Diplomarbeit

„Entwicklung einer Netzwerkboot-Umgebung für Thin-Clients“

Ausgeführt zum Zweck der Erlangung des akademischen Grades eines
Dipl.-Ing. (FH) für Telekommunikation und Medien
am Fachhochschul-Diplomstudiengang Telekommunikation und Medien St. Pölten
Vertiefungsrichtung Telekommunikation

unter der Leitung von

Bakk.rer.soc.oec. Bernhard R. Fischer

ausgeführt von

Hans-Peter Brandstätter

tm0210038017

St. Pölten, am 08. September 2006 Unterschrift:

Ehrenwörtliche Erklärung

Ich versichere, dass

- ich diese Diplomarbeit selbständig verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich auch sonst keiner unerlaubten Hilfe bedient habe.
- ich dieses Diplomarbeitsthema bisher weder im Inland noch im Ausland einem Begutachter/in zur Beurteilung oder in irgendeiner Form als Prüfungsarbeit vorgelegt habe.

Diese Arbeit stimmt mit der vom Begutachter beurteilten Arbeit überein.

Ort, Datum Unterschrift

Kurzfassung

Durch die ständig steigende Komplexität von dezentral verwalteten Computernetzen, besonderes in Mittel- und Großbetrieben, steigt auch die damit verbunden Fehleranfälligkeit solcher Systeme. Aber nicht nur durch den erhöhten Administrationsaufwand, der meist recht autonomen Computerarbeitsplätze, nehmen jährlich die Kosten der IT-Abteilungen zu, sondern auch durch die stetig wachsenden Anforderungen an die Leistungsfähigkeit der Arbeitsplatzrechner und die damit einhergehenden kürzeren Austauschzyklen dieser Computer. Alternativ zu den dezentralen Computernetzen, gibt es die Möglichkeit die Betriebssysteme und Anwendungsprogramme zentral zu verwalten und bereitzustellen und die Arbeitsplätze mit plattenlosen Computern auszustatten, welche sich während des Systemstarts die benötigte Software eigenständig über das Computernetzwerk laden.

Um einen Überblick über das umfangreiche Themengebiet von *Thin-Clients* zu geben, werden unterschiedliche Endgeräte, Server, Bootkonzepte und Protokolle kurz vorgestellt.

Weiters werden die Protokolle *ICA (Citrix)*, *RDP (Microsoft)* und *X (Linux/Unix)*, welche häufig in Verbindung mit Terminals eingesetzt werden, etwas genauer erörtert und anschließend einem Performancevergleichstest unterzogen, mit einem Hauptaugenmerk auf die dabei auftretende Netzwerklast.

Abschließend in dieser Arbeit, wird ein Grobkonzept für den Einsatz von Thin-Clients in der Austria Presse Agentur vorgestellt.

Abstract

By the constantly rising complexity of decentralized administered computer networks, especially in medium-sized and large-scale enterprises, rises also the fault liability of such systems connected with it. But not only by the increased administration expenditure of the usually quite autonomous workstations the annual costs of the IT-departments increase, but also by the constantly growing requirements to the efficiency of the personal computers and the shorter exchange cycles of these computers. Alternatively to the decentralized computer networks, there is the possibility to administer and to provide the operating system and applications programs from a central place and to equip the jobs with diskless-computers, which load the necessary software independently over the computer network during the system start.

In order to give an overview of the extensive topic area *Thin-Clients*, different terminals, server, boot concepts and protocols are briefly introduced.

Further the protocols *ICA (Citrix)*, *RDP (Microsoft)* and *X (Linux/Unix)*, which are frequently used in connection with terminals, become somewhat more exactly discussed and afterwards submitted to a performance comparison test, with a special attention on the arising network load.

Finally in this work, a rough conception for the deployment of Thin-Clients in the Austria Press Agency will be presented.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ADS	Active Directory Service
APA	Austria Presse Agentur
API	Application Programming Interface
BIOS	Basic Input/Output System
BSD	Berkeley Software Distribution
CPU	Central Processing Unit
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DOS	Disk Operating System
FDDI	Fiber Distributed Data Interface
FIPS	Federal Information Processing Standard
GDI	Graphics Device Interface
GDM	Gnome Display Manager
GPL	GNU General Public License
GRUB	Grand Unified Bootloader
HTTP	Hypertext Transfer Protocol
KDE	K Desktop Environment
KDM	KDE Display Manager
LAN	Local Area Network
PC	Personal Computer

ICA	Independent Computing Architecture
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPC	Inter-Process Communication
IPX	Internetwork Packet eXchange
ISDN	Integrated Services Digital Network
ITU	International Telecommunications Union
iSCSI	Internet Small Computer System Interface
SBC	Server Bases Computing
SMTP	Simple Mail Transfer Protocol
SPX	Sequence Packet eXchange
SSH	Secure Shell
IT	Information Technology
LILO	Linux Loader
LBA	Linear Block Array
MAC	Media Access Control
MBR	Master Boot Record
MIT	Massachusetts Institute of Technology
MTFTP	Multicast TFTP
NBP	Network Bootstrap Program
NFS	Network File System
NIC	Network Interface Card

NTLDR	NT Loader
OSI	Open System Interconnection
PDF	Portable Document Format
POSIX	Portable Operating System Interface
POST	Power On Self Test
POTS	Plain Old Telephone Service
PXE	Preboot eXecution Environment
RAM	Random Access Memory
RDC	Remote Desktop Connection
RDP	Remote Desktop Protocol
RFC	Request For Comment
ROM	Read Only Memory
SAN	Storage Area Network
SCSI	Small Computer System Interface
SMB	Server Message Block
SSL	Secure Socket Layer
TCP	Transfer Control Protocol
TDTCP	Transport Driver TCP
TFTP	Trivial File Transfer Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
UNDI	Universal Network Device Interface

USB	Universal Serial Bus
UUID	Universally Unique Identifier
VNC	Virtual Network Computing
WAN	Wide Area Network
WfM	Wired for Management
X, X11	X Window System
XDM	X Display Manager
XDMCP	X Display Manager Control Protocol

1 Einleitung

Die Verwaltung und Betreuung von vernetzten Computerarbeitsplätzen in großen Betrieben gestaltet sich von Jahr zu Jahr immer komplexer und aufwendiger. Während sich die Personal Computer (PC) zu Beginn der Computervernetzung als die beste Lösung etablierten, stellt das Konzept zur Vernetzung von Arbeitsplätzen mit PCs heutzutage bereits ein großes Problem dar, da die Anzahl von Computern in Firmennetzen schnell auf mehrere Hundert steigen kann.

Administratoren betreuen heutzutage nicht mehr nur eine Vielzahl großer rechenstarker Server, sondern müssen immer häufiger eine Fülle verteilter PCs betreuen. Dies bedeutet nicht nur explodierende Budgetkosten durch steigenden Bedarf an Personal zur Betreuung und Wartung, sondern ein ebenfalls steigendes Sicherheitsrisiko für Ihre IT Infrastruktur. Weiters hängen diese PCs in immer leistungsfähigeren Netzen, nützen diese jedoch nur zur Authentifizierung über eine zentrale Benutzerverwaltung sowie für Internet-, Datei- und Druckdienste.

Um die Mitarbeiter in den IT Abteilungen effizient zu entlasten, setzen Betriebe immer häufiger auf alternative Betriebskonzepte, bei denen zentral verwaltete Betriebssysteme und Anwendungsprogramme an plattenlose Rechner bei deren Systemstart über das Firmennetzwerk verteilt werden können. Diese Verfahren, bei dem laufwerkslose Computer nach deren Einschalten das für ihren Betrieb notwendige Betriebssystem von Servern über das Firmennetz beziehen, nennt man Netzwerk-Boot.

Wie bei jeder Betriebsform, sind auch bei über das Netzwerk bootenden plattenlosen Computern Vorteile und Nachteile vorhanden und aufzeigbar. Zu den Vorteilen zählen unumstritten die zentrale Daten- und Softwareverwaltung, an Hand dieser es möglich wird eine Vielzahl von Arbeitsplätzen ohne nennenswerten Mehraufwand von zentraler Stelle aus zu administrieren und zu betreuen. Weiters entsteht bei PCs, bei denen auf eine Festplatte verzichtet wurde, eine geringer Temperatur- und Geräuschemission, was wiederum zu einem angenehmeren Betriebsklima, im speziellen bei Großraumbüros, führen kann. Beschränkt man sich als Administrator überhaupt nur mehr auf den Einsatz von so genannten Terminals, werden nur mehr Rechner benötigt die genug Rechenleistung besitzen um eine Dateneingabe sowie –ausgabe zu bewerkstelligen. Die Datenver-

arbeitung erfolgt somit nicht mehr lokal auf jedem einzelnen Arbeitsplatz, sondern zentralisiert auf einem oder meist in einen ganzen Verbund zusammengeschlossener Server. Neben vielen Vorteilen gibt es aber auch einige Nachteile. Probleme können durch die gesteigerte Abhängigkeit von Servern und vom Netzwerk, sowie durch einen erhöhten Ressourcenbedarf auf der Seite der Server und eines eventuellen steigenden Netzwerkverkehrs zwischen dem Server- und Clientsystems entstehen. Die Abwägung der einzelnen Vor- und Nachteile muss aber für jede IT Landschaft von den entsprechenden Entscheidungsträgern selbst erfolgen und kann nicht pauschaliert durchgeführt werden.

So wie heutzutage bereits eine große Vielzahl an unterschiedlichen Betriebssystemen am Markt verfügbar sind, gibt es ebenfalls ein breites Spektrum an verschiedenen Netzwerk-Boot-Konzepten. Angefangen bei der BootROM-Technologie, wie sie bei verschiedenen Projekten wie NetBoot, EtherBoot oder Preboot eXecution Environment (PXE) zum Einsatz kommt, bis hin zur Virtualisierung von lokalen Festplatten, wobei sich das Speichermedium aber tatsächlich auf einem entfernten Storage-Server befindet.

Im ersten Teil dieser Arbeit geht es in erster Linie darum, die verschiedenen Ansätze, Varianten und dahinter liegenden Techniken von plattenlosen Computersystemen, welche über ein Computernetzwerk booten, zu erörtern. Es werden verschiedene Formen von plattenlosen Endgeräten, sowie unterschiedliche Arten von Servern vorgestellt, welche in mannigfachen Zusammenstellungen ein System bilden, um einen netzwerk- und serverbasierten Bootprozess und Produktionsbetrieb ermöglichen. Weiters werden verschiedene am Markt befindliche Produkte und für einen reibungslosen Netzbootvorgang notwendige Programme und Protokolle kurz beleuchtet.

Im dritten Abschnitt findet eine genauere Vorstellung und Gegenüberstellung von so genannten Remote Display Protokollen statt, welche häufig in Verbindung mit Thin-Clients eingesetzt werden, welche nur dazu verwendet werden, um Computerein- und ausgaben zu kontrollieren, wobei die eigentlichen Programme aber auf entfernten Servern laufen. Im anschließenden direkten Vergleich werden die drei zuvor vorgestellten Protokolle ICA (Citrix), RDP (Microsoft) und X (Unix/Linux) auf deren Effizienz bei der Netzwerkübertragung während normalen Computerarbeitstätigkeiten überprüft, wie sie im üblichen Büroalltag auftreten.

Im vierten Kapitel dieser Arbeit wird ein Grobkonzept für den möglichen Betrieb von plattenlosen Computern für die Austria Presse Agentur (APA) erstellt und ausgearbeitet, welches helfen soll, den derzeitigen Administrationsaufwand zu verringern. Darin werden Vorgaben seitens der APA, sowie die momentane Infrastruktur berücksichtigt, um eine für den Benutzer möglichst transparente Migration durchführen zu können.

Im letzten Teil dieser Arbeit werden nochmals alle wichtigen Erkenntnisse dieser Diplomarbeit kurz zusammengefasst und Vorschläge und Ideen gegeben, mit welchen sich zukünftige Arbeiten befassen könnten.

2 Grundlagen

Dieses Kapitel soll ein grobes Verständnis über Computersysteme und -modelle vermitteln, welche im Gegensatz zu anderen herkömmlichen Computern keine lokale Festplatte besitzen, ihr Betriebssystem und die für einen ordentlichen Betrieb weiters notwendigen Programme ausschließlich über ein Netzwerk beziehen.

Dazu wird zu Beginn ein Überblick über die Kategorisierung und über die verschiedenen Varianten von plattenlosen Computern gegeben. Weiters werden in diesem Kapitel Arten von Servern vorgestellt, welche für den Aufbau und Betrieb einer netzwerkbasierten Bootumgebung mit plattenlosen Computern wichtig und notwendig sind. Anschließend werden kurz verschiedene Konzepte vorgestellt, an Hand welcher es möglich ist, plattenlose Computer über ein Netzwerk in Betrieb zu nehmen. Abschließend werden die grundlegenden Protokolle, Dienste und Programme, die für den reibungslosen Betrieb von Netzwerkbootumgebungen typisch und nötig sind, kurz erklärt.

2.1 Einteilung der Endgeräte nach deren Aufgaben

Der Begriff *Thin-Client* deckt ein sehr breit gefächertes Spektrum an verschiedenen Varianten von Computern eines Netzwerksystems ab. Angefangen bei extrem einfach gehaltenen Thin-Clients, welche nur dazu verwendet werden, es einem Benützer zu ermöglichen sich mit deren Hilfe an Terminal Servern einzulogen, bis hin zu Thin-Clients mit bereits lokal installierter Software, die Server nur mehr als Datenspeicher verwenden, oder auf diesen spezielle Programme auszuführen, welche auf den Clients nicht vorhanden sind. Durch den fließenden Übergang zwischen den unterschiedlichen Thin-Client Konzepten und den traditionellen Computern im herkömmlichen Sinn, welche ebenfalls Verbindungen mit Anwendungsservern für gelegentliche spezielle Tätigkeiten verwenden, wird es sehr schwierig klare Grenzen zwischen den verschiedenen Systemen zu ziehen. Unterschiedliche Menschen ziehen diese Trennlinien an unterschiedlichen Orten an Hand unterschiedlicher Definitionen. (vgl. [28])

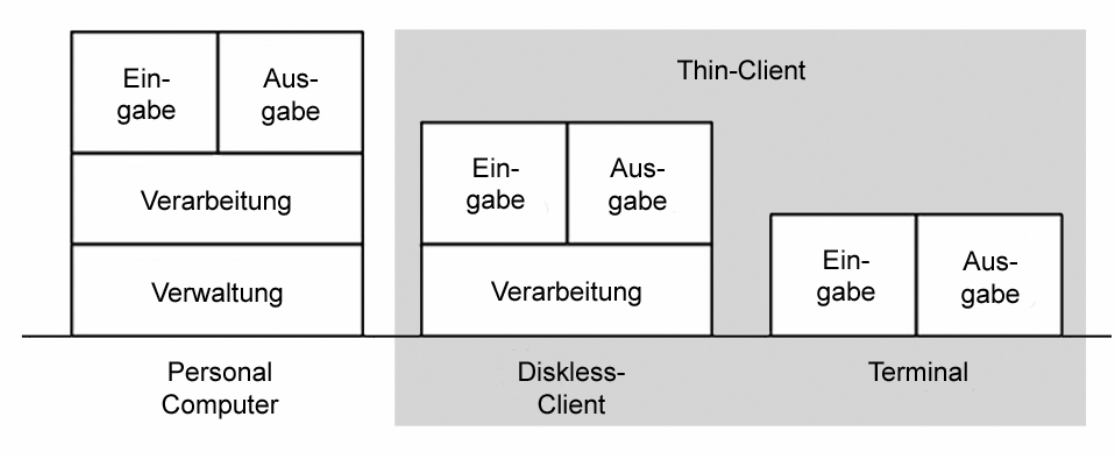

Abbildung 1: Einteilung der Endgeräte nach deren Aufgaben

Innerhalb dieser Arbeit bestimme ich die Anwesenheit oder Abwesenheit einer Festplatte als Kriterium zur Unterscheidung zwischen Thin-Clients und anderen Computerformen. Besitzt ein Computer keine fest integrierte Festplatte in seiner Hardwarekonfiguration, fällt er unter die Kategorie der Thin-Clients.

Wie in Abbildung 1 ersichtlich unterscheiden sich Thin-Clients von herkömmlichen PCs dadurch, dass die Verwaltung des lokalen Systems und der installierten Software, sprich die Speicherung von Konfigurationen sowie sämtlicher zum Betrieb weiterer notwendiger Daten und Programmen, nicht mehr lokal auf dem Endgerät erfolgt.

Auch unter der Gruppe der Thin-Clients gibt es wieder unterschiedliche Konzepte für deren Betrieb.

Bei einer Variante des Thin-Clients erstreckt sich die Aufgabe des Endgerätes von der Abwicklung der Eingaben des Anwenders per Tastatur, Maus oder anderer Eingabegeräte und der Ausgaben per Bildschirm, Drucker oder weiterer Ausgabegeräte bis hin zur Verarbeitung der Daten lokal auf dem Rechner. Dieses Betriebsmodel wird hier als bezeichnet und unter 2.1.2 näher erklärt.

Eine weitere Variante der Thin-Clients sind Computer mit minimaler Hardwareanforderung, deren Aufgabe es ist nur für die Ein- und Ausgabe der Informationen zu sorgen, wobei die tatsächliche Verarbeitung der Informationen auf so genannten Anwendungsservern erfolgt. Solche Computer werden allgemein als *Terminal* bezeichnet und finden unter 2.1.3 eine nähere Betrachtung.

2.1.1 Vor- und Nachteile von Thin-Client Computing

In den folgenden Tabellen wird nur ein kleiner Auszug der vielen Vor- und Nachteile des Betriebskonzeptes von Thin-Clients dargestellt. (vgl. [4])

Tabelle 1: Vorteile von Thin-Client Computing

Vorteile	Erläuterung
Zentrale Verwaltung	Betriebssysteme und Anwendungen können auf einfache Art und Weise zentralisiert auf Servern bereitgestellt werden. Die Pflege der Software kann zentral erfolgen (Einspielen von Patches und Updates). Keine administrativen Arbeiten vor Ort an den Clients mehr nötig. Betreuung der Anwendungen findet zentralisiert statt, die Fehlerdiagnose und Fehlerbehebung wird vereinfacht. Erhöhung der Produktivität für den Anwender und die Administratoren. Durch die vereinfachte Administration erfolgt eine Beschleunigung bei der Bereitstellung von Applikationen zugunsten der Endanwender. Durch die Verringerung der personalintensiven Fehlerbehebungseinsätze vor Ort kann der administrative Mehraufwande reduziert werden.
Verringerte Hardwareanforderungen bei Terminallösung	Die Client-Systeme benötigen weniger Hardwareressourcen (Netzwerkkarte, Grafikkarte, Tastatur, Maus). Ein regelmäßiger Ausbau der Client-Hardware, wegen steigender Anforderungen durch Betriebssysteme oder Anwendungen ist nicht mehr notwendig. Die regelmäßigen Austauschzyklen der Hardware können durch die geringeren Anforderungen vergrößert werden.

Erhöhte Sicherheit	Durch den Einsatz von Thin-Clients wird der Verlust von Daten verringert, weil Daten nur mehr auf zentralen Servern gespeichert werden. Die Angriffspunkte durch Virenattacken werden stark verringert, da auf dem Client-System keine Daten gespeichert werden können.
Unabhängigkeit vom Client	Arbeitsplatzrechner können schnell und problemlos Ausgetauscht werden, da keine persönlichen Daten oder Einstellungen von Anwendern darauf gespeichert sind. Benutzer können die Arbeitsplätze wechseln ohne auf deren persönlichen Einstellungen verzichten zu müssen.

Tabelle 2: Nachteile von Thin-Client Computing

Nachteile	Erläuterung
Abhängigkeit	Server-Sessions werden beim Ausfall des Terminal Servers abgebrochen und die Arbeitsaufnahme ist erst wieder möglich, wenn der Fehler behoben wurde. Durch den Einsatz von Serverfarmen können solche Ausfälle minimiert werden. Beim Abbruch von Server-Sessions kann es zu Datenverlusten kommen.
Erhöhter Ressourcenbedarf	Da alle Anwendungen auf den Terminal Servern laufen, müssen diese über deutlich hohe Ressourcenausstattung verfügen. In Relation zum Gesamtbedarf (Server und Clients) werden jedoch weniger Ressourcen benötigt, da bestimmte Operationen auf einem Server nur einmal für alle Benutzer ausgeführt werden müssen und nicht auf jedem einzelnen Client.

Erhöhter Netzwerkverkehr	Die Kommunikation zwischen den Thin-Clients und den Servern erfolgt auf der Netzwerkebene. Übertragen werden bei Terminalsitzungen die Inhaltsdifferenzen für den Bildaufbau oder die Anweisungen für den Bildaufbau. Durch bestimmte Dienstprogramme (Grafikprogramme) kann es zu einer starken Erhöhung der Netzauslastung kommen. Durch andere Programme (Textverarbeitung) kann sich die Netzlast aber auch verringern.
Anpassungen verwendeter Anwendungen	Leider gibt es auch eine Vielzahl von Programmen, vor allem im Windows-Bereich, welche nicht für einen Terminal Serverbetrieb geeignet sind. Manche diese Probleme können aber durch spezielle Techniken, wie das Applikation Isolation Environment von Citrix wieder umgangen werden.
Mobile Endgeräte	Mobile Endgeräten (Notebooks) sind in den meisten Firmen immer mehr am Vormarsch. Eine Integration dieser Computer in eine reine Thin-Client Umgebung stellt ein großes Problem dar.

2.1.2 Diskless-Client

„*Unter Diskless versteht man einen Computer in dem weder eine Festplatte noch ein CD-ROM, DVD-LW, Floppy oder sonstiges Laufwerk eingebaut ist. Solche Computer können über das Netzwerk booten, z.B. über das PXE Protokoll. (Boot on LAN)*“ [2]

Natürlich gibt es auch Konzepte von s in denen die Verwendung von CD-ROM, DVD-LW, Floppy und andere Laufwerke, sei es ob intern oder extern, unterstützt werden. Diese Laufwerke werden aber in solchen Fällen hauptsächlich zum Sichern und Übertragen von Daten verwendet und nicht zum Booten von Betriebssystem verwendet.

Ein festplattenloser Arbeitsplatzrechner, besitzt wie jeder andere voll ausgestattet Computer einen Prozessor sowie einen Arbeitsspeicher, aber keine integrierte Festplatte. Weiters müssen solche Rechner mit einer Netzwerkkarte zum Anbinden an ein Firmennetzwerk ausgestattet sein. Die Dateneingabe und -ausgabe erfolgt wie üblich über Tastatur, Maus und Bildschirm. Der Grundgedanke besteht darin, das für den Betrieb notwendige Betriebssystem solcher festplattenloser Computer von einem dafür vorgesehenen Bootserver über das Computernetzwerk zu beziehen. Ebenfalls die Anwendungssoftware wird zentral auf Servern verwaltet und von diesen bezogen, aber lokal auf dem Rechner ausgeführt. Die Datenverwaltung und -speicherung muss ebenfalls wegen der fehlenden Speichermöglichkeit vor Ort auf Netzwerkservern durchgeführt werden.

Solche Konzepte lassen sich auf relativ einfache Art und Weise kostengünstig mit UNIX oder Linux Betriebssystemen umsetzen. Dabei wird ein kleiner Bertriebssystemkern per Trivial File Transfer Protocol (TFTP) oder Network File System (NFS) von einem Bootserver geladen und zur Ausführung gebracht. Das Root-Verzeichnis wird entweder mit Hilfe einer RAM-Disk (siehe 2.1.2.2) direkt vollständig in den Arbeitsspeicher geladen, oder per NFS (siehe 2.1.2.1) von einem dafür vorgesehen Server dem Betriebssystem verfügbar gemacht. In Abbildung 2 ist dieses Prinzip in groben Zügen dargestellt. Alle weiteren Dateisysteme können ebenfalls per NFS im Anschluss eingehängt werden.

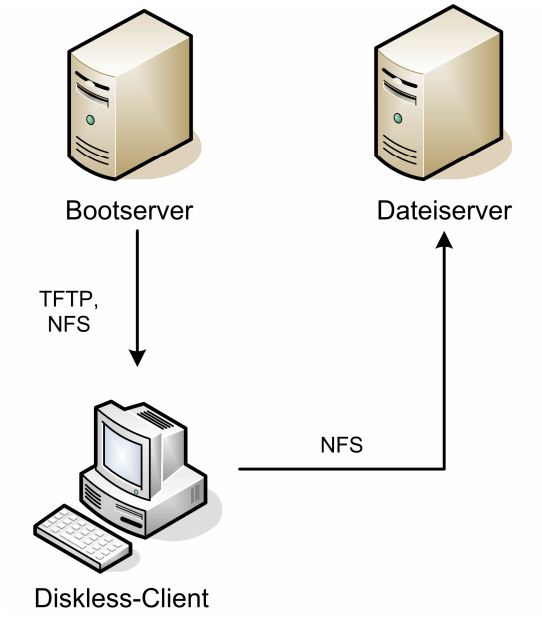

Abbildung 2: Das Diskless-Client Prinzip

Der Vorteil dieser Methode liegt darin, dass der oder die Bootserver nur während des Bootvorganges zum downloaden des Kernels verwendet wird. Das Root-Verzeichnis wird anschließend per NFS entweder ebenfalls vom Bootserver, oder von einem Dateiserver, wie hier in Abbildung 2 zusehen, bereitgestellt, was die Hauptlast für diese Server darstellt. Die ganze Prozesslogik des Systems und der Applikationen läuft aber vor Ort auf den s und belastet somit die Server nicht. Dies hat weiters den Vorteil, dass Betriebe die Möglichkeit einer zentralen Administration nutzen können, weiterhin günstige handelsübliche PCs als Endgeräte einsetzen können und keine riesigen Terminal Server benötigen.

Einen ähnlichen Ansatz verfolgen die Konzepte von Kontron[1] (siehe 2.5.4) oder IBM[2] (siehe 2.5.5), welcher darauf abzielt ein Betriebssystem von einem Serverlaufwerk zu booten. In diesem Fall würde ebenfalls die gesamte Systemlogik lokal im Arbeitsspeicher des Thin-Clients ablaufen.

[1] siehe http://www.kontron.com/products/pdproductdetail.cfm?keyProduct=31731
[2] siehe http://www.haifa.ibm.com/projects/storage/iboot/index.html

2.1.2.1 Root over NFS

In der Unix-Welt sind Diskless-Clients, welche ihren Betriebssystemkern und auch das Root-Verzeichnis über das Netzwerk beziehen, bereits weit verbreitet. Dadurch wird eine lokale Installation eines Betriebssystems samt Software hinfällig, da alle Programme und Verzeichnisse auf den Servern verbleiben.

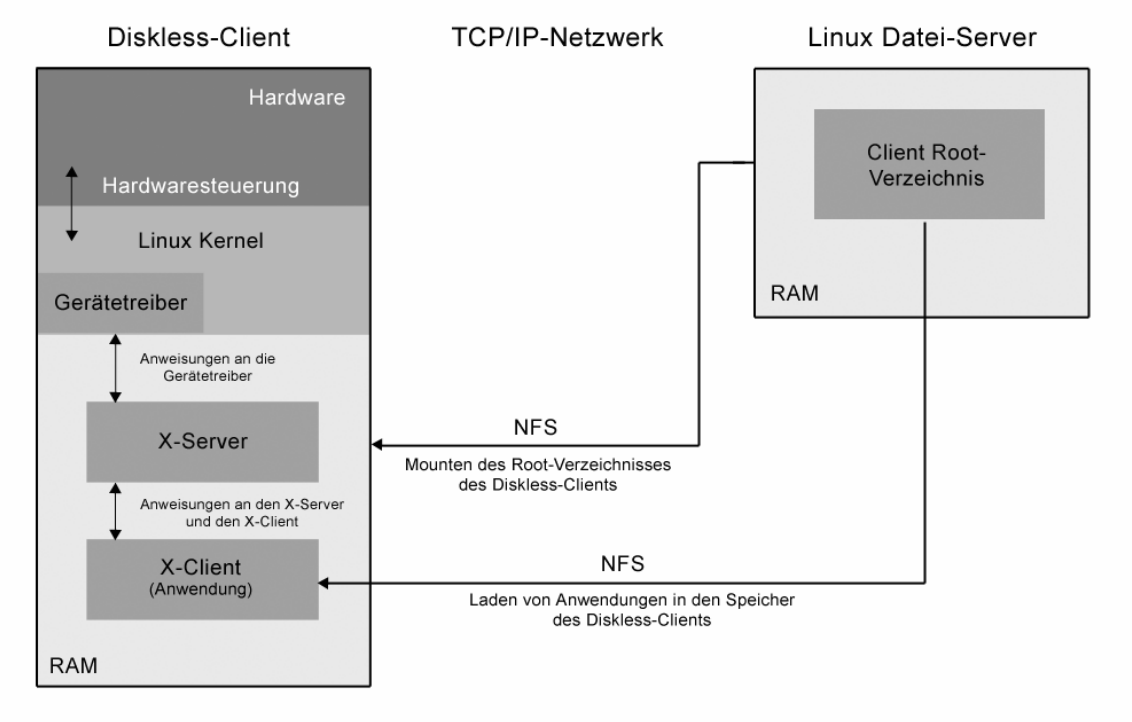

Abbildung 3: Root over NFS (vgl. [4])

Wie in Abbildung 3 ersichtlich befindet sich das Root-Verzeichnis für den Diskless-Client auf einem Linux Datei-Server, mit welchem der Client über ein Netzwerk verbunden ist. Bei dem Systemstart des Clients wird während des Bootvorganges das Root-Verzeichnis per NFS von dem Client gemountet und so alle in diesem Verzeichnis befindlichen Daten, welche physisch auf dem Server liegen, dem Client zugänglich gemacht. Somit werden sämtliche Anwendungen (X-Client sowie auch X-Server), welche von dem Benutzer und dem System verwendet werden, per NFS in das Random Access Memory (RAM) des Diskless-Client geladen und lokal ausgeführt.

Vorteil dieser Variante ist, dass bereits angeschaffte Arbeitsplatzrechner weiterverwendet werden können und die meist große Rechenleistung dieser Computer voll ausgeschöpft werden kann. Dadurch werden leistungsstarke Server überflüssig, da die Systemlogik auf den Diskless-Clients abläuft.

2.1.2.2 Root RAM-Disk

Ein weiterer Ansatz beim Thema Netzwerkboot im Bezug auf Thin-Clients ist die Möglichkeit das Root-Verzeichnis nicht per NFS zu mounten, sondern das Root-Verzeichnis in ein Dateisystem-Image zu packen und dieses während des Bootvorganges vom Server auf den Client zu kopieren, dort zu entpacken und in die RAM-Disk zu laden.

Mounten (deut. einhängen) wird bei einem Computer jener Vorgang genannt, bei dem ein Dateisystem an einer bestimmten Stelle (Mount Point) dem Betriebssystem verfügbar gemacht wird.

Bei der Verwendung einer RAM-Disk wird ein Teil des Arbeitsspeichers in einem Computer wie eine Festplatte verwendet. Dieser Teil des Arbeitsspeichers wird dann dazu Verwendet um dort das Root-Verzeichnis abzulegen.

Vorteile dieser Variante sind, dass auch bei instabilen Netzwerken der Rechner stabil weiter arbeitet und der Zugriff auf das Root-Verzeichnis sehr schnell ist, da es sich direkt im Arbeitsspeicher des Clients befindet.

Ein Nachteil ist, dass bereits alle Programme die Verwendung finden sollen bereits in das RAM-Disk Image integriert werden müssen. Bei vielen Anwendungen werden auch die zu ladenden Images relativ schnell sehr groß. Deshalb findet diese Variante hauptsächlich für Terminals eine Verwendung.

Im Internet findet sich eine Fülle von Anleitungen zum Erzeugen von Thin-Clients, bei denen das Root-Verzeichnis mit Hilfe einer RAM-Disk geladen werden kann. Eine Informationsquelle dazu findet sich unter [7].

2.1.3 Terminal

Das Grundprinzip der Terminallösung basiert auf dem der serverbasierten Datenverarbeitung (Server Based Computing), bei dem alle Anwendungen zentralisiert auf Servern laufen.

In Abbildung 4 ist ein grober Überblick über die Client/Server-Architektur einer Terminal Serverlösung zu sehen. Im Falle von Terminals laden diese ein kleines Image per TFTP oder NFS von einem entsprechenden Bootserver über das Netzwerk, welches das Betriebssystem für das Terminal und ein entsprechendes Programm zur Kommunikation mit dem Terminal Server enthält. Das Programm auf dem Terminal bietet dem Benutzer einen grafikorientierten Login auf einem Terminal Server an. Die Kommunikation zwischen Server und Client erfolgt je nach eingesetztem Programm zum Beispiel per X Window System Version 11 (X11), Remote Desktop Protocol (RDP) oder Independent Computing Architcture (ICA). Auf dem zumeist leistungsstarken Terminalserver wird dann dem Anwender der gewohnte Desktop mitsamt den Anwenderprogrammen bereitgestellt, welche die Thin-Clients am Bildschirm darstellen. Jeder am Terminal Server angemeldete Benutzer erhält seine eigene von anderen Benutzern unabhängige Server-Session, sowie den Zugriff auf alle verfügbaren Ressourcen des Betriebssystems.

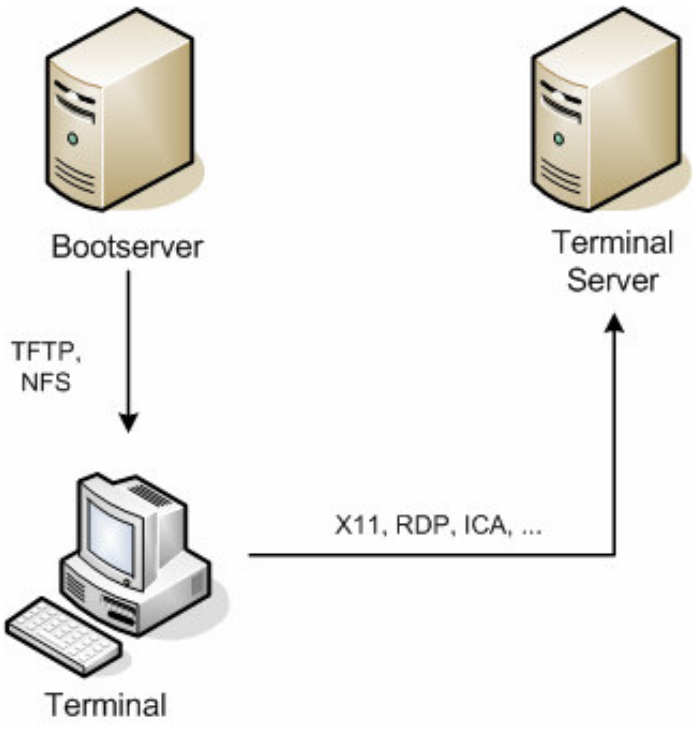

Abbildung 4: Das Terminal Prinzip

Bei einer Terminal Serverlösung werden nur die Tastatureingaben und Mausbewegungen mittels eines speziellen Protokolls (X11, RDP, ICA) von den Clients an den Server übermittelt, welcher im Gegenzug nur Änderungen der Bildschirmanzeige an den Client überträgt. Die komplette Systemlogik läuft dabei auf dem Terminal Server.

Ein Vorteil dieser Variante liegt darin, dass an den Prozessor und Arbeitsspeicher der Clients nur bescheidene Mindestanforderungen gestellt werden. Die gesteigerte Rechenleistung durch Speicher- und CPU-Aufrüstungen in den Servern kommen zu gleichen Teilen sofort allen beteiligten Clients zugute.

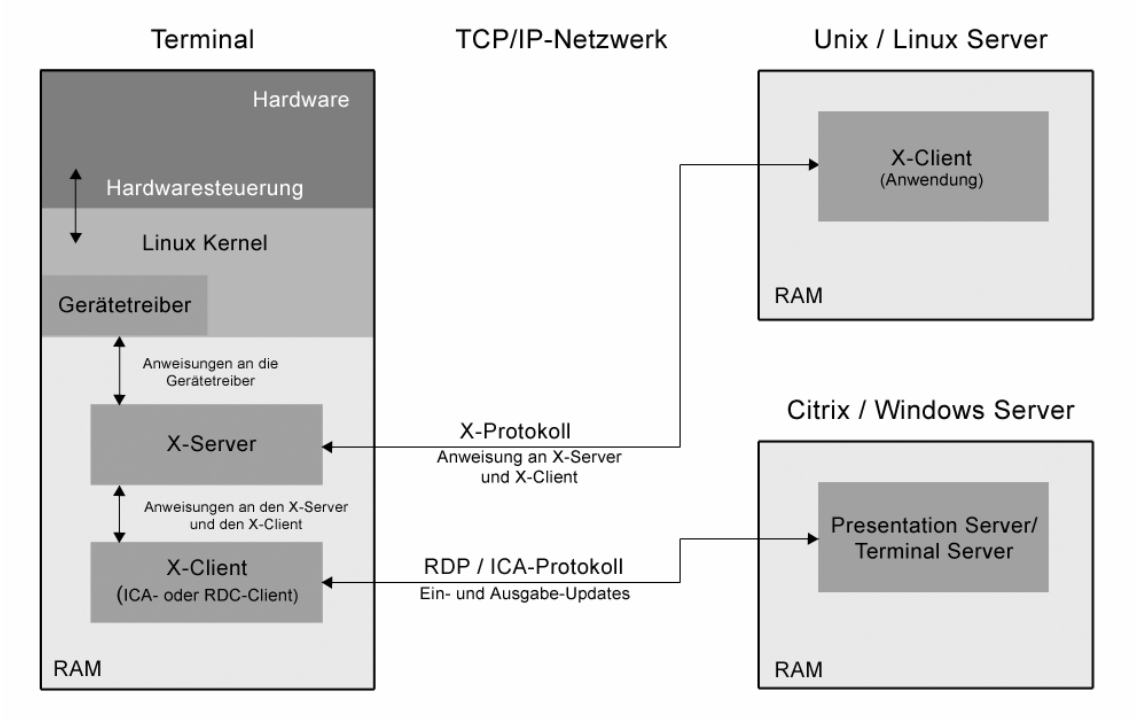

Abbildung 5: Kommunikation eines Linux Clients mit Terminal Servern (vgl. [4])

Abbildung 5 zeigt die wesentlichen Komponenten beim Betrieb eines Linux Clients als Terminal, welche zur Verbindung mit verschieden Terminal Servern notwendig sind. Bei einer Verbindung zu einem Unix oder Linux Server erfolgt die Kommunikation direkt zwischen dem X-Server des Terminals und dem X-Client auf dem Server. Bei einer Verbindung zu einem Citrix oder Windows Server, muss die Kommunikation noch über ein weiters Dienstprogramm auf dem Terminal erfolgen, nämlich einen Citrix (ICA)- oder Windowsclient (RDC), welcher in Folge als X-Client mit dem X-Server kommuniziert.

2.1.4 Hardwareanforderungen

Je nach Verwendung lassen sich Thin-Clients individuell ausstatten und softwareseitig beliebig anpassen. Lokal angeschlossene Geräte wie CD-ROM Laufwerke oder Brenner sind genauso nutzbar wie andere Wechselmedien, Scanner oder Grafikkarten. Interne Festplatten können ebenso für Swapping oder zum Ablegen von temporären Dateien weiter verwendet werden. Da je nach Verwendung und geplanten Anzahl von Anwendern pro Server die Hardwareanforderung sehr stark variieren, müssen diese die Administratoren für jeden einzelnen Fall gesondert entscheiden und können nicht pauschal bestimmt werden. Die folgende Tabelle soll nur einen groben Überblick über die einzelnen Anforderungen geben. [1], [21]

Tabelle 3: Hardwareanforderungen in einer Thin-Client Umgebung

Komponente	Diskless-Client	Terminal
Endgeräte	aktuelle, handelsübliche Computer; Speicherausstattung wie bei normalen Arbeitsplatzrechnern	ältere Computer, oder neuere mit langsamen Prozessor, Speicherausstattung kann relativ bescheiden ausfallen (~ 64 MB).
Boot Server	bescheiden ausgestatteter Server nur für die Bootservices BootP oder Dynamic Host Configuration Protocol (DHCP), TFTP, eventuell NFS Bei einer *root-over-nfs* Lösung, muss die Bereitstellung des Root-Verzeichnisses für die Clients per NFS berücksichtigt werden.	bescheiden ausgestatteter Server nur für die Bootservices (BootP oder DHCP, TFTP, eventuell NFS)

Terminal Server		sehr rechenstarke Server (2 – 4 Prozessoren) mit sehr viel Hauptspeicher (ca. 128 MB für das Basissystem und 20 -50 MB pro Anwender je nach verwendeter Anwendungen).
Netzanbindung	breitbandige Anbindung der Clients mit 100 Mbps, Server mit mind. 1 Gbps.	schmalbandige Anbindung der Clients, auch über Wide Area Network (WAN) möglich. Server mit 100 Mbps.

2.2 Bootserver

Da Thin-Clients wegen der fehlenden Festplatte lokal kein Betriebssystem vorhalten können, muss dieses während der Startphase des Computers über das Netzwerk von einem Bootserver bezogen werden. Dazu muss der Server je nach verwendetem Bootkonzept die Dienste TFTP oder NFS zur Verfügung stellen, über welche der Betriebssystemkern an die Clients verteilt werden kann. Die Dienste TFTP und NFS werden dabei immer nur kurz verwendet und beeinflussen die Gesamtlast des Servers kaum.

Als Bootserver zum Bereitstellen der Bootimages eignen sich sowohl MS Windows sowie Unix/Linux Server.

Im Falle einer PXE-Boot Variante kann, um den regulären DHCP Serverbetrieb nicht zu beeinflussen, auch ein DHCP Proxy (siehe 2.6.3.3) auf dem Bootserver vorgesehen werden.

Bei einer Thin-Client Variante, wobei das Root-Verzeichnis für die Clients von einem NFS Server eingebunden wird (siehe 2.1.2.1), stellt dieser Prozess die Hauptbelastung des Servers dar und trägt daher maßgeblich zur Designentscheidung für den Server bei. Bei einer solchen Bootlösung wird das subjektive Empfinden der Anwender bei der Arbeit mit den Thin-Clients unmittelbar durch die NFS Leistung des Servers bestimmt.

Da bei einem Ausfall des Bootservers kein booten von Thin-Clients mehr möglich ist und bei einer root-over-nfs Lösung selbst der Betrieb im Extremfall zum stehen kommt, muss dieses Problem bereits bei der Planung der Serverlandschaft berücksichtigt werden und für ausreichend Serverredundanzen gesorgt werden.

2.3 Terminal Server

Ein Terminal Server ist ein Server, welcher Anwendern mit Hilfe von Terminals (Thin-Clients) eine vollständigen Computerarbeitsumgebung zur Verfügung stellt, ohne dass die verwendeten Programme auf den Arbeitsstationen selbst installiert sein müssen. Terminals kommunizieren hierbei entweder direkt über ein mit dem Server verbundenes Kabel, oder über ein Computernetzwerk.

Im Folgenden sollen drei Terminal Servervarianten kurz etwas näher vorgestellt werden.

2.3.1 Unix / Linux

Unix, sowie Linux zielte seit Beginn seiner Entwicklung darauf ab ein echtes Mehrbenutzer-Betriebssystem zu sein, welches von mehreren Benutzern gleichzeitig unabhängig von einander genutzt werden kann. Anfänglich stellten die Server hauptsächlich textbasierte Arbeitsoberflächen zur Verfügung, welche nur Zeichen und Buchstaben darstellen konnten. Später wurden auch grafikbasierte X-Terminals entwickelt, die einen Arbeitsplatz wie er heute auf jedem Computer üblich ist darstellen können.

Im Gegensatz zu anderen Betriebssystemen weiß der Unix- / Linux-Kern nichts von Grafiken. Dazu wird das *X Window System* verwendet, welches eine Sammlung von Standards, Protokollen und Programmen darstellt, um Ein- und Ausgabegeräte, wie Maus, Tastatur und Bildschirm, anzusteuern.

2.3.2 MS Windows Terminal Server 2000/2003

Standard Windows Server erlauben es mehreren Anwendern lediglich nur lokale Ressourcen wie Dateien, Drucker, Dienste usw. gleichzeitig zu nutzen, aber nicht sich gleichzeitig interaktiv auf einem Server anzumelden und auf virtuellen Arbeitsplätzen zu arbeiten.

Die ersten Ansätze eines Terminal Servers von Microsoft gab es in den Zeiten von Windows NT 4.0. Diese damalige Terminal Server Edition war komplett von der restlichen Server-Familie getrennt. Mit dem Windows Server 2000 wurden dann auch die Dienst und die Betriebssystemkomponenten, welche für einen Mehrbenutzer Terminal

Server notwendig sind, in den Standard Server integriert. Auch in dem jetzigen Produkt von Microsoft, dem Windows Server 2003 ist das Terminal Server Service direkt enthalten.

Im Windows Server 2003 sind zwei verschieden Versionen der Terminal Server Komponente enthalten. Eine davon ist die richtige Terminal Server Variante, welche mehrere simultane Usersessions auf einem Server erlaubt. Die zweite Variante, der so genannte „Remote Desktop“, ist eine etwas abgestufte Variante des richtigen Terminal Servers und erlaubt nur zwei gleichzeitig konkurrierende Anwendersitzungen. Diese Variante ist hauptsächlich zur Fernadministration der Windows Server gedacht.

Um eine Verbindung von einem Client zu einem Windows Terminal Server aufzubauen benötigt der Anwender einen Remote Desktop Connection Client (RDC Client), sowie das RDP, welches unter 3.1.1 noch näher behandelt wird. Weiters besteht ein Windows Terminal Server, wie in Abbildung 6 zu sehen, aus vier wichtigen Komponenten. Zum einen aus dem Windows Kernel, welcher die Basis des Terminal Servers bildet. Darauf setzen direkt die Terminal Server Services auf. Verwaltet werden die einzelnen Usersessions vom Session Manager, welcher vom Terminal Server Service initiiert wird. Der „Listener“ ist ein Subsystem des Connection Ports, welcher eingehende Verbindungsanforderungen von RDCs an den Session Manager weiterleitet und sofort wieder auf neue eingehende Verbindungen „horcht“.

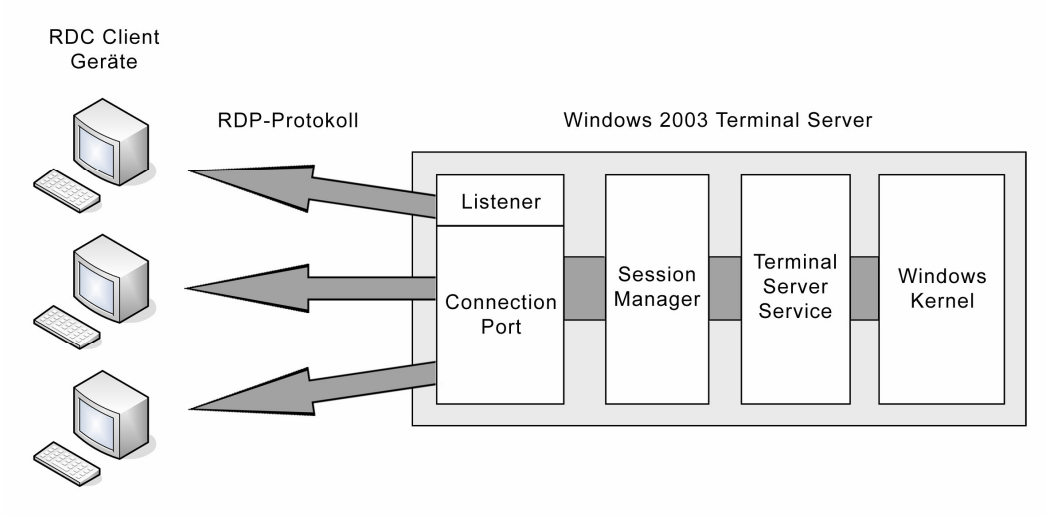

Abbildung 6: Terminal Server 2003 Komponenten [16]

Weiterführende Information zu diesem Thema finden sich in [16], [17], [19], [20] und [21].

2.3.3 Citrix Presentation Server

Die in der Terminal Serverlösung von Microsoft implementierte Mehrbenutzer-Technologie ist ein ursprünglich von der Firma Citrix entwickeltes Produkt. Citrix und Microsoft verbindet eine bereits jahrelange Kooperation auf dem Gebiet der Terminal Server Lösungen.

Citrix entwickelt aber auch eine eigene Terminal Serverlösung unter dem Namen Meta-Frame Presentation Server bzw. Citrix Presentation Server. Der Einsatz des Citrix Presentation Servers basiert auf dem Einsatz der Terminal Serverdienste eines Windows-Servers. Der Citrix Presentation Server würde ohne den speziellen Dienst nicht funktionieren, da der Windows Terminal Serverdienst die Basis dafür bildet. Citrix Presentation Server selbst ist kein eigenständiges Betriebssystem, sondern ist eine Software, welche auf das Betriebssystem Windows-Server auf setzt. Der Presentation Server von Citrix unterstütz aber nicht nur die Windows-Server von Microsoft, sondern kann ebenfalls zur zentralen Anwendungs- und Informationsbereitstellung auf den Serverplattformen Sun Solaris, IBM AIX und HP-UX eingesetzt werden.

Die Kerntechnologie, welche hinter dem Presentation Server steht, ist das von Citrix entwickelte Protokoll ICA. Das ICA-Protokoll ist vollkommen unabhängig von dem Microsoft RDP und legt eine Spezifikation fest, um Daten zwischen Client- und Serversystemen übertragen zu können. ICA ist auf der Clientseite an keine bestimmte Betriebssystemplattform gebunden.

Weiterführende Informationen zu den Grundlagen und Administration von Citrix Presentation Server ist unter [21], [22], [23] und [24] zu finden.

2.4 Bootprozess eines Thin-Clients

Während eines typischen Systemstarts, wird ein sehr kleines Bootprogramm geladen und ausgeführt, welches es erst im Anschluss ein viel größeres und komplexeres Betriebssystem lädt.

Nach dem Einschalten eines Rechners wird das Basic Input/Output System (BIOS), welches sich auf einem PROM oder EPROM direkt auf der Hauptplatine befindet, direkt von diesen Speicherbausteinen aus ausgeführt und erledigt einige Diagnosetests der Computerhardware durch. Diese Tests werden auch Power On Self Test (POST) genannt. Beim POST wird der Hauptspeicher, der Prozessor, sowie andere Hardwarekomponenten überprüft und die Kontroller initialisiert um das System betriebsbereit zu machen. Danach dekomprimiert sich das BIOS vom BIOS-Speicher in den Systemhauptspeicher und startet mit der Ausführung des Programmcodes. Im Anschluss liest das BIOS den ersten Datenblock (512 Bytes) von dem festgelegten Bootgeräte, den Master Boot Record (MBR). Der MBR enthält eine Partitionstabelle des Datenträgers und den Boot-Loader, welcher wiederum den Betriebssystemkern in den Hauptspeicher lädt und startet.

Im Falle eines Netzwerkbootprozesses mittels PXE, wird die Kontrolle aber zuvor an das BIOS der Netzwerkkarte abgegeben, welches mit Hilfe des DHCP versucht eine Internet Protocol (IP) Adresse für die Netzwerkkarte und die Adresse eines Bootservers von einem DHCP Server zu beziehen. Auf dem Bootserver befindet sich ein Network Bootstrap Program (NBP), welches wie das Bootprogramm bei Festplattenbootumgebungen dafür verantwortlich ist, einen Betriebssystemkern in den Speicher zu laden, um wiederum das Betriebssystem hoch zu booten. Die Informationen, welcher der Client vom DHCP Server erhält, beinhalten auch den Dateinamen des NBP. Mittels TFTP (Trivial File Transfer Protocol) lädt der Client das Bootprogramm vom Bootserver in seinen eigenen Speicher. Das NBP ist vergleichbar mit dem Grand Unified Bootloader (GRUB), dem LInux LOader (LILO) oder dem NT Loader (NTLDR). Als NBP zum Booten eines Linuxkernels per PXE kann zum Beispiel PXELinux (siehe 2.6.4) verwendet werden.

2.5 Unterschiedliche Bootkonzepte

Unter diesem Punkt möchte ich einen kurzen Überblick über verschiedene Möglichkeiten geben, um Thin-Clients deren Betriebssystem über ein Netzwerk booten zu lassen.

Nach den so genannten BootPROM Varianten wie PXE, Etherboot und Netboot werden auch andere Bootverfahren wie von WEBasDisk oder von iBoot kurz vorgestellt und erklärt.

2.5.1 Preboot eXecution Environment (PXE)

Nahezu jeder seit 1998 verkaufte PC ist mit „Wired for Management“ (WfM) kompatibel. WfM ist eine open-industry Spezifikation, welche von Intel initiiert wurde um ein besseres und komfortableres Management von Client PC-Systemen zu ermöglichen. Weiters ist WfM Teil der PC98-Spezifikation von Intel und Microsoft. PXE ist Bestandteil der WfM-Spezifikation.

PXE ist ein offener Industriestandard, welcher von einigen Software- und Hardwareherstellern entwickelt wurde. Anfänglich wurde er von Intel entworfen und mit Unterstützung von verschiedensten Herstellern wie HP, Dell, 3Com, Compaq und Phoenix Technologies weiterentwickelt. PXE funktioniert in Zusammenarbeit mit der Netzwerkkarte des PCs und befähigte diese als ein weiters Bootgerät zu fungieren. Die Grundidee hinter PXE war die Computernetzwerkschnittstelle als von der Industrie akzeptiertes Bootgerät zu etablieren. Dies bedeutete die Netzwerkkarte in die Liste der traditionellen Bootgeräte wie den Floppylaufwerken, Festplatten und CD-ROM Laufwerken einzugliedern, welche dazu verwendet werden um Betriebssysteme auf Computern zu starten. Somit kann es dazu verwendet werden, einen Rechner von einem Netzwerk zu booten.

PXE bootet einen Client PC vom Netzwerk, indem es eine Boot-Image Datei von einem Server auf den Computer transferiert. Diese Datei kann entweder ein Betriebssystem für den Client sein, oder auch ein Hilfsprogramm wie ein Speichertest, Virenscanner oder ein Backupprogramm, welches Clientmanagementaufgaben übernimmt. Ein großer Vorteil ist, dass PXE nicht betriebssystemspezifisch ist und somit jedes Betriebssystem über die Image-Datei geladen werden kann.

Da PXE in Verbindung mit einer Netzwerkkarte arbeitet, muss die Netzwerkkarte auch PXE-tauglich sein. Die meisten derzeit im Handel verfügbaren Karten, angefangen bei 3COM, Intel, Digital, RealTek bis hin zu SMC, unterstützen bereits PXE. Weiters ist PXE als BootPROM Chip erhältlich oder ist bei auf Hauptplatinen integrierten Netzwerkschnittstellen bereits im BIOS verankert. Vorsicht ist bei Boot-ROMs geboten, denn PXE ist für einzelne Netzwerkkartentypen spezifisch. Somit funktionieren Boot-ROMs immer nur auf einer Type von Karten. (vgl. [29])

Eine ausführliche PXE-Protokollbeschreibung finden Sie unter 2.6.3.

2.5.2 EtherBoot

Die ersten Zeilen Programmcode von Etherboot wurde 1995 von Markus Gutschke geschrieben. Er begann damals aus reinem Forschungsdrang heraus ein Netzwerk in seinem damaligen Studentenzimmer aufzubauen. Damals experimentierte er an Programmcode von Gero Kuhlmann, dem Entwickler von Netboot. Da aber dieser Code auf eine bestimmte Netzwerkkarte zurechtgeschnitten war, entwickelte Gutschke sein eigenes BootPROM Programm auf Basis einer RemoteBoot Lösung von FreeBSD. Ab 1997 übernahm Ken Yap das Projekt von Markus Gutschke und immer mehr Entwickler schlossen sich Etherboot an. (vgl. [1])

Etherboot ist ein quelloffenes und kostenloses Softwarepaket, welches unter der GNU General Public License Version 2 (GPL2) veröffentlicht wird. Mit Hilfe von Etherboot können BootROM Images erstellt werden, welche anschließend auf EPROM Chips gebrannt werden. Viele Netzwerkadapter haben entsprechende Sockel um solche Speicherbausteine aufnehmen zu können. Mit Hilfe dieser modifizierten Netzwerkkarten kann nun zur Bootzeit von Bootservern Programmcode via Netzwerk auf Client PCs geladen und auf dem Rechner anschließend ausgeführt werden. Weiters ist es auch möglich ein Etherboot-Image in das System-BIOS eines Computers zu integrieren. Etherboot arbeitet zurzeit auf x86, Itanium und Hammer Architekturen und wird hauptsächlich dazu verwendet Linux, FreeBSD oder Disk Operating System (DOS) Betriebssystemplattformen über das Netzwerk zu booten. Grundsätzlich können aber auch andere Betriebssysteme geladen werden, da Etherboot an kein bestimmtes System gebunden ist.

2.5.2.1 Etherboot BootROM Image

Eine einfache Möglichkeit, um ein Etherboot BootROM Image zu kreieren, bietet die Internetseite http://rom-o-matic.net/. Auf dieser Seite kann man sich ein fertiges Etherboot BootROM Image selber zusammenstellen, konfigurieren und anschließend downloaden. Ein weiterer Weg um ein Boot-Image zu erzeugen ist die Kompilierung des Quellcodes auf einem Linuxsystem. Neben dem Read Only Memory (ROM) Image gibt es auch die Möglichkeit ein Floppy- oder ein PXE Image erzeugen zu lassen. Bevor man ein fertiges Image auf einen ROM Chip brennt, sollte man auf alle Fälle zuerst ein Floppy Image erzeugen, mit welchem man seine zusammengestellte Konfiguration testen kann.

2.5.2.2 Etherboot Boot-Image

Ein Vorteil von Etherboot gegenüber von PXE ist, dass Etherboot direkt einen Linuxkernel ohne Umweg über einen „2nd Level Bootloader“, wie zum Beispiel PXELinux, laden und starten kann. Damit der Kernel nach dem Netzwerktransfer auf dem Client startet, muss dieser aber für Etherboot noch speziell bearbeitet werden. Etherboot erwartet als Boot-Image ein so genanntes „tagged image“. Ein „tagged image“ ist ein normales Kernelimage, welches mit einem besonderen Header versehen wird. In diesem Header stehen Informationen in welchen Speicherbereich die Daten geladen werden müssen und an welcher Speicheradresse das Programm gestartet werden muss. Solche Images lassen sich mit dem Linuxprogrammen *mknbi (make network bootable image)* erzeugen. Dazu wird mit Hilfe dieses Programmes eine einzige Datei generiert, welche aus einem Header, einem Betriebssystemkern und eine RAM-Disk (initrd) besteht.

2.5.2.3 Multiboot

Etherboot kann um Routinen erweitert werden, die es dem Administrator ermöglichen ein einfaches Menü anzeigen zu lassen, welches neben dem klassischen Start über das Netzwerk auch Optionen zum Booten von Festplatte, Diskette oder CD-ROM anbietet. Dieses System ist aber leider nicht sehr flexibel, da bei Änderungen an der Konfiguration sämtliche BootPROMs ausgetauscht werden müssten.

Als Alternative dazu ist aber auch möglich Etherboot aufwändigere Menüs über DHCP-Optionen anzeigen zu lassen.

Um die Benutzer der Computer aber nicht zu verwirren sollte man sich aber nur auf eine Variante festlegen.

2.5.2.4 Bootabfolge

Ein typischer Bootvorgang mittels Etherboot läuft auf folgender Weise ab. Nach dem Starten des plattenlosen Computers wird der Bootstrap-Loader ausgeführt, welcher sich auf dem ROM Chip der Netzwerkkarte befindet. Mittels DHCP oder BootP wird dem Client seine IP Adresse mitgeteilt. Bei Etherboot gibt es zwei Varianten wie das Bootimage vom Bootserver bezogen werden kann. Bei der ersten Variante wird mittels TFTP das Bootimage in den Speicher des Rechners kopiert und anschließend lokal ausgeführt. Bei der zweiten Variante ist es möglich das NFS dazu zu verwenden, um das Bootimage auf den Client zu transferieren und anschließend auszuführen. Die Kontrolle über den Computer wird dann vom Bootstrap Programm an den Betriebssystemkern übergeben.

2.5.2.5 Treiber

Da Etherboot während des Bootprozesses die Netzwerkkarte steuern muss, ist es auf Gerätetreiber angewiesen, welche diese Aufgabe übernehmen. Etherboot enthält Treiber für fast alle gängigen Netzwerkadapter. Für Etherboot müssen die Kartentreiber eigens programmiert werden, falls dies nicht schon durch die Entwicklergemeinschaft erfolgt ist.

Weitere interessante Informationen zum Thema Etherboot sind unter [1] und [5] erhältlich.

2.5.3 Netboot

Zu Beginn der 90er Jahre begann Gero Kuhlmann, der heute als Anästhesist arbeit, mit der Vernetzung von Computern zu experimentieren. Er wollte damals einen plattenlosen Druckserver von einem anderen PC über das Netzwerk booten und benötigte dazu ein BootPROM für seine Netzwerkkarte. Bei dem Australier Jamie Honan fand er eine für seine Zwecke passenden Quellcode und erste Versuche mit einem selbst gebrannten BootEPROM verliefen erfolgreich. Um Linux jetzt noch über das Netzwerk booten zu können, musste Kuhlmann den Linuxkernel an das Boot Imageformat von Jamie Honan mit Hilfe eines Hex-Editor anpassen. Um dies zu automatisieren schrieb er das Programm *mknbi-linux*, welches zur Basis des Netboot-Projektes wurde. Netboot wird bis heute von Gero Kuhlmann unter Mithilfe eines kleinen Entwicklerteams weitergeführt und weiterentwickelt. (vgl. [1])

Netboot ist ein quelloffenes Softwarepaket, welches alle notwendigen Werkzeuge enthält um plattenlose Computer über ein IP Netzwerk von einem Server booten zu lassen. und wird unter der GNU Public License (GPL) veröffentlicht. Gegenwärtig ist es möglich Linux, FreeBSD und unterschiedliche DOS-Varianten, wie MS-DOS, PC-DOS, PTS-DOS, OpenDOS, NovellDOS und FreeDOS, mit Netboot zu starten. Das Netbootpaket unterstützt alle BSD- und POSIX-Plattformen.

2.5.3.1 Netboot BootROM Image

Mit Hilfe des Netboot-Dienstprogrammes *makerom* kann ein BootROM Image für eine Floppy Diskette oder ein EPROM erzeugt werden.

2.5.3.2 Netboot Boot-Image

Netboot verwendet wie Etherboot ein eigenes Boot-Image-Format, welches ebenso einen Bootlader, den Betriebsystemkern und ein RAM-Disk enthält. Um ein solches Netboot-Image zu erzeugen, wird das Programm *mknbi-linux* verwendet, dessen Output anschließend nur mehr in das TFTP Verzeichnis kopiert werden muss.

2.5.3.3 Bootabfolge

Netboot arbeitet mit der BootPROM Technologie mit deren Hilfe Netzwerkkarten netzwerkbootfähig gemacht werden können. Das BootROM verwenden nach dem Einschalten des Computers BOOTP oder DHCP um die IP-Adresse des Client und die IP Adresse des Bootservers zu beziehen. Anschließend wird per TFTP das Netboot-Image vom TFTP Server downgeloaded in den Systemspeicher kopiert und ein spezieller Bootlader ausgeführt, welcher wiederum anschließend den Kernel in den Speicher lädt.

2.5.3.4 Treiber

Die Netzwerkkartentreiber werden nicht wie bei dem unter Punkt 2.5.2 behandelten Etherboot Projekt vom Entwicklerteam selbst programmiert, sondern Netboot verwendet die Treiber der Netzwerkkartenhersteller. Netboot schafft zu Beginn eine Umgebung in der die mitgelieferten NDIS- beziehungsweise DOS-Treiber geladen werden können und startet erst dann DHCP und TFTP. Dieser Ansatz hat den Vorteil das Netboot somit jede Karte unterstützt zu der der Hersteller auch Treiber liefert.

Weitere interessante Informationen zum Thema Netboot sind unter [6] zu finden.

2.5.4 WEBasDisk

Ein Produkt von der Firma Kontron Embedded Modules GmbH namens WEBasDisk erlaubt es einen festplattenlosen Computer verschiedene Betriebssysteme, wie Windows XP oder Linux, über ein Ethernet Netzwerk von einem Server zu booten.

Wie aus Abbildung 7 zu entnehmen, funktioniert Kontrons WEBasDisk aus der Sichtweise eines PCs wie eine lokale IDE-Festplatte. Anstatt auf die lokale Festplatte des Computers zu zugreifen, greift der mit WEBasDisk ausgestattete Rechner per Transfer Control Protocol /Internet Protocol (TCP/IP) aber über das Local Area Network (LAN) oder WAN auf ein Laufwerk eines entfernten Servers zu. Dieser Zugriff kann mittels den Kommunikationsprotokollen Hyper Transfer Text Protokoll (HTTP) oder Server Message Block (SMB) erfolgen. Dazu wird die WEBasDisk-Karte, welche die Größe einer 2,5 Zoll-Festplatte besitzt, mit der IDE-Schnittstelle des Computers verbunden und weiters über den 10/100Base-T Ethernet-Kontroller an das Netzwerk gekoppelt. WEBasDisk arbeitet dabei völlig transparent, ohne das der Rechner von dem dahinter liegendem Netzwerk erfährt.

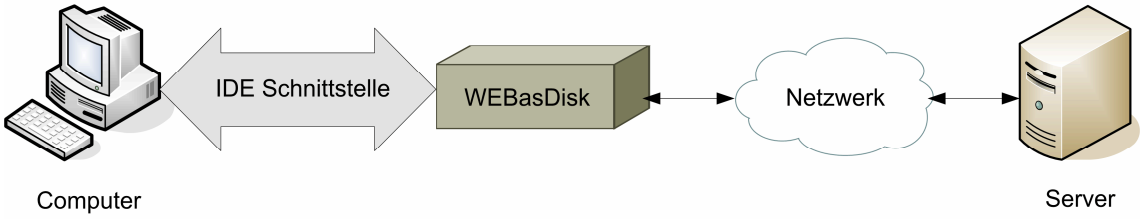

Abbildung 7: WEBasDisk [8]

Bei einem Systemstart bootet der Client mit Hilfe seiner virtuellen Festplatte über das Netz. Während des laufenden Betriebes greifen Anwendungen die lokal auf dem PC laufen dann auf die Daten über die IDE-Schnittstelle, vergleichbar wie auf die Daten einer lokalen installierten Festplatte, zu. Der Net+ARM-Prozessor der WEBasDisk-Karte wandelt dabei die Lese- und Schreibzugriffe bidirektional in TCP/IP-Paket um und versendet diese über das Ethernet-Netzwerk. Der Datentransfer über das Netzwerk erfolgt dabei in einer Geschwindigkeit von 500 KByte/Sekunde, was nicht mit einem Datentransfer beim Zugriff auf eine echte Festplatte vergleichbar ist. (vgl. [1], [8])

2.5.5 iBoot – remote Boot over Internet Small Computer System Interface (iSCSI)

iBoot wurde von den IBM Forschungslaboren in Haifa Israel entwickelt und ermöglicht es einem plattenlosen Computer über das Netzwerk zu starten. Der plattenlose Computer wird dabei via Netzwerk mit Festplatten eines entfernten Servers verbunden, von denen der Clientcomputer sein Betriebssystem bootet. Neben Linux Betriebssystemen können ebenfalls Betriebssysteme von Microsoft, wie Windows 2000 oder Windows XP, über diese Technologie von einem Storage Area Network (SAN) gestartet werden. Das Betriebssystem sowie die Anwendungen laufen bei Verwendung dieser Technologie lokal auf dem PC.

IBM setzt bei Ihrem Bootkonzept von iBoot auf das von der Internet Engineering Task Force (IETF) standardisierte Internetprotokoll iSCSI, welches Standard Small Computer System Interface (SCSI) Befehle kapselt und über IP-Netze überträgt. Bei dem iBoot Projekt verwendet IBM nach eigenen Angaben ein ROM-Image, welches den iSCSI Client Code, einen TCP/IP-Stack und BIOS Interrupt Code enthält.

2.5.5.1 iBoot Konzept

Der Initialisierungscode von iBoot konfiguriert die Bootumgebung und setzt dabei die Parameter für das iSCSI-Login, sowie die Ziel IP-Adresse und den Namen des Initiators. Mit iBoot ist es möglich, unterschiedliche Betriebssysteme auf Grund von Firmenorganisationsregeln den Benutzern dynamisch zuzuweisen.

Bei einem Systemstart wie unter 2.4 beschrieben, nutzt das BIOS eine Reihe von Funktionen um in einem frühen Stadium den Computer zu steuern. Einige dieser im BIOS integrierten Befehle sind die Interrupt 13h (int 13h) Befehle, welche zum Verwalten der Disketten- und Festplattenlaufwerke verwendet werden. Das BIOS verwendet unter anderem diese Festplatten-Lese-Schreibdienste, um auf den MBR im ersten Sektor der Festplatte zugreifen zu können. Danach wird die Kontrolle über das System vom BIOS an den Bootloader im MBR übergeben, welcher den Bootvorgang fortsetzt. Die int 13h Befehle werden vom Bootloader solange verwendet, bis das Betriebssystem und die Treiber soweit geladen sind, um den Startprozess weiter zu führen.

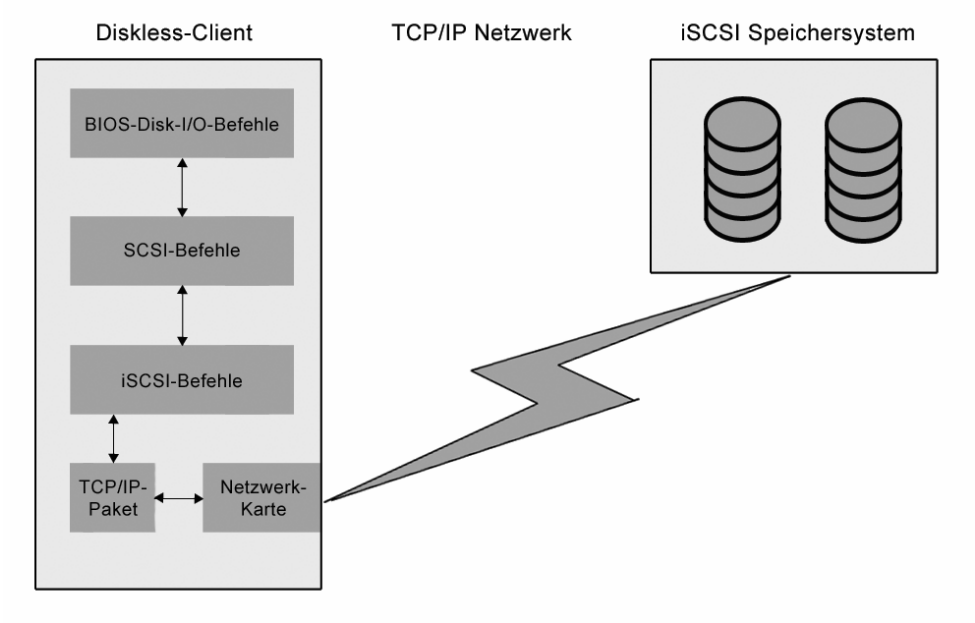

Abbildung 8: iBoot Architektur (vgl. [1])

Bei dem von iBoot verfolgtem Konzept wird der zuvor beschriebene Bootprozess aber soweit abgeändert, damit per iSCSI Protokoll entfernte Speichersysteme als Bootmedien verwendet werden können.

Abbildung 8 zeigt die prinzipielle Funktionsweise einer iBoot Architektur. Der im ROM-Image gespeicherte int 13 hook-Code fängt alle BIOS Festplatten I/O Befehle ab, übersetzt das Cylinder, Header, Sector-Format (CHS-Format) in ein Linear Block Array (LBA) und führt die Befehle der SCSI-Schicht zu. Die SCSI-Schicht generiert Lese/Schreib-Befehle und leitet diese wiederum an die iSCSI-Schicht weiter. Die iSCSI-Schicht übersetzt die SCSI-Befehle in iSCSI-Befehle und schickt sie zur weiteren Verarbeitung an die TCP/IP-Stack Schicht, welche den Netzwerkadapter benutzt um die erzeugten Datenpakete über das Netzwerk zu übertragen. (vgl. [1], [9])

2.6 Protokolle und Dienste

Dieser Abschnitt der Arbeit soll einen Überblick über die wichtigsten Protokolle und Dienste im Zusammenhang mit der Errichtung einer Netzboot-Umgebung geben.

2.6.1 Bootstrap Protocol (BootP)

Das Bootstrap Protocol (BootP), wurde bereits 1985 in dem Request for Comment (RFC) 951 festgelegt und war ursprünglich dafür vorgesehen, um plattenlosen Clients während des Bootprozesses deren Netzwerkkonfigurationsparameter wie IP-Adresse, die Adresse des Bootservers und den Namen einer Datei, welche in den Speicher geladen und ausgeführt werden muss, mitzuteilen. BootP verwendet dabei die User Datagram Protocol (UDP) Ports 67 und 68 für die Anforderung der Informationen (BOOTREQUEST) und für die Antwort des Servers (BOOTREPLY). Der Bootserver muss dabei aber nicht zwingend in demselben Netzwerksegment liegen wie der Client (cross-gateway booting). Bei netzwerkboot-tauglichen Clients ist das BootP meist in einem BootPROM-Chip auf einer Netzwerkkarte oder direkt in dem System-BIOS integriert.

Bei der Anforderung der Netzwerkkonfigurationsparameter sendet der Client einen Broadcast immer auf den Zielport 67. Anschließend horcht der Client auf dem Port 68 auf eine Antwort von einem Bootserver. Die wichtigsten Daten die ein Client bei Anforderung mit sendet sind die eigene Hardwareadresse der Netzwerkkarte und eine zufällig generierte Zahl, welche sich in der Antwort des Servers wieder finden muss.

Erhält ein Bootserver eine Anforderung von einem Client, muss er an Hand der Hardwareadresse feststellen, welche IP-Adresse für den Client vorgesehen ist, ein Antwortpaket generieren und die weiteren Parameter in das Paket eintragen. Ein Antwortpaket enthält folgende Informationen:

- Hardwareadresse des Clients
- IP-Adresse des Clients
- Die Zufallszahl
- IP-Adresse und Hostname des Boot Servers

- Name und Pfad der Bootdatei
- weiter Optionen (z.B.: Verzeichnisname, welches als Root-Verzeichnis per NFS vom Kernel eingebunden werden soll)

Der festplattenlose Client „horcht“ auf dem Port 68 und wartet auf die Antwort des Bootservers. Empfängt er ein dementsprechendes Paket auf diesem Port, vergleicht er seine Media Access Control (MAC)-Adresse mit derer in dem Paket und prüft auch ob die Zufallszahl, welche von ihm generiert wurde mit der aus dem Paket übereinstimmt. Falls beides Übereinstimmt, weiß er, dass das Paket für ihn bestimmt ist und er kann die zugestellten Informationen aus dem Paket weiterverarbeiten.

Die weiteren Optionen, welche ebenfalls per BOOTP dem Client übermittelt werden können, sind in dem *RFC 1497 – BOOTP Vendor Information Extensions* genau erklärt.

Das ältere BOOTP wird aber heutzutage immer häufiger durch das neuere DHCP, welches mehr Möglichkeiten zur Clientkonfiguration bietet, ersetzt.

Weitere Details zu dem Thema BootP sind unter [10] nach zu lesen.

2.6.2 Dynamic Host Configuration Protocol (DHCP)

Das DHCP wurde 1997 mit dem RFC 2131 standardisiert und basiert auf dem älteren BOOTP. DHCP wurde so designed damit es auch BOOTP-Clients ohne Einschränkungen bedienen kann. Wie das BOOTP wird auch das DHCP dafür verwendet, Clients ihre Netzwerkkonfigurationsparameter automatisch über das Netzwerk mitzuteilen. DHCP wurde nicht speziell für den Einsatz mit plattenlosen Clients entwickelt, enthält aber zahlreiche nützliche Eigenschaften für die Verwendung von Thin-Clients.

Wie sein Vorgänger das BOOTP verwendet auch das UDP basierte DHCP die Ports 67, für Anfragen an den Server, und 68, für die Antworten an den Client. Die IP-Adressen können im Gegensatz zu dem BOOTP auch dynamisch an die Clients vergeben werde, aber auch statisch wenn dies gefordert wird.

Neben den für das Booten über ein Netzwerk wichtigen Standardeinträgen wie *next-server* (Adresse des TFTP Servers) und *filename* (Dateiname der per TFTP zu downloadenden Bootdatei) gibt es noch eine Reihe von Zusatzinformationen, welche per DHCP Optionen an die Clients weitergeleitet werden können.

2.6.2.1 DHCP Optionen

Am Ende eines DHCP-Paketes ist ein Optionsfeld mit variabler Länge definiert, mit welchem dem Client komplett individuelle Konfigurationsparameter übermittelt werden können. An Hand dieses Feldes kann DHCP 256 verschiedene Optionen definieren, von denen bereits etliche als Standardoptionen vordefiniert sind. Mit Hilfe diesem Optionsfeld können weiters Information wie Subnetzmaske, IP-Adressen von Gateway, Zeit- und Name Server an den Client übermittelt werden. Von diesen 256 verschiedenen Optionen wurden bereits die Optionen 0 – 223 und 255 klassifiziert. Somit bleiben nur mehr 31 Optionen (224 -254) zur freien Verwendung. Siehe dazu auch [15].

Diese Optionen machen DHCP zu einem zentralen Konfigurationswerkzeug beim Einsatz von Thin-Clients. Administratoren können mit Hilfe dieses Dienstes alle relevanten Parameter für den Betrieb von Thin-Clients an diese übermitteln.

Die folgende Tabelle 4 gibt einen Überblick über einige nützliche Optionen zur Konfiguration von Thin-Clients.

Tabelle 4: DHCP Optionen

DHCP-Option	Datentyp	Erklärung
root-path	String	Pfad des NFS-Rootverzeichnisses
routers	IP-Adresse	Liste von IP-Adressen der Gateways
log-servers	IP-Adresse	Liste von IP-Adressen der Log-Servern
domain-name-servers	IP-Adresse	Liste von IP-Adressen der Domain Name System (DNS) Server
domain-name	String	Domainname für Namensauflösungen
host-name	String	Name des Thin-Client
font-servers	IP-Adresse	Liste von X Window System Font Server
x-display-manager	IP-Adresse	Liste von Systemen, welche einen XDMCP Login anbieten
swap-server	IP-Adresse	Swap-Server für den Client
subnet-mask	IP-Adresse	Subnetzmaske des Clients

Neben diesen standardisierten fest definierten Optionen erlaubt DHCP im Codenummernbereich von 128 – 223 so genannte Vendor-Optionen mit auf zu nehmen. Zum Beispiel kann bei Verwendung von Etherboot ein Bootmenü per DHCP konfiguriert werde, um für die Clients eine Multibootumgebung zu schaffen (siehe auch 2.5.2.3). Da bei muss für jede Menüzeile ein eigener Optionseintrag im DHCP Server konfiguriert werden.

Weiters können mittels der frei verwendbaren Optionen 224 – 254 Informationen zur Runlevel- und Hardwarekonfiguration an die Clients übermittelt werden.

2.6.3 PXE

PXE wurde auf Basis von den standardisierten Internetprotokollen und Diensten TCP/IP, DHCP und TFTP definiert. Die aktuelle Spezifikation von Intel Corporation und SystemSoft liegt in der Version 2.1 vor. [3]
PXE ist eine von Clientseitig verfügbarem Massenspeicher und von Betriebssystemen unabhängige Umgebung, welche einen netzwerkbasierten Bootvorgang von Rechnern erlaubt.

PXE kann per TFTP nur kleine Dateien von maximal 32 kByte laden. Aus diesem Grund muss zuerst ein kleines Programm (Bootlader) von PXE geladen werden, welches erst im Anschluss den viel größeren Kernel laden kann. Durch diesen Mechanismus bleibt PXE unabhängig von dem zu bootenden Betriebssystem, weil diese Prozeduren noch vor dem eigentlichen Booten des Betriebssystems passiert.

PXE ist an eine Netzwerkkarte gebunden, da es deren Treiber benötigt, um über das Netzwerk zu kommunizieren. Es ist in einem EPROM auf der Netzwerkkarte gespeichert oder bei onboard-Netzwerkadaptern Bestandteil des Mainboard-BIOS.

2.6.3.1 Funktionsweise von PXE

Nachdem die Kontrolle während des Bootvorganges an das Network Interface Card (NIC) BIOS übergeben wurde, sucht der PXE Client über das Netzwerk nach einem DHCP- oder Proxy DHCP Service, welches ihm Informationen über im Netz befindliche Bootserver übermittelt. Im nächsten Schritt kontaktiert der PXE Client den Bootserver, welcher dem Client entsprechende Informationen über Dateiname und Pfad des zu downloadenden NBP mitteilt. Per Multicast Trivial File Transfer Protocol (MTFTP) oder TFTP wird vom Client anschließend das NBP vom Bootserver über das Netzwerk in den Arbeitsspeicher des Clients geladen und dort ausgeführt.

2.6.3.2 Protokolldetails

Das Protokoll ist eine Kombination von Erweiterung des DHCP (durch die Verwendung verschiedener neuer DHCP Option Tags) und der Definition von einfachen Pakettransaktionen, welche die DHCP-Paketformate- und Optionen verwenden, um Informationen zwischen dem Client und dem Server auszutauschen. Eine Anforderung dabei ist, das bestehende DHCP Service nicht zu stören.

Die DHCP Optionsfelder werden in diesem Protokoll dazu verwendet, um folgende Anforderung zu ermöglichen:

- Unterscheidung von als Teil des PXE Protokolls gesendete DHCPDISCOVER und DHCPREQUEST Paketen zu anderen DHCP Paketen, welche der DHCP- oder Boot Server empfängt.
- Unterscheidung von als Teil des PXE Protokolls gesendete DHCPOFFER und DHCPACK Pakete zu anderen Paketen, welche der Client empfängt.
- Übertragung der Client System ID (UUID) an den DHCP- und Boot Server.
- Übertragung der Client Systemarchitekturtype an den DHCP- und Boot Server.
- Übertragung der Bootservertype, von welchem der Client eine Antwort anfordert.

2.6.3.3 Proxy DHCP

Die PXE DHCP Optionen können entweder von einem DHCP Service oder von einem Proxy DHCP Service bearbeitet werden. Das Proxy DHCP Service kann sich entweder auf demselben Server wie das DHCP Service befinden, oder auf einem separaten Server. Befindet sich der Proxy auf dem gleichen Server wie das DHCP Service, wird der UDP Port 4011 verwendet, und die Kommunikation erfolgt erst nach Beendigung des DHCP Protokolls. Port 4011 wird aus dem Grund verwendet da sich beide DHCP Services nicht den Port 67 teilen können.

2.6.3.4 PXE Boot

Die folgende Grafik veranschaulicht die prinzipiellen Schritte des PXE Protokolls beim Netzwerkbootvorgang.

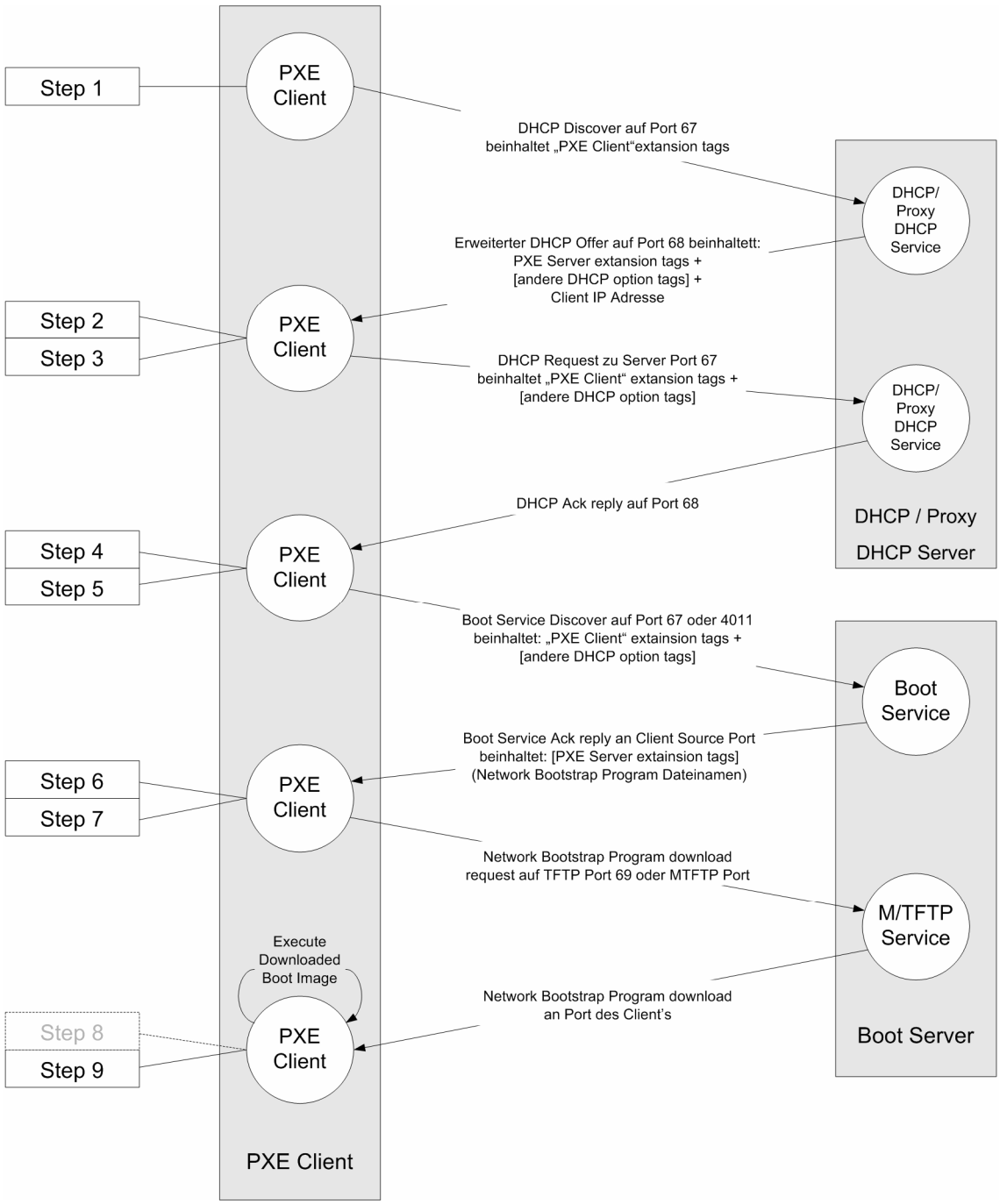

Abbildung 9: PXE Boot (vgl. [3])

Step 1:

Der PXE Client verschickt eine DHCPDISCOVER Nachricht an den Standard-Port 67. Das Optionsfeld in diesem DHCP-Paket beinhaltet folgende Einträge:

- Einen Tag für den „Client Identifier“ (UUID).
- Einen Tag für die Client Universal Network Device Interface (UNDI) Version.
- Einen Tag für die Client Systemarchitektur.
- Eine DHCP Option 60, Class ID, gesetzt auf „PXEClient:Arch:xxxxx:UNDI:yyyzzz“.

Step 2:

Der DHCP oder Proxy DHCP Dienst sendet als Antwort eine DHCPOFFER Nachricht retour an den Client auf den Standard DHCP-Port 68. Falls es sich um einen Antwort von einem Proxy DHCP Service handelt, ist das Client IP Adressfeld NULL (0.0.0.0). Bei einer Antwort von einem DHCP Service beinhaltet das IP Adressfeld die gültige IP-Adresse für den Client.

Step 3:

Der Client übernimmt von dem DHCPOFFER, welche er von den Servern erhält, folgende Punkte:

- Die Client IP-Adresse und weiter Parameter, welche von Standard DHCP oder BOOTP Services angeboten werden.
- Eine Bootserverliste aus dem Bootserverfeld in den PXE Tags vom DHCPOFFER.
- Discover Control Options (falls verwendet).
- Multicast Discover IP-Adresse (falls verwendet).

Step 4:

Falls sich der Client eine von einem DHCP Service angeboten Adresse einträgt, muss er das Standard DHCP-Protokoll komplettieren, indem er wieder einen Request an den Server retourniert und auf eine Bestätigung des DHCP Services wartet. Falls der Client eine Adresse von einem BOOTP-Service auswählt, kann er diese sofort verwenden.

Step 5:

Der Client wählt sich einen Bootserver aus der ihm angebotenen Liste und sendet ein Bootserverdiscoverpaket. Das Paket wird entweder per Broadcast (Port 67), Multicast (Port 4011) oder Unicast (Port 4011) versendet. Dies ist abhängig von den „discovery control options“ in den „PXE extension tags“ des vorangegangenen DHCPOFFER Paketes. Dieses Paket beinhaltet folgende Punkte:

- Die dem Client von einem DHCP Service zugewiesene IP-Adresse.
- Einen Tag für den „Client Identifier“ (UUID).
- Einen Tag die Client UNDI Version.
- Einen Tag für die Client Systemarchitektur.
- Eine DHCP Option 60, Class ID, gesetzt auf „PXEClient:Arch:xxxxx:UNDI:yyyzzz“.
- Den Bootservertyp in einem PXE Optionsfeld.

Step 6:

Der Bootserver retourniert per Unicast ein DHCPACK Paket an den Client auf den Client Source Port. Dieses Paket enthält folgende Informationen:

- Boot File Name
- M/TFTP Konfigurationsparameter
- Weitere Optionen, welche das NBP benötigt, bevor es erfolgreich ausgeführt werden kann.

Step 7:

Der Client lädt anschließend die ausführbare Datei vom Bootserver per TFTP (Port 69) oder per MTFTP (Port wurde im Boot Server Ack Paket zugewiesen). Die zu ladende Datei, sowie der Ort im Hauptspeicher, in welchem das File geladen wird, sind von der Clientarchitektur abhängig.

Step 8:

Nach dem vollständigen Download der Datei, bestimmt der Client ob eine Authentizitätsprüfung der Datei erforderlich ist oder nicht. Falls ja, sendet der Client eine weiter DHCPREQUEST Nachricht an den Bootserver, um ein „credentials files“ für das zuvor geladenen Boot File anzufordern. Der Client startet einen Download der „credentials“ via TFTP oder MTFTP und führt anschließend die Authentizitätsprüfung durch.

Step 9:

Schließlich leitet der PXE Client die Ausführung des soeben geladenen Codes ein.

Weitere genauere Informationen und Erklärungen zu PXE sind unter [3] nach zu schlagen.

2.6.4 PXELinux

Um ein Linux Betriebssystem über das Netzwerk von einem Server mittels PXE zu booten, benötigt man einen hierfür geeigneten Bootloader. PXELinux ist ein solches Network Bootstrap Program, welches dazu verwendet werden kann, um einen Linuxkernel über das Netz auf den Client-Rechner zu laden. PXELinux ist Bestandteil des mehrschichtig aufgebauten SYSLinux Paketes von Peter Anvin. (vgl. [12])

Für das Zusammenspiel von PXELinux mit dem PXE-Protokoll muss ein DHCP Service dem Client eine IP-Basiskonfiguration, den Namen der Bootdatei (*pxelinux.0*), sowie die Adresse eines Servers, von wo die Bootdatei bezogen werden kann, mitteilen. Ein weiterer wichtiger Bestandteil ist ein TFTP Service, welches die Bereitstellung des PXELinux-Programms, des Kernels- und des Inital Ramdisk Images übernimmt.

2.6.4.1 Funktionsweise von PXELinux

Wie bereits in Punkt 2.6.3 erläutert wird mittels des PXE-Protokolls ein Bootloader unter der Verwendung des TFTP Services von einem Bootserver auf den Client geladen. In diesem Falle handelt es sich um die Datei *pxelinux.0*. Nachdem *pxelinux.0* gestartet wurde, gibt PXE die Kontrolle ab. Das NBP bezieht nun Information wie es sich dem Benutzer gegenüber darstellen und verhalten soll von einer Konfigurationsdatei im Unterverzeichnis *pxelinux.cfg*. Die Konfigurationsdatei folgt dabei einem speziellen Bezeichnungsschema, sodass sie für jeden einzelnen Client, oder für ganze Gruppen von PCs eine eigene Konfigurationsdatei hinterlegen können. Im folgenden Beispiel wurde auf dem TFTP Server ein Verzeichnis „/tftpboot“ angelegt und pxelinux.0 in dieses Verzeichnis kopiert. Weiters wurde ein Verzeichnis „/tftpboot/pxelinux.cfg“ angelegt.

Da man ja für jeden Client ein eigenes Konfigurationsfile anlegen kann, beginnt *syslinux.0* mit der Suche nach einem Dateinamen, welcher sich aus dem Hardwaretype- und adresse in hexadezimalen Kleinbuchstaben mit Minuszeichen getrennt zusammensetzt. Zum Beispiel würde pxelinux.0 bei einem Rechner mit der Hardwaretype 01 (Ethernet = Type 1) und der Hardwareadresse 00:0B:6A:4A:97:57 zuerst nach einem File mit dem Namen 01-00-0b-6a-4a-97-57 suchen.

Im nächsten Schritt würde das Programm nach der hexadezimalen Form der eigenen IP-Adresse suchen. Bei einer IP-Adresse von 192.168.0.1 würde *pxelinux.0* nach einem File namens *C0A80001* suchen.

Falls das Programm wieder nicht fündig geworden ist, entfernt es eine hexadezimale Stelle und beginnt mit einer erneuten Suche. In diesem Falle nach *C0A8000*. Falls der Name komplett aufgelöst wurde und noch immer kein Treffer erfolgte, sucht das Programm nach einem Konfigurationsfile namens *default*.

Die Suchschritte des Programms für oben genannte Hardware- und IP-Adresse würden folgendermaßen aussehen:

/tftpboot/pxelinux.cfg/01-00-0b-6a-4a-97-57
/tftpboot/pxelinux.cfg/C0A80001
/tftpboot/pxelinux.cfg/C0A8000
/tftpboot/pxelinux.cfg/C0A800
/tftpboot/pxelinux.cfg/C0A80
/tftpboot/pxelinux.cfg/C0A8
/tftpboot/pxelinux.cfg/C0A
/tftpboot/pxelinux.cfg/C0
/tftpboot/pxelinux.cfg/C
/tftpboot/pxelinux.cfg/default

Auf diese Art und Weise habe ich als Administrator die Möglichkeit Rechner in Gruppen zusammenzufassen, um beispielsweise eine Gruppe von Maschinen vom Netz boot zu lassen, hingegen PCs eines anderen Subnetzes lokal von der Festplatte.

2.6.4.2 Konfiguration von PXElinux

An Hand der Konfigurationsdatei von PXElinux kann man sehen, dass sie sich sehr stark am Aufbau von Konfigurationen anderer Bootloader orientiert.

In Tabelle 5 ist eine stark vereinfachte, aber typische PXElinux - Defaultkonfigurationsdatei, welche automatisch unter Verwendung des Open Source Project – Thinstation 2.1[3] erzeugt wurde, dargestellt. Im ersten Teil befinden sich Optionen des Bootloaders betreffend und im zweiten Abschnitt Bootoptionen.

Tabelle 5: Beispiel einer PXElinux-Konfigurationsdatei

00	default linux
01	prompt 0
03	label linux
04	kernel vmlinuz
05	append ramdisk_blocksize=4096 initrd=initrd root=/dev/rd/0 \ ramdisk_size=262144 splash=silent vga=791 console=ttyS3

Zeile 00 zeigt an, dass defaultmäßig das Linux Image geladen werden soll.

Zeile 01 gibt an, dass PXElinux auf keine Benutzereingabe warten soll, sonder unmittelbar mit dem Laden des Kernel Images beginnen soll.

In Zeile 04 steht der Pfadname der Imagedatei, welche per TFTP vom Bootserver in den Speicher des Clients geladen wird. In diesem Fall liegt *vmlinuz* direkt im Root-Verzeichnis des TFTP Servers.

In Zeile 05 finden sich die Bootoptionen, welche dem Kernel beim Booten übergeben werden.

[3] siehe http://thinstation.sourceforge.net/wiki/index.php/ThIndex

2.6.5 TFTP (Trivial File Transfer Protocol)

Das TFTP ist wie der Name bereits verrät eine sehr vereinfachte Form des File Transfer Protocol (FTP). TFTP bietet eine zuverlässige Übertragung von Dateien auf der Basis des UDPs (User Datagram Protocols) auf Port 69. Da es sich um ein sehr einfach gehaltenes Dateiübertragungsprotokoll handelt, ist nur ein sehr geringer Bedarf an Speicherplatz notwendig, um einen TFTP-Client im System abzulegen. Solche Clients finden daher sehr leicht auf dem EPROM einer Systemplatine oder eines Netzwerkadapters platz. TFTP ist vorwiegend dazu gedacht, um in LANs (Local Area Networks) Systeme über das Netzwerk in Betrieb zu nehmen (Bootstrapping).

2.6.5.1 Protokolldetails

Abbildung 10 zeigt in vier Schritten wie prinzipiell eine Datenübertragung zwischen einem TFTP-Client und einem Server initiiert, durchgeführt und beendet wird.

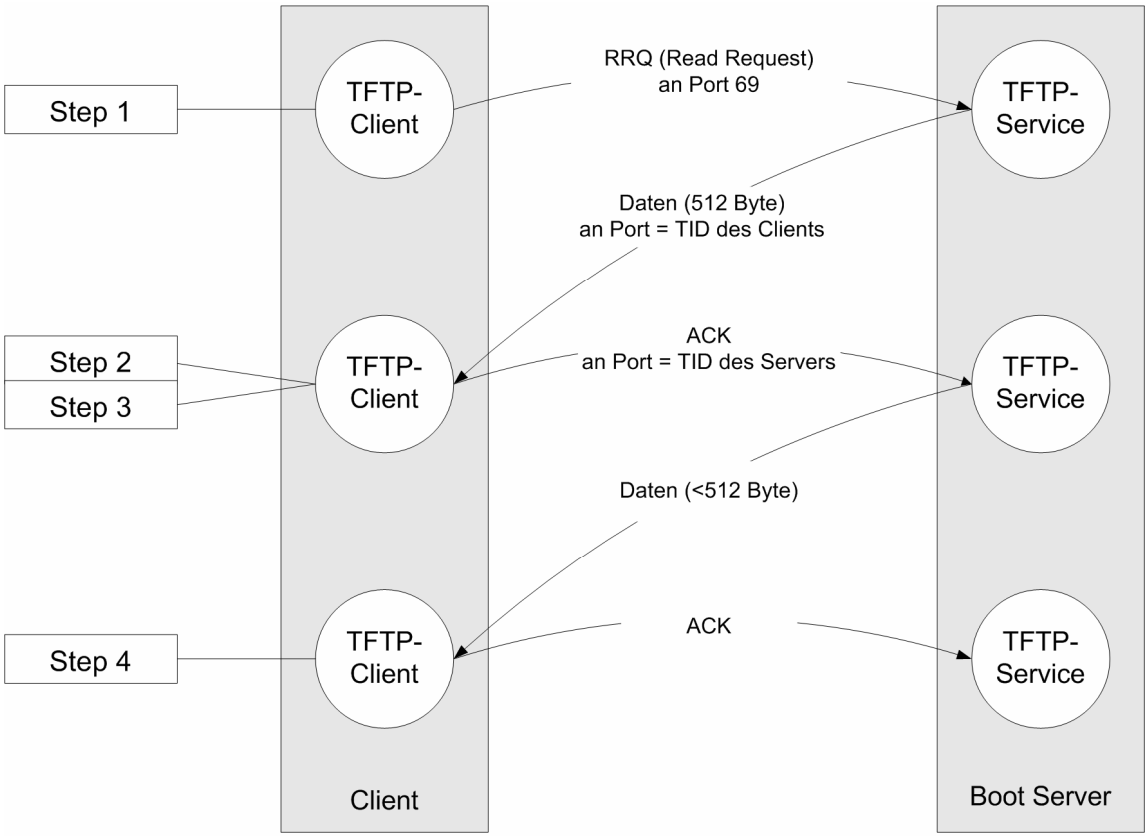

Abbildung 10: TFTP-Übertragungsmechanismus (vgl. [11])

Step 1:

Jeder Datentransfer beginnt mit einem read oder write request auf eine Datei am Server, welcher vom Client initialisiert wird. Dabei sendet der Client ein Requestpaket an Port 69 des Servers. Der Client wählt eine zufällige Zahl zwischen 0 and 65,535 als TID (Transfer Identifier), welche zugleich dem UDP-Layer als Source Port übergeben wird.

Step 2:

Falls das TFTP-Service die Verbindung zulässt, wird vom Server ebenfalls ein TID erzeugt, welche auch zugleich als Source Port des Servers fungiert und im Datenpaket eingefügt. Die versendeten Datenblöcke haben immer eine fixe Größe von 512 Byte.

Step 3:

Jedes vom Server versendete Datenpaket, muss vom Client mit einem Acknowledgment-Paket bestätigt werden, bevor das nächste Paket versendet werden kann.

Step 4:

Datenpaket die *kleiner* als 512 Byte sind, signalisieren dem Client das Ende des Datentransfers. Hätte das letzte Datenpaket ebenfalls genau die Größe von 512 Bytes würde ein eigenes Paket generiert um ein Transferende zu signalisieren.

2.6.5.2 tsize Option

PXElinux benötigt für eine reibungslose Funktion einen TFTP Server, welcher die „tsize“ Option unterstützt. (vgl. [12])

3 Vergleich von Kommunikationsprotokollen

Eine Schlüsselkomponente zur Bereitstellung von Terminaldiensten ist das zur Kommunikation zwischen den Clients und den Servern notwendige Remote Display Protokoll. Dieses Protokoll ermöglicht es Bildschirmdaten, sowie Daten von Tastatur und Maus über ein Netzwerk zu transportieren.

Im Zuge dieser Arbeit möchte ich die drei Protokolle RDP, ICA und X, welche in enger Verbindung mit Thin-Clients und dem Server Based Computing (SBC) stehen, auf deren Performance bei der Netzwerkübertragung hin vergleichen.

3.1 Überblick über die Protokolle

3.1.1 Remote Desktop Protocol (RDP)

Das RDP ist ein proprietäres Anwendungsprotokoll von Microsoft und bildet eine Schlüsselkomponente für die Terminalservices von Microsofts Windows Server 2000/2003. An Hand des RDP wird es dem Endanwender ermöglicht sich von einem RDC-Client mit einem Windows Terminal Server (siehe 2.3.2) zu verbinden.

Microsoft lizenzierte 1997 die von Citrix entwickelte MetaFrame Technologie für deren Terminal Servervariante. Die Lizenzierung enthielt aber nicht das Anwendungsprotokoll ICA von Citrix und so entwickelte Microsoft deren eigenes Protokoll, das RDP. Seit Windows XP Service Pack2 und Windows Server 2003 ist die letzte aktuelle Version RDP 5.2

3.1.1.1 RDP-Basisarchitektur

Das Anwendungsprotokoll arbeitet auf der obersten Ebene des TCP/IP-Referenzmodell (Anwendungsschicht) wie auch zum Beispiel HTTP, Simple Mail Transfer Protocol (SMTP) oder FTP. RDP wurde zwar für den Einsatz unter verschieden Typen von Netzwerktopologien (Integrated Services Digital Network (ISDN), Plain Old Telephone Service (POTS), verschieden LAN-Protokolle wie Internetwork Packet eXchange (IPX), NetBIOS, TCP/IP) designed, unterstützt bis weilen bei der Übertragung aber nur das Transportprotokoll TCP/IP und verwendet dabei standardmäßig den TCP Port 3389,

obwohl auch ein anderer Port zur Kommunikation zwischen Server und Client verwendet werden kann.

RDP verwendet seinen eigenen Videotreiber, um Bildschirmausgaben serverseitig zu rendern. Die Rendering-Information wird anschließend in Datenpakete gepackt und per RDP über das Netzwerk dem Client zugestellt. Auf der Clientseite werden die Informationen anschließend an Hand der entsprechenden Win32 Graphics Device Interface (GDI) Application Programmer Interface (API) Aufrufen interpretiert (vgl. [20]). Die Bildschirmausgabe wird beim RDP-Protokoll als Rastergrafik (Bitmap) vom Terminal Server zum Client übertragen. (vgl. [22])

3.1.1.2 RDP-Datenfluss

Der Datenfluss beim RDP-Protokoll zwischen dem Server und dem Client kann in zwei Komponenten aufgeteilt werden, in Bildschirmdaten und in Maus/Tastaturdaten.

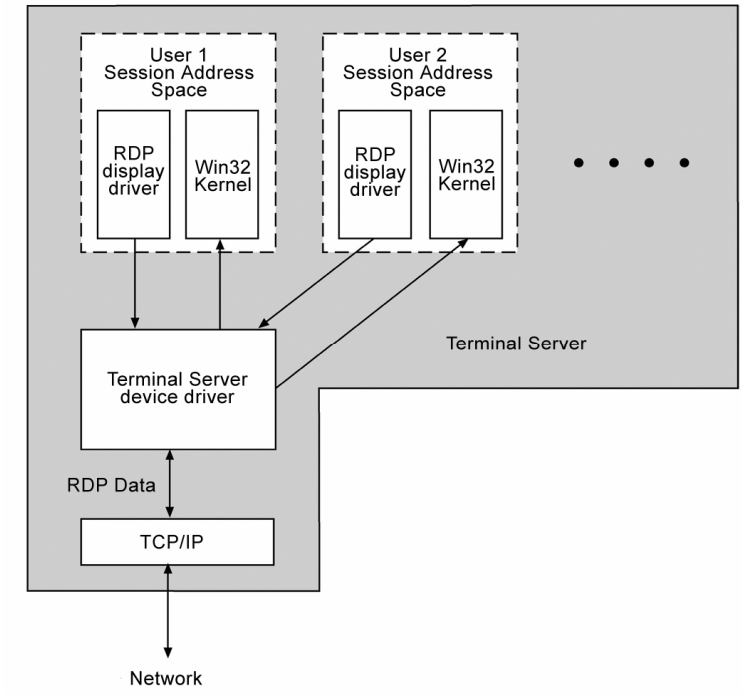

Abbildung 11: RDP-Datenfluss (vgl. [21])

Wie in Abbildung 11 ersichtlich, hat jede Usersession ihren eigenen Session Address Space, welche wiederum einen eigenen Win32 Kernel, einen RDP-Displaytreiber und einen Druckertreiber enthält. Jeder RDP-Displaytreiber erhält seine Anweisungen von dem GDI und leitet diese Informationen anschließend an den Terminal Server Device

Driver weiter. Dieser Treiber verarbeitet die Daten zu RDP-Daten, welche dann zu den Clients geschickt werden. Im Anschluss werden die Daten an den TCP/IP-Protokollstack gesendet und weiter an die Clients, wo die Daten wieder verarbeitet werden und am Bildschirm angezeigt werden. Im Gegenzug werden die Daten von Maus und Tastatur am Client in RDP-Daten umgewandelt und an den Server gesendet. Der Terminal Server Gerätetreiber verarbeitet die ankommenden Daten von den Clients und sendet diese an den Win32-Kernel im entsprechenden Sitzungsadressraum zur weiteren Verarbeitung.

3.1.1.3 Virtuelle Kanäle

Das RDP basiert auf und ist auch zugleich eine Erweiterung der seit langem verfügbaren T.120 International Telecommunications Union (ITU) Standardprotokollfamilie. Das Protokoll unterstützt aber mehr als nur die Übertragung von virtuellen Computerarbeitsplätzen. RDP ist, wie in Abbildung 12 dargestellt, ein Mulitkanal-Protokoll, welches auf mehreren separaten virtuellen Datenkanälen Präsentationsdaten (Audio), serielle Gerätekommunikation, Lizenzinformationen, verschlüsselte Daten (Tastatur, Mausaktivitäten), usw. übertragen kann. Damit kann es auf diesen virtuellen Kanälen Daten zwischen einer Terminal Serversitzung (Session) und einem Client transportieren. Theoretisch sind bei einer RDP Verbindung bis zu 64.000 separate Kanäle zur Datenübertragung möglich.

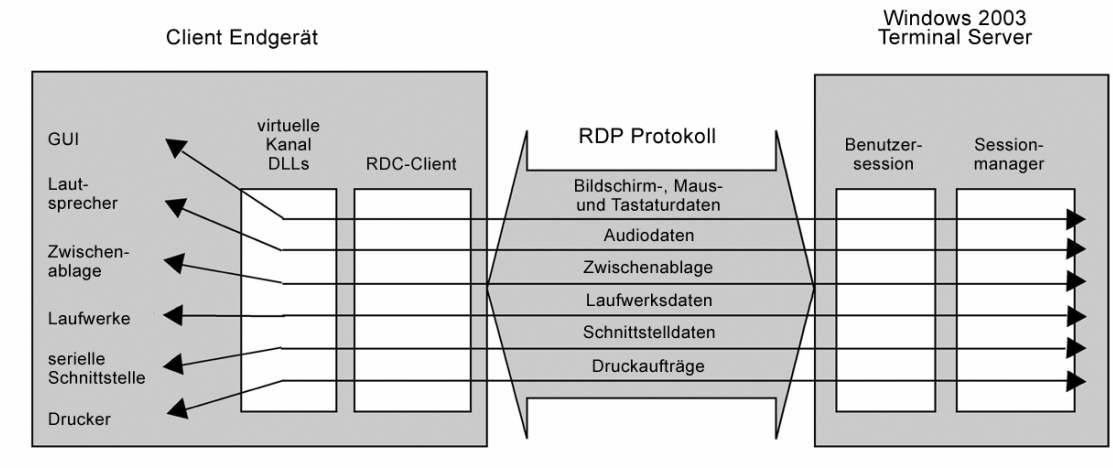

Abbildung 12: RDP Virtuelle Kanal Architektur [16]

In der ersten Veröffentlichung des Windows Terminal Servers wurde hauptsächlich auf eine hohe Zuverlässigkeit und Schnelligkeit des Protokolls geachtet und nur ein Kanal zur Datenübertragung verwendet (vgl. [17]). Standardmäßig werden beim Windows Terminal Server 2003 mehrere virtuelle Kanäle verwendet.

- *Audio:* Soundereignisse, welche bei einer Serversitzung generiert werden, werden über das RDP Protokoll an den RDC-Client übertragen, wo diese über die PC-Lautsprecher wiedergegeben werden können.

- *Client Laufwerke:* Lokale Laufwerke auf den Thin-Clients können über den Terminal Server per RDP Protokoll für die Anwender verfügbar gemacht werde. Somit können Benutzer nicht nur die Serverlaufwerke, sondern auch die lokalen Laufwerke auf den Thin-Clients weiter benützen.

- *Drucker:* Über das RDP Protokoll ist es auch möglich lokal angeschlossene Drucker weiter zu verwenden.

- *Serielle Ports:* Über den virtuellen Kanal, ist es möglich an die serielle Schnittstelle des Thin-Client angeschlossene Geräte bei einer Terminal Serversitzung weiter zu betreiben.

- *Windows Zwischenablage:* Bei Endgeräten auf denen Programme lokal ausgeführt werden, können Daten in der Zwischenablage auch auf der Terminal Serversitzung übernommen werden. Anwender können nahtlos die Kopier- und Einfügefunktion zwischen lokalen Anwendungen und Anwendungen auf dem Terminal Server verwenden.

3.1.1.4 Bandbreite

RDP unterstützt unterschiedliche Mechanismen, um das Datentransfervolumen einer Netzwerkverbindung zwischen einem Client und einem Terminal Server deutlich zu senken. Dazu benutzt Microsoft im RDP-Protokoll ein Datenkompressionsverfahren und Cachen von Bitmaps und Glyphen im Arbeitsspeicher.

Durchschnittlich benötigt eine Terminal Serversitzung um die 31 KBit/s, abhängig von den verwendeten Applikationen. (vgl. [16])

3.1.1.5 Verschlüsselung

Microsoft verwendet bei jeder Version des RDP Protokolls den Stromchiffre RC4[4] (Ron's Code 4) welcher 1987 von Ronald L. Rivest für RSA Data Security Inc. (heute RSA Security) entwickelt wurde (vgl. [18]), und unterstützt seit Windows Server 2003 SP1 auch Secure Socket Layer (SSL) / Transport Layer Security (TLS) 1.0 zur Serverauthentifizierung.

Die Verschlüsselungsstufe wird beim Erstellen einer Verbindung zwischen Server und Client ausgehandelt und wird von den Einstellungen auf dem Server bestimmt. Bei der Verschlüsselung des Datenverkehrs einer Terminal Serversitzung stehen dem Administrator vier Varianten zur Auswahl. Folgende Tabelle gibt eine Übersicht über die unter Windows Servers 2003 und RDP 5.2 implementierten Verschlüsselungsstufen. (vgl. [19])

Tabelle 6: RDP-Verschlüsselungsstufen

Verschlüsselungsstufe	Beschreibung
FIPS-konform	Bei einer Verschlüsselung auf dieser Stufe, werden die Daten, welche vom Server zum Client, beziehungsweise vom Client zum Server übertragen werden, mit den Verschlüsselungsalgorithmen des FIPS (Federal Inforamtion Processing Standard)[5] mit Hilfe der Microsoft Krypthografiemodule ver- und entschlüsselt.
hoch	Bei dieser Verschlüsselungsstufe werden alle Daten in beide Richtungen unter Verwendung eines 128-Bit Schlüssels verschlüsselt. Falls aber Clients diese Stufe nicht unterstützen, können diese mit dem Server keine Verbindung aufbauen.

[4] http://www.rsasecurity.com/rsalabs/node.asp?id=2250
[5] http://www.itl.nist.gov/fipspubs/

clientkompatibel	In diesem Fall werden die Kommunikationsdaten auf der maximalen vom Client unterstützten Stufe verschlüsselt. Diese Stufe ist sinnvoll bei einer Umgebung mit verschiedenen Clientversionen.
niedrig	Bei der Verwendung dieser Verschlüsselungsstufe wird nur der Datenstrom vom Client zum Server mit einem 56-Bit Schlüssel verschlüsselt. Daten die vom Server an dem Client gesendet werden, werden bei dieser Stufe nicht verschlüsselt.

3.1.1.6 Remote Desktop Connection (RDC) Client

Ein RDC Client ist bereits in Windows XP und Windows Server 2003 implementiert. Für weitere Microsoft Betriebssysteme, wie Windows 95, Windows 98, Windows 98 Second Edition, Windows ME, Windows NT 4.0, Windows 2000 oder Windows CE können entsprechende Clients nachinstalliert werden. Für das einzige nicht von Microsoft stammende Betriebssystem, für welches von Microsoft selbst ein Client bereitgestellt wird, ist das Betriebssystem von Apple, nämlich MAC OS X.

Eine weitere Variante wäre die Verbindung zu einem Terminal Server über einen Webbrowser mit Hilfe der ActiveX-Technologie.

3.1.1.6.1 Drittanbieter

Seit einiger Zeit sind auch RDC Clients für andere Plattformen verfügbar, mit denen es möglich ist sich mit einem native Windows Terminal Servern zu verbinden, ohne dass dafür eine Serversoftware eines Drittanbieters installiert werden muss.

Für Linux- oder Unixplattformen gibt es einen quelloffenen RDP Client von www.rdesktop.org. Dieser Client ist kostenlos verfügbar und steht unter der GPL.

Selbst eine kostenpflichtige Version eines RDC Client (DOSRDP 2.0)[6], für den Einsatz auf einer DOS-Plattform, ist im Internet per download verfügbar.

Von der Firma HOB[7] ist eine plattformunabhängige Java-Version eines RDC Clients erhältlich. Mit Hilfe dieses Clients ist man nicht mehr an eine bestimmte Hardware- oder Betriebsystemplattform gebunden.

Eine weiter Firma namens DDH Software[8] stellt einen RDC Client für Palm OS Handhelds bereit, über welchen man sich mit einem Windows Terminal Server verbinden kann.

[6] http://www.soft14.com/Utilities_and_Hardware/Remote_computing/DOSRDP_2811_Review.html
[7] http://www.hobsoft.com/
[8] http://www.ddhsoftware.com/

3.1.2 Independent Computing Architecture (ICA)

Die Grundlage zur Verwendung aller über einen Citrix Presentation Server bereit gestellter Ressourcen stellt ein von Citrix selbst entwickeltes proprietäres Protokoll. Dieses Protokoll trägt den Namen Independent Computing Architecture und spezifiziert die Übertragung von Daten zwischen Servern und Clients. Der Zugriff über das ICA-Protokoll erfolgt von eigenen ICA-Clients aus, welche Citrix für eine Vielzahl von Plattformen zur Verfügung stellt.

3.1.2.1 ICA-Basisarchitektur

Bei dem ICA-Protokoll handelt es sich um ein Anwendungsprotokoll auf oberster Ebene des TCP/IP-Referenzmodell (Anwendungsschicht). Das ICA-Protokoll unterstützt bei der Netzwerkkommunikation mehrere Netzwerkprotokolle, wie TCP/IP, IPX, Sequence Packet eXchange (SPX) oder NetBEUI. Weiters kann auch eine direkte asynchrone Verbindung, beispielsweise über Nullmodemkabel, verwendet werden (vgl. [23]). Dies macht ICA auch für Novell-Netzwerke sehr interessant. Bei der Kommunikation zwischen einem ICA-Client und einem Presentation Server über ein TCP/IP-Netz erfolgt der Datenaustausch standardmäßig über den TCP-Port 1494. (vgl. [24])

3.1.2.2 ICA-Datenfluss

Bei der Datenübertragung, wie in Abbildung 13 zu sehen, erhält zuerst ein Input-Buffer Daten von einem virtuellen ICA-Kanal. Beim Erstellen eines ICA Datenpaketes werden immer nur ICA-Daten eines Kanals zusammengepackt. Diese Daten werden abhängig von den Konfigurationseinstellungen komprimiert oder nicht und an einen Output-Buffer weitergeleitet. Anschließend werden die Daten an den Encryption Protocol Driver weitergeleitet. Dies erfolgt ebenfalls nur wenn dies in der Konfiguration vorgesehen ist. Danach wird an das Datenpaket, falls notwendig, noch ein Reliability-Header vorgesetzt und weiter an den Framing Protocol Driver geschickt. Danach werden die Daten durch den Transport Driver TCP (TDTCP) an den TCP/IP-Stapel weitergeleitet. Der TDTCP ist die ICA-Schnittstelle zu dem TCP/IP-Stapel und fügt dem Datenpaket keine weiteren Informationen hinzu. Bei dem Erhalt eines ICA-Datenpaketes, nehmen die ICA-Daten den gleichen Weg durch die ICA Protokollebenen, aber logischerweise in umgekehrter Reihenfolge. Alle ICA-Protokollebenen befinden sich auf der Darstellungsschicht des Open System Interconnection (OSI) Referenzmodells. (vgl. [26])

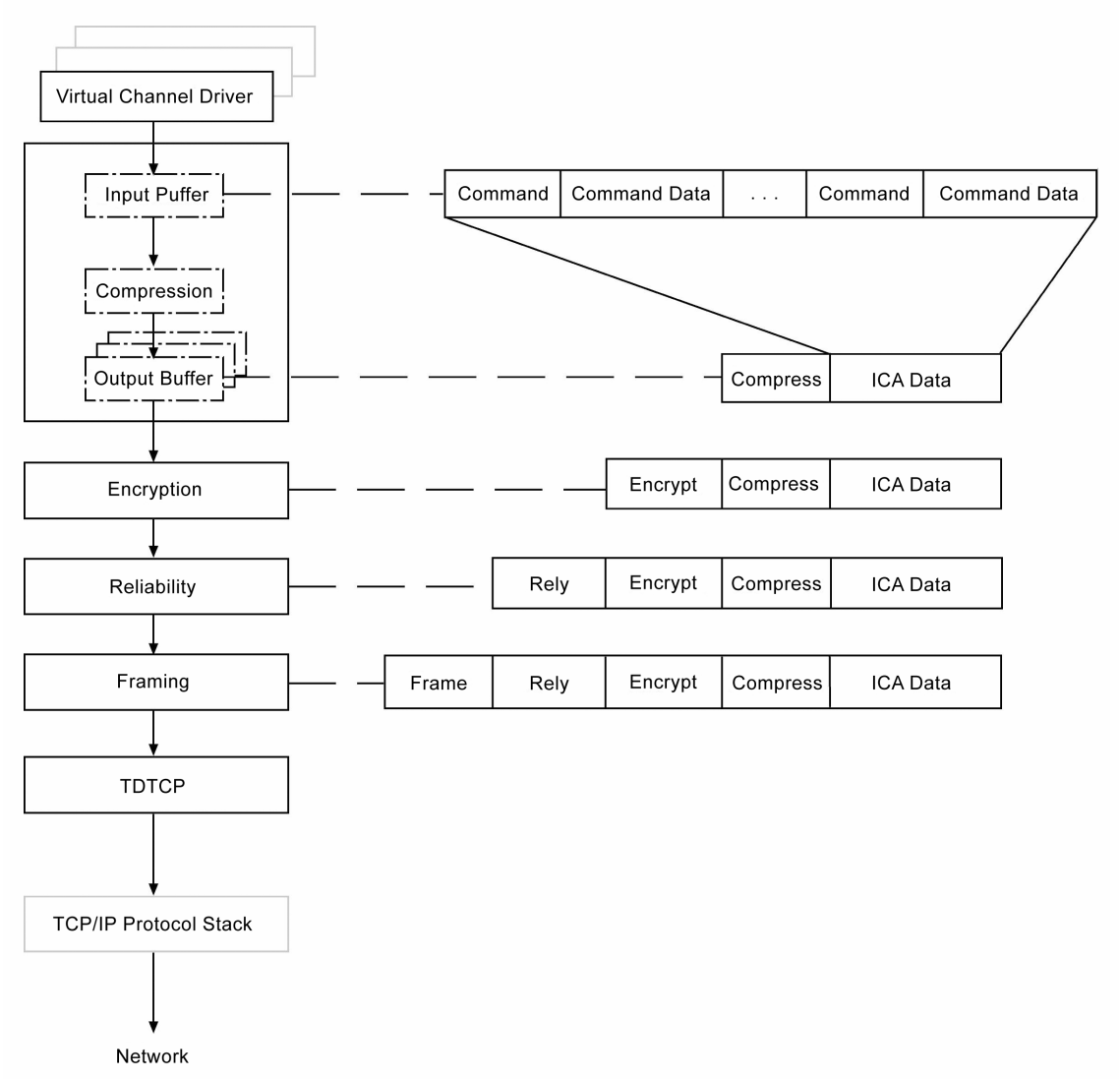

Abbildung 13: ICA Datenfluss (vgl. [26])

3.1.2.3 ICA-Datenpaket

Abbildung 14 zeigt die Komponenten eines ICA-Datenpaketes, welches es von dem TDTCP an den TCP/IP-Stack weitergegeben wird. So ein Datenpaket enthält immer nur Daten von einem virtuellen Kanal. Die maximale Größe für ein ICA-Datenpaket beträgt 2 KByte (2.048 Bytes).

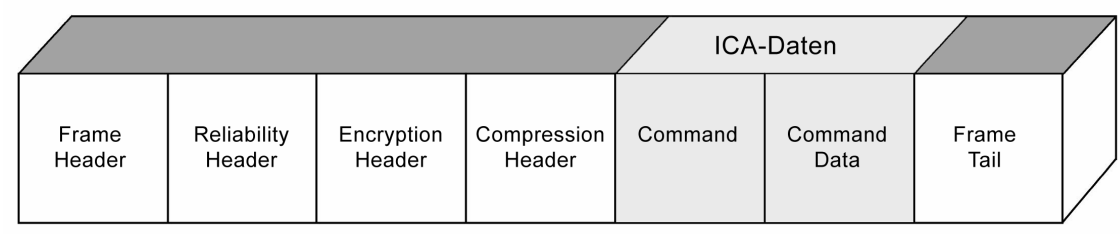

Abbildung 14: Das ICA-Datenpaket [21]

- *Frame Header:* Der Frame-Header besteht aus 16 Bit und enthält unter anderen zwei Bit, mit welchen die ICA-Daten unterschiedlich priorisiert werden können (ICA Priority Packet Tagging). So ist zum Beispiel möglich Bildschirmdaten eine höhere Priorität zu geben, als Daten welche für den Drucker bestimmt sind.

- *Reliabilty Header:* Der Header für das Reliabilty Protocol wird verwendet, wenn der Datentransfer über ein unzuverlässiges Transportprotokoll wie IPX erfolgt.

- *Encryption Header:* Ein Encryption-Header wird eingefügt, falls eine Verschlüsselung der Daten erfolgt. Somit sind alle Daten im Anschluss an den Encryption-Header verschlüsselt.

- *Compression Header:* Falls die Daten auch komprimiert werden, wird noch ein Compression-Header angefügt.

- *Command:* Enthält das Kommandobyte für das ICA-Paket (erforderlich).

- *Command Daten:* Nach dem Kommandobyte folgen die Kommandodaten, welche aber optional sind.

- *ICA Daten:* Die ICA-Daten bestehen aus einem oder mehreren Kommandos und Kommandodatenpaketen. Die ICA-Daten können auch nur aus einem einzigen Kommandobyte ohne Kommandodaten bestehen. Umgekehrt können aber ICA-Daten nicht nur aus Kommandodaten bestehen. Kommandodaten brauchen immer ein Kommandobyte.

- *Frame Tail:* Signalisiert das Ende des Rahmen

3.1.2.4 Virtuelle Kanäle

Zur Kommunikation zwischen Client und Server verwendet das ICA-Protokoll, wie das RDP-Protokoll von Microsoft, virtuelle Datenkanäle. Das ICA-Protokoll von Citrix besteht aus 32 Kanälen, von denen jeder für eine bestimmte Funktion genutzt werden kann. Zurzeit werden in der aktuellen Version aber noch nicht alle der 32 Kanäle genutzt, was noch einiges an freien Ressourcen für zukünftige Entwicklungen bedeutet (vgl. [24]). Den einzelnen virtuellen Übertragungskanälen können vier verschiedene Prioritätsstufen zugewiesen werden (vgl. [26]). Folgende separate virtuelle Kanäle werden unter anderem für folgende Daten verwendet:

- *Bildschirmdaten:* Bei der Übertragung der Bildschirmdaten werden nur die Änderungen im Bildschirmaufbau vom Server zum Client übertragen. Unterstützte das ICA-Protokoll anfänglich eine Auflösung von 1024 x 768 Bildpunkten bei einer Farbtiefe von 8 Bit (256 Farben), kann heute eine Auflösung von 64000 x 64000 Bildpunkten bei einer Farbtiefe von 24 Bit (16 Millionen Farben) erreicht werden. (vgl. [23])

- *Tastatur- und Mausdaten:* Auf einem weiteren Kanal wird der Tastatur- und Mausdatenstrom vom Client zum Presentation Server übertragen.

- *Audio:* Die Audiodatenübertragung erfolgt seit Presentation Server 4 bidirektional. Das heißt das Audiodaten vom Server zum Client und auch vom Client zum Server übertragen werden können. Das ist vor allem beim Einsatz von Mikrofonen auf der Clientseite interessant. Es kann zwischen drei Tonqualitätsstufen bei der Übertragung gewählt werden. (vgl. [25])

    - *geringe Tonqualität*: für Verbindungen mit geringer Bandbreite empfohlen. Die Audiodaten werden dabei um maximal 16 KBit/s komprimiert.

    - *mittlere Tonqualität:* für die meisten LAN-basierten Verbindungen empfohlen. Audiodaten werden auf maximal 64 KBit/s komprimiert.

    - *hohe Tonqualität*: nur für Verbindungen mit großer Bandbreite empfohlen. Audiodaten werden auf höchster Qualitätsstufe bei ungefähr 1,3 MBit/s übertragen.

- *Drucker:* Druckdaten können ebenfalls wie bei dem RDP-Protokoll auf Druckern, welche auf den Clients angeschlossen sind, umgeleitet werden.

Die virtuellen Kanäle werden bei dem Verbindungsaufbau zwischen Client und Server ausgehandelt. Durch die Nichtbenützung von einigen Kanälen, wie zum Beispiel des Laufwerks-, Drucker- oder Audiomappings kann das Protokoll schlank gehalten werden.

3.1.2.5 Bandbreite

Citrix beziffert die durchschnittlich benötigte Bandbreitenanforderung eines ICA-Clients pro Serversitzung mit ungefähr 10 – 20 KBit/s, macht dazu aber keine genaueren Angaben unter welchen Bedingungen diese Daten ermittelt wurden.

Somit eignet sich das ICA-Protokoll von Citrix besonders für eine Anbindung von Anwendern über ein 56 KBit/s Modem oder über ein ISDN-Modem an einen Presentation Server. (vgl. [23])

3.1.2.6 Verschlüsselung

Citrix implementierte in fast allen ICA-Clients eine RC5-Verschlüsselung[9] für Anmelde- und Sitzungsdaten. Bei der Konfiguration kann man hierbei zwischen fünf Verschlüsselungsstufen wählen:

Tabelle 7: ICA Verschlüsselungsstufen

Verschlüsselungsstufe	Beschreibung
Basic	Bei der Verschlüsselungsstufe Basic wird ein einfacher Verschlüsselungsalgorithmus mit einem Schlüssel kleiner als 40 Bit verwendet. Diese Stufe kann nicht mehr als sicher betrachtet werden, da bereits Programme zum Entschlüsseln der Basisstufe im Internet verfügbar sind.

[9] http://www.rsasecurity.com/rsalabs/node.asp?id=2251

nur Anmeldung mit 128-Bit	Bei dieser Stufe werden nur die Anmeldeinformationen bei der Übertragung verschlüsselt. Die Daten einer Terminalsitzung werden unverschlüsselt übertragen.
40-, 56- und 128-Bit	In diesem Fall werden die Daten des Anmeldeprozesses mit 128 Bit verschlüsselt. Sitzungsdaten werden mit der gewählten Schlüssellänge verschlüsselt.

3.1.2.7 ICA-Client

Citrix stellt eine Reihe von ICA-Clients für unterschiedliche Hardware- und Betriebssystemplattformen bereit. Citrix unterstützt mit seinen Clients nahezu allen Windowssysteme, sowie Unix- und Linuxplattformen, bis hin zu Apple und IBM. Diese Clients können kostenlos von der Citrix Homepage[10] downgeloaded werden.

3.1.2.7.1 Drittanbieter

Die Firma Thinstall Corp.[11], eine Partnerfirma von Citrix, entwickelte einen virtualisierten ICA-Client, welcher auf Windowsplattformen ab Windows 98, ohne lokale Installation ausgeführt werden kann.

[10] http://www.citrix.com
[11] http://www.thinstall.com

3.1.3 X Window System (X, X11)

Von den hier behandelten Protokollen ist X das älteste System zur Übertragung von Ein- und Ausgabedaten zwischen Thin-Clients und Servern. X kommt hauptsächlich im Unix-und Linux Umfeld zur Bereitstellung einer grafischen Benutzeroberfläche zum Einsatz. Die Entwicklung von X wurde bereits 1984 am Massachusetts Institute of Technology (MIT) begonnen. X11 wurde 1987 veröffentlicht und seit her immer weiter entwickelt. Die aktuelle Version von X liegt zurzeit in dem Release X11R7.1 vor.

3.1.3.1 X-Basisarchitektur

Die Kommunikation von X basiert auf dem Client Server Computermodell, wobei die Rollen anders verteilt sind als man üblich vermutet. Im Falle von X läuft der X-Server nämlich auf dem Endgerät, wobei der X-Client entweder ebenfalls lokal auf dem Endgerät, oder auf einem entfernten Rechner seinen Dienst erledigt. Standardmäßig verwendet X den TCP Port 6000 für das erste Display zur Übertragung seiner Daten. Falls auf dem System ein zweites Display verwendet wird horcht dieses auf Port 6001. Ein weiters auf Port 6002 und so weiter.

Der **X-Server** hält die Kontrolle über die Benutzerschnittstellen, wie den Bildschirm, die Tastatur und die Maus. Er kennt drei Nachrichten Typen:

- Event
- Reply
- Error

Der **X-Client** ist das Anwendungsprogramm, welches vollkommen vom X-Server getrennt ist. Der X-Client benutzt die Dienste des X-Servers um mit dem Anwender zu interagieren. Der X-Client verwendet nur eine Nachrichtentype:

- Request

Die vollständige Trennung von Server und Client ist das Fundament von X, was es dadurch zu einem sehr flexiblen System macht.

X ist an keinen bestimmten Netzwerktyp gebunden und ist aus diesem Grund netzwerkunabhängig. Voraussetzung ist aber die Verwendung von zuverlässigen Transportprotokollen wie TCP/IP, DECnet, AppleTalk oder OSI.

X kann sowohl über Ethernet, Token Ring, Fiber Distribution Data Interface (FDDI) oder auch X.25 Netzwerke übertragen werden. Weiters kann X auch über serielle Datenverbindungen eingesetzt werden.

3.1.3.2 X Datenfluss

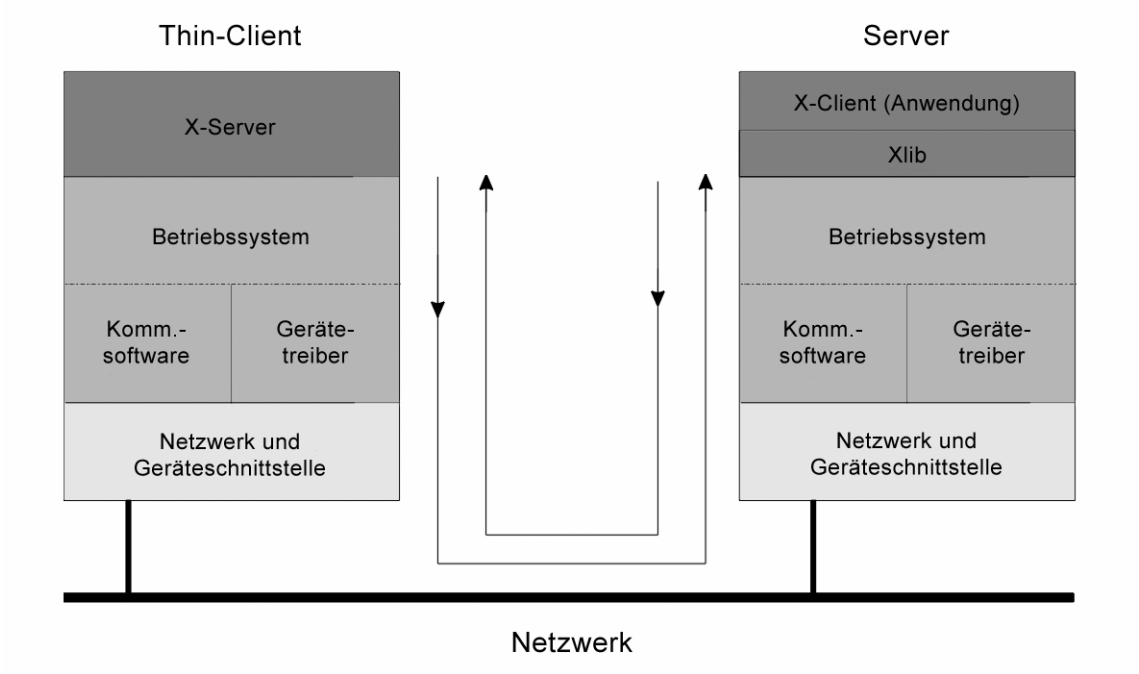

Abbildung 15: X Datenfluss (vgl. [27])

X ermöglicht eine Kommunikation zwischen einem X-Server und einem X-Client (Anwendung) entweder direkt auf dem Rechner, oder über eine Netzwerkverbindung. Befinden sich X-Server und X-Client auf demselben Computer, wird die Kommunikation über eine Inter-Prozess Communication (IPC) Software abgewickelt. Bei einer Kommunikation über eine Netzwerkverbindung wie in Abbildung 15, werden die Requests und Events an eine Netzwerktransportsoftware (zum Beispiel TCP/IP, DECnet oder OSI) übergeben, welche die Kommunikation über das Netzwerk regelt. Dabei verwaltet auf der X-Client Seite die *Xlib* Programmbibliothek die Kommunikation mit der Netzwerksoftware.

3.1.3.2.1 Request

Requests stellen den fundamentalen Mechanismus bei der Übertragung von Daten vom X-Client zum X-Server dar. Mit Hilfe dieser Requests kann der X-Client dem Server Anweisungen übermitteln, welche Grafik-, Fenster- oder Managementoperationen auf dem Bildschirm durchgeführt werden sollen. Darstellungen auf dem Bildschirm werden als Objekte und nicht als Bitmaps übertragen.

Es gibt verschiedene Klassen von Requests. Nachfolgend sind nur einige Klassen von vielen angeführt:

- Erzeugen und Löschen von Fenstern.
- Ändern von Fensterattributen, wie Fensterrahmen oder Colormap.
- Andern von Fensterkonfigurationen, wie Breite, Höhe und Position
- Anfordern von Fensterinformationen, wie Fensterattribute oder Fensterkonfigurationen
- Ändern und Abfragen von Eigenschaften. Eigenschaften sind Daten, welche eine Anwendung an ein Fenster anhängen kann. Diese Daten können von anderen Anwendungen gelesen oder geschrieben werden, um so mit dem X-Server zu kommunizieren.
- Auflisten und Laden von Schriften.

Requests werden asynchron vom Client zum Server übertragen. X- Der Server bestätigt nicht den Erhalt von Daten, somit muss der X-Client nicht auf eine Antwort des X-Servers warten, sondern kann umgehend weitere Requests an den Server schicken. Deshalb erwartet X auch ein zuverlässiges Transportprotokoll, welches sich um die garantierte Zustellung der Datenpakete kümmert.

3.1.3.2.1.1 high-level Kodierung

Request sind objektbasiert. Das heißt, das anstatt von Bitmaps oder eine Liste von zu zeichnenden Pixel ganze Anweisungen an den Server übertragen werden. Solche Anweisungen können einen Operationscode, Fenster-ID, Koordinaten, Text, usw. enthalten.

3.1.3.2.2 Events

So wie Requests zur Übertragung von Informationen vom X-Clients zum X-Server verwendet werden, so werden so genannte Events zur Übermittelung von Informationen vom X-Server zum X-Client genutzt. Events sind das Gegenstück zu den Requests und sind Mitteilung, dass sich Änderungen ereignet haben. Über diesen Mechanismus werden Benutzereingaben, wie Tastaturanschläge oder Mausbewegungen an den X-Client weitergeleitet. Aber es werden auch andere Veränderungen, wie Änderungen der Fenstersichtbarkeit, an den X-Client gemeldet.

Die Meldung an den X-Client über Veränderungen erfolgt ebenfalls asynchron wie die Requests. Der X-Client fragt den X-Server nicht regelmäßig nach Veränderung, sondern der X-Server schickt sofort Änderungsupdates sobald diese eintreten und der ordentliche Erhalt dieser wird vom X-Client aber nicht bestätigt.

Events beinhalten ein gewisses Set an Basisinformationen, zum Beispiel wann und wo ein Ereignis aufgetreten ist. Weiters werden aber noch zusätzliche Daten mit übergeben, welche mehr Details über die Veränderungen im System beinhalten. Wie auch bei den Requests gibt es auch bei den Events verschiedene Typen:

Mausevents

Einige Eventtypen, welche von der Maus generiert werden.

- ButtonPress und ButtonRelease bei Mausklicks. Hier wird unter anderem auch festgehalten, welche Maustaste gedrückt wurde. Zu dem wird auch die Zeit aufgezeichnet über die man Doppelklicks von einfachen unterscheiden kann.
- MotionNotify wenn die Maus bewegt wurde.
- Enter und Leave wenn der Cursor in oder aus dem Fenster bewegt wird.

Tastaturevents

Einige Eventtypen, welche von der Tastatur generiert werden.

- KeyPress wird erzeugt wenn eine Taste gedrückt wird. Hier wird übermittelt um welche Taste es sich dabei handelt und ob zusätzliche Umschalttasten gedrückt sind.

- KeyRelease wird beim Loslassen einer Taste erzeugt.

Wie auch andere Events, werden Tastaturevents nur dann an Anwendungen gemeldet, wenn diese auch solche Meldungen von Events anfordern. Zum Beispiel wird von dem Programm *xclock* keine Texteingabe akzeptiert und deshalb werden an dieses Programm keine Tastaturevents gemeldet.

3.1.3.3 Bandbreite

Angaben zur benötigten Bandbreite zum Darstellen von X-Anwendungen über ein Netzwerk, belaufen sich im Durchschnitt auf zirka 20 KBit/s. Diese Angaben sind aber sehr abhängig von den eingesetzten Anwendungen.

Einige Anwendungen sind für eine Verwendung über eine Netzwerkverbindung weniger gut geeignet. Dies sind vor allem Anwendungen, welche hauptsächlich mit Bildern arbeiten. Diese Anwendungen arbeiten meist auf einer Pixel-Basis, welche einen großen Datenaustausch zwischen Server und Client hervorruft.

3.1.3.4 Verschlüsselung

In X selber ist kein Mechanismus zur Verschlüsselung der über ein Netzwerk transportierten Daten implementiert.

Eine einfache aber effektive und vielfach verwendete Variante ist, das X-Protokoll über eine Anwendungen namens *Secure Shell* (*ssh)* zu tunneln und so eine Verschlüsselung der Daten zu erreichen.

3.1.3.5 X Display Manager (xdm)

Der X-Server verwaltet nach seinem Start, wie bereits in Punkt 3.1.3.1 erwähnt, die Benutzerschnittstellen Bildschirm, Tastatur und Maus. Dies führt besonders bei X-Terminals zu einem Problem, wenn der Server ohne X-Clients gestartet wird, weil dann noch keine Anwendung mit dem Display verbunden wurde. Dies hat zur Folge, dass sämtliche Eingaben des Benutzers sofort vom X-Server verworfen werden, da sich noch kein X-Client registriert hat, welcher Interesse an den vom X-Server erzeugten Events hätte.

Um dieses Problem zu umgehen und die Verwaltung von X zu erleichtern, wurde ein spezielles Programm, der X Display Manager (*xdm*), entwickelt. Der *xdm*, wie der Name bereits sagt, managt das Display, indem es den X-Server und einen X-Client startet, welcher dem User ein Login Fenster bereitstellt. Nach dem Abmelden vom System, sorgt *xdm* dafür, dass der X-Server wieder zurückgesetzt wird und wieder ein Login Fenster präsentiert wird, damit sich wieder ein Benutzer am System anmelden kann.

Standardmäßig verwendet X den *xdm*, aber es gibt auch Alternativen. Zum Beispiel können diese Managementaufgaben ebenfalls von dem KDE Display Manager (*kdm*), oder dem Gnome Display Manager (*gdm*) übernommen werden.

3.1.3.6 X Display Management Control Protocol (XDMCP)

Das X Display Management Control Protocol (XDMCP) ist ein eleganter Weg um X-Terminals über ein Netzwerk mit einem anderen Rechnersystem zu verbinden.

Dabei überwacht das *xdm* Programm ständig das Netzwerk, ob ein X-Terminal per XDMCP einen *xdm* Dienst anfordert. Empfängt *xdm* eine Bitte von einem X-Terminal per XDMCP für einen Login auf dem System, kontrolliert *xdm* zuerst seine Konfigurationsfiles, ob der Client berechtigt ist auf diesem System einen Login durchzuführen. Wenn ja, bietet *xdm* dem X-Terminal seinen Dienst an und falls nicht, wird er abgewiesen.

XDMCP und xdm sind nicht Bestandteil des X Basis Systems, erleichtern die Benutzung und Administrierung eines X Systems aber bedeutend.

Weiterführende Informationen zu dieser Thematik ist unter [27] zu finden.

3.2 Performancevergleich

Für den Performancevergleich verwendete ich drei populäre Thin-Client Plattformen: Microsoft Windows 2003 Terminal Services, Citrix MetaFrame Presentation Server 4.0 und X.Org/X11R6.8.2.auf Linux, sowie deren zugehörigen Remote Display Protokoll RDP, ICA und X.

Tabelle 8: Charakteristiken von Server Based Computing Plattformen

Plattform	Display Encoding	Screen Updates	Kompression	Client Caching	Max. Client Display	Transport Protokoll
Microsoft Terminal Services (RDP)	Low-level graphics	Server-push, lazy	RLE	Glyphs, small bitmaps in memory (1,5MB)	24-Bit	TCP/IP
Citrix MetaFrame (ICA)	Low-level graphics	Server-push, lazy	RLE	Glyphs, small bitmaps in memory (3MB)	32-Bit	TCP/IP
X	High-level graphics	Server-push, eager	None	usually none	24-Bit	TCP/IP

Wie in Tabelle 8 zusammengefasst, kann man sehen das sich die Plattformen von Microsoft und Citrix sehr ähnlich sind, sich aber gegenüber zu dem X Window System in allen Belangen sehr unterscheiden. Deshalb erwarte ich in meiner Untersuchung auch große Ähnlichkeiten zwischen dem RDP-und ICA-Protokoll, im Gegensatz zu dem X-Protokoll.

3.2.1 Testumgebung

Um das Testnetzwerk nicht durch andere Einflüsse zu stören und eine gemeinsame Basis für alle Testläufe zu schaffen, wird ein isoliertes Netzwerk aufgebaut. Die Testumgebung besteht aus drei Geräten. Dabei handelt es sich um ein VMware Gastsystem, auf welchem die einzelnen Terminalserver installiert sind, einen Thin-Client, und einen Netzwerkswitch. Um während der Entwicklung möglichst flexibel und mobil zu bleiben, wurden die einzelnen Terminal Server, mit gleicher Hardwarekonfiguration, in einer VMWare[12] aufgesetzt.

Tabelle 9: Maschinenkonfiguration der Testumgebung

Verwendung	Hardware	Betriebssystem	Software
Thin-Client	1800+ AMD Athlon 512 MB RAM 40 GB Festplatte 1 10/100BaseT NIC	Microsoft Windows XP Professional SP 2 Thinstation 2.0	Citrix ICA-Client 9.2 RDP-Client 5.2
Windows Terminal Server	1 VMware Prozessor 8 GB Festplatter 256 MB RAM 1 10/100BaseT NIC	Microsoft Windows 2003 Terminal Server SP 1	OpenOffice 2.0 Adobe Reader 7.0
Citrix Terminal Server	1 VMware Prozessor 8 GB Festplatte 4 GB Festplatte 256 MB RAM 1 10/100BaseT NIC	Microsoft Windows 2003 Terminal Server SP 1	Citrix MetaFrame Presentation-Server 4.0 OpenOffice 2.0 Adobe Reader 7.0

[12] siehe http://www.vmware.com

Linux Terminal Server	1 VMware Prozessor 8 GB Festplatte 256 MB RAM 1 10/100BaseT NIC	Linux 2.6.15 Kernel Gentoo 2005.1	OpenOffice 2.0 KPDF 7.0
VMware Gast	3200+ AMD Athlon 64 512 MB RAM 2 10/100BaseT NIC ASUS 9600 Serie	Microsoft Windows XP Professional SP 2	VMware Workstation 5.5.1 WinDump 3.9.3

Als Thin-Client wurde ein handelsübliches Notebook verwendet. Das Notebook wurde für die Tests mit zwei Betriebssystemen betrieben. Für die Testläufe mit dem RDP und dem ICA Protokoll wurde das lokal installierte Betriebssystem Windows XP mit SP2 eingesetzt. Für die Testläufe mit dem X Protokoll wurde Thinstation, eine Linux Distribution für Thin-Clients, benützt, welches von dem Linux Terminalserver gebootet wurde. Anschließend wurde jeweils mit den entsprechenden Clientprogrammen eine Terminal Server Verbindung initialisiert und die Tests durchgeführt.

Als Hardwareplattform für den VMware Host der Terminal Server wurde ebenfalls ein handelsüblicher x86 Arbeitsplatzrechner verwendet. Die Betriebssysteme der Terminal-Server wurden aber in einer VMWare aufgesetzt. Die Maschinenkonfigurationen der einzelnen VMware Maschinen sind in Tabelle 9 zusammengefasst. Neben der VMware wurde auch das Kommandozeilen-Netzwerkanalyse Programm WinDump auf dem Gastsystem installiert, um die Netzwerkpaket, welche zwischen dem Terminalserver und dem Client ausgetauscht wurden, aufzuzeichnen und anschließend auszuwerten.

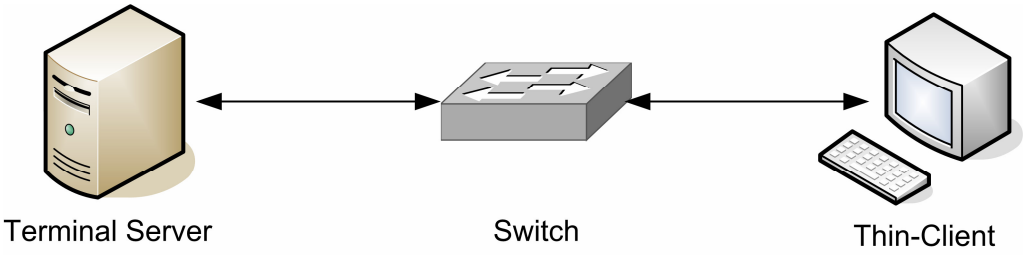

Abbildung 16: Testumgebung

Die einzelnen Komponenten der Testumgebung (Terminal Server und Thin-Client) werden über ein 100BASE-T Ethernet Netzwerk mittels einen Cisco LAN-Switch, so wie in Abbildung 16 ersichtlich, gemeinsam verbunden. Somit steht zur Übertragung zwischen den Komponenten während der Testläufe eine Netzwerkbandbreite von 100 MBit/s zur Verfügung.

Die Terminal Server wurden in einer VMware aufgesetzt. Dazu wurde das von VMware erhältliche Softwareprodukt VMware Workstation 5.1.1 verwendet. Um eine bessere Skalierbarkeit von Servern zu erreichen, werden Produktionsserver im Echtbetrieb heutzutage bereits häufig in einer VMware Umgebung betrieben.

3.2.2 Performancetests

Um die Performance der einzelnen Protokolle (ICA, RDP und X) zu testen, verwende ich sechs selbst erstellte Testvarianten. Dabei werden auf dem Thin-Client häufige im Büroarbeitsalltag vorkommende Tätigkeiten wie Textbearbeitung oder Präsentationen durchgeführt und die auftretenden Netzwerkdaten zwischen Client und Server aufgezeichnet. Für jedes der drei Protokolle wird jeder der sechs unterschiedlichen Tests absolviert.

Die Bildschirmauflösung des Thin-Clients beträgt bei allen Tests 1024 x 768 Pixel bei einer Farbtiefe von 24 Bit. Diese Einstellung wird auch von allen Protokollen unterstützt. Die Arbeitsfläche nach dem Login erstreckt sich ohne Fensterrahmen über den vollen Bildschirm.

3.2.2.1 Verbindungsleerlauf

Bei diesem Testlauf wird zuvor eine Verbindung zwischen dem Thin-Client und dem Terminal Server hergestellt und ein Login durchgeführt, sodass der Benutzer den Arbeitsplatz des Computers vor sich sieht. Anschließend wird für drei Minuten der auftretende Datentransfer zwischen dem Client und dem Server aufgezeichnet. In dieser Zeit findet keine Interaktion zwischen Server und Client statt. Mittels dieses Verfahrens möchte ich den grundlegenden Datenverkehr, wie etwa die Verbindungsaufrechterhaltung, erfassen.

3.2.2.2 Text schreiben

In dieser Teststellung soll ein vorgegebener Text jeweils mit möglichst gleichmäßiger Schreibgeschwindigkeit geschrieben werden. Dabei wird der Datenverkehr aufgezeichnet, welcher beim Übertragen der Tastenanschläge und der Bildschirmupdates anfällt..

Um auf der Windows-, sowie der Linuxplattform dasselbe Textverarbeitungsprogramm verwenden zu können, habe ich mich für das Programm *OpenOffice.org 2.0 Writer* von

OpenOffice.org[13] entschieden, welches eben für beide Betriebssystemplattformen verfügbar und kostenlos erhältlich ist.

Der Text besteht aus 62 Wörtern mit gesamt 444 Zeichen (inklusive Leerzeichen), welche zusammen zirka sechs Dokumentzeilen ergeben. Als Schriftart wurde Arial mit einem Schriftgrad von 12 Punkten gewählt.

3.2.2.3 Scrollen in langen Texten

Bei diesem Test wird ein größeres Schriftdokument vom Anfang bis zum Ende per Mausscrollrad durchgescrollt. In diesem Test wird auf den auftretenden Datenverkehr zwischen dem Endgerät und dem Server geachtet, welcher aus der Übertragung von Änderungen der Bildschirmausgabe und von den erzeugten Daten der Maus resultiert.

Bei dem Dokument handelt es sich um eine PDF-Datei mit 30 Seiten Text und einer Dateigröße von 170 KByte. Das Dokument wird mit dem kostenlosen Programm *Adobe Reader 7.0* von Adobe[14] unter Windows und KPDF unter Linux zur Anzeige gebracht. Der Zoomfaktor beim Betrachten des Dokuments wurde immer so gewählt das auf beiden Systemen der gleiche Bildschirminhalt zu sehen war. Das System wurde so konfiguriert, dass eine Einbuchtung auf dem Mausrad einen Bildlauf von drei Zeilen bewirkt.

3.2.2.4 Fensterbewegungen

In einem weiteren Testdurchlauf wurde zuvor ein Fenster, welches zirka auf die Größe eines Viertels des Ganzen Arbeitsplatzes verändert wurde, in dem rechten oberen Bereich des Desktops platziert. Dann wurde das Fenster per Maus über den Bildschirm bewegt und der dabei anfallende Datentransfer wieder von dem Netzwerkanalyseprogramm aufgezeichnet. Die Fensterbewegung erfolgt einmal rund um den Arbeitsplatz entlang der Aussenkante, so dass der Fensterrahmen immer zur Gänze sichtbar bleibt. In den Ecken wurde das Fenster immer kurz abgesetzt, damit sich der Fensterinhalt neu aufbauen musste.

---

[13] siehe http://www.openoffice.org

[14] siehe http://www.adobe.com

3.2.2.5 Präsentation mit abwechselnden Bildern

Bei dieser Testvariante wird eine Präsentation mit *OpenOffice.org 2.0 Impress* automatisiert abgespielt. Die Präsentation besteht aus 20 Präsentationsfolien, welche unterschiedlichen farbige Fotoaufnahmen aus der Natur enthalten. Jede Folie wird immer für drei Sekunden angezeigt und ohne Überblendungseffekte anschließend automatisch weitergeschaltet werden. Jedes der Bilder hat ein Format von 1024 x 768 Bildpunkten bei einer Auflösung von 72 Bildpunkten pro Inch. Die Präsentation hat eine Dateigröße von 12,561 MByte. Die übertragenen Daten zwischen Server und Client werden ebenfalls mit dem Programm *Windump* aufgezeichnet und mit *Ethereal* anschließend analysiert.

3.2.2.6 Videowiedergabe

Bei diesem Verfahren wird ein DivX Video, welches auf dem jeweiligen Terminal Server gespeichert ist, mit Hilfe des *VLC Media Player 0.8.4a* auf den Windows Plattformen und mit dem *MPlayer* auf der Linux Plattform abgespielt und die dabei Übertragenen Daten aufgezeichnet. Bei dem Video handelt es sich um einen Kinotrailer in einem DivX4 kodierten Format. Die Spieldauer des Trailers beträgt eine Minute und 34 Sekunden bei einer Bildfrequenz von 23 Bildern pro Sekunde. Die Datei selbst hat eine Größe von 11,179 MByte. Die Bildauflösung des Films beträgt 640 x 272 Bildpunkten. Neben der Aufzeichnung der Übertragenen Daten während des Abspielens, wird auch auf die subjektive visuelle Qualität während der Wiedergabe auf dem Client geachtet.

3.2.3 Testergebnisse

Die Abbildung 17 bis Abbildung 28 zeigen die Messergebnisse des angefallenen bidirektionalen Netzwerkverkehres zwischen dem Thin-Client und dem Server während der einzelnen Testläufe.

3.2.3.1 Verbindungsleerlauf

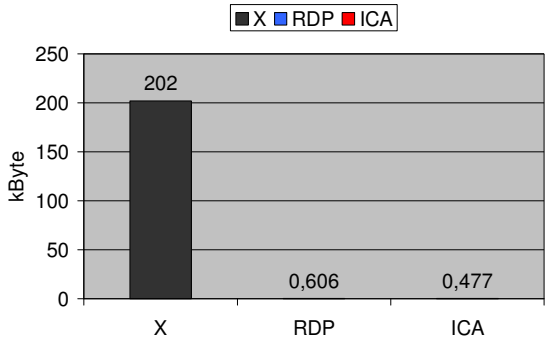

Abbildung 17: Übertragene Gesamtdaten im Verbindungsleerlauf.

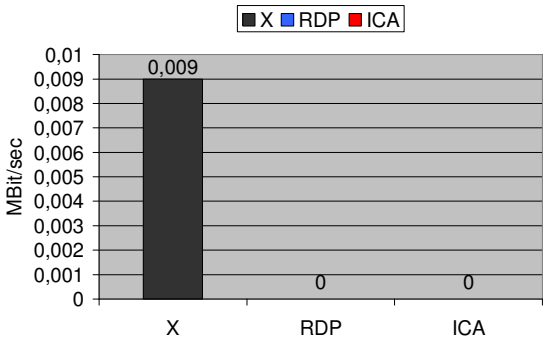

Abbildung 18: Durchschnittliche Datenrate im Verbindungsleerlauf

Abbildung 17 zeigt die angefallenen Verkehrsdaten zwischen dem Thin-Client und dem Server im Verbindungsleerlauf. Dabei wurden bei der X Verbindung 201.976 kByte an Daten aufgeteilt auf 2290 Datenpaketen übertragen. Im Vergleich dazu wurden bei der RDP Verbindung 606 Bytes in insgesamt sechs Datenpaketen und bei der ICA Verbindung 477 Bytes in ebenfalls sechs Datenpaketen übertragen. Schön zu sehen war, dass es sich Falle der RDP- und ICA Verbindung um Bildschirmupdates nach erfolgter Änderung der Uhrzeit handelte, weil die Datenpaket in 60 Sekunden Intervallen auftraten.

Abbildung 18 zeigt die errechnete durchschnittliche Datenrate zwischen dem Thin-Client und dem Server in einem Zeitfenster von rund 180 Sekunden. Die X Verbindung kam dabei auf eine Datenrate von rund 0,009 MBit/sec und die RDP- und ICA Verbindung auf praktisch 0 MBit/sec. Dies ergibt, dass der anfallende Datenverkehr in einem 100 MBit LAN praktisch vernachlässigbar ist.

3.2.3.2 Text schreiben

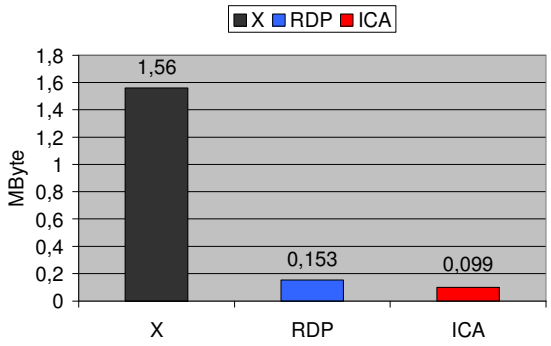

Abbildung 19: Übertragene Gesamtdaten beim Schreiben eines Textes

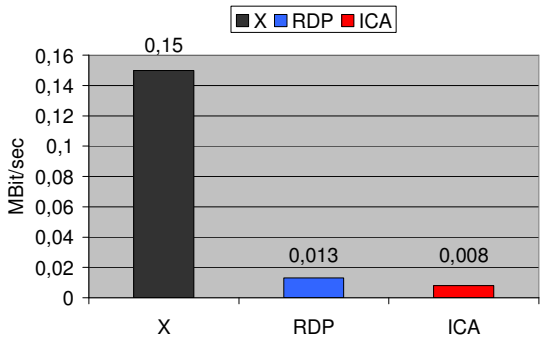

Abbildung 20: Durchschnittliche Datenrate beim Schreiben eines Textes

Abbildung 19 zeigt die angefallenen Verkehrsdaten zwischen dem Thin-Client und dem Server während dem Schreiben eines Textes. Der Vorgang dauerte zirka 70 Sekunden und in dieser Zeit wurden von dem X Protokoll rund 1,56 MByte an Daten zwischen den Rechnern ausgetauscht. Das angefallene Datenvolumen lag bei den anderen beiden Protokollen wesentlich darunter. RDP erzeugte dabei rund 0,153 MByte und ICA rund 0,099 MByte. Das X Protokoll erzeugte dabei 7255 Datenpaket um die notwendigen Daten zuzustellen. Das RDP Protokoll fand dabei mit 1968 Paketen und das ICA Protokoll mit 1215 Paketen das auslangen, um die erforderlichen Daten zu übertragen.

Abbildung 20 verdeutlicht die auftretende durchschnittliche Datenrate der jeweiligen Verbindung während des Testvorganges. Diese lag bei der X Verbindung bei rund 0,15 MBit/sec. Bei der RDP Verbindung waren es nur mehr 0,013 MBit/sec. Ungefähr die gleiche Datenrate wie das RDP Protokoll erzielte ICA mit rund 0,008 MBit/sec.

Das subjektive Empfinden bei diesem Test war bei allen drei Protokollen sehr zufrieden stellen. Es war während dem Schreiben des Textes kaum ein merklicher Unterschied zu einem vergleichbaren Arbeitsvorgang auf einem PC System zu erkennen. Der Schreibvorgang am Bildschirm wirkte sehr flüssig.

3.2.3.3 Scrollen in langen Texten

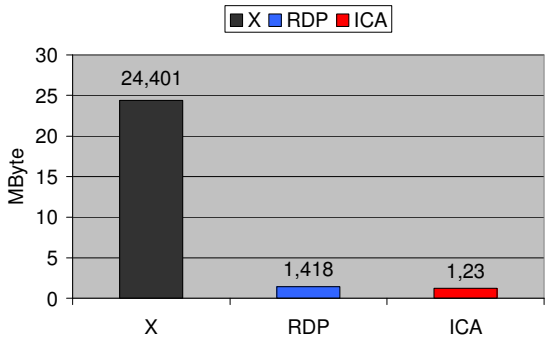

Abbildung 21: Übertragene Gesamtdaten beim Scrollen in langen Texten

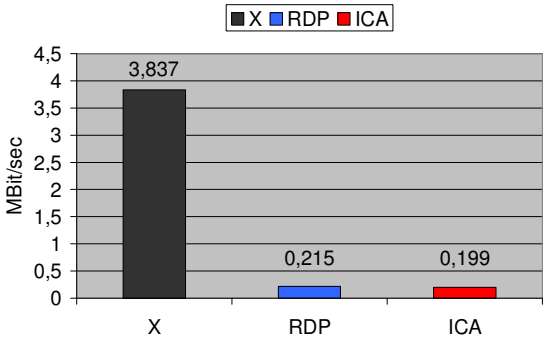

Abbildung 22: Durchschnittliche Datenrate beim Scrollen in langen Texten

In Abbildung 21 ist zu sehen, welche Datenmenge während dem Scrollen in langen Textstücken von den jeweiligen Protokollen erzeugt wurde. Bei dem X Protokoll waren es rund 24,4 MByte bis das gesamte Dokument durchgescrollt war. Das RDP und das ICA Protokoll lagen deutlich darunter. Das RDP Protokoll benötigte für diesen Prozess rund 1,4 MByte und das ICA Protokoll rund 1,23 MByte. X erzeugte hierbei 34776 Datenpakete, RDP 3009 und ICA 2569 Paket.

Abbildung 22 zeigt, welche Datenraten bei den einzelnen Protokollen während des rund 50 Sekunden dauernden Tests auftraten. Die X Verbindung benötigte für das durchscrollen des Textes rund 3,8 MBit/sec. Die beiden anderen Protokolle, RDP und ICA hielten sich wieder ziemlich die Waage. RDP benötigte eine Datenrate von rund 0,215 MBit/sec zur Übertragung und ICA rund 0,19 MBit/sec.

Im Gegensatz zu dem Schreiben einer Textpassage, wurde in diesem Test einklarer Unterschied im subjektiven Empfinden zwischen den Protokollen sichtbar. Während der Scrollvorgang bei den X Protokoll relativ gleichmäßig verlief, kam es bei den beiden anderen Protokollen zu relative ungleichmäßigen Scrollabläufen, wo der Seitenvorschub einmal schneller und einmal langsamer erfolgte.

3.2.3.4 Fensterbewegungen

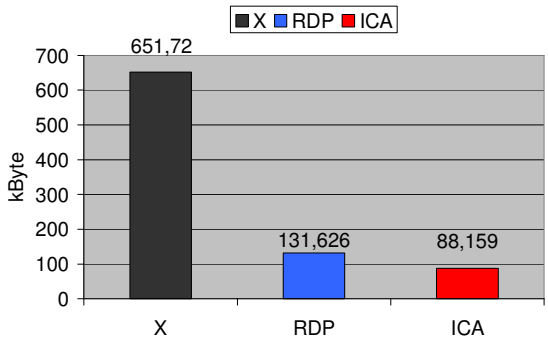

Abbildung 23: Übertragene Gesamtdaten beim verschieben eines Fensters

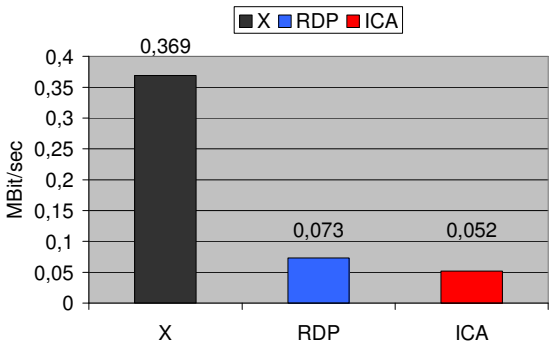

Abbildung 24: Durchschnittliche Datenrate beim verschieben eines Fensters

Welche Gesamtdatenmenge bei diesem Test übertragen wurde, lässt sich aus der Abbildung 23 entnehmen. Hierbei wurde mit dem X Protokoll mit 2832 Datenpaketen ein Datenvolumen von rund 0,65 MByte zwischen dem Thin-Client und dem Server transportiert. Mit dem RDP Protokoll wurde mit 325 Datenpaketen ein Datenvolumen von 0,13 MByte transferiert und mit dem ICA Protokoll mit 844 Paketen 0,08 MByte an Daten.

An Hand von Abbildung 24 lässt sich zusätzlich ablesen, welche Datenrate dabei von welchem Protokoll erzeugt wurde. Der Zeitrahmen bei diesem Test betrug in etwa 14 Sekunden. Das X Protokoll erreichte dabei ein Datenrate von zirka 0,37 MBit/sec. Durch das mehrmalige Verschieben eines Fensters wurde mit dem RDP Protokoll eine Datenrate von ungefähr 0,07 MBit/sec erzielt. Ebenfalls auf gleichem Niveau bewegt sich das ICA Protokoll mit einer Datenrate von rund 0,052 MBit/sec.

An Hand dem subjektiven Empfinden konnten keine gravierenden Unterschiede zwischen den einzelnen Protokollen festgestellt werden.

3.2.3.5 Präsentation mit abwechselnden Bildern

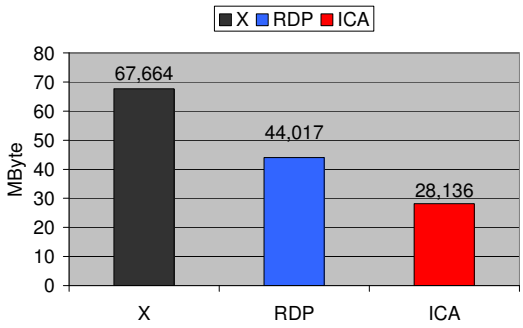

Abbildung 25: Übertragene Gesamtdaten bei der Bilderpräsentation

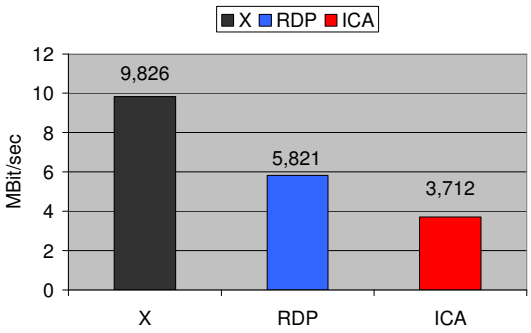

Abbildung 26: Durchschnittliche Datenrate bei der Bilderpräsentation

Wie in Abbildung 25 zu erkennen, wurden bei diesem Test rund 67,6 MByte an Daten mit dem X Protokoll übertragen. Diese wurden in 64870 Datenpaket gepackt und versendet. Das RDP Protokoll benötigte für die Erledigung dieser Aufgabe rund 44 MByte bei einer Anzahl von 45212 übertragenen Datenpaketen, während das ICA Protokoll rund 28,1 MByte in 28149 Paketen übertragen hat.

X kam bei diesem Testlauf auf eine Bandbreite von ungefähr 9,8 MBit/sec bei einem Testlauf von zirka 55 Sekunden, wie in Abbildung 26 abzulesen. RDP kam dabei auf zirka 5,8 MBit/sec, sowie ICA auf zirka 3,7 MBit/sec.

Subjektiv konnte kein größerer Unterschied in der Performance der Präsentation bei den einzelnen Protokollen festgestellt werden.

3.2.3.6 Videowiedergabe

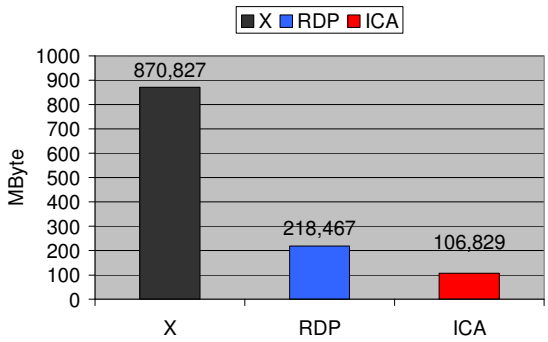

Abbildung 27: Übertragene Gesamtdaten bei der Videowiedergabe

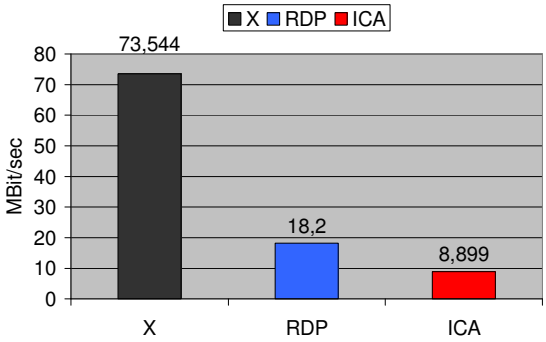

Abbildung 28: Durchschnittliche Datenrate bei der Videowiedergabe

Wie der Grafik aus Abbildung 27 zu entnehmen, wurde während der rund 95 Sekunden dauernden Videowiedergabe 870,827 MByte an Daten von dem X-Protokoll übertragen. Dabei wurden die zu transportierenden Daten in 815.709 Paketen zwischen Server und Client ausgetauscht. Effizienter arbeitete das RDP-Protokoll, welches für die Übertragung der Bildschirmdaten nur mehr 253,067 MByte in 253.067 Paketen benötigte. Die geringste Datenmenge erzeugte das ICA-Protokoll mit 106,829 MByte in 109.708 Datenpaketen.

Abbildung 28 verdeutlicht die bei der Übertragung auftretende durchschnittliche Datenrate der einzelnen Protokolle während der Übermittlung der Displaydaten. Dabei ergab sich durch die große Menge an Daten auch eine hohe Datenrate von 73,544 MBit/sec durch das X-Protokoll. Mit 18,2 MBit/sec verursachte das RDP-Protokoll eine wesentlich geringe Datenrate. Die geringste Datenrate verursachte das ICA-Protokoll mit rund 8,899 MBit/sec.

Von der visuellen Qualität des wiedergegebenen Videos schnitt am besten die Wiedergabe über das X-Protokoll ab, gefolgt von dem RDP-Protokoll. Das schlechteste Ergebnis lieferte die Wiedergabe über das ICA-Protokoll.

3.3 Zusammenfassung

Der experimentelle Vergleich der drei Protokolle X, RDP und ICA hat gezeigt, dass der durchschnittliche Datentransfer aller Protokolle nicht sehr hoch ist, wobei aber die transportierten Daten des X Protokoll meistens weit über denen der RDP und ICA Protokollen liegt. Ausgenommen ist der Test mit der Wiedergabe eines Videos, bei dem doch recht beträchtliche Datenmengen übertragen wurden, welche in einem Produktivnetz hohe Lasten verursachen würden.

An Hand dieser Daten kann man auch die Ähnlichkeiten der eingesetzten Techniken zwischen dem RDP und ICA Protokoll erahnen.

Interessant ist auch, dass bei den RDP und ICA Protokollen im Leerlauf im Gegensatz zu dem X Protokoll kaum Daten transportiert wurden.

In Punkte subjektives Empfinden während der Interaktion mit dem Client liegt das X Protokoll deutlich vor den beiden anderen Protokollen, was sich aber in der verursachten Datenrate wieder spiegelt.

4 Grobkonzept für die APA

Im Zuge dieser Diplomarbeit soll auch ein mögliches technisches Design einer Netzwerkboot-Umgebung für die Austria Presse Agentur entworfen werden. Dieses Kapitel befasst sich mit der Entwicklung eines solchen Designs.

4.1 Vorgaben seitens der APA

Die Einführung einer Netzwerkboot-Umgebung soll möglichst transparent für den Anwender erfolgen und größtmöglich und weitestgehend keine Änderungen des täglichen Arbeitsprozesses der Mitarbeiter der APA nach sich ziehen.

Weiters unterhält die APA auf mehreren Standorten in Österreich kleine Außenstellen, welche wenn möglich ebenfalls in die Netzwerkboot-Umgebung miteinbezogen werden sollen.

4.1.1 Ausfallsicherheit

Da es sich bei der APA um einen Nachrichtendienstleister handelt, welcher rund um die Uhr (24 x 7 x 365) im produktiven Einsatz steht, muss eine Hochverfügbarkeit der gesamten Netzwerkboot-Umgebung erreicht werden.

4.1.2 Endgeräte

Die bestehende Hardware der Endgeräte solle in der neuen Netzwerkboot-Umgebung weiterhin verwendet werden können. Weiters soll der Datenaustausch zwischen dem Rechnersystemen und externen Geräten, wie zum Beispiel Universal Serial Bus (USB) Massenspeichersystemen oder Diktiergeräten, weiterhin gewährleistet werden. Ein weiterer wichtiger Punkt ist die weiterhin mögliche Synchronisation von E-Mail- und Kalenderdaten mit PDAs über die Endgeräte.

Den Benützern der Arbeitsstationen soll es weiterhin möglich sein CDs oder DVDs, je nach vorhandenen Laufwerken, auf ihren Computern verwenden zu können.

Das Wiedergeben von Audiodateien auf den Endgeräten, muss ebenfalls weiterhin möglich sein.

4.1.3 Server

Die neuen Server sollen in die bestehende Serverlandschaft nahtlos eingefügt werden können. Das neue Serversystem soll in das bestehende Serverüberwachungssystem integriert werden können.

4.1.4 Software

4.1.4.1 Anwendersoftware

Da sich für die Anwender keine großen Veränderungen im Arbeitsprozess ergeben dürfen, sollen möglichst die bereits eingesetzten Softwareprodukte weiter verwendet werden.

Im Anhang findet sich eine Auflistung der momentan eingesetzten Software in der APA.

4.2 IST Konzept APA

Im Folgenden soll die IT Landschaft der APA etwas genauer betrachtet werden, um das Design der Netzboot-Umgebung besser darauf abstimmen zu können.

4.2.1 Netzwerkinfrastruktur

Abbildung 29 gibt einen Überblick über die momentane Infrastruktur der EDV-Landschaft in der APA.

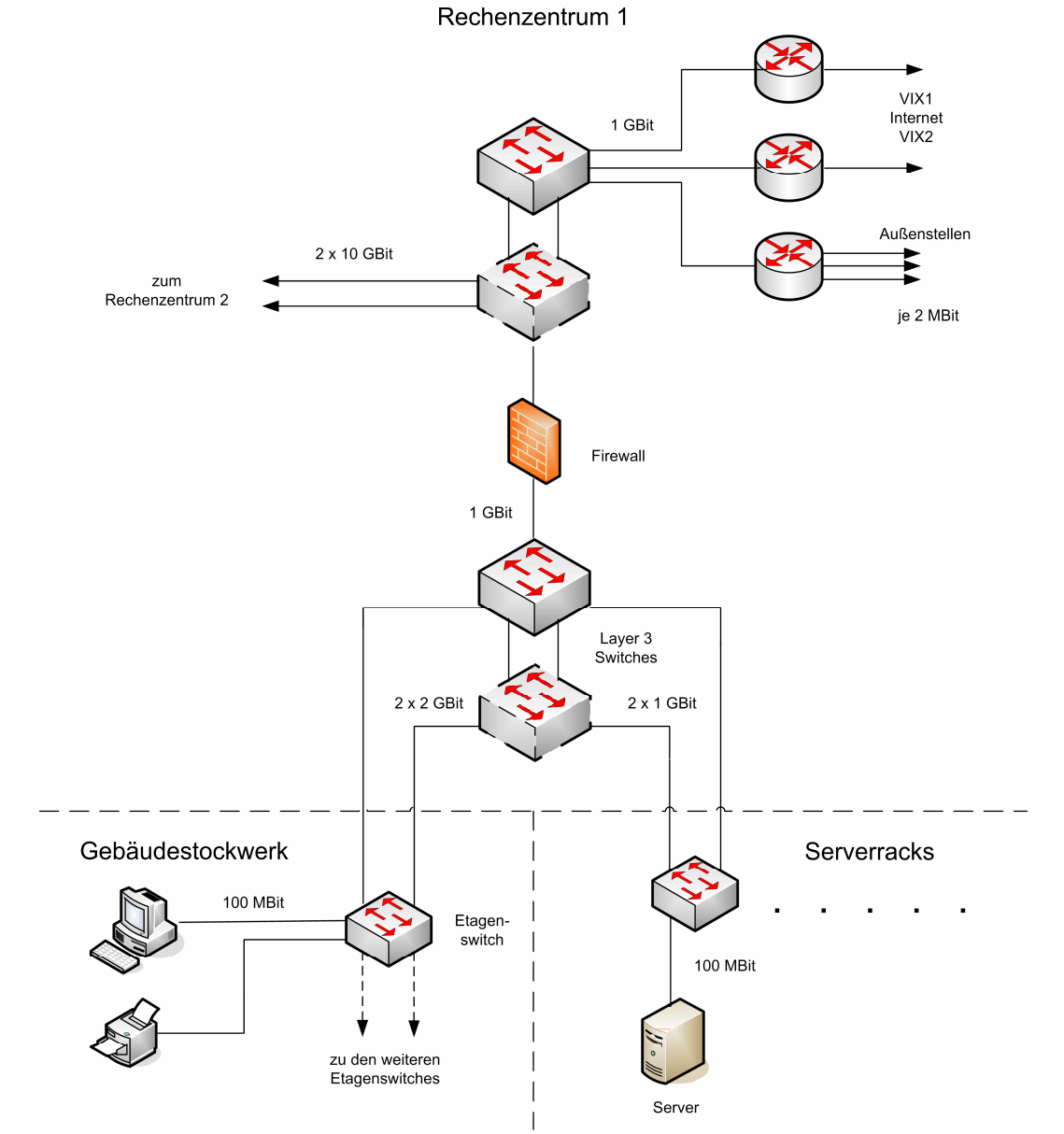

Abbildung 29: Netzwerkplan APA IST

4.2.1.1 LAN

Die Anbindung der Arbeitsplatzrechner erfolgt über ein voll geswitchtes 100 BASE-T Ethernet.

Die einzelnen Etagenswitches in den Stockwerksverteilern sind redundant über einen 2 GBit Glasfaserring an den Gebäude Backbone angebunden.

In jedem Serverrack befindet sich ein 100 MBit Switch, welcher wiederum mit einem redundanten 1 GBit Uplink an den Gebäude Backbone angebunden ist.

4.2.1.2 WAN

Die Außenstellen der APA in den Bundesländern sind über eine 2 MBit ATM Leitung angebunden.

4.2.1.3 Internet

Der Anschluss an das Internet erfolgt über eine Firewall mit einer 1GBit Anbindung.

4.2.2 ADS

Alle Arbeitsplatzrechner, sowie die Server der APA sind Mitglieder einer Active Directory Service Domäne und werden über den Verzeichnisdienst Active Directory Service (ADS) verwaltet.

Wie auch die Computer werden auch die Benutzer über das ADS verwaltet. Die Benutzerauthentifizierung erfolgt ebenfalls über das ADS.

4.2.3 DHCP

Die IP Konfiguration der Computer während der Startphase erfolgt über das DHCP Service.

4.2.4 Drucker

Die Anwender drucken ausschließlich über Netzwerkdrucker, welche beim Login automatisch dem Anwender zugewiesen werden.

4.2.5 DFS

Die Anwenderverzeichnisse befinden sich auf eigenen Dateiservern

4.3 Grobkonzept APA

Das hier für die APA vorgeschlagene technische Design für eine Netzwerkboot-Umgebung mit Thin-Clients, basiert auf dem Prinzip des Server Based Computing mit so genannten X-Terminals. Auf diesen X-Terminals läuft ein ICA-Client, welcher sich nach dem Bootprozess automatisch mit einem Citrix MetaFrame Presentation Server verbindet und einen grafischen Login auf diesem Server ermöglicht und anbietet.

Als X-Terminals können die bestehenden Arbeitsplatzrechner der APA ohne hardwaremäßige Adaption weiter verwendet werden. Da bereits alle Computer, welche in der APA zum Einsatz kommen, über bereits PXE-fähige Netzwerkkarten verfügen, wird ein Bootkonzept mittels PXE gewählt. Dieser gewählte Ansatz erfordert ebenfalls, keinen Eingriff in die bestehende Computerhardware der Arbeitsplatzrechner.

Prinzipielle Voraussetzungen für diesen Ansatz sind ein Bootserver und ein Terminal Server wie auch unter Punkt 2.1.3 bereits vorgestellt. Der Bootserver stellt den Linux Kern sowie die RAM-Disk mit der notwendigen Systemsoftware zur Verfügung, welcher der Client für ein lauffähiges System benötigt. Nach dem Bootvorgang kann sich der Anwender auf einem Terminal-Server vergleichbar wie auf einem lokalem System anmelden.

4.3.1 Netzwerkinfrastruktur

Als Netzwerk kann das bereits bestehende Netzwerk der APA ohne größere Adaptierungen weiterverwendet werden. Wie in Abbildung 30 abgebildet, sind die einzigen größeren Eingriffe in die Netzwerkstruktur der APA nur im Rechenzentrum durchzuführen. Dort wird ein eigener Server Backbone eingerichtet, über welchen die Kommunikation zwischen den Terminal- und Anwendungsservern und den dahinterliegenden File- und Datenbankserver läuft.

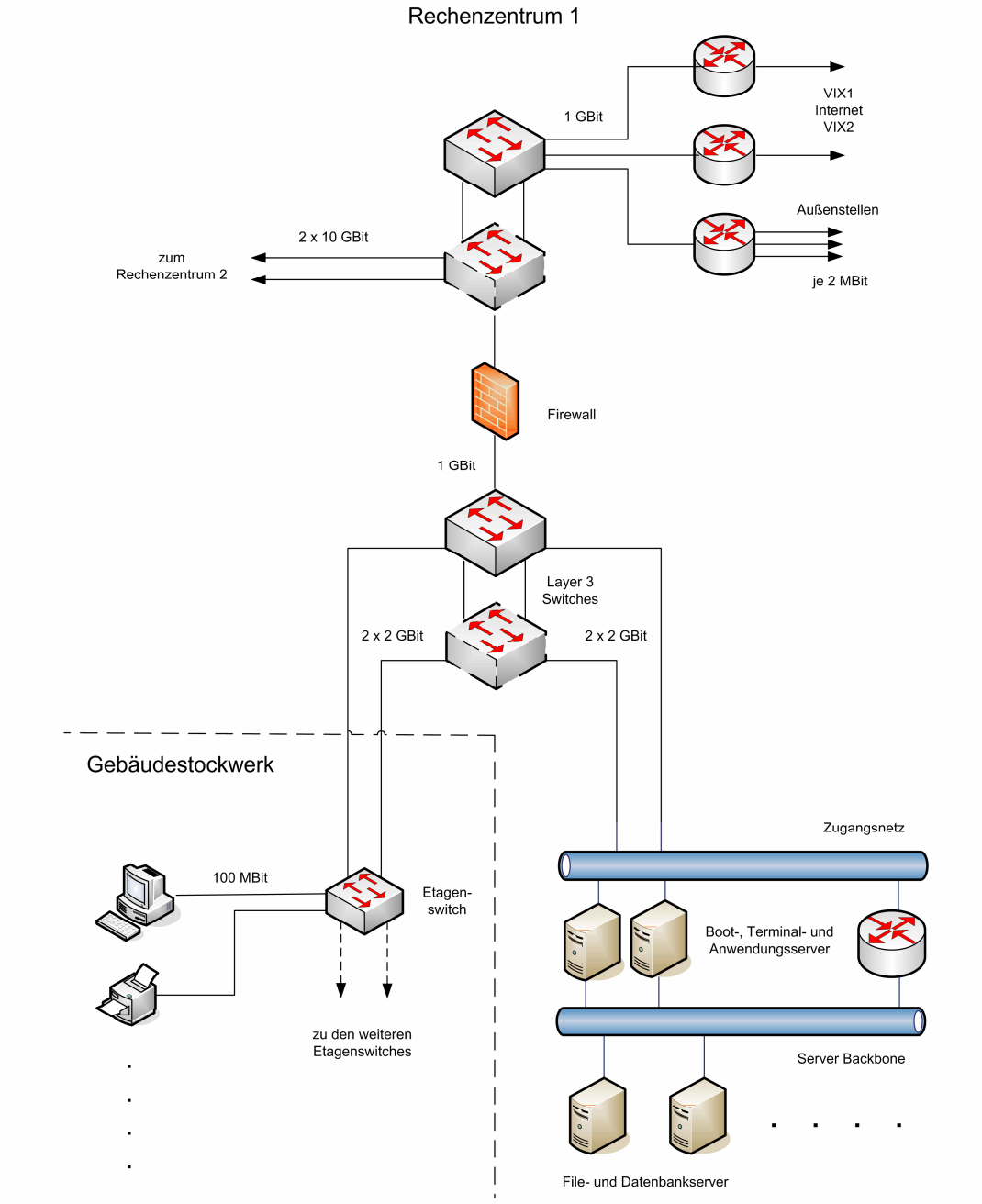

Abbildung 30: Netzwerkplan APA SOLL

4.3.1.1 LAN

Die Anbindung der Clients an das LAN der APA erfolgt über das bereits bestehende voll geswitchte 100 BASE-T Ethernet Netzwerk.

Die Anbindung der Server erfolgt über ein 1 Gbit Ethernet an den Gebäude Backbone der APA, an welchen ebenfalls die Switches der Stockwerksverteiler angebunden sind. Über eine zweite 1 GBit Ethernet Netzwerkkarte erfolgt eine Anbindung an den Server Backbone, über welchen die Verbindung zu den Dateiservern und Datenbanken erstellt wird.

4.3.1.2 WAN

Die Anbindung der Außenstellen der APA mit einer jeweils bestehenden 2 MBit ATM Anbindung ist ebenfalls ausreichend.

4.3.1.3 Internet

Die Internetanbindung erfolgt ebenfalls wie bei der bereits bestehenden Netzwerkinfrastruktur.

4.3.2 Endgeräte

Bei den Endgeräten (Thin-Client) kann die bestehende verwendete Hardware weiterhin eingesetzt werden. Die eingesetzte Software (Thinstation) unterstützt eine Vielzahl von handelsüblichen Hardwarekomponenten (Grafikkarten, Netzwerkkarten, usw.).

Als einziger Eingriff bei den Clients, muss nur die BIOS-Einstellung geändert werden. Hier muss die Netzwerkkarte als erstes Bootgerät gesetzt werden. Dies hat den Vorteil, dass bei Problemen bei Umstieg auf die Netzwerkbootumgebung jederzeit wieder auf das lokale System zurück gestiegen werden kann.

Betriebssystem

Als Betriebssystem für die Thin-Clients wird eine quelloffene Linux Distribution namens Thinstation verwendet. Thinstation unterstützt das für eine Verbindung zu einem Citrix Terminal Server notwendige Protokoll ICA.

Der Betriebssystemkern und die notwendigen Treiber sowie weitere Software (ICA-Client) werden von dem Bootserver per TFTP bereitgestellt. Dabei werden die Treiber und die Software in eine RAM-Disk integriert.

Netzanbindung

Die für einen Netzwerkbootvorgang und den anschließenden Betrieb notwendige Netzwerkanbindung an das LAN der APA erfolgt über die bereits bestehende 100 MBit Ethernet Verbindung der Clients.

4.3.3 DHCP-Server

Als DHCP Server werden die bestehenden Server der APA IT Infrastruktur verwendet. Neben den Standardinformationen, wie zum Beispiel Client IP-Adresse oder Standard-Gateway, müssen den Clients noch weitere Informationen zugestellt werden.

Um den Clients die Netzwerkadresse des TFTP Servers und den Namen des zu ladenden Bootladers mit zu teilen, müssen auf dem DHCP Server die Optionen 66 (Hostname des Startservers) und 67 (Name der Startdatei) aktiviert und konfiguriert werden.

- Hostname des Startservers ist jener des Bootservers (TFTP Server).
- Name der Startdatei ist *pxelinux.0* (NBP).

4.3.4 Bootserver

Der Bootserver dient dazu, um den plattenlosen Clients während des Netzwerkbootvorganges deren Betriebssystem und die dazugehörige Software bereitzustellen.

Betriebssystem

Als Betriebssystem für den Bootserver wird Red Hat Enterprise Linux ES verwendet. Grund dafür ist, dass die APA die Produktlinie von Red Hat einsetzt und auch Serviceverträge vorhanden sind. Weiters ist auch das Administrationsteam mit diesem System bereits bestens vertraut.

Thinstation

Als spezielle Software wird auf dem Bootserver das Programm *Thinstation* installiert, welches zum Erstellen der Bootimages für die Thin-Clients verwendet wird. Thinstation wird in erster Line dazu verwendet um die Bootimages für die Thin-Clients zu erstellen. Weiters wird auch die Hardwarekonfiguration der Clients über Thinstation abgewickelt.

Mit Hilfe des Konfigurationsfiles *build.conf* im Verzeichnis */opt/Thinstation2.1* wird das Bootimage an die Hardware der Clients angepasst und anschließend mit dem Programm *build*, ebenfalls in dem selben Verzeichnis das Image mit samt der RAM-Disk erstellt. Der Kernel und die RAM-Disk müssen anschließend nur mehr an die Stelle kopiert werden, von wo sie der Client während des Bootvorganges per TFTP download. Die zu übertragende Bootimagegröße für einen Thin-Client ist abhängig von den mit einkompilierten Programmen und Systemtreibern und beträgt zirka 7 – 8 MB für einen Client.

Die Konfiguration der Clients kann über mehrere hierarchisch aufgebaute Konfigurationsfiles erfolgen, welche im TFTP-Verzeichnis des Servers abgelegt werden und von den Clients während deren Bootprozess downgeloaded und ausgewertet werden.

Im ersten Schritt sucht der Client während der Bootprozedur nach dem Standardkonfigurationsfile (thinstation.conf.network). In dieser Datei werden alle Einstellungen festgelegt, welche standardmäßig für alle Clients gelten sollen (Verbindungstype, Serveradressen, …).

Im zweiten Schritt sucht der Client nach der *thinstation.conf.hosts* Datei, in welcher die Clients nach Namen oder Gruppen zusammengefasst werden.

Im dritten Schritt sucht der Client nach die für ihn passenden Konfigurationsdateien in folgender Reihenfolge:

- thinstation.conf-<hostname>
- thinstation.conf-<IP_Adresse>
- thinstation.conf-<MAC_Adresse>

Beim Auswerten der Konfigurationsdateien werden bereits bestehende Konfigurationen von zuvor gelesen Dateien überschrieben und gegen die neuen Werte ersetzt. Die Clients können zur besseren und übersichtlicheren Verwaltung in verschiedene logische Gruppen zusammengefasst werden.

Bei dem Bootprozess eines Thin-Clients mittels PXE wird ein eigener Bootloader benötigt der den eigentlichen Betriebssystemkern über das Netzwerk downloaded (siehe 2.6.4). Als Bootloader wird bei dieser Variante *pxelinux.0* von PXELINUX verwendet, welches auf den Betrieb von PXE in Zusammenhang mit einem Linuxkernel zugeschnitten ist. Der Bootloader (*pxelinux.0*) wird im TFTP-Verzeichnis des Bootservers abgelegt, von wo er von den Clients geladen werden kann.

Der Betriebssystemkern, sowie die RAM-Disk, müssen sich in demselben Verzeichnis wie der Bootloader befinden.

Für einzelne Endgeräte oder Gruppen kann man spezielle Konfigurationsdateien vorsehen (siehe 2.6.4.2). Um den Administrationsaufwand aber möglichst gering und die Anzahl der PXELINUX-Konfigurationsdateien überschaubar zu halten, wird nur ein Bootimage erzeugt, welches für alle Clients verwendet werden kann. Da alle Clients mit derselben Software ausgestattet werden, vergrößert sich das Bootimage nur um die Treiber die für jede Hardware fest mit einkompiliert werden müssen. Die Konfigurationsdateien müssen im Unterverzeichnis *pxelinux.cfg* liegen, welches wiederum im selben Verzeichnis wie das Ladeprogramms *pxelinux.0* zu finden sein muss.

Die Konfigurationsinformationen werden in der Konfigurationsdatei *default* gespeichert, von wo das Ladeprogramm den Namen des Kernel und die dazugehörigen Kerneloptionen bezieht.

TFTP

Ein wichtiges Service, welches vom Bootserver angeboten werden muss, ist TFTP, denn über diesen Dienst werden der Kernel, die RAM-Disk und zuvor das NBP an den Client übertragen. Deshalb muss auf dem Bootserver ein TFTP-Server installiert sein.

Da TFTP keinen Schutz durch Passwortauthentifizierung während des Betriebes bietet, sollte bereits beim Aufsetzen größte Vorsicht geboten werden und TFTP in einer chroot-Umgebung betrieben werden. Dadurch kann der Zugriff auf höher gelegenen Verzeichnisebenen verhindert werden.

Der Name und der Verzeichnispfad zu dem NBP werden dem Client per DHCP übermittelt (Option 67). Der Kernel und die RAM-Disk müssen sich im selben Verzeichnis wie der Bootlader *pxelinux.0* befinden.

Ausfallsicherheit und Lastverteilung

Da bei einem Ausfall des Bootservers, keine Clients mehr den Betrieb aufnehmen können, muss das Bootservice gegen einen Totalausfall abgesichert werden. Da das Rechnernetz der APA, durch ein Ausfallsrechenzentrum, vollkommen redundant ausgelegt ist, wird auch ein redundanter Bootserver im Ausfallsrechenzentrum installiert.

Zwischen den beiden Bootservern wird während des normalen Betriebes ein Lastausgleich eingerichtet.

Die Netzanbindung des Servers erfolgt über eine Gigabit-Ethernet Netzwerkkarte an den Gigabit-Backbone der APA, welcher ebenfalls die Switches der Stockwerksverteiler verbindet. Die beiden Netzwerkkarten werden gebündelt ausgeführt. Bei Hewlett-Packard[15] wird dies als *Teaming* bezeichnet, wodurch eine höhere Ausfallsicherheit und Lastverteilung erreicht werden kann.

---

[15] http://www.hp.com/

4.3.5 Terminal Server

Mit Hilfe von Terminal Servern wird den Benutzern der Thin-Clients eine vollständige Computerarbeitsplatzumgebung bereitgestellt. Dem Anwender werden so sämtliche Ressourcen der Terminal Server zur Verfügung gestellt. Die Bereitstellung der Computerarbeitsplatzumgebung erfolgt mit Citrix Presentation Server 4.0. Die Zuteilung, welche Benutzer sich auf dem Terminal Server anmelden dürfen, erfolgt in Verbindung mit den im Verzeichnisdienst ADS angelegten Benutzergruppen.

Alle Terminal Server werden als Mitgliedsserver in die bestehende ADS Domäne aufgenommen.

Betriebssystem

Als Betriebssystem kommt Microsofts Windows Server 2003 x64 auf den Terminal-Servern zum Einsatz.

Neben der Installation des Grundbetriebssystems, muss auf dem Server noch die Funktion des Terminal Servers von Microsoft installiert und aktiviert werden.

Citrix Presentation Server 4.0 x64

Die Bereitstellung der Arbeitsplatzumgebung, oder unter Citrix als „Veröffentlichung des Desktops“ genannt, erfolgt mit dem Softwareprodukt von Citrix, dem Presentation Server 4.0 x64 Advanced Edition.

Neben dem Presentation Server, wird auf den Terminal Server auch jene Software installiert, welche allen Benutzern zur Verfügung gestellt werden soll.

Anwendersoftware

Wichtig zu beachten dabei ist, dass jede Software, welche auf den Terminal Servern installiert werden soll, zuvor auf ihre Terminal Servertauglichkeit überprüft wird.

Auf dem Terminal Server wird ein Grundpaket an Software installiert, welche jedem Anwender der APA zur Verfügung stehen soll. Dieses Paket wurde bereits von der APA definiert.

Ausfallsicherheit und Lastausgleich

Citrix hat bereits eine Funktionalität für einen Lastausgleich unter Servern in einer Serverfarm implementiert. Dabei wird der Anwender bei der Anmeldung am System an den Server mit der geringsten Belastung zugewiesen.

Da bei der APA höchster Wert auf die Ausfallsicherheit des Systems gelegt wird, werden die Terminal Server vollkommen redundant im Ausfallrechenzentrum ausgeführt.

Die Netzanbindung der Server erfolgt jeweils über zwei 1 GBit Netzwerkkarten an den Gebäude-Backbone des APA-LAN und über zwei 1 GBit Netzwerkkarten an den Server-Backbone. Die Netzwerkkarten werden wieder jeweils in Teams zusammengefügt. Details sind aus Abbildung 30 zu entnehmen.

4.3.6 Anwendungsserver

Mit Hilfe den Anwendungsservern wird den Benutzern der Thin-Clients zu den bereits mit den Terminal Servern allgemein veröffentlichten Desktops, die individuelle Software, welche nur einzelnen Benutzergruppen zur Verfügung stehen soll, veröffentlicht. Dem Anwender werden auf diese Art und Weise jene Programme bereitgestellt, welche nicht auf den Terminal Servern zur Verfügung stehen. Die Bereitstellung dieser Anwendungen erfolgt mit dem Citrix Presentation Server 4.0 x64. Die Zuteilung, welche Benutzer auf welche Programme Zugriff haben, erfolgt in Verbindung mit den im Verzeichnisdienst ADS angelegten Benutzergruppen.

Alle Anwendungsserver werden als Mitgliedsserver in die bestehende ADS Domäne der APA aufgenommen.

Betriebssystem

Als Betriebssystem kommt Microsofts Windows Server 2003 x64 auf den Anwendungsservern zum Einsatz.

Neben der Installation des Grundbetriebssystems, muss, wie auf den Terminal Servern, noch die Funktion des Terminal Servers von Microsoft installiert und aktiviert werden.

Citrix Presentation Server 4.0 x64

Die Bereitstellung der zusätzlichen Anwendungssoftware, unter Citrix als „Anwendungen veröffentlichen“ bezeichnet, erfolgt mit dem Softwareprodukt von Citrix, dem Presentation Server 4.0 x64 Advanced Edition.

Neben dem Presentation Server, wird auf den Anwendungsserver auch jene Software installiert, welche denn einzelnen Benutzergruppen zur Verfügung gestellt werden soll.

Anwendersoftware

Wichtig zu beachten ist, dass jede Software, welche auf den Terminal Servern installiert werden soll, zuvor auf ihre Terminal Servertauglichkeit überprüft wird.

Auf den Anwendungsservern wird jene Software installiert, welche den Anwendern der APA zur Verfügung stehen soll, welche noch nicht per Terminal Server bereitgestellt wurde.

Die erforderliche Software für die einzelnen Benutzergruppen wurde bereits seitens der APA vordefiniert.

Ausfallsicherheit und Lastausgleich

Citrix hat bereits eine Funktionalität für einen Lastausgleich unter Servern in einer Serverfarm implementiert. Dabei wird der Anwender bei dem Zugriff auf für ihn veröffentlichte Software, immer an den Server mit der geringsten Ressourcenauslastung zugewiesen.

Da bei der APA höchster Wert auf die Ausfallsicherheit des Systems gelegt wird, werden die Anwendungsserver vollkommen redundant im Ausfallrechenzentrum ausgeführt.

Die Netzanbindung der Server erfolgt jeweils über zwei 1 GBit Netzwerkkarte an den Gebäude-Backbone des APA-LAN und über zwei 1 GBit Netzwerkkarte an den Server-Backbone. Die Netzwerkkarten werden wieder jeweils in Teams zusammengefügt. Details sind aus Abbildung 30 zu entnehmen.

4.3.7 Benutzer

An der Verwaltung der Benutzer sind keine Änderungen erforderlich, denn diese werden wie bisher über den Verzeichnisdienst ADS verwaltet.

Authentifizierung

Die Authentifizierung der Benutzer erfolgt ebenfalls wie bisher gegen das ADS des APA-LAN.

5 Zusammenfassung und zukünftige Arbeiten

Während der Ausarbeitung dieser Diplomarbeit hat sich gezeigt, dass es ein breites Spektrum an verschiedenen Varianten und Ansätze zur Entwicklung und zum Betrieb von plattenlosen Computern gibt. Angefangen von kommerziellen Produkten bis hin zu verschiedenen kostenlosen und bereits weit ausgereiften Projekten der Open Source Communities.

Der Vergleich der drei Remote Display Protokolle X, RDP und ICA hat gezeigt, dass absolut alle drei Protokolle geeignet sind, um Thin-Clients in heutigen LANs mit Terminal Servern effizient zu verbinden, wobei das von Citrix entwickelte Protokoll ICA am effizientesten mit den vorhanden Netzwerkressourcen umzugehen weiß und das X Protokoll durch fehlende Mechanismen, wie Kompression und Caching, weit höhere Netzlasten erzeugt.

Für zukünftige Arbeiten wäre es interessant, weitere Vergleiche mit anderen Protokollen, wie der Weiterentwicklung des X Protokolls für Übertragungen auf schmallbandigen Netzen, dem Low Band X (LBX) Protokoll, oder dem Virtual Network Computing (VNC) Protokoll, anzustellen. Auch eine Untersuchung der Protokolle mittels einer Langzeitanalyse in einem Echtbetrieb wäre sehr aufschlussreich. Weiters wäre es interessant sich auch mit der verursachten Serverauslastung (CPU, RAM und Disk I/O) unter Verwendung der einzelnen Produkte (X, Microsoft Terminal Services und Citrix Presentation Server) sich etwas genauer zu beschäftigen.

Literaturverzeichnis

[1] Kretschmar, B., Burbach, H., Dirk von Suchodoletz (2005): Linux-Terminal Server. Galileo Press GmbH, Bonn, Deutschland

[2] Wikipedia (2006): Diskless. http://de.wikipedia.org/wiki/Diskless. (Datum des Zugriffs: 15. Februar 2006)

[3] Intel Corporation (1999). Preboot Execution Environment (PXE) Specification: Version 2.1 http://www.pix.net/software/pxeboot/archive/pxespec.pdf. (Datum des Zugriffs: 15. März 2006)

[4] Bundesministerium des Inneren. (2006). Leitfaden für die Migration der Basissoftwarekomponenten auf Server- und Arbeitsplatzsystemen: http://www.kbst.bund.de/cln_011/nn_837406/SharedDocs/Anlagen-kbst/Migrationsleitfaden/migrationsleitfaden__version__2__1__pdf,templateId=raw,property=publicationFile.pdf/migrationsleitfaden_version_2_1_pdf.pdf

[5] Etherboot Project: http://www.etherboot.org/. (Datum des Zugriffs: 11. Mai 2006)

[6] Kuhlmann,G. Netboot: http://netboot.sourceforge.net/. (Datum des Zugriffs: 11. Mai 2006)

[7] Coldwell,C. M. (2002). Diskless Linux: http://frank.harvard.edu/~coldwell/diskless/. (Datum des Zugriffs: 24. Mai 2006)

[8] Kontron AG. (2003). Technical Manual WEBasDisk Rev 2.2: http://emea.kontron.com/downloads/manual/net1m122.pdf. (Datum des Zugriffs: 13. Mai 2006)

[9] IBM Corporation. iBoot – Remote Boot over iSCSI: http://www.haifa.il.ibm.com/projects/storage/iboot/index.html. (Datum des Zugriffs: 13. Mai 2006)

[10] Croft, B., Gilmore, J. (1985). RFC-951 Bootstrap Protocol:
http://www.rfc-archive.org/getrfc.php?rfc=951

[11] Sollins, K. (1992). RFC-1350 TFTP Protocol (Revision 2):
http://www.rfc-archive.org/getrfc.php?rfc=1350

[12] Anvin, H.P. SYSLINUX/PXELINUX:
http://syslinux.zytor.com/pxe.php.
(Datum des Zugriffs: 17. Mai 2006)

[13] Sarkar P., Missimer D., Sapuntzakis C. (2005) RFC-4173 Bootstrapping Clients using the Internet Small Computer System Interface (iSCSI) Protocol:
http://www.rfc-archive.org/getrfc.php?rfc=4173

[14] Droms R. (1997). RFC-2131 Dynamic Host Configuration Protocol:
http://www.rfc-archive.org/getrfc.php?rfc=2131

[15] Volz, B. (2004). RFC-3942 Reclassifying Dynamic Host Configuration Protocol version 4 (DHCPv4) Options:
http://www.rfc-archive.org/getrfc.php?rfc=3942

[16] Madden, B. S., Oglesby, R. (2004). Terminal Services for Microsoft Windows Server 2003 - Advanced Technical Design Guide:
Brianmadden.com Publishing Group, USA

[17] Microsoft Corporation (2004). Understanding the Remote Desktop Protocol.
Online im Internet: http://support.microsoft.com/kb/186607/en-us.
(Datum des Zugriffs: 2. Juni 2006)

[18] Thöing, C. (2002). Moderne Kryptographie - RC4. Online im Internet:
http://www.kuno-kohn.de/crypto/crypto/rc4.htm.
(Datum des Zugriffs: 3. Juni 2006)

[19] Microsoft Corporation (2005). Konfigurieren der Authentifizierung und Verschlüsselung. Online im Internet: http://technet2.microsoft.com/WindowsServer/de/Library/8be5bfb5-b652-49aa-8ac4-f6c2b01f35101031.mspx?mfr=true. (Datum des Zugriffs: 3. Juni 2006)

[20] Microsoft Corporation. Remote Desktop Protocol (RDP) Features and Performance. Online im Internet: http://www.microsoft.com/technet/prodtechnol/Win2KTS/evaluate/featfunc/rdpfperf.mspx. (Datum des Zugriffs: 3. Juni 2006)

[21] Mathers, T.W. (2004). Windows Server 2003/2000 Terminal Server Solutions, Third Edition: Addison Wesley Professional

[22] Larisch, D. (2005). Citrix Presentation Server – Grundlagen und Profiwissen: Carl Hanser Verlag, Bonn, Deutschland

[23] Dreyer, C. (2004). Citrix MetaFrame XP Presentation Server und Windows Terminal Services: mitp-Verlag, Bonn, Deutschland

[24] Lüdeman, N. (2006). Citrix Presentation Server 4 – Grundlagen, Praxis, Refernez: Galileo Computing, Bonn, Deutschland

[25] Citrix Systems, Inc. (2005). MetaFrame Presentation Server - Administratorhandbuch: Citrix Systems Inc., Ft. Lauderdale, U.S.A.

[26] Citrix Systems, Inc. (2001). Citrix ICA Priority Packet Tagging: Citrix Systems Inc., Ft. Lauderdale, U.S.A.

[27] Mansfield N. (1993). The Joy of X – An Overview of the X Window System: Addison Weley Publishing Company Inc.

[28] Smith, R. W. (2005). Linux in a Windows World: O'Reilly Media Inc., Sebastopol. U.S.A.

[29] Cowan, P. (2001). What is PXE? – White Paper: 3Com Corporation, Santa Clara, U.S.A.