Zeiten aus. So dauert jetzt im Durchschnitt der Migrationsprozess 10,49 Sekunden und die Downtime verlängert sich ebenfalls, wie Tabelle 4.23 zu entnehmen ist. Allerdings ist sie im Vergleich zu den Tests ohne Last ziemlich gering.

	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**
Anzahl nicht beantworteter Requests	9	6	6	6	6
Ping Ausfallzeit (in Sekunden)	1,8	1,2	1,2	1,2	1,2
zeit Skript Ausfallzeit (in Sekunden)	1,66	1,06	1,05	1,07	0,99

Abbildung 4.23: Ausfallzeiten mit Netperf

Die CPU Auslastung wird durch den Einsatz von Netperf während der Migration wenig verändert. Allerdings belastet Netperf die CPU auf Kirsche in der Domain 0 auch nachdem die VM migriert ist (Abbildung 4.24). Dies wird verursacht durch den zwischen Kirsche und Apfel durchgeführten Streaming Test, der nicht mit der VM zusammenhängt. Folglich wird auch auf Apfel bereits vor der Migration eine deutliche CPU Last in der Domain 0 gemessen (Abbildung 4.25).

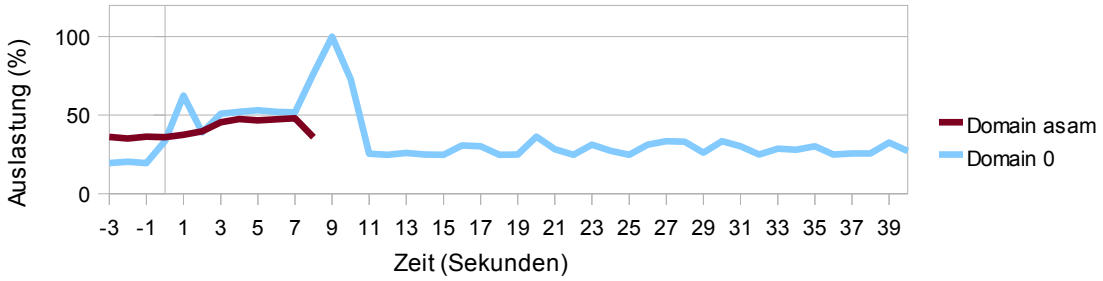

Abbildung 4.24: Pingtest mit Netperf: Prozessor Auslastung auf Kirsche (xentop)

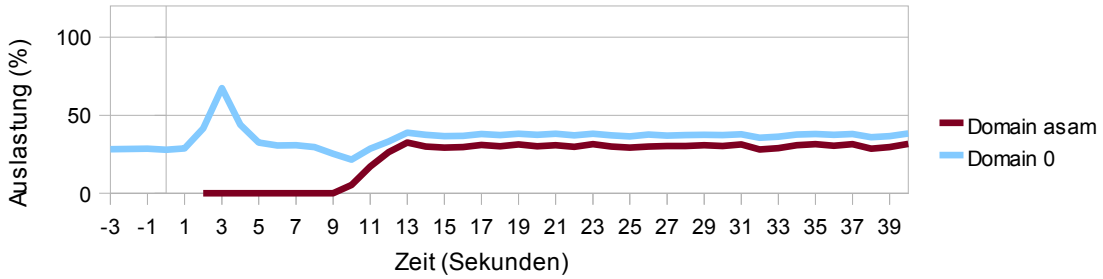

Abbildung 4.25: Pingtest mit Netperf: Prozessor Auslastung auf Apfel (xentop)

Ebenfalls deutlich fällt der Einsatz von Netperf bei den Software Interrupts ausserhalb der Migrationszeit zumindest auf dem Kirsche System auf, wie Abbildung 4.26 zeigt. Auf Apfel hingegen gibt es keine Häufung an Software Interrupts ausserhalb der Migrationszeit (Abbildung 4.27).

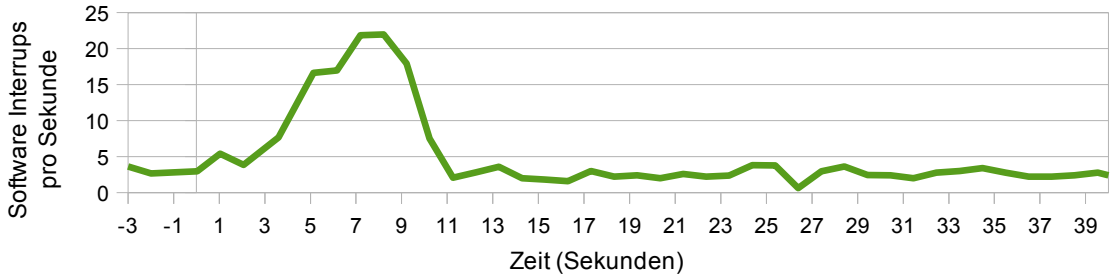

Abbildung 4.26: Pingtest mit Netperf: Software Interrupts auf Kirsche

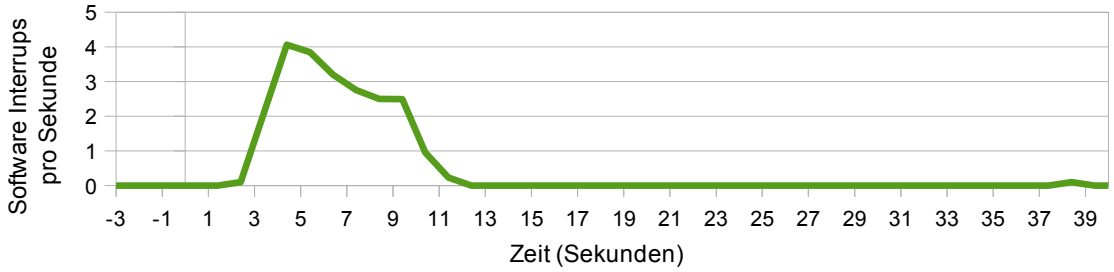

Abbildung 4.27: Pingtest mit Netperf: Software Interrupts auf Apfel

Die Round Trip Time zeigt sich während der Migration durchaus erhöht (Abbildung 4.28), etwa zwei bis dreimal so hoch wie im Test ohne Last. Da sie aber vor und nach der Migration auf konstant niedrigem Niveau wie ohne Last ist, erzeugt nicht Netperf selbst die Erhöhung, sondern nur die Kombination aus starker Netzlast mit gleichzeitiger Live Migration.

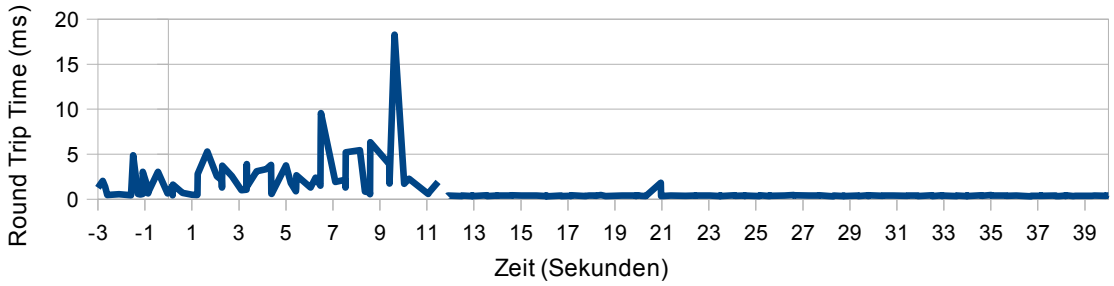

Abbildung 4.28: Pingtest mit Netperf: Ping Round Trip Time

Eine hohe Netzlast hat somit durchaus Auswirkungen auf eine Live Migration, allerdings in eher geringem Maß. Problematisch könnte jedoch der stellenweise recht dramatische Einbruch in der Datenübertragungsgeschwindigkeit auch für andere aktive VMs und deren Dienste sein.

4.1.4 Ping mit CPU Last

Als weiterer Test wird die VM unter extreme Prozessor Last gesetzt. Dazu wird ein sehr einfaches Tool namens CPU Burn-in verwendet [Mie], welches versucht, den Prozessor durch intensive Rechenaufgaben auf die durch Software maximal erreichbare Temperatur zu erhitzen. Eigentliches Einsatzgebiet ist das Testen von übertakteten Systemen auf Stabilität. Hier ist lediglich die dabei erzeugte intensive Prozessorlast entscheidend. Da die hier verwendeten Systeme nicht übertaktet sind, wurden während des Tests auch keine Inkonsistenzen festgestellt. Gestartet wird CPU Burn-in wie üblich über ssh aus dem main Skript (Listing 4.4). Der zweite Parameter lässt cpuburn-in für 2 Minuten laufen, während die Ausgabe mangels relevanter Informationen unterdrückt wird.

Listing 4.4: cpuburn-in Aufruf

```
ssh root@192.168.2.70 "~/bench/cpuburn-in 2 >/dev/null &"
```

Trotz dieser aussergewöhnlich hohen CPU Beanspruchung liegt die durchschnittliche Zeit für den Migrationsprozess bei nur 9,35 Sekunden. Interessanterweise liegen allerdings die Downzeiten hier über denen des Migrationsprozesses. Weshalb hier die Tabelle 4.29 über die Ausfallzeiten noch um eine Zeile erweitert wird, um die deutliche Differenz herauszustellen.

	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**
Anzahl nicht beantworteter Requests	73	58	44	39	101
Ping Ausfallzeit (in Sekunden)	14,6	11,6	8,8	7,8	20,2
zeit Skript Ausfallzeit (in Sekunden)	14,97	11,78	8,95	8,39	20,61
Dauer Migrationsprozess (in Sekunden)	9,86	10,22	8,96	8,88	8,84

Abbildung 4.29: Ausfallzeiten und Migrationsprozess Dauer mit cpuburn-in

Auch wenn die Zeiten des Migrationsprozesses in relativ ähnlichen Größen bleiben, driften die Ausfallzeiten doch recht deutlich auseinander. Der Einfluss von CPU Burn-in ist sogar so stark, dass es dem zeit Skript nicht gelingt in der üblichen Weise die Ausgabe von date mitzuprotokollieren. Statt der üblichen 50 Einträge pro Sekunde werden nur noch zwischen 5 und 10 Einträgen pro Sekunde erstellt, mit erkennbarem Unterschied vor und nach der Migration: Zum Ende der Pre-Copy Phase werden es immer weniger, bis runter auf 5 Einträge pro Sekunde, während sonst meistens 10 Einträge erstellt werden. Mehr als einen gewissen Verlust an Messgenauigkeit hat dies aber zum Glück nicht zur Folge.

Die CPU Auslastung in der Domain 0 unterscheidet sich nicht weiter von der ohne Last. Nur die Daten von xentop zeigen den Einsatz von CPU Burn-in in den Abbildungen 4.30 und 4.31 deutlich. Der treppenartige Anstieg der Prozessorauslastung nach der Aktivierung der VM auf Apfel ist dabei auf die sehr unterschiedlichen Downzeiten bei diesem Test zurückzuführen und bedeutet nicht eine langsame sukzessive Steigerung der Prozessorlast. Vielmehr wird die CPU sofort nach Aktivierung der VM wieder zu 100% ausgelastet.

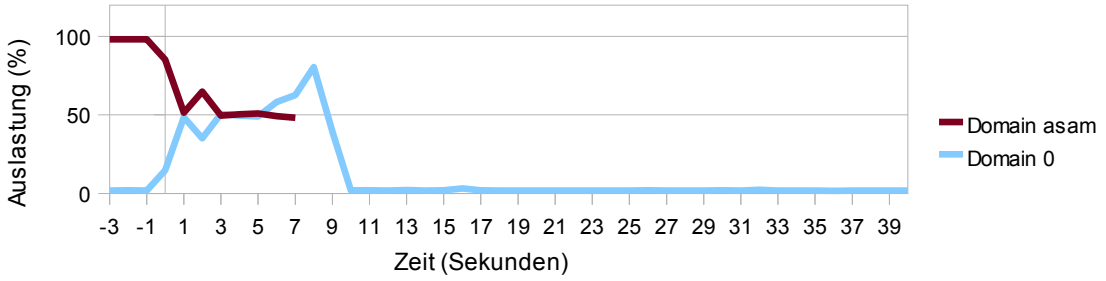

Abbildung 4.30: Pingtest mit CPU Burn-in: Prozessor Auslastung auf Kirsche (xentop)

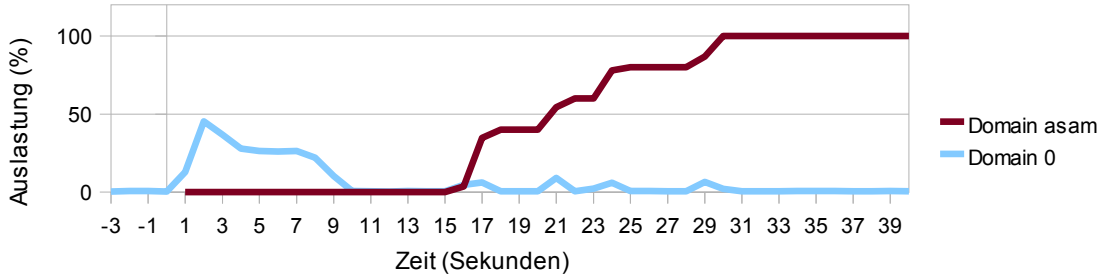

Abbildung 4.31: Pingtest mit CPU Burn-in: Prozessor Auslastung auf Apfel (xentop)

Dabei fällt auf, dass nach Sekunde 7 die VM auf Kirsche schon nicht mehr vorhanden ist, die Wiederaktivierung der VM aber erst frühestens zu Sekunde 16 geschieht. Auch das Diagramm zur Netzlast in Abbildung 4.32 zeigt, dass nach maximal 11 Sekunden keine Daten mehr ausgetauscht werden. Somit ist anzunehmen, dass die reine Wiederaktivierung der VM volle 5 Sekunden benötigt. Vermutlich ist das Wiederherstellen der CPU Status bei dieser starken CPU Last besonders aufwendig und zeitintensiv.

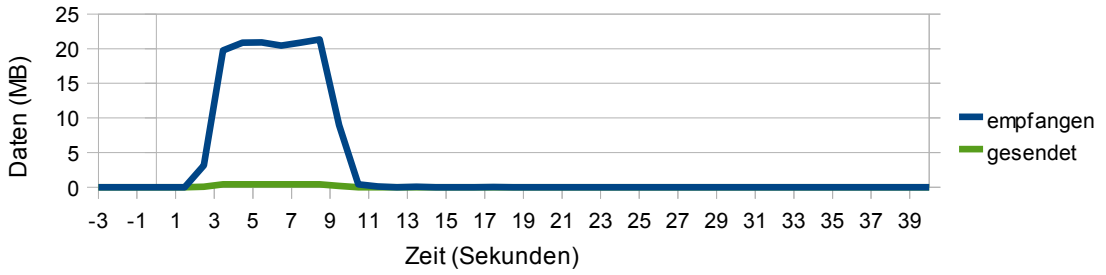

Abbildung 4.32: Pingtest mit CPU Burn-in: Netzwerk Auslastung auf Apfel

Die bereits oben ausführlich vermerkte lange Downtime wird zusätzlich während der Migrationsdauer von ziemlich hohen Round Trip Times begleitet (Abbildung 4.33). Auch hier ist wieder nur die Kombination aus Migration und CPU Last dafür verantwortlich. CPU Burn-in alleine verschlechtert die Round Trip Time nicht, wie die Werte vor (Zeitpunkt 0) und nach (Zeitpunkt 20) der Migration zeigen.

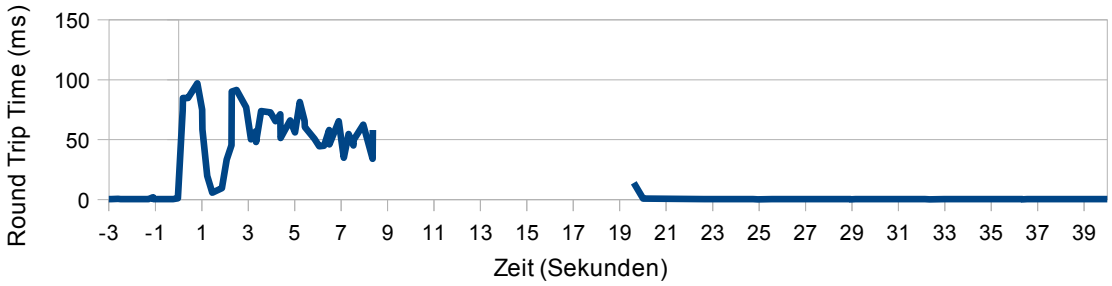

Abbildung 4.33: Pingtest mit CPU Burn-in: Ping Round Trip Time

4.1.5 Ping mit I/O Last

Als letztes wird nun der Einfluss von I/O Operationen untersucht. Zum Einsatz kommt hier das Tool Iometer [QDS+], welches auch in zahlreichen anderen Publikationen als I/O Benchmark eingesetzt wird. Das Tool arbeitet nach dem Client Server Prinzip um viele Systeme gleichzeitig und zentralisiert testen zu können. Der Client heißt Dynamo, der Server Iometer. Iometer bietet eine komfortable Benutzeroberfläche, von der aus alle Dynamo Prozesse gesteuert werden, egal ob lokal oder verteilt im Netzwerk. Da Iometer nur auf Microsoft Windows Systemen läuft, wird hier zusätzlich ein Laptop mit Microsoft Windows XP an das Steuerungsnetz angeschlossen. Dynamo dagegen ist relativ Plattform unabhängig und funktioniert auf dem in der VM installiertem Debian System. Zwar wird Dynamo auch aus dem main Skript heraus automatisch gestartet, welches danach aber auf eine Eingabe wartet, um in der Zwischenzeit Iometer manuell starten zu können. Dynamo wird wie üblich über ssh gestartet und benötigt über die Option '`-i`' die IP Adresse des Rechners, auf dem Iometer läuft, sowie über '`-m`' die IP Adresse unter der Dynamo selbst erreichbar ist (Listing 4.5). Die Ausgabe von Dynamo wird zur eventuellen Fehleranalyse in eine Logdatei geschrieben, ist aber nicht von Belang, da alle relevanten Daten von Iometer standardmäßig in eine csv Datei geschrieben werden.

Listing 4.5: dynamo Aufruf

```
ssh root@192.168.2.70 ~/dynamo -i 192.168.2.2 -m 192.168.2.70 >~/dynamo.
    log &
read -p "Iometer gestartet?"
```

Vor dem ersten Testlauf erzeugt Dynamo im Root Verzeichnis eine Datei 'iobw.tst', auf welche die Zugriffe ausgeführt werden. Standardmäßig versucht Dynamo die Datei so groß wie möglich zu machen, was zur Folge hatte, dass das zeit Skript nicht mehr ausreichend Platz für seine Logdatei hatte. Deshalb wurde die Größe manuell etwas darunter eingestellt, nämlich auf 40000 Sektoren (156,25 MByte). Die Größe ist in diesem Test auch nicht entscheidend, da nicht die Performance der Festplatte oder des NAS Systems getestet werden soll, sondern nur möglichst viele I/O Operationen ausgeführt werden sollen.

Als Test wird ein sogenannter Access Pattern nach der Definition von StorageReview.com erstellt, der einer typischen Workstation entsprechen soll [Ra]. Die dafür eingestellten Werte sind Tabelle 4.34 zu entnehmen. Die Anzahl an gleichzeitig anstehenden Anfragen (Number of Outstanding I/Os) beeinflusst die Gesamtlast unter der das I/O Gerät steht. Mit 64 ist hier eine als moderat eingestufte Last gewählt, die zwar hoch ist, aber gleichzeitig noch in der Realität auftritt.

% of Access Specification	**Transfer Size Request**	**% Reads**	**% Random**	**Number of Outstanding I/Os**
100%	8 KB	80%	80%	64

Abbildung 4.34: Workstation Access Pattern (nach der Definition von StorageReview.com)

Bei einer durchschnittlichen Dauer des Migrationsprozesses von 12,01 Sekunden ist zwar hier ein deutlicher Einfluss messbar, auf die Downzeiten hat dies aber einen eher geringen Einfluss. Mit 0,69 Sekunden Downtime und nur 3 unbeantworteten Ping Requests erzielte ein Testdurchlauf sogar die bisher kürzeste gemessene Downtime, kürzer als ohne Last. Wie

Tabelle 4.35 allerdings zeigt, variieren die Downtimes stärker und liegen im Mittel doch ein bisschen über denen ohne Last.

	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**
Anzahl nicht beantworteter Requests	8	8	5	3	5
Ping Ausfallzeit (in Sekunden)	1,6	1,6	1,0	0,6	1,0
zeit Skript Ausfallzeit (in Sekunden)	1,59	1,56	0,95	0,69	0,88

Abbildung 4.35: Ausfallzeiten mit Iometer

Entsprechend ähnlich zeigen sich daher auch die meisten Diagramme zu denen ohne Last. In den Diagrammen 4.36 und 4.37 zur Prozessorauslastung ist, neben den insgesamt mit etwa 3 Sekunden etwas länger ausschlagenden Werten, vorallem die grüne Linie interessant, die den Status 'wartend' der CPU darstellt. Dieser Status zeigt an, wann die CPU auf I/O Geräte warten muss. In allen bisherigen Tests verlief diese stets bei 0, während sie hier auf Kirsche über 90% und auf Apfel um die 50% erreicht. Die 50% auf Apfel erklären sich wiederum durch den Dual Prozessor, bei dem nur ein Prozessor warten muss, während sich der andere im Leerlauf befinden kann.

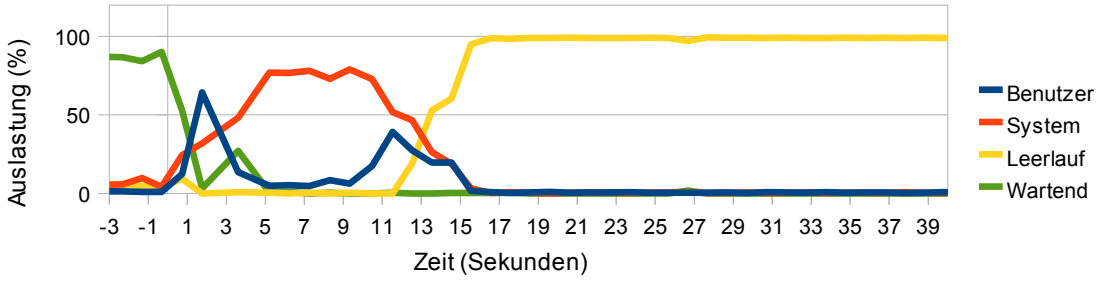

Abbildung 4.36: Pingtest mit Iometer: Prozessor Auslastung auf Kirsche (dstat)

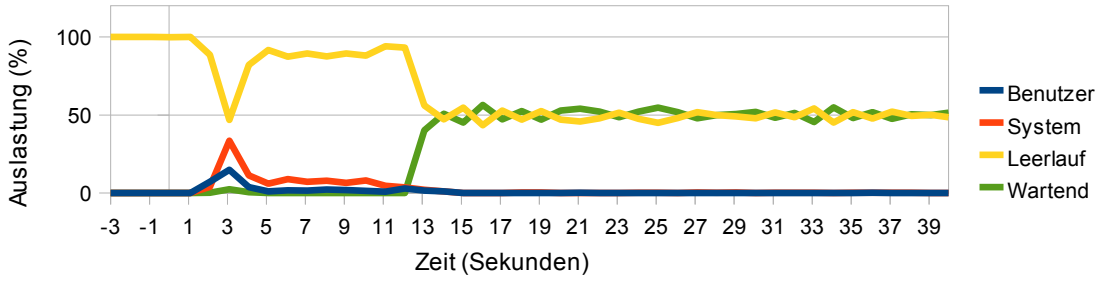

Abbildung 4.37: Pingtest mit Iometer: Prozessor Auslastung auf Apfel (dstat)

Dynamo selbst erzeugt keine bemerkenswerte zusätzliche CPU Last, weshalb die Prozessor Auslastung innerhalb der virtuellen Maschine der während des Tests ohne Last entspricht. Auf ähnlichem Niveau zeigen sich die Software Interrupts und die Netzwerkauslastung. Die in Abbildung 4.38 dargestellte Round Trip Time fällt durch die I/O Last dann allerdings doch merkbar höher aus, bewegt sich aber noch in unproblematischen Größenordnungen.

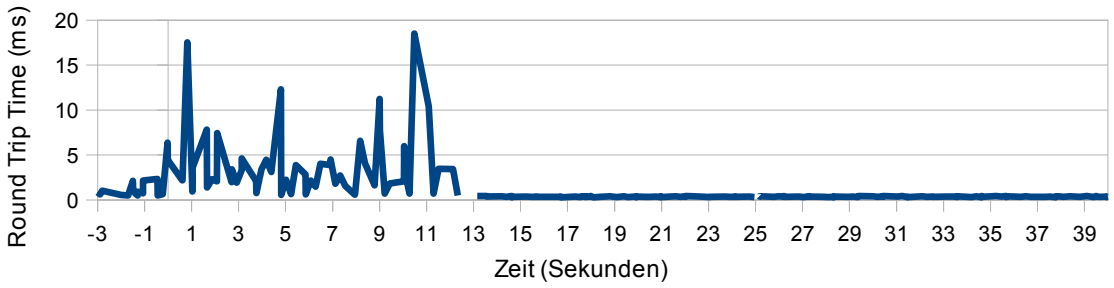

Abbildung 4.38: Pingtest mit Iometer: Ping Round Trip Time

Die starke I/O Last hat allerdings auch noch einen gewissen Einfluss auf den Speicher. Üblicherweise verlaufen die beiden Graphen für den freien und den genutzten Speicher weitgehend inversiv, die nun in den Abbildungen 4.39 und 4.40 auftretenden Diskrepanzen erklären sich durch die hier auftretende intensivere Nutzung des Caches.

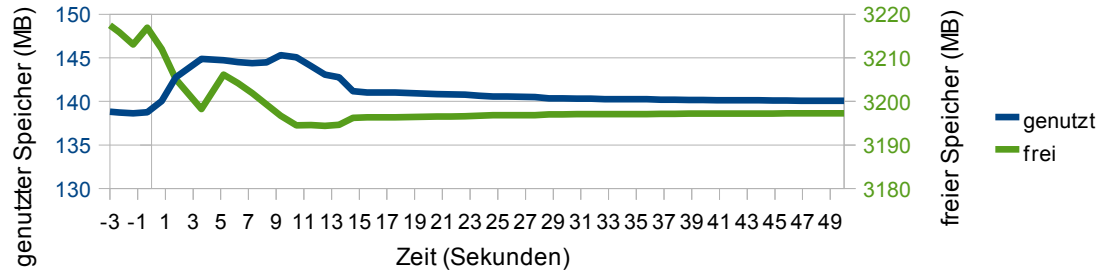

Abbildung 4.39: Pingtest mit Iometer: Speichernutzung auf Kirsche

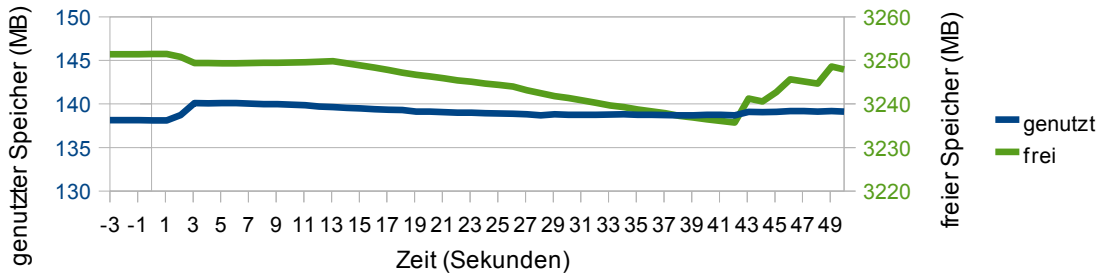

Abbildung 4.40: Pingtest mit Iometer: Speichernutzung auf Apfel

4.2 Streaming Test

Im zweiten Test wird der Einfluss einer Live Migration auf eine Echtzeitanwendung untersucht. Die Open Source Software VLC Media Player des VideoLAN Projects [Tea] bietet hierfür alle wichtigen Funktionen und ist zudem sowohl unter Linux als auch unter Microsoft Windows funktionsfähig. Als Video wird ein Trailer des aktuellen Kinofilmes „Planet 51“ eingesetzt, welches vorab in ein streaming-fähiges Format codiert wurde. Die genauen Video Eigenschaften zeigt Tabelle 4.41.

Video	
Codec	h.264
Auflösung	640x360
Framerate	25 Bilder pro Sekunde
Bitrate	2000 Kbit/s
Audio	
Codec	mp4a
Kanäle	2 (Stereo)
Abtastrate	48000 Hz
Bitrate	128 Kbit/s
Gesamt	
Container	ts (MPEG Transport Stream)
Dateigröße	37513 KB

Abbildung 4.41: „Planet 51“ Video Eigenschaften

Dieses Video wird durch VLC von der VM asam aus per UDP Stream in das Steuerungsnetz an die Broadcast Adresse 192.168.2.255 gestreamt und gleichzeitig von dem Projekt3 Rechner und dem Windows Laptop ebenfalls über VLC angezeigt. Die gleichzeitige Wiedergabe auf 2 unterschiedlichen Systemen ermöglicht es, interessante Unterschiede auf der Client Seite festzustellen, die mit dem VLC Server nicht in Verbindung stehen.

Um das Starten von VLC in der VM und auf dem Projekt3 wieder soweit wie möglich zu automatisieren, wird das main Skript entsprechend angepasst. Wie schon im Abschnitt 3.5.1 beschrieben, werden dafür einige Zeilen auskommentiert und andere dagegen wieder aktiviert. In Zeile 86 aus Listing 3.2 wird der VLC Media Player lokal auf Projekt3 gestartet. Über den Parameter '`udp://@`' sucht VLC nach UDP Broadcast Streams und gibt diese gegebenenfalls wieder. Die restlichen Parameter '`--verbose 1`' und '`--extraintf logger`' dienen der Generierung einer Protokoll Datei, während '`--quiet`' die direkte Konsolen Ausgabe einschränkt. Anschließend wird in Zeile 89 wie üblich über einen ssh Remote Aufruf der VLC Server in der VM asam gestartet und die Datei 'Planet_51.ts' und das VLC Kommando '`vlc://quit`' (zum automatischen Beenden von VLC nach erfolgtem Stream) in die Wiedergabeliste aufgenommen. Über den Parameter '`--sout`' wird die Art und das Ziel des Streams übergeben, sowie die Verwendung des SAPs (Session Announcement Protocol) zum verbesserten Auffinden der angebotenen Streaming Services. Mit '`--intf dummy`' wird ein Dummy Interface ausgewählt, das keine graphische Oberfläche benötigt. Der VLC Media Player auf dem Windows Laptop muss bereits vorher manuel gestartet worden sein und bedarf eines leicht modifizierten Aufrufes mit '`udp://@192.168.2.71:1234`' als zu öffnende Adresse und ohne '`--quiet`', da dies hier auch die Protokollausgabe deaktiviert.

4.2.1 Video-Stream ohne Last

Zuerst wird auch das Streamen des Trailers ohne zusätzliche Last durchgeführt. Wie die Abbildungen 4.42 bis 4.47 für die CPU Belastungen bereits zeigen, ist der Unterschied zum Pingtest marginal. Lediglich eine minimal allgemein höhere Belastung lässt sich ausmachen. Dies zeigt sich zum Beispiel in der um etwa eine Sekunde länger andauernden CPU Auslastung auf Kirsche in Abbildung 4.43 und den etwas deutlicher ausschlagenden Graphen für den Apfel Rechner in Abbildung 4.46.

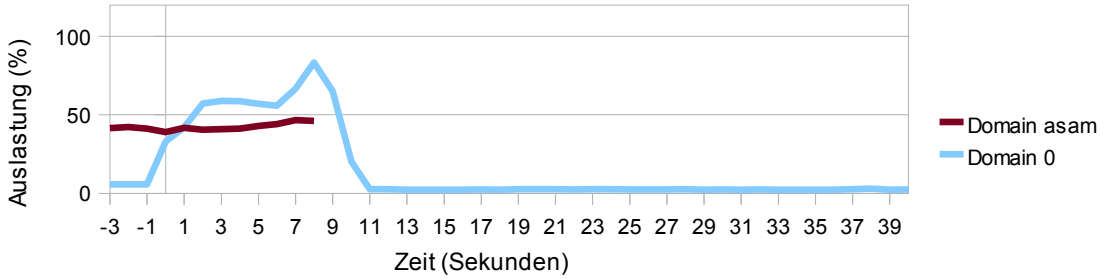

Abbildung 4.42: Streaming Test ohne Last: Prozessor Auslastung auf Kirsche (xentop)

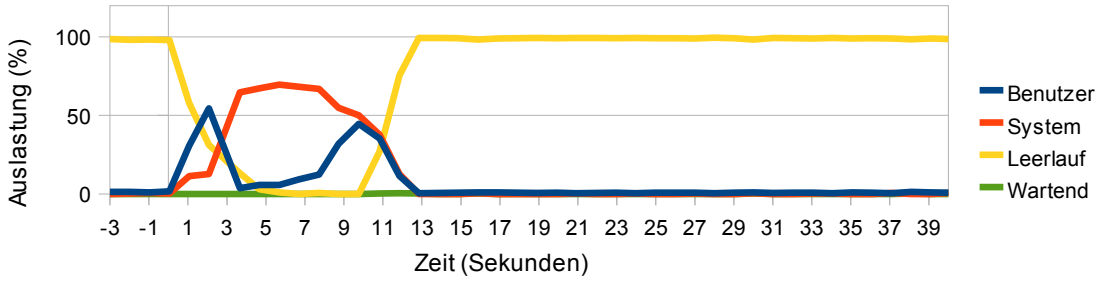

Abbildung 4.43: Streaming Test ohne Last: Prozessor Auslastung auf Kirsche (dstat)

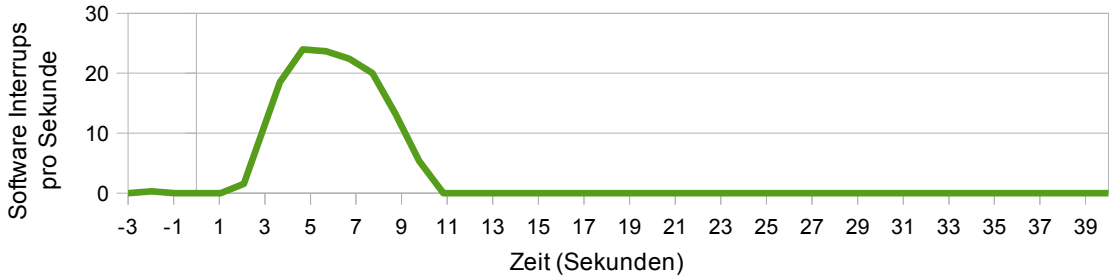

Abbildung 4.44: Streaming Test ohne Last: Software Interrupts auf Kirsche

Die CPU Auslastung der VM ist sogar geringer. Während sie auf Kirsche im Pingtest bei bis zu 49,07% liegt, erreicht sie im Streaming Test nur maximal 46,6%. Verkehrtes Bild allerdings auf dem Apfel Rechner. Hier werden im Pingtest maximal 27,08%, im Streaming Test jedoch 30,4% gemessen. Dies ist wohl darauf zurückzuführen, dass das Wiederaufneh-

men des Videostreams kurzzeitig mehr Prozessorlast erzeugt, während das Beantworten von Ping Anfragen auch bei Wiederaktivierung der VM nicht mehr CPU Leistung benötigt. Das bestätigen auch die von xentop gemessenen Werte für die Domain 0. Während auf Kirsche eine ähnliche CPU Auslastung angezeigt wird, schlägt auf dem Apfel Rechner die Kurve kurzzeitig deutlich stärker aus (71,1% in Abbildung 4.45 gegenüber 54,63% in Abbildung 4.5). Das Abflachen der roten Kurve für die Domain asam nach der 19. Sekunde wird durch das manuelle Deaktivieren des zeit Skriptes verursacht.

Abbildung 4.45 zeigt noch eine weitere interessante Beobachtung. Ab Sekunde 26 ist das zeit Skript in keinem Testdurchlauf mehr aktiv und der Graph illustriert ab da schön, dass der VLC Server alleine die CPU lediglich zu etwa 1,5% auslastet.

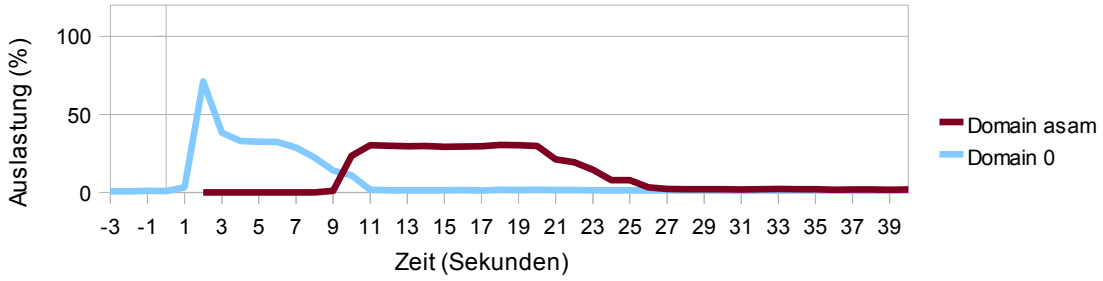

Abbildung 4.45: Streaming Test ohne Last: Prozessor Auslastung auf Apfel (xentop)

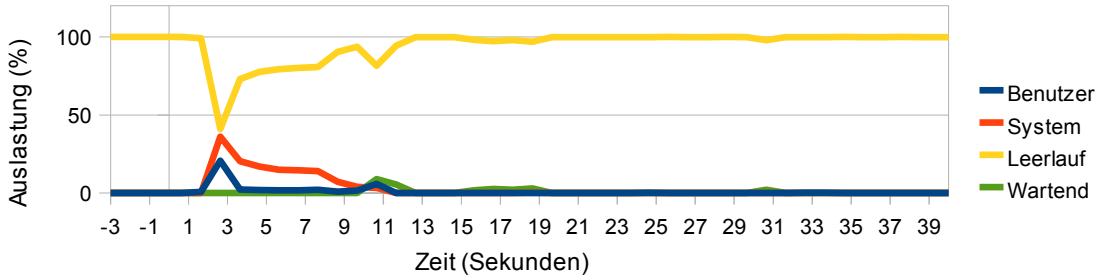

Abbildung 4.46: Streaming Test ohne Last: Prozessor Auslastung auf Apfel (dstat)

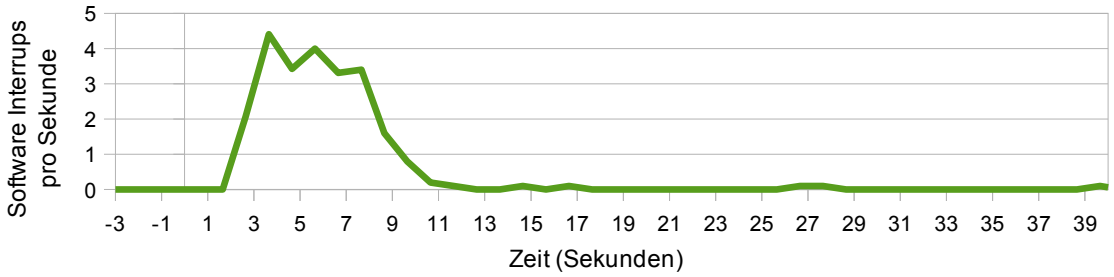

Abbildung 4.47: Streaming Test ohne Last: Software Interrupts auf Apfel

Die Graphen zu den Software Interrupts zeigen wie gewohnt den deutlichen Unterschied zwischen Kirsche und Apfel Rechner und gehen mit den bisherigen Beobachtungen einher,

dass die Belastung im Streaming Test geringfügig länger und stärker ist (Abbildungen 4.44 und 4.47).

Ähnliches lässt sich auch bei der Speicher Auslastung beobachten. Auf Kirsche entspricht der Maximalwert des genutzten Speichers während des Pingtestes (151,29 MB) in etwa dem Minimalwert des Streaming Tests (151.09 MB). Dieser Versatz ist dem VLC Server zuzuschreiben, der offensichtlich mit erstaunlich wenig Ressourcen auskommt. Während der Migration werden dann sogar etwa 2 MB weniger zusätzlich genutzt (Abbildung 4.48). Auf Apfel hingegen wird sogar weniger Speicher benutzt als im Pingtest und mit minimal höherem Speichermehraufwand migriert (Abbildung 4.49).

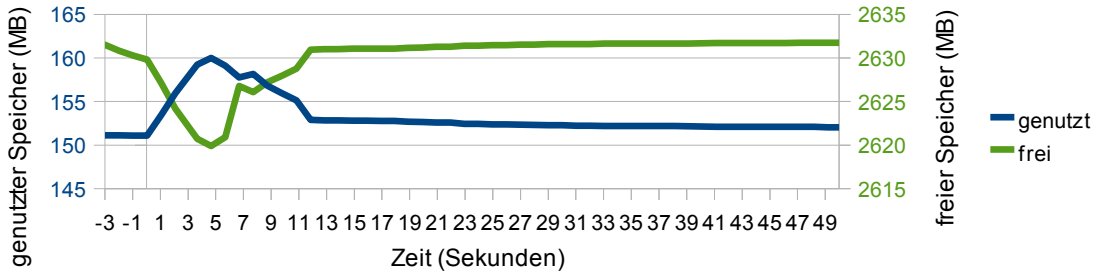

Abbildung 4.48: Streaming Test ohne Last: Speichernutzung auf Kirsche

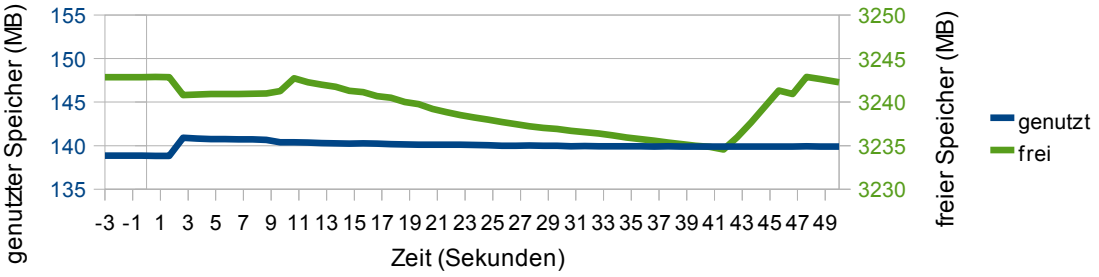

Abbildung 4.49: Streaming Test ohne Last: Speichernutzung auf Apfel

Die Netzwerkauslastung fällt wie zu erwarten durch den Videostream während der gesamten Testdauer ein bisschen höher aus, bleibt aber mit Werten um 1 MB/s auf sehr niedrigem Niveau und ist kaum anhand der Graphen auszumachen. Nur auf Kirsche schlägt der Graph einmal kurz deutlich stärker aus und erreicht Werte von 41,17 MB/s wie Abbildung 4.50 belegt. Dies ist jedoch nicht in allen Tests durchgängig. Genauer wurden zweimal mit 69,71 MB/s und 64,80 MB/s aussergewöhnlich hohe Raten erzielt, während in den restlichen drei Tests keine besonders hohen Werte auftauchten. Somit kann die Netzlast kurzfristig mal ansteigen, muss es aber nicht. Auf Apfel hingegen verläuft alles normal und zeigt, bis auf die allgemein erhöhte Grundlast durch den Videostream, ein sehr ähnliches Bild zum Pingtest (Abbildung 4.51).

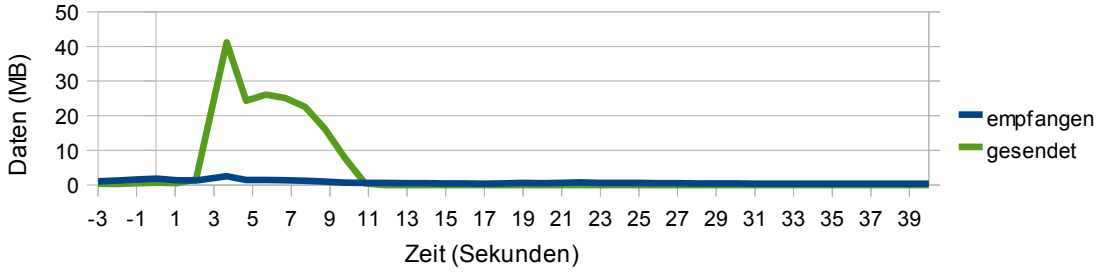

Abbildung 4.50: Streaming Test ohne Last: Netzwerkauslastung auf Kirsche

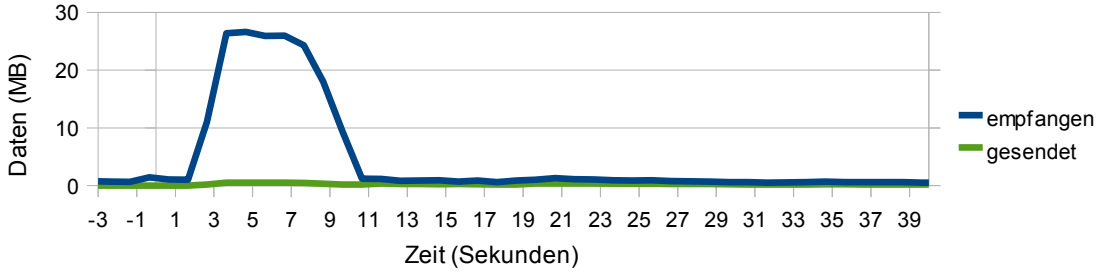

Abbildung 4.51: Streaming Test ohne Last: Netzwerkauslastung auf Apfel

Nachdem in diesem Test kein Ping zur Messung der Downzeit herangezogen werden kann, stützt sich hier die Messung rein auf die Ergebnisse des zeit Skriptes. Wie die Pingtests aber gezeigt haben, korrelieren die Werte sowieso ausgezeichnet und bieten eine ausreichend genaue Messung der Downtime. Diese ist hier mit im Mittel 0,9 Sekunden trotz gleichzeitigem Stream recht niedrig geblieben, bleibt aber jederzeit für den Anwender bemerkbar und stört das Videoerlebnis.

	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**
zeit Skript Ausfallzeit	1,14	1,05	0,82	1,05	1,49
verlorene Frames (Linux)	5	5	38	5	24
verlorene Frames (Windows)	9	8	13	9	9
verlorene Puffer (Linux)	2	3	47	2	31
verlorene Puffer (Windows)	64	65	109	64	96

Abbildung 4.52: Ausfallzeit und Streaming Fehler ohne Last

Zusätzlich zeigt Tabelle 4.52 nun weitere Details der VLC Clienten. Die verlorenen Frames beziehen sich auf nicht angezeigte Bilder, während der verlorene Puffer ausgefallenen Ton repräsentiert. Wie sich im Laufe der weiteren Tests auch bestätigen wird, liefern die VLC Clienten unter Linux und Windows sehr unterschiedliche Statistiken. Die Anzahl der angeblich verlorenen Frames passt dabei allerdings nicht zu den Downzeiten. Bei einer Framerate von 25 Bildern pro Sekunde müssten also im Mittel etwa 22,5 Bilder verloren gehen. Sehr merkwürdig ist auch die Beobachtung, dass ausgerechnet bei der kürzesten Downtime die meisten Frames verloren gehen.

Um dies zu erklären muss man sich das Video ansehen, dass der Client ausgibt. Hier scheint

das größte Problem zu sein, nach der Migration den Stream wieder zu synchronisieren. Dies hängt von dem Zeitpunkt der Wiederaufnahme des Streams ab und gelingt teilweise recht schnell, aber im worst case überhaupt nicht (passierte während aller Tests nur einmal). Die hier gezeigten Werte sind also nicht Frames, die während der Downtime gezeigt werden hätten müssen, sondern Frames die empfangen wurden, aber nicht im richtigen Moment angezeigt werden konnten. Sie liefern damit ein Indiz dafür, wie schwer es für den jeweiligen Client ist, den Stream wieder korrekt anzuzeigen. Diese ganze Problematik hat zur Folge, dass das Video nicht nur, wie vielleicht zu erwarten wäre, für eine gewisse Dauer stoppt, sondern springt und somit dem Anwender Szenen verloren gehen können. Bei der Wiederaufnahme der Wiedergabe erklingt dabei zuerst nur Ton und nach unterschiedlich langer Zeit kommt das dazugehörige Bild dazu. Dabei treten anfangs für kurze Zeit Bildstörungen auf, wie der Screenshot in Abbildung 4.53 exemplarisch zeigt.

Abbildung 4.53: Screenshot nach Wiederaufnahme der Wiedergabe
(Planet 51 © 2009 Ilion Animation Studios)

Interessanterweise erzeugt hier der Test mit der kürzesten Downtime die schlechtesten Werte bei den verlorenen Frames und Puffer. Scheinbar kann die VM zu schnell aktiviert werden, wodurch Probleme für die laufenden Anwendungen auftreten.

4.2.2 Video-Stream mit Speicherlast

Der Test mit Memtester benötigt eine kleine Modifikation im Streaming Test, da der VLC Server selbst ein bisschen Speicher benötigt. Die Speichertests von Memtester werden hier nur auf 100 MB Speicher durchgeführt, da nach dem Start von VLC Memtester keine 110 MB Speicher mehr reservieren kann (Listing 4.6).

Listing 4.6: Memtester Aufruf während Streaming Test

```
memtester 100 1 >~/memtester.log &
```

Zusammen mit VLC wird so trotzdem gewährleistet, dass möglichst viel Speicher genutzt wird. Die nun auftretenden Downtimes, aus Tabelle 4.54 zu entnehmen, bestätigen die aussergewöhnlich hohe Last in diesem worst case Szenario.

	Test 1	Test 2	Test 3	Test 4	Test 5
zeit Skript Ausfallzeit	26,73	30,36	11,05	21,2	27,47
verlorene Frames (Linux)	591	598	316	596	601
verlorene Frames (Windows)	14	13	14	12	81
verlorene Puffer (Linux)	1136	1392	486	906	1217
verlorene Puffer (Windows)	215	592	190	181	432

Abbildung 4.54: Ausfallzeit und Streaming Fehler mit Memtester

Ähnlich wie schon im Pingtest gibt es auch hier starke Schwankungen in den gemessenen Werten, generell sind sie aber noch einmal um ein Vielfaches höher. Wie aber auch schon im Pingtest ist die Zeit für den Migrationsprozess mit einer Standardabweichung von 0,36 Sekunden erstaunlich einheitlich über allen Tests und liegt im Mittel bei 27,55 Sekunden. Interessant ist die Auswirkung auf die Videowiedergabe nach der Migration. Unerwartet ähnlich verhalten sich hier die Werte für die verlorenen Frames trotz teilweiser Downtime Unterschieden von fast 10 Sekunden. Nur im besten Fall mit 11,05 Sekunden Downtime erweist sich auch die Anzahl an verlorenen Frames sowie verlorenen Puffer als deutlich niedriger. Soweit zumindest die Beobachtungen auf dem Linux System. Unter Windows protokolliert VLC völlig andere Werte. Die Anzahl an verlorenen Frames liegt dabei nur minimal über denen im Test ohne Last und damit um mehr als das 42 fache unter denen auf Linux gemessenen. Dies deckt sich auch mit der Beobachtung, dass unter Windows das Bild in der Regel deutlich schneller wieder angezeigt wird. Dafür setzt der Ton unter Linux etwas früher ein, trotz höherer Werte für die verlorenen Puffer.

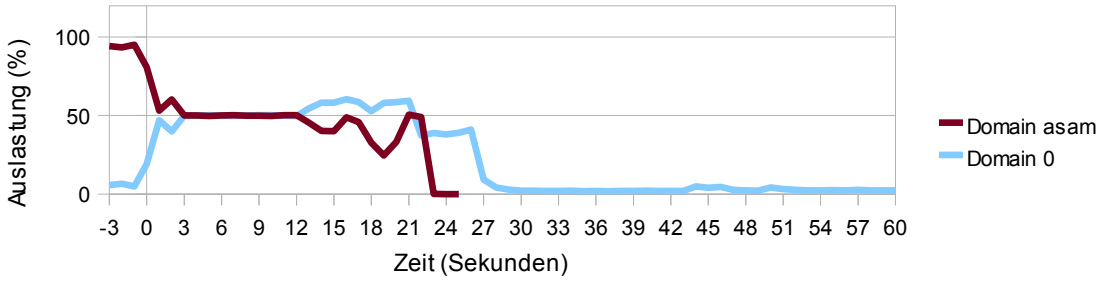

Abbildung 4.55: Streaming Test mit Memtester: Prozessor Auslastung auf Kirsche (xentop)

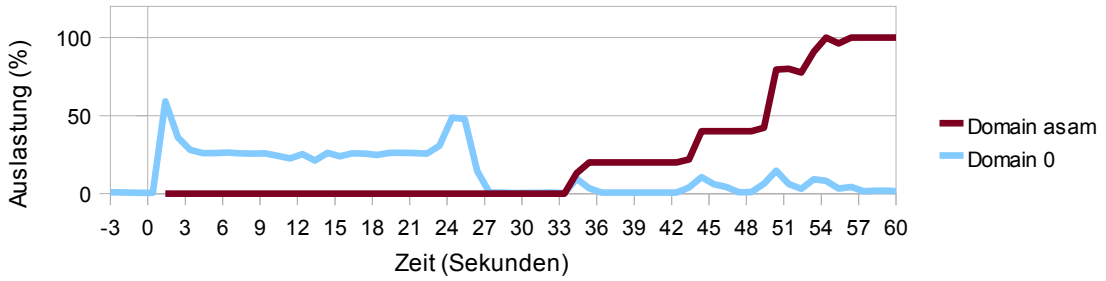

Abbildung 4.56: Streaming Test mit memtester: Prozessor Auslastung auf Apfel (xentop)

Diese interessanten Unterschiede zwischen Linux und Windows zeigen vor allem die hohen Anforderungen an den Client nach der Migration. Während der VLC Server weitgehend unbeeindruckt von der Migration seinen Dienst wieder aufnimmt, müssen die VLC Clienten nach dem Ausfall ziemlich aufwendig wieder eine synchrone Wiedergabe von Bild und Ton gewährleisten. Die eingesetzten Codecs unter Windows und Linux arbeiten dabei anscheinend ziemlich unterschiedlich effizient. Dass der Server keine Mehrbelastung bei Wiederaktivierung der VM erfährt, zeigte auch schon Abbildung 4.45 im Streaming Test ohne Last. In Verbindung mit Memtester allerdings benötigt der Apfel Rechner wieder ziemlich viel Zeit, bis er die VM aktiviert. In Abbildung 4.56 ist schön zu erkennen, wie etwa 23 Sekunden nach Start der Migration die pre-copy Phase endet und alle restlichen noch schmutzigen Seiten so schnell wie möglich übertragen werden. Dies ist spätestens 4 Sekunden später beendet. Es dauert jedoch noch weitere 7 Sekunden bis die VM im schnellsten Testdurchlauf startet. Durch die hier auftretende lange pre-copy Phase in Verbindung mit den langen Downzeiten sind für die Diagramme größere Wertebereiche für die Abszissen notwendig. Neben der schon erwähnten Abbildung zeigt auch Abbildung 4.55 den üblichen starken Einfluss von Memtester auf die CPU. Während der gesamten pre-copy Phase ist allerdings beim Clienten noch eine problemlose Wiedergabe des Videostreams möglich und zumindest hier von der bereits laufenden Live Migration noch nichts zu bemerken.

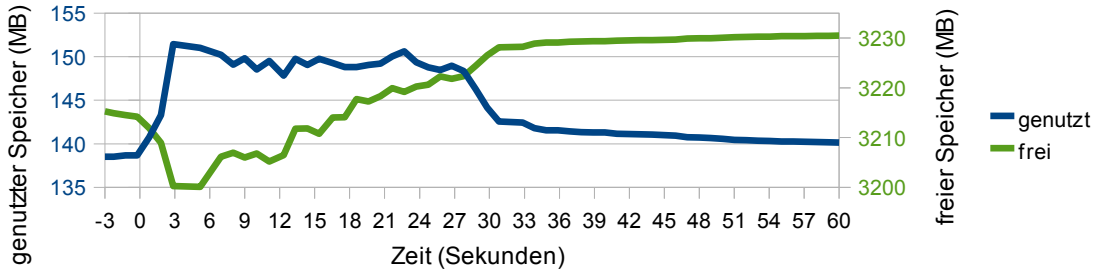

Abbildung 4.57: Streaming Test mit Memtester: Speichernutzung auf Kirsche

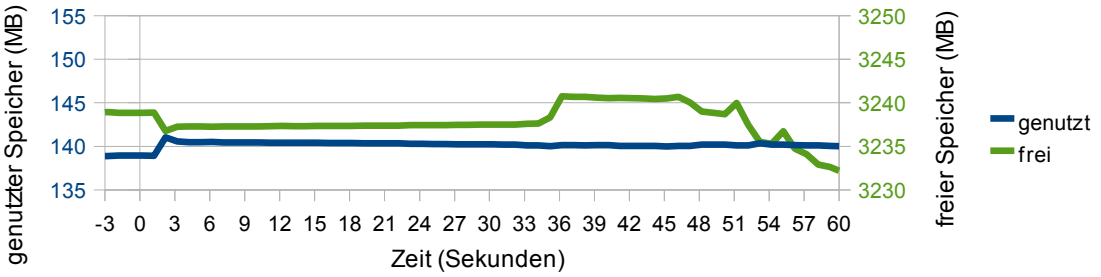

Abbildung 4.58: Streaming Test mit Memtester: Speichernutzung auf Apfel

Wie schon im Pingtest wird der zusätzliche Speicherverbrauch durch Memtester nicht protokolliert. Die Abbildungen 4.57 und 4.58 für den Speicherverbrauch der Domain 0 zeigen daher ein ähnliches Bild. Maximal 12,95 MB zusätzlicher Speicher wird hier für die Migration auf Kirsche benötigt, Apfel begnügt sich mit 2,15 MB.

4.2.3 Video-Stream mit Netzwerklast

Ein sehr unterschiedliches Bild zum Pingtest liefert nun der Streaming Test mit Netzwerklast. Das Netzwerk wird natürlich durch den Videostream stärker belastet als es durch den Pingtest der Fall war, wie in Kapitel 4.2.1 aber schon beschrieben, ist der durch den Stream verursachte Netzwerkverkehr jedoch ziemlich gering. Dennoch beeinflusst diese Kombination die Migration erheblich. Schon anhand der Abbildung 4.59 erkennt man deutlich den langen Einbruch in der Netzwerkübertragung zwischen der 12. und 19. Sekunde nach Migrationsbeginn. Anfangs zeigt die Messung noch die üblichen, auch aus dem Pingtest bekannten Werte. So wird vor Beginn der Migration das Netz mit etwa 30 MB/s belastet. Etwa 3 Sekunden nach Start der Migration schlägt die Auslastung bis auf 53,6 MB/s aus und senkt sich anschließend stetig ab, bis sie zum oben genannten Zeitpunkt Null erreicht. Auch auf Apfel lässt sich dieser lange Komplettausfall beobachten, allerdings ohne dem auf Kirsche gemessenen vorhergehenden Ausschlag (Abbildung 4.60).

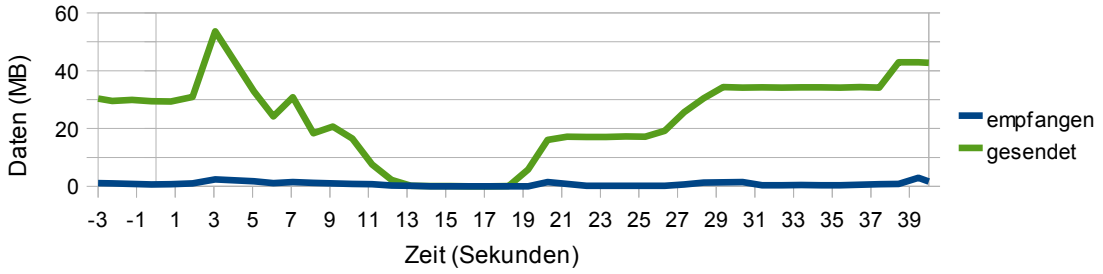

Abbildung 4.59: Streaming Test mit Netperf: Netzwerkauslastung auf Kirsche

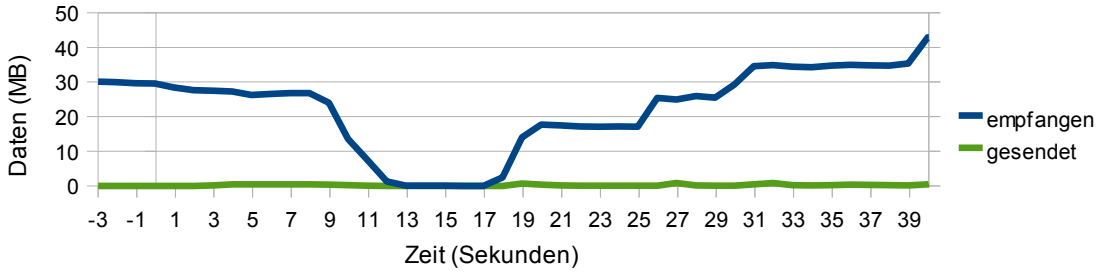

Abbildung 4.60: Streaming Test mit Netperf: Netzwerkauslastung auf Apfel

Die anschließend bei beiden Graphen auftretenden treppenartigen Anstiege spiegeln wieder einmal die recht unterschiedlichen Downtimes wider, welche in Tabelle 4.61 aufgelistet sind. Die Downtime ist hier weit entfernt von den noch recht guten Werten im Pingtest und streut zudem stark. Die Dauer für den Migrationsprozess dagegen ist mit durchschnittlich 10,57 Sekunden nur minimal länger als im Pingtest und mit 1,17 Sekunden Standardabweichung auch erstaunlich gering streuend.

Entsprechend hoch sind auch die Bild und Ton Ausfälle. Interessanterweise scheint Windows hier aussergewöhnlich starke Probleme zu bekommen und verliert mehr Frames als dies im Speichertest mit noch höheren Downtimes der Fall war. Dies war schon während der Tests beim Beobachten des Streams auffällig und bestätigt sich hier. Ebenfalls auffällig

	Test 1	Test 2	Test 3	Test 4	Test 5
zeit Skript Ausfallzeit	17,11	28,24	7,86	21,45	8,67
verlorene Frames (Linux)	406	585	208	602	196
verlorene Frames (Windows)	28	28	9	33	13
verlorene Puffer (Linux)	630	1226	349	979	298
verlorene Puffer (Windows)	532	480	336	624	207

Abbildung 4.61: Ausfallzeit und Streaming Fehler mit Netperf

sind die nach Abschluss der Migration auftretenden Software Interrupts. Abbildung 4.62 zeigt, wie Interrupts über die gesamte Laufzeit von Netperf auftreten. Dies deckt sich jedoch noch mit den Ergebnissen aus dem Pingtest mit Netperf (Abbildung 4.26). Auf Apfel traten ausserhalb der Migrationszeiten jedoch bisher keine Software Interrupts auf. Der Graph in Abbildung 4.63 zeigt nun aber genau dies. Während vor der Migration trotz laufendem Netperf Benchmark wie üblich keine Interrupts auftreten, kommt es in diesem Test auch nach dem Abschluss der Migration weiterhin zu Interrupts, die im Streaming Test ohne Last nicht auftraten. Dies ist ein Indiz für die hier auftretenden Probleme im korrekten Wiederaufnehmen des Videostreams.

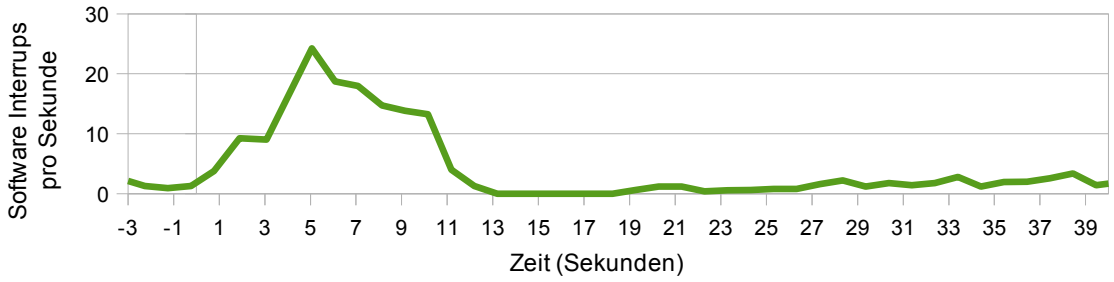

Abbildung 4.62: Streaming Test mit Netperf: Software Interrupts auf Kirsche

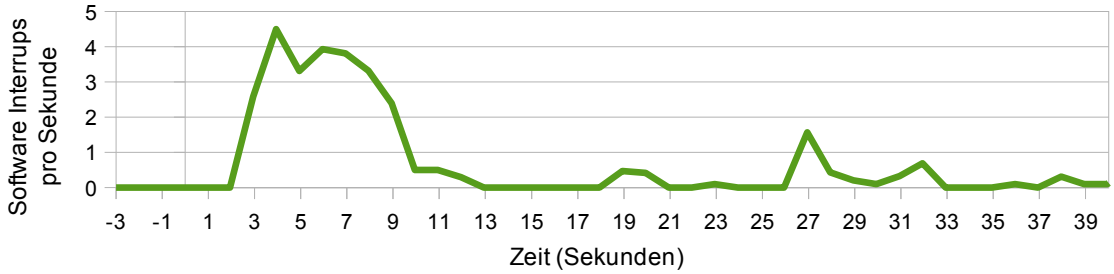

Abbildung 4.63: Streaming Test mit Netperf: Software Interrupts auf Apfel

Die CPU der Domain 0 ist auch in diesem Test wieder über die gesamte Testdauer stark beansprucht, wie in den Abbildungen 4.64 und 4.65 zu sehen ist, jedoch unterscheiden sich die Werte nicht besonders von denen aus dem Pingtest. Da der Stream Test zwischen Kirsche und Apfel durchgeführt wird, bleibt Kirsche auch nach der Migration belastet, genauso wie Apfel schon vor der Migration belastet ist. Jedoch bleibt die Last auf moderatem Niveau, weshalb die CPU hier nicht das limitierende Element darstellt.

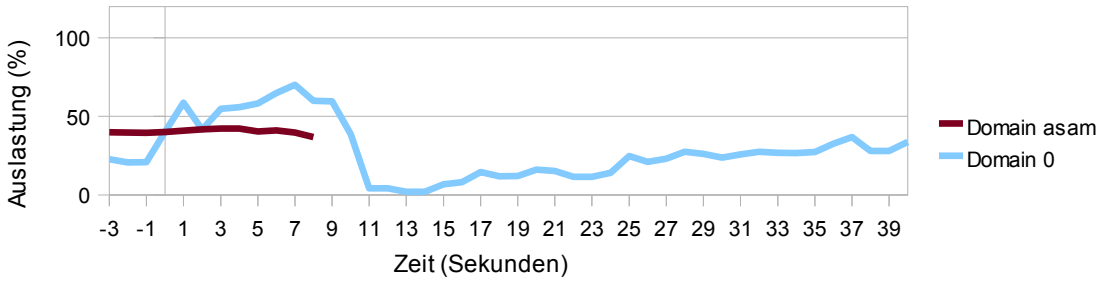

Abbildung 4.64: Streaming Test mit Netperf: Prozessor Auslastung auf Kirsche (xentop)

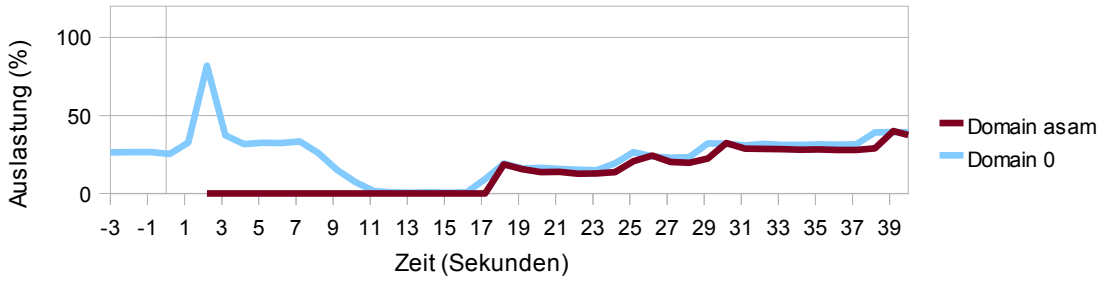

Abbildung 4.65: Streaming Test mit Netperf: Prozessor Auslastung auf Apfel (xentop)

4.2.4 Video-Stream mit CPU Last

Als ein Test mit ziemlich starken Einfluss, erwies sich der Test mit cpuburn-in bereits im Pingtest. Wie gehabt zeigt sich dies am deutlichsten bei den Messungen von xentop. Unterschiede zum Pingtest sind aber auch hier marginal.

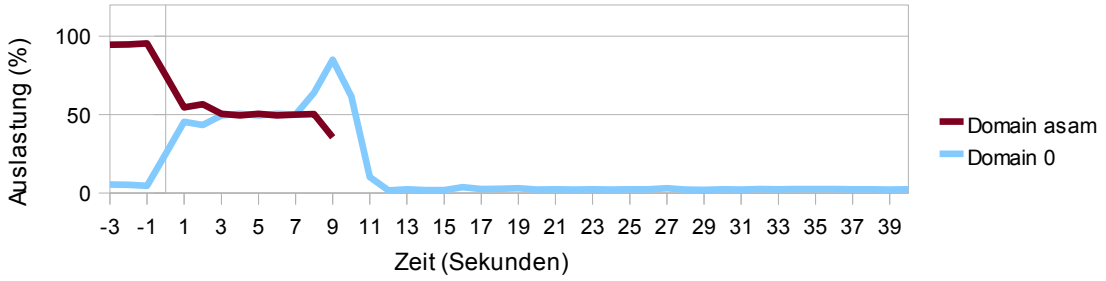

Abbildung 4.66: Streaming Test mit CPU Burn-in: Prozessor Auslastung auf Kirsche (xentop)

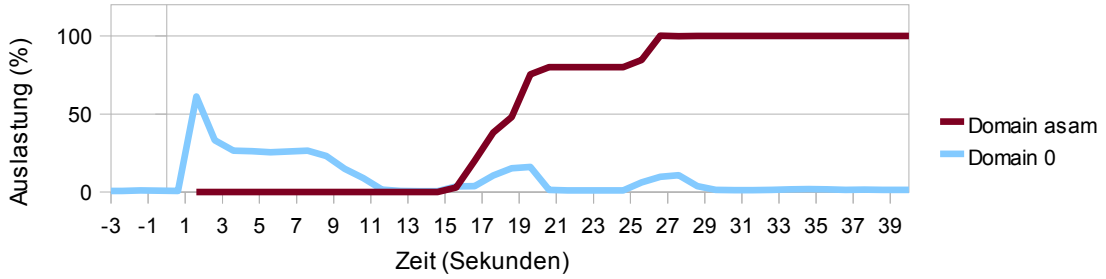

Abbildung 4.67: Streaming Test mit CPU Burn-in: Prozessor Auslastung auf Apfel (xentop)

So dauert es auf Kirsche nur eine Sekunde länger, bis die Auslastung der Domain 0 auf 0% sinkt und die Domain asam existiert zwei Sekunden länger (Abbildung 4.66). Die Graphen verlaufen ansonsten sehr ähnlich. Auf Apfel schlagen sie für die Domain 0 lediglich ein bisschen deutlicher aus, während der Graph für die Domain asam steiler ansteigt und damit den ähnlicheren Werten für die Downtime Rechnung trägt (Abbildung 4.67). Der letzte und deutlich spätere Anstieg zu den 100% ist dem einen Ausreißer in Test 2 zu zuschreiben, welcher mit fast 16 Sekunden Downtime doppelt so hoch liegt wie der Rest und Tabelle 4.68 entnommen werden kann.

	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**
zeit Skript Ausfallzeit	7,14	15,98	8,45	8,33	8,57
Dauer Migrationsprozess	10,08	11,86	9,72	12,30	11,75
verlorene Frames (Linux)	165	429	237	226	240
verlorene Frames (Windows)	14	19	10	13	73
verlorene Puffer (Linux)	250	732	358	337	366
verlorene Puffer (Windows)	144	791	184	116	238

Abbildung 4.68: Ausfallzeit und Streaming Fehler mit cpuburn-in

Diese Tabelle zeigt noch eine Reihe weiterer interessanter Werte. Die Downtime liegt hier mit 9,69 Sekunden im Mittel überraschend meistens deutlich unter den Werten der gleichen Testreihe im Pingtest mit 13,14 Sekunden. Bei beiden Tests kann es aber zu erheblichen Differenzen bei den Werten der einzelnen Tests kommen. Die wesentlich höhere Downtime als im Streaming Test ohne Last erzeugt auch mehr Schwierigkeiten bei der korrekten Wiederaufnahme der Wiedergabe. Die Werte für die verlorenen Frames und die verlorenen Puffer sind wesentlich höher als im Test ohne Last, befinden sich aber unter den im Streaming Test gemessenen Werten im Mittelfeld.

Einen einzigartigen Verlauf nimmt hier der Graph für die Software Interrupts, allerdings nur auf dem Apfel Rechner. Während der Graph dafür auf dem Kirsche Rechner den üblichen Verlauf nimmt (Abbildung 4.69), schlägt er auf dem Apfel Rechner auch nach Abschluss der Migration ein paar mal deutlich aus, allerdings beläuft sich die Anzahl der Software Interrupts pro Sekunde mit maximal drei auf gewohnt niedrigem Niveau für den Rechner Apfel. Nach Wiederaktivierung der VM löst der VLC Server einige in Abbildung 4.70 gezeigte Software Interrupts aus, die so bisher noch nicht auftraten.

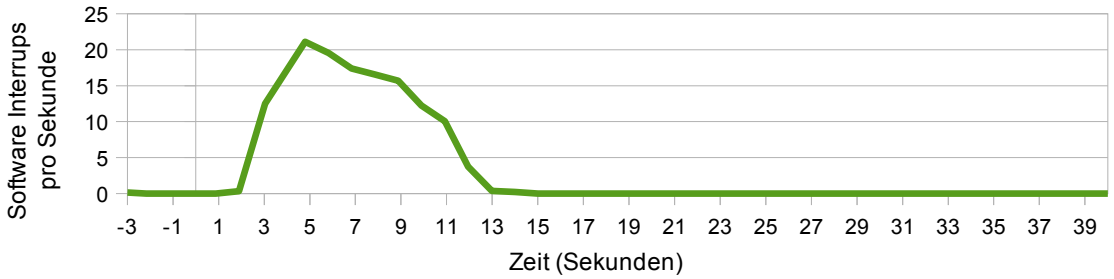

Abbildung 4.69: Streaming Test mit CPU Burn-in: Software Interrupts auf Kirsche

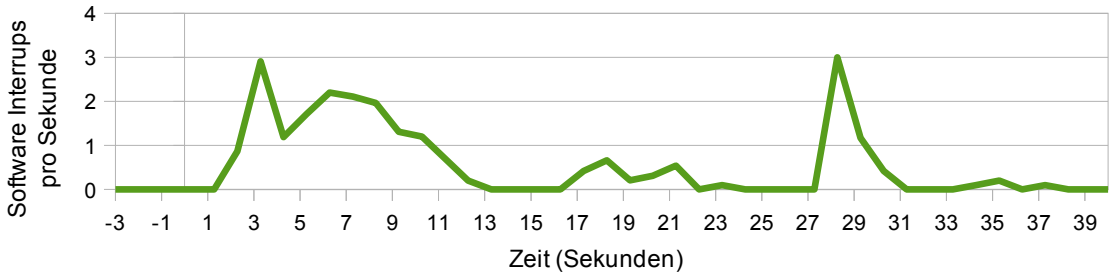

Abbildung 4.70: Streaming Test mit CPU Burn-in: Software Interrupts auf Apfel

4.2.5 Video-Stream mit I/O Last

Der einzige Benchmark mit dem trotz laufendem Videostream kurze Downtimes zu erreichen waren, ist Iometer. Wie Tabelle 4.71 zu entnehmen, sind dabei Werte von knapp über einer Sekunde erreichbar, die manche Werte im Test ohne Last übertreffen. Allerdings kann es auch über 2,5 Sekunden dauern. Einhergehend mit der im Allgemeinen geringfügig längeren Downtime (im Durchschnitt 0,6s länger) ergeben sich verhältnismäßig geringe Ausfälle bei

Frames und Puffer. Wie schon im Test ohne Last fallen die Werte für die verlorenen Frames und Puffer im Test mit der kürzesten Downtime auffallend hoch aus.

	Test 1	**Test 2**	**Test 3**	**Test 4**	**Test 5**
zeit Skript Ausfallzeit	2,66	2,58	1,22	1,01	1,08
verlorene Frames (Linux)	64	14	5	23	7
verlorene Frames (Windows)	16	9	10	8	8
verlorene Puffer (Linux)	90	18	2	26	2
verlorene Puffer (Windows)	156	80	65	93	64

Abbildung 4.71: Ausfallzeit und Streaming Fehler mit Iometer

Trotzdem belastet die I/O Last die Live Migration und den dabei laufenden Videostream mit Abstand am geringsten. Die Auswirkungen von Iometer sind auch hier deutlich in den Abbildungen 4.72 und 4.73 anhand des grünen Graphen für den wartenden CPU Status erkennbar, allerdings stört dies offensichtlich VLC nicht weiter. Der Zugriff auf das Videofile funktioniert trotzdem noch reibungslos.

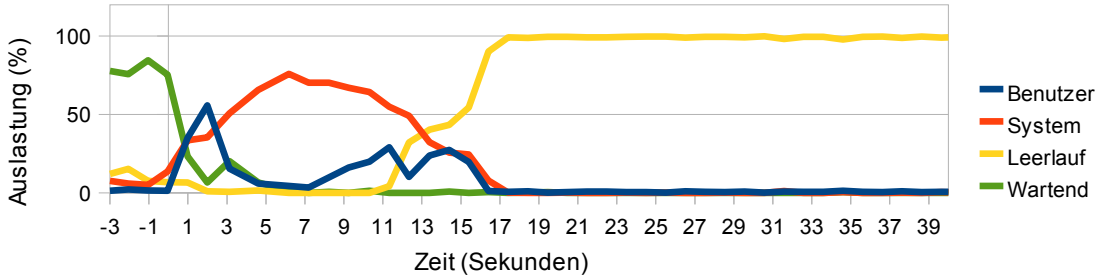

Abbildung 4.72: Streaming Test mit Iometer: Prozessor Auslastung auf Kirsche (dstat)

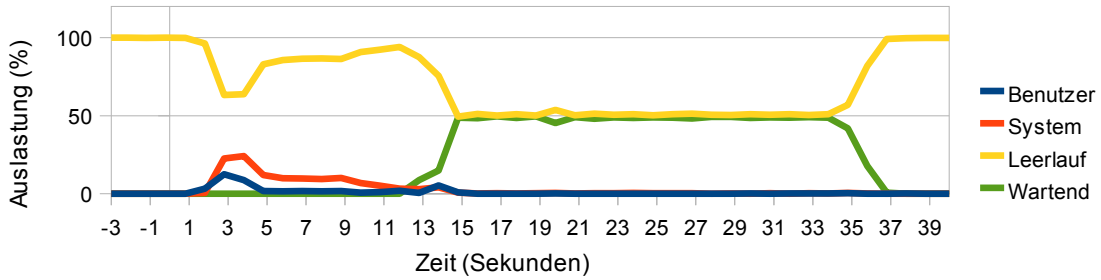

Abbildung 4.73: Streaming Test mit Iometer: Prozessor Auslastung auf Apfel (dstat)

Dies wohl auch aufgrund stärkerer Speichernutzung. Der grüne Graph aus Abbildung 4.74 für den freien Speicher weist ein auffallend starkes Auf und Ab auf, zumindest solange die VM noch auf Kirsche existiert. Auf Apfel wird ab dem Zeitpunkt der Aktivierung der VM stetig mehr Speicher gebraucht, bis der Iometer Testlauf beendet ist (Abbildung 4.75). Da beides nur anhand des Graphen für den freien Speicher feststellbar ist und der für den genutzten Speicher kaum ausschlägt, wird hier, wie schon im Pingtest, überwiegend der Cache benutzt. Weil die Ausschläge hier aber noch einmal deutlicher sind und um den

Cache Gebrauch einmal zu illustrieren, ist dieser in den Abbildungen ausnahmsweise über den gelben Graph extra dargestellt. Darunter leidet allerdings etwas die Darstellung des genutzten Speichergebrauchs, da sich beide Graphen die linke Ordinate teilen.

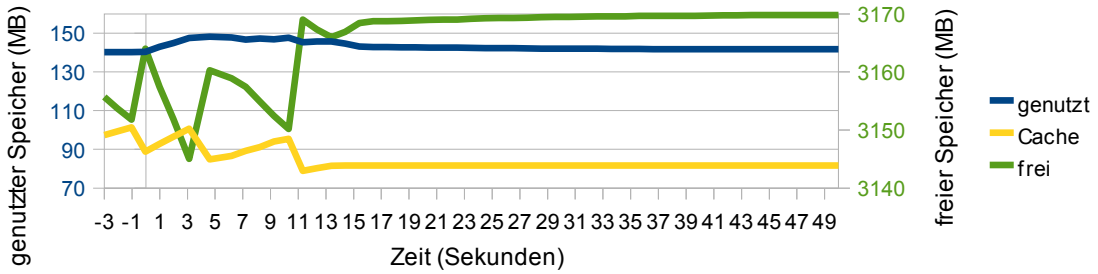

Abbildung 4.74: Streaming Test mit Iometer: Speichernutzung auf Kirsche

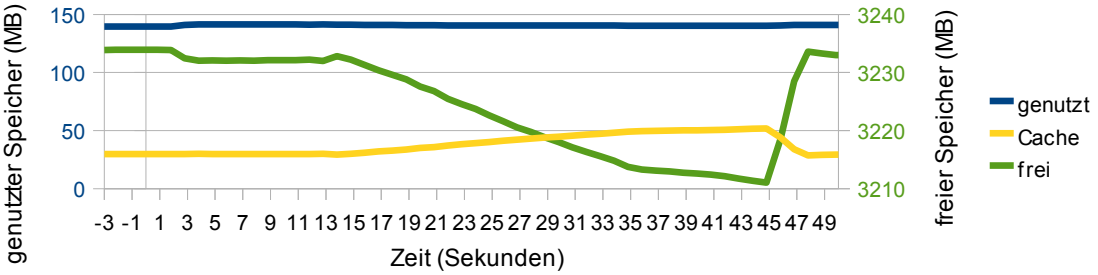

Abbildung 4.75: Streaming Test mit Iometer: Speichernutzung auf Apfel

5 Zusammenfassung

Alle hier durchgeführten Tests zeigen, dass Echtzeitanwendungen durch eine Live Migration so stark behindert werden, dass die Migration Nutzern auffallen muss. Selbst in dem optimalen Fall ohne Last erreicht die Downtime keine Werte unterhalb 0,7 Sekunden und der beste gemessene Wert liegt mit 0,69 Sekunden nur knapp darunter. Die publizierten Downtimes aus dem Paper 'Live Migration of Virtual Machines' werden damit deutlich verfehlt, trotz ähnlicher Hardware.

Aber selbst wenn die reine Downtime 0,25 Sekunden und weniger betragen würde, lassen die Erkenntnisse aus dem Streaming Test erwarten, dass die Clienten noch zusätzlich einige Zeit benötigen, um den weiteren Betrieb wieder aufzunehmen. Diese Zeiten wurden in dem Paper schlicht nicht genannt, bzw. wurden Dienste getestet, die diese Problematik nicht aufweisen. In dieser Hinsicht ist ein Video Stream sicher ein aussergewöhnlich fordernder Test. Trotzdem ist die Live Migration auch mit den hier gemessenen Downtimes eine äusserst nützliche Funktionalität der Virtualisierung und mit vielen Diensten problemlos einsetzbar.

Die Testergebnisse zeigen durchaus sehr interessante Abhängigkeiten von spezifischen Lasten. So sind I/O Lasten in keinem Test problematisch und erzeugen minimal höhere Downtimes als ohne Last. Andererseits haben CPU und Speicher Lasten sehr starken Einfluss und erhöhen die Downtime erheblich. Mit den Ergebnissen unter Speicher Last muss man allerdings vorsichtig sein, da der Speichertest auch gleichzeitig hohe CPU Lasten erzeugt und nicht isoliert durchführbar ist. Die Tests mit Netzwerklast zeigen schön, wie unterschiedlich Dienste mit Lasten umgehen können. Im Ping Test reiht sich die Testreihe mit Netzwerklast von den Downtimes her bei den guten Werten der I/O Tests ein, während sich beim Streaming Test schlechtere Werte als unter CPU Last ergeben. Dass ein Videostream stärker auf Netzlast reagieren würde, war vorherzusehen, dass der Einfluss aber derart dramatisch ausfällt, überrascht dann schon. In allen Tests dagegen erweist sich der Apfel Rechner als weniger ausgelastet als der Kirsche Rechner, was den Nutzen von mehreren Prozessor Kernen auch für Live Migrationen unterstreicht. Auch wenn der eigentliche Migrationsprozess daraus scheinbar keine Vorteile ziehen kann, hilft es, die parallelen VM Lasten während der Migration besser zu verarbeiten.

Ein großes Problem stellen die teilweise recht unterschiedlichen Ergebnisse innerhalb eines Tests dar. Während im Pingtest die Ergebnisse noch relativ nah beeinander liegen, driften sie im Streaming Test teilweise sehr stark auseinander. Da die Ausgangssituation immer gleich bleibt, lassen sich diese Diskrepanzen schlecht erklären. Auffallend war lediglich, dass die ersten Migrationen oft längere Downtimes lieferten als folgende. Dies mag daran liegen, dass der Zielhost beim ersten Mal noch nicht bekannt war oder die für die Migration nötigen Prozesse noch nicht im Hauptspeicher geladen waren.

Völlig unerklärlich bleibt der Zusammenhang der Downtimes mit dem zeit Skript. Warum eine im Leerlauf befindliche VM wesentlich mehr Zeit benötigt zu migrieren als eine, die alle 0,01 Sekunden einen Zeitstempel protokolliert, ist nicht nachzuvollziehen. Als das zeit Skript per simpler Schleife nach einer bestimmten Anzahl an Zeitstempel Einträgen automatisch beendet werden sollte, scheiterten sogar alle Migrationsversuche. Hier handelt es sich

womöglich um einen Bug in der hier verwendeten Xen Version.

Bis auf dieses merkwürdige Verhalten liefen jedoch der Großteil der Tests ohne Probleme, so dass eine Live Migration durchaus als stabil und sicher bezeichnet werden kann. Und die hier gewonnen Erkenntisse bezüglich des Einflusses bestimmter Ressourcen auf Migrationen können in jedem Fall die Auswahl zu migrierender VMs optimieren. Eine unter starker Speicher oder CPU Last stehende VM ist somit eher ein schlechter Migrationskandidat.

Die Live Migration selbst benötigt durchaus über die gesamte Dauer der Migration nicht unerhebliche Ressourcen, womit sie ein System, das bereits unter starker Last steht, zumindest für diesen Zeitraum zusätzlich stark belastet. Allerdings konnten ausserhalb der Downtimes nur geringe, für den Nutzer kaum feststellbare Beeinträchtigungen der getesteten Dienste festgestellt werden, da die migrierende VM die einzige VM auf den Servern darstellte und so die Server ohne weitere Last liefen.

Interessant wäre in diesem Zusammenhang, die Auswirkungen beim Einsatz mehrerer VMs und hoch ausgelasteter Server mit gleichzeitiger Last auf mehreren Ressourcen zu untersuchen. Ebenso wäre es sinnvoll, noch weitere Dienste zu testen, um so ein differenzierteres Bild erstellen zu können, welche Arten von Diensten wie stark von einer Live Migration beeinträchtigt werden.

Abbildungsverzeichnis

Literaturverzeichnis

[BDF+] Barham, Paul, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt und Andrew Warfield: *Xen and the Art of Virtualisation*. http://www.cl.cam.ac.uk/research/srg/netos/papers/2003-xensosp.pdf.

[bem04] bemowski: *ssh shared-key authentication setup*, September 2004. http://jmatrix.net/dao/case/case.jsp?case=7F000001-1777B1-FF02AC7A18-E7.

[BI10] Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e.V.: *IT- und Telekommunikations-Trends 2010*, Januar 2010. http://www.bitkom.org/files/documents/BITKOM-Presseinfo_IT-Trends_2010_-_13_01_2010.pdf.

[Caz09] Cazabon, Charles: *memtester*, 2009. http://pyropus.ca/software/memtester/.

[CFH+05] Clark, Christopher, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt und Andrew Warfield: *Live Migration of Virtual Machines*, 2005. http://www.cl.cam.ac.uk/research/srg/netos/papers/2005-migration-nsdi-pre.pdf.

[HFV96] Hartvigsen, G., V. Farsi und B. Vinter: *The Virtual Secretary Project: Some Parts of the Project's Theoretical Background*, Februar 1996. http://www.vise.cs.uit.no/vise/publications/reports/tr-95-02.pdf.

[Hol07] Holzer, Jan Mark: *Bug 243934 - Provide summary fields for CPU and Memory utilization in xentop command*, Dezember 2007. https://bugzilla.redhat.com/show_bug.cgi?id=243934.

[Jon09] Jones, Rick: *The Netperf Homepage*, 2009. http://www.netperf.org/netperf/.

[Lip] Lipinski, Klaus: *IT-Lexikon: Virtualisierung*. http://www.itwissen.info/definition/lexikon/Virtualisierung-VT-virtualization-technology.html.

[Mie] Mienik, Michal: *CPU Burn-in Instructions (readme)*. http://www.eatsushi.org/content/dump/dox/cpuburn-in.tar.gz.

[oCCL08] Cambridge Computer Laboratory, University of: *The Xen virtual machine monitor*, 2008. http://www.cl.cam.ac.uk/research/srg/netos/xen/.

[QDS+] Querol, Carlos, Dan Bar Dov, Daniel Scheibli, Joe Eiler, Ming Zhang, Richard Riggs, Rick Altherr, Thayne Harmon und Vedran Degoricija: *Iometer project*. http://www.iometer.org/.

[Ra] RA, EUGENE: *Operating Systems and Benchmarks - Part 5: Storage-Review.com's IOMeter Tests*. http://www.storagereview.com/articles/200003/20000313OSandBM_5.html.

[Rad06] RADONIC, ANDREJ: *Xen 3 wird zur VMware-Alternative*, Januar 2006. http://www.computerwoche.de/software/software-infrastruktur/571065/index.html.

[SK06] SCHUBERT, KLAUS und MARTINA KLEIN: *Das Politiklexikon*, 2006. http://www1.bpb.de/popup/popup_lemmata.html?guid=LYFP96.

[Sti07] STILLER, ANDREAS: *Xen und die virtuelle Zeit*. c't, Heft 26:180, 2007.

[Tea] TEAM, VIDEOLAN: *VideoLAN - VLC media player*. http://www.videolan.org/.

[Wie] WIEERS, DAG: *All you ever wanted to know about counter-rollovers and dstat*. http://svn.rpmforge.net/svn/trunk/tools/dstat/docs/counter-rollovers.txt.

[Wie08] WIEERS, DAG: *Dstat: Versatile resource statistics tool*, Dezember 2008. http://dag.wieers.com/home-made/dstat/.