Vysoká škola báňská – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Využití prostředí Preboot eXecution Environment pro správu počítačových systémů

Leveraging the Preboot eXecution Environment for Managing Computer Systems

2014 Petr Antončík

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání bakalářské práce

Student: **Petr Antončík**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R059 Mobilní technologie

Téma: Využití prostředí Preboot eXecution Environment pro správu počítačových systémů
Leveraging the Preboot eXecution Environment for Managing Computer Systems

Zásady pro vypracování:

1. Popis prostředí Preboot eXecution Environment (PXE).
2. Návrh řešení pro správu počítačové učebny založené na prostředí PXE a OS Linux.
3. Ověření funkčnosti s využitím serveru a bezdiskových stanic.

Pro vypracování závěrečné práce bude použit typografický systém LaTeX.

Seznam doporučené odborné literatury:

Nemth E., Snyder G., Hein T. R. *Linux. Kompletní příručka administrátora*. Brno: Computer Press. 2008. ISBN 80-722-6919-4.
Oetiker, T., a kol. *Ne příliš stručný úvod do systému LaTeX 2e.*

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Miroslav Bureš**

Datum zadání: 16.11.2012
Datum odevzdání: 07.05.2014

doc. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava.*

V Ostravě 30. dubna 2014 ..... Petr Antonáč .....

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 30. dubna 2014 ..... Petr Antonáč .....

Rád bych tímto poděkoval panu Ing. Miroslavu Burešovi, který mi pomohl porozumět jednotlivým oblastem v konfiguracích Linuxu a dopomohl tak ke vzniku bakalářské práce. Dále bych rád poděkoval rodině a profesorům, kteří mě podporovali při sepisování této práce.

Abstrakt

Tato práce je sepsána jako návod pro jednoduchou konfiguraci serveru, který by měl fungovat na principu PXE, a spouštět tak bezdiskové stanice nacházející se v lokální síti. Jedná se o shrnutí základních informací získaných z různých zdrojů a jejich aplikaci v praxi. Práce obsahuje informace o tom, co všechno potřebujeme pro spuštění PXE, a také návody, jak nastavit dílčí části serverů.

Klíčová slova: PXE, informace, konfigurace, DHCP, TFTP, NFS, ISO, Debootstrap, Dstat, Phoronix

Abstract

This work is write like a manual of server configuration which has functionality like PXE server and it should run clients in local network without discs. It is summary of main information collected from diferent sources and aplication them in real. The work has information what we need for PXE server and a manual how configure each part of servers.

Keywords: PXE, information, configuration, DHCP, TFTP, NFS, ISO, Debootstrap, Dstat, Phoronix

Seznam použitých zkratek a symbolů

CD	–	Compact Disc
DHCP	–	Dynamic Host Configuration Protocol
DNS	–	Domain Name System
FTP	–	File Transfer Protocol
HDD	–	Hard Disk Drive
IP	–	Internet Protocol
ISO	–	International Organization for Standardisation
JPEG	–	Joint Photographic Experts Group
PC	–	Personal Computer
PCI	–	Peripheral Component Interconnect
PCI- Express	–	Peripheral Component Interconnect Express
PNG	–	Portable Network Graphics
PXE	–	Preboot eXecution Environment
RAM	–	Random Access Memory
SSD	–	Solid State Drive
TCP	–	Transmission Control Protocol
TFTP	–	Trivial File Transfer Protocol
UCK	–	Ubuntu Customization Kit
UDP	–	User Datagram Protocol

Obsah

1 Úvod

V dnešní době moderních a výkonných počítačů nedělá lidem žádný problém pořídit si vhodný, cenově dostupný a výkonný počítač. Obzvláště teď, kdy se nám na pultech objevují stále výkonnější počítače se širokou škálou možností. Přesto všechny tyto počítače, ať jsou jakkoli výkonné, potřebují místo pro ukádaní svých dat, aby s nimi mohli později pracovat. K tomu nám standardně slouží HDD neboli harddisk. Dnes se oproti dřívějším dobám začínají objevovat tyto harddisky s terabajtovými oddíly, a to nemluvím o nových typech harddisků SSD, které jsou několikanásobně rychlejší ve čtení a zapisování dat než jejich předchůdci HDD. Nevýhodou ovšem může být, že námi ukládaná data jsou dostupná pouze lokálně na daném počítači, tudíž máme přístup k těmto datům pouze na konkrétním počítači, na kterém jsme s daty pracovali a ukládali je. Problém může nastat v případě, kdy s těmito daty potřebujeme náhle pracovat, ale nemáme přístup k danému počítači, na kterém jsou data obsažena, a který obsahuje program pro práci s těmito daty. V takovém případě existuje řešení v podobě bezdiskových stanic, které vznikly z dřívějších bezdiskových terminálů. Díky těmto stanicím můžeme s daty pracovat na kterémkoliv počítači, aniž bychom potřebovali data nějak externě přenášet například pomocí různých flashdisků. To je způsobeno tím, že jsou tyto stanice napojeny na centrální server, který obsahuje všechna naše data a potřebné programy pro práci s nimi. Tento způsob spouštění počítačů jako bezdiskových stanic byl vhodný především v dřívějších dobách, kdy hardwarové součástky byly příliš drahé a nebyly příliš výkonné. Výhodou těchto bezdiskových stanic jsou nízké náklady na správu, protože jsou prakticky spravovány serverem. Také nedochází ke ztrátě dat v případě poškození klientského počítače, protože data jsou uložena na serveru. Příkladem pro nastavení takové sítě, ve které se počítače budou spouštět ze serveru, je PXE rozhraní.

Ve své bakalářské práci se zaměřím především na to, jak vytvořit takovou bootovací síť mezi serverem a jedním či více klienty. Zkusím popsat, jaké prostředky budu potřebovat, abych tuto síť vytvořil, jaké serverové daemony budu muset nastavit a jakým způsobem. V první části bakalářské práce si rozeberu teorii jednotlivých serverových daemonů, jako je DHCP, TFTP, NFS a samotnou teorii PXE. Popíšu zde, k čemu jednotlivé serverové daemony slouží, jak fungují a proč jsou důležité. Další část mé práce bude obsahovat praktický postup jak nastavit jednotlivé serverové daemony v linuxu. Ukážu, jaké příkazy se musí pro nastavení těchto daemonů zadat, a jaký je význam jednotlivých příkazů. Dále pak uvedu, jak jsem vytvořil své vlastní bootovací prostředí, jak jsem ho nastavil, a jak samotné prostředí fungovalo.

2 PXE

2.1 Co je PXE

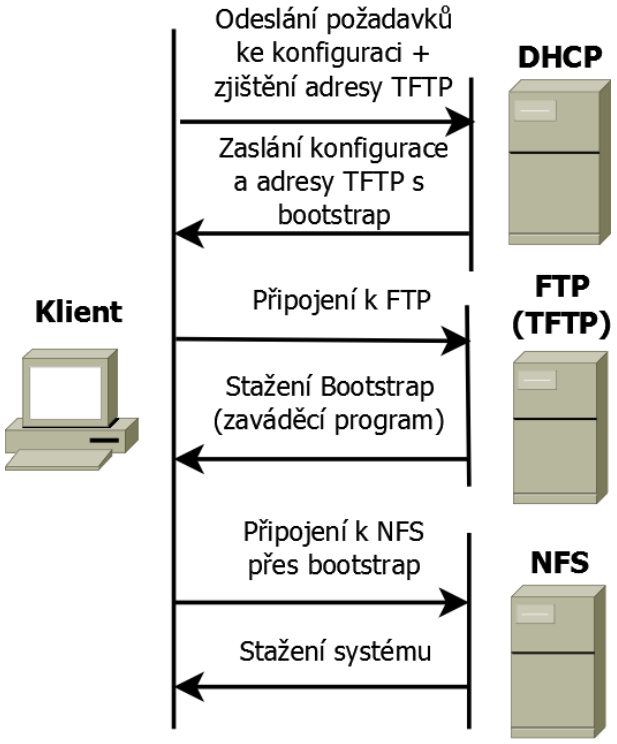
Preboot eXecution Environment neboli PXE je označení technologie určené pro spouštění systému počítače z počítačové sítě. Také se dá využít pro automatické instalace operačních systémů. Tento standard byl vytvořen firmou Intel v září roku 1999 a nahradil dosud používané technologie například BOOTP. PXE rozšiřuje možnosti startu počítače tím, že síťová karta obsahuje flash paměť s kódem, který rozšiřuje schopnosti BIOSu o zavedení operačního systému z počítačové sítě. V dnešní době je PXE dodáván jako součást BIOSu základních desek s integrovanou síťovou kartou, a také externích síťových karet, které jsou připojeny pomocí slotů PCI nebo PCI-Express. Hlavní výhodou PXE je, že je nezávislé na počítačové platformě jak z pohledu hardwaru, tak i softwaru. Takže funguje na jakékoli počítačové sestavě a jakémkoli operačním systému.

2.1.1 Komunikace v PXE

Samotná komunikace mezi síťovou kartou a servery je vytvářena pomocí IP protokolu. Při zapnutí počítače dochází k vybuzení síťové karty, ke které DHCP server přiřadí údaje pro komunikaci v síti, jako je například IP adresa. Tyto údaje získá klient tak, že počítač pomocí DHCP vyšle broadcastovou zprávu v UDP datagramu spolu se správou DHCPDISCOVER a PXE příkazem. Na tuto zprávu odpoví DHCP server, který je nastaven pro spouštění zařízení v síti. Tento server odešle zprávu DHCPOFFER s údaji a adresou k FTP (TFTP) serveru zpět na klientské PC. Když klient zná tyto údaje, vyšle druhou zprávu na DHCP server, a to DHCPREQUEST pro zjištění kompletní cesty k zaváděcímu programu (bootstraping), pomocí kterého dochází k bootování ze sítě. DHCP server odešle informace o této cestě ve zprávě DHCPACK. Následně klient začne stahovat pomocí UDP protokolu zaváděcí program z FTP (TFTP) serveru a uloží si jej do paměti RAM. Tento zaváděcí program funguje jako zavaděč jádra operačního systému. Když je tento program stažen, dochází k jeho spuštění. Poté si sám řídí síťovou komunikaci a zavádí do paměti RAM jádro operačního systému, který následně spustí. Obrázek 2.1 ukazuje, jak spouštění probíhá.

2.1.2 Co je potřebné pro spouštění PXE

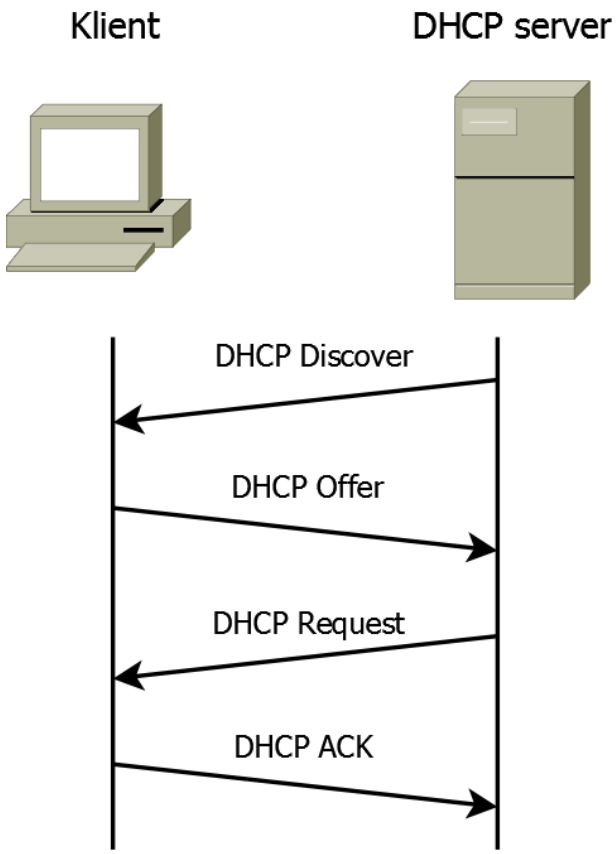
Abychom vytvořili plnohodnotné bootovací prostředí pro Linux, tedy PXE prostředí, potřebujeme mít nakonfigurováno několik serverových daemonů, které nám spouštění přes síť umožní. Prvním z nich je DHCP daemon. Ten nám bude sloužit pro přiřazování IP adres a informací k jednotlivým síťovým rozhraním. Dále potřebujeme nakonfigurovat FTP (TFTP) daemon. Ten bude zprostředkovávat přenos souborů, které jsou potřebné pro spouštění ze sítě. Nakonec nakonfigurujeme NFS daemon sloužící pro uložení a sdílení souborů stahovaných při spouštění systému. Když máme všechny daemony nakonfigurovány, nastavíme ještě syslinux neboli samotný systém PXE, který nám bude bootovat počítače.



Obrázek 2.1: Komunikace se servery.

2.2 DHCP

Název DHCP pochází z anglického Dynamic Host Configuration Protocol. Jedná se o protokol z rodiny TCP/IP, avšak tato zkratka se také používá pro označení odpovídajícího DHCP serveru nebo klienta. Tento protokol slouží k automatické konfiguraci počítačů a zařízení, která jsou připojena k počítačové síti. Hlavní úlohou DHCP je přidělování IP adres, síťových masek, defaultní brány a adresy DNS serveru, který nám IP adresy mapuje na jména. DHCP nám tyto informace poskytuje pouze po omezenou dobu, a tedy říkáme, že dochází k zapůjčení těchto adres. Po uplynutí poloviny doby zápůjčky dochází k tomu, že klient, pokud potřebuje, zapůjčení adresy znovu obnoví. Sám si tedy vyžádá prodloužení platnosti adres. Hlavní výhodou automatického přidělování IP adres a informací pro jednotlivá zařízení je, že není zapotřebí externího zásahu administrátora. To je obzvláště výhodné pro stanice, které nejsou k síti připojeny trvale a dochází k jejich vypínání, jako je například osobní počítač. Další výhodou automatického přiřazování adres je konfigurace zařízení po výpadku v síti. V takovém případě při opětovném spuštění zařízení, které během výpadku nefungovalo, rozešle DHCP informace vhodné pro připojení do sítě, aniž by byla potřeba zásahu správce dané sítě. Na obrázku 2.2 můžeme vidět komunikaci, ke které dochází mezi klientem a DHCP serverem.



Obrázek 2.2: Ukázka DHCP komunikace.

2.3 FTP

Název FTP pochází z anglického File Transfer Protocol. Jedná se protokol, který slouží pro přenos souborů mezi počítači pomocí počítačové sítě. Tento protokol se hlavně využívá pro anonymní přístup, takže ke sdíleným datům může mít přístup kdokoli. To může být považováno za narušení bezpečnosti na serveru. Příkladem narušení bezpečnosti může být třeba, že přihlašovací údaje pro připojení k serveru jsou přenášeny v běžné textové podobě a při jejich odchycení v komunikaci mohou být zneužity. Samotná komunikace je prováděna pomocí TCP protokolu. U TFTP (Trivial File Transfer Protocol) je naopak využit UDP protokol.

2.4 TFTP

TFTP (Trivial File Transfer Protocol) je jednoduchý protokol pro přenos souborů, který obsahuje pouze základní funkce protokolu FTP. Tento standard byl stanoven v roce 1980 a je určen pro přenos souborů v případech, pro které je standardní protokol FTP příliš komplikovaný. Jeden z těchto příkladů je právě spouštění systému přes síť, kdy musíme přenášet pouze omezené množství dat kvůli omezené velikosti paměti RAM.

2.4.1 Odlišnosti TFTP protokolu oproti FTP

1. Neumožněné procházení adresářů.
2. Nemožnost přihlašování uživatelů a zadávání hesel.
3. Maximální velikost souboru je 32 MB.
4. Používá se pro čtení nebo zápis dat na server.
5. Používá tři přenosové metody: netascii, octet a mail.

2.5 NFS

NFS neboli Network File System je protokol sloužící ke vzdálenému přístupu k souborům přes počítačovou síť. Poprvé se tento protokol objevil v roce 1984 a byl představen firmou Sun Microsystems. Funkčnost tohoto protokolu je hlavně zaměřena nad UDP protokolem, avšak dokáže také využít i TCP protokolu. Jeho hlavní úkol spočívá v tom, že dokáže připojit vzdálený disk, který se nachází na nějakém serveru v síti, k počítači tak, že se nám tento disk jeví jako lokální a je součástí našeho počítače. Pokud použijeme NFS server, tak si také na tomto serveru můžeme nastavit k čemu nám bude sloužit, a podle toho přiřadit práva přístupu na tento server. Příkladem může být, že si server nastavíme pouze pro čtení, a tedy kdokoliv kdo se na tento server připojí, bude moci data pouze číst a stahovat, ale nikoliv zapisovat.

2.6 UDP

UDP neboli User Datagram Protocol je jedním z internetových protokolů využívajících se pro přenos dat v počítačových sítích. Zmiňuji ho, protože se jedná o hlavní protokol, který nám zprostředkovává všechen přenos dat, ke kterému dochází při spouštění systému ze sítě. Oproti TCP protokolu se jedná o protokol méně spolehlivý z hlediska přenosu. Tento protokol, zjednodušeně řečeno, pouze zasílá data na jiná zařízení připojená v síti a neřeší jejich doručení. To znamená, že nám pouze odesílá data a neručí nám, zda dorazí na zařízení, na které chceme. Ani nám neručí v jakém pořadí dorazí nebo zda dorazí duplicitní data. Toto je hlavní rozdíl oproti protokolu TCP, který je mírně řečeno striktní a kontroluje jednotlivé pakety zda dorazily, v jakém pořadí, a zda některé nechybí. V takovém případě nám TCP přenese chybějící pakety ještě jednou.

Protokol UDP je vhodný pro jednoduché aplikace, které fungují pouze na principu otázek a odpovědí, a také pro zmíněné sdílení dat. Využívá se hlavně v aplikacích, ve kterých nevadí ztráta nepatrných dat, jako je například VOIP telefonie nebo online hry netolerující časové prodlevy. Kdyby se pro tyto aplikace použil protokol TCP, tak by docházelo k časovým ztrátám.

3 Konfigurace

3.1 DHCP-Ubuntu

Jak již bylo řečeno v úvodu, tak abychom mohli PXE server využívat ve své lokální síti, potřebujeme mít nainstalováno několik Linuxových daemonů, tedy programů, běžících na pozadí spuštěného systému, které nám umožní využívat námi definované potřebné služby. Jedním z nich je dhcp3-server daemon, který nám spustí na pozadí běžícího systému DHCP server starající se o přidělování IP adres a konfigurace sítě jednotlivých klientů. Jeho hlavní role bude spočívat, mimo jiné, převážně při prvotním spuštění klientského počítače, kdy nám automaticky nastaví síťová rozhraní klientů pro spouštění systému ze sítě.

V následujících řádcích popíšu podrobný postup, jak si nastavit vlastní DHCP server, jaké příkazy budeme muset použít a podám vysvětlení jednotlivých příkazů.

1. Prvním příkazem, který bychom měli použít, je příkaz **sudo**. Tento příkaz nám umožní přepnutí z našeho účtu uživatele na účet super-uživatele. Takto se nám rozšíří práva a my budeme moci zasahovat a měnit konfiguraci v našem systému, do kterého bychom z běžného účtu nemohli zasahovat. V případě, že bychom jsme se nepřepli do účtu super-uživatele, tak bychom museli před každým příkazem psát předponu sudo.

   ```
   sudo -i
   ```

2. Jakmile jsme v účtu super-uživatele, tak si nejdříve nainstalujeme ovladače pro DHCP server pomocí příkazu **apt-get install**. Ovladače se nacházejí v balíčku **dhcp3-server**. To provedeme příkazem:

   ```
   apt-get install dhcp3-server
   ```

   Při instalaci se nám vytvoří složka **/etc/default/** se souborem **isc-dhcp-server** a složka **/etc/dhcp/** se souborem **dhcpd.conf**.

3. Předtím, než začneme samotnou konfiguraci, si ovšem zjistíme, jaké rozhraní mají přiřazené naše síťové karty, abychom mohli dále v konfiguraci nastavit, ze kterého rozhraní bude DHCP server přidělovat jednotlivé IP adresy. Zjištění provedeme příkazem:

   ```
   ifconfig
   ```

4. Pokud bychom nastavovali statickou adresu přes příkaz **ifconfig**, tak bychom si měli zastavit network-managera, který by nám mohl naši statickou adresu přepisovat. Zastavení network-managera provedeme pomocí:

   ```
   service network-manager stop
   ```

5. Nyní přikročíme k samotné konfiguraci rozhraní, která se nachází v adresáři **/etc/-network/interfaces**, kde nastavíme defaultní rozhraní serveru/pc, nebo-li statickou IP adresu serveru, kterou budeme používat pro dhcp server. Konfiguraci provedeme přes textový editor **gedit** příkazem:

   ```
   gedit /etc/network/interfaces
   ```

   V případě, že by tento editor měl problémy spustit se v rozhraní super-uživatele, tak na začátek příkazu vložíme ještě předponu **sudo** a spusíme ho z režimu běžného uživatele.

   Zde doplníme následující údaje, které nám říkají, že při spuštění systému se nám automaticky nastaví síťové rozhraní eth0 s přidělenou statickou adresou 192.168.1.1. Bude se jednat o síť 192.168.1.0 s maskou 255.255.255.0 a výchozí branou 192.168.1.1.

   ```
   auto eth0
   iface eth0 inet static
   address 192.168.1.1
   network 192.168.1.0
   netmask 255.255.255.0
   gateway 192.168.1.1
   ```

   Pak tuto konfiguraci uložíme a uzavřeme soubor.

6. Abychom mohli nově nastavenou konfiguraci použít, musíme si **restartovat síť a rozhraní** , kde se mají nové parametry načíst. Restartování provedeme příkazem:

   ```
   /etc/init.d/networking restart
   nebo
   service networking restart
   ```

   Při spuštění příkazu **ifconfig**, bychom měli vidět nově nastavené parametry z kroku číslo 5.

7. Nyní nastavíme rozhraní, přes které nám dhcp server bude přidělovat adresy. To provedeme úpravou souboru **/etc/default/isc-dhcp-server** opět přes textový editor **gedit** příkazem:

   ```
   gedit /etc/default/isc-dhcp-server
   ```

   Upravíme soubor tak, že vložíme všechna **rozhraní**, přes která může DHCP server přidělovat síťové nastavení. Zde je výpis:

   ```
   INTERFACES="eth0"
   ```

8. Posledním souborem, který při své cestě za úspěšným spuštěním DHCP serveru budeme nastavovat je **dhcpd.conf**, kde nastavíme pravidla pro samotný DHCP server. Jedná se o adresy, které se budou přidělovat, po jak dlouhou dobu budou platit u daného klienta a další informace pro dhcp server. I zde se dostaneme přes textový editor **gedit** příkazem:

   ```
   gedit /etc/dhcp/dhcpd.conf
   ```

   Uvnitř tohoto souboru si nastavíme doménu pro DHCP server a základní **čas pro zapůjčení IP adresy s maximální dobou zápůjčky**. Dále zde vložíme námi známé parametry, jako je **maska sítě, IP adresa broadcastu, výchozí bránu DHCP serveru a adresu DNS serveru**, který můžeme také vytvořit. V poslední části pak nastavíme **podsíť s maskou a rozsahy přidělovaných adres**. Nesmíme také zapomenout definovat soubor **pxelinux.0**, který bude klientům připojeným k této síti oznamovat, že si zde mohou spustit systém bootovaný přes síť. Níže vidíme příklad výpisu ze souboru **dhcpd.conf**:

   ```
   option domain-name "petr.dhcp";
   default-lease-time 86400;
   max-lease-time 86400;

   option subnet-mask 255.255.255.0;
   option broadcast-address 192.168.1.255;
   option routers 192.168.1.1;
   option domain-name-servers 192.168.1.1;

   subnet 192.168.1.0 netmask 255.255.255.0 {
   range 192.168.1.11 192.168.1.100;
   filename "pxelinux.0";
   option subnet-mask 255.255.255.0;
   }
   ```

   V případě, že bychom chtěli mít u daného klienta stálou IP adresu, která by byla přidělována DHCP, můžeme ji pro daný počítač nastavit tak, že do souboru připíšeme IP adresu, kterou provážeme s MAC adresou klienta. Provázání provedeme například:

   ```
   host PC1 {
   hardware ethernet 00:11:22:AA:BB:CC;
   fixed-address 192.168.1.10;
   }
   ```

   Soubor si opět uložíme a zavřeme textový editor.

9. Nyní máme vše nastaveno pro bezchybnou funkci dhcp serveru. Jediné, co nám ještě zbývá, je **restartovat server**, aby načetl nově nabytou konfiguraci a pracoval podle našich stanovených pravidel. Restart provedeme příkazem:

   ```
   service isc-dhcp-server restart
   nebo
   /etc/init.d/isc-dhcp-server restart
   ```

Po provedení všech těchto kroků, bychom měli mít funkční DHCP server, který by nám měl automaticky přidělovat adresu v rozsahu od 192.168.1.11 do 192.168.1.100. Celkově by měl být schopen obsloužit až 90 zařízení v síti. Funkčnost si můžeme ověřit tak, že k jeho rozhraní připojíme například jiný počítač a zjistíme, zda v síťovém nastavení obdržel jednu z našich definovaných adres.

3.2 TFTP

Další daemon, který je důležitý pro funkčnost PXE serveru, je **tftpd-hpa daemon**, který nám poskytne ovladače pro funkčnost FTP serveru, v našem případě spíše pro funkčnost **TFTP serveru**. Díky těmto ovladačům budeme moci přenášet naše data, která budeme chtít přenést ze serveru na klienta při spouštění klienta. Konkrétněji systémová data potřebná pro spuštění a funkci systému, která budeme mít zabalena v ISO souboru na serveru, nebo v nějaké přednastavené složce. Pomocí TFTP protokolu se nám budou tato data přenášet s ohledem na omezenou paměť RAM, která je přidružená k síťové kartě.

1. Pro používání FTP a TFTP serveru si nejdříve budeme muset ovladače nainstalovat. To provedeme standardně pomocí **apt-get install** [9]. Zde vidíme příkaz pro instalaci:

   ```
   apt-get install tftpd-hpa
   ```

   Když proběhne instalace, tak se nám vytvoří adresář se složkami **/var/lib/tftpboot/**. Do tohoto adresáře pak budeme připojovat soubory, které budeme chtít přenést TFTP protokolem a spustit tak systém.

2. Po instalaci TFTP serveru, kdy už máme potřebné ovladače a složky, musíme ještě nainstalovat hlavní **bootovací soubory**, které nám zajistí komunikaci a to, že budeme moci spouštět systém ze sítě. Tyto soubory se nacházejí v instalačním balíčku s názvem **syslinux**. Tento balíček nainstalujeme následujícím příkazem:

   ```
   apt-get install syslinux
   ```

   Po instalaci balíčku syslinux se nám vytvoří adresář **/usr/lib/syslinux/** se soubory. My pro náš server budeme potřebovat tři soubory s názvy **pxelinux.0, menu.c32 a memdisk**, které se nacházejí v tomto adresáři. Později si ukážeme, jak nahradíme soubor **menu.c32 za vesamenu.c32**, který nám dovolí upravit vzhled spouštěcího menu.

3. Dalším krokem, který musíme udělat, je předpřipravení TFTP serveru pro bootování. To provedeme tak, že zmíněné soubory **pxelinux.0, menu.c32 a memdisk** překopírujeme do systémového bootovacího adresáře **/var/lib/tftpboot/**. Kopírování provedeme příkazem **cp**, kde určíme cestu a název souboru, který chceme kopírovat, a poté cestu, kde soubor chceme vložit. Zde jsou příkazy ke kopírování těchto souborů:

   ```
   cp /usr/lib/syslinux/pxelinux.0 /var/lib/tftpboot/
   cp /usr/lib/syslinux/menu.c32 /var/lib/tftpboot/
   cp /usr/lib/syslinux/memdisk /var/lib/tftpboot/
   ```

4. Poslední věcí, kterou si předpřipravíme TFTP server je, že si vytvoříme konfigurační složku **pxelinux.cfg** v adresáři TFTP serveru. V této složce pak vytvoříme konfigurační soubor, který bude obsahovat cesty k datům. Tato data budou zaslána TFTP serverem při spuštění. Složku vytvoříme pomocí příkazu **mkdir** (název složky) pokud budeme v adresáři tftpboot, nebo příkazem:

   ```
   mkdir /var/lib/tftpboot/pxelinux.cfg
   ```

Provedením těchto příkazů a nastavením TFTP serveru, jsme připraveni provést nastavení pro samotný PXE server, a tak zajistit, že budeme moci spouštět systém ze sítě a nebudeme potřebovat mít nainstalovaný systém na lokálním harddisku.

3.3 NFS

Díky externímu NFS serveru budeme moci ukládat svá data na tento server a později k nim také přistupovat a pracovat s nimi jako na lokálním disku. [14]

3.3.1 Nastavení serveru

1. Nejdříve si nainstalujeme ovadače pro NFS server příkazem:

   ```
   apt-get install nfs-kernel-server
   ```

   Kromě složky **/srv**, kde budeme připojovat spouštěcí soubory, se nám také vytvoří složka **/etc** se souborem **exports**. Do tohoto souboru pak definujeme jednotlivá práva přístupu pro jednotlivé klienty. Musíme si ovšem uvědomit, že NFS není zcela bezpečný systém a má řadu bezpečnostních děr, které zkušení uživatelé mohou snadno obejít. Je to tím, že na úrovni souborů je přístup řízen stejně jako na lokálních souborových systémech pomocí hodnot UID a GID. V případě, že by dva uživatelé měli na různých počítačích stejné UID nebo by si jeden uživatel přepsal své UID na svém počítači na jiné totožné s jiným uživatelem, pak by tento uživatel mohl disponovat se soubory prvního uživatele. Proto by se tento systém měl používat primárně v místních a důvěryhodných sítí a firewallem by se mělo řešit zabránění přístupu z vnější sítě.

2. Když je server nainstalovaný, tak si pro kontrolu zjistíme, zda nám běží na defaultním **portu 2049**. Použijeme příkaz:

   ```
   rpcinfo -p | grep nfs
   ```

3. Také si můžeme zkontrolovat, jaké verze nám NFS server podporuje pomocí:

   ```
   cat /proc/filesystems | grep nfs
   ```

   Většinou se jedná o verze **nfs, nfs4, nfsd**.

4. Posledním ověřovacím příkazem, který můžeme pro kontrolu použít, a který nám zjistí na jakém portu běží takzvané **odposlouchávače** mapující přístup k NFS je:

   ```
   cat /proc/filesystems | grep nfs
   ```

5. Jestliže používáme starší systém Linux, tak bychom měli **integrovat nfs službu do běžícího linuxového jádra**. Jedná se o zavedení potřebných ovladačů do právě běžícího systému. V novějších verzích Linuxu se tato akce provádí zpravidla automaticky:

   ```
   modprobe nfs
   ```

6. Hlavní soubor, který musíme nastavit je **exports**, ve kterém definujeme **práva k přístupu na NFS server** jednotlivým klientům, kteří k NFS budou přistupovat:

   ```
   gedit /etc/exports
   ```

   Uvnitř souboru pak nastavíme jednotlivá práva přístupu pro dané IP adresy. Já si zde nastavil plný přístup pro všechny klienty v mé síti tím, že jsem zadal:

   ```
   /srv/boot/iso/ubuntu-12.04.2-desktop-i386
   192.168.1.1/255.255.255.0(rw,sync,no_subtree_check,
   no_root_squash)
   /mnt/debiana 192.168.1.1/255.255.255.0(rw,sync,
   no_subtree_check,no_root_squash)
   /home/petr/nfs 192.168.1.1/255.255.255.0(rw,sync,
   no_subtree_check,no_root_squash)
   ```

   První řádek povoluje spouštění live distribuce, kterou připojím k této složce. Druhý řádek představuje místo s rozbaleným plnohodnotným systémem a v posledním řádku mám místo na ukládaní souborů.

7. Když máme vše nastaveno, tak restartujeme server následujícím příkazem:

```
/etc/init.d/nfs-kernel-server restart
nebo
service nfs-kernel-server restart
```

Popřípadě můžeme exportovat souborové systémy uvedené v exports (doporučuje se provést po každé změně v exports) příkazem:

```
exportfs -a
```

3.3.2 Připojení klienta

Abychom mohli používat vzdálené serverové úložiště jako lokální, musíme mít nainstalovaného NFS klienta. Zvolíme si místo, kde se nám budou ukazovat soubory uložené na serveru a na toto místo připojíme NFS.

1. Nejdříve si tedy nainstalujeme klienta příkazem:

   ```
   apt-get install nfs-common
   ```

   Tohoto klienta doporučuji nainstalovat do upravené Linuxové distribuce. Podrobněji se jí zabývám v kapitole *Vytvoření vlastního ISO CD*.

2. Výsledné připojení pak provedeme jediným příkazem, po kterém můžeme s daty na serveru disponovat. Připojíme se tedy pomocí příkazu:

   ```
   mount -t (co) (adresa):(cesta na serveru) (cesta na klientovi)
   mount -t nfs 192.168.1.1:/home/petr/nfs  /home/petr/nfs
   ```

   Zde pak můžeme ukládat všechny naše soubory.

3.4 PXE

Teď bych se rád věnoval hlavní konfiguraci PXE serveru pro systém Linux. Popíšu zde jednotlivé soubory, které musíme nastavit, jaké hodnoty jim musíme přiřadit, a jaké soubory máme kde připojit, abychom vytvořili plně funkční bootovací PXE server.

1. Jestliže chceme bootovat náš operační systém, tak si musíme vytvořit složku, kde bude uložen tento operační systém. Pro svůj projekt jsem si vybral, že si uložím **ISO obraz** se systémem ve složce na svém účtu. Složku, kterou budu používat pro ukládaní systémových ISO obrazů, jsem si nazval iso a vytvořil ji ve svém domovském adresáři **/home/petr/**. Na školním počítači bych tuto složku vytvořil v adresáři **/home/student/**. Složku iso vytvoříme příkazem:

   ```
   mkdir /home/petr/iso
   ```

2. Další iso složku si vytvoříme do adresáře **/var/lib/tftpboot/**, a potom do této složky vložíme **složku s názvem systému**, do které budeme připojovat systémové soubory určené pro přenos TFTP protokolem. Zde jsou příkazy pro vytvoření:

```
mkdir /var/lib/tftpboot/iso
mkdir /var/lib/tftpboot/iso/ubuntu-12.04.2-desktop-i386
```

3. Poslední složku iso musíme vytvořit v adresáři **/srv/boot/**. Tento adresář se bude vyznačovat tím, že k souborům systému, které budeme spouštět přes síť, bude mít **přístup NFS server**, na který si budeme moci ukládat svá vytvořená data. Pokud se nám vytvoří pouze adresář **/srv/** bez boot složky, pak si tuto složku pro přehlednost vytvoříme sami následujícími příkazy:

```
mkdir /srv/boot
mkdir mkdir /srv/boot/iso
```

4. Když máme všechny adresáře připraveny, můžeme do našeho hlavního adresáře **/home/petr/iso/ nebo /home/student/iso/** nakopírovat ISO obraz systému, který chceme spouštět po síti. Druhý způsob je přejít do složky iso, a poté do ní **stáhnout** systém z internetu. Zde je příklad odkud lze stáhnout systém linux:

```
cd /home/petr/iso
wget http://releases.ubuntu.com/precise/
ubuntu-12.04.2-desktop-i386.iso
```

5. Nyní, když máme připravený systém, který budeme spouštět po síti, si připravíme soubor **default**, který vytvoříme v našem předpřipraveném adresáři pro FTP server **/var/lib/tftpboot/pxelinux.cfg/**. V tomto souboru si pak připravíme menu, které nám ukáže systémy spustitelné po síti, a ke každému z těchto systému přiřadíme cestu ke spouštěcím souborům. Celou konfiguraci provedeme opět přes textový editor gedit:

```
gedit /var/lib/tftpboot/pxelinux.cfg/default
```

Soubor nakonfigurujeme tímto způsobem:

```
DEFAULT menu.c32
TIMEOUT 600
ONTIMEOUT localboot
MENU TITLE PXE server - Petr Antoncik
LABEL localboot
MENU LABEL ^Local Boot (bootovat z CD nebo HDD)
MENU DEFAULT
LOCALBOOT 0
```

```
LABEL Ubuntu 12.04.2 32-bit LIVE
MENU LABEL ^Ubuntu 12.04.2 32-bit LIVE
LINUX iso/ubuntu-12.04.2-desktop-i386/casper/vmlinuz
INITRD iso/ubuntu-12.04.2-desktop-i386/casper/initrd.lz
APPEND boot=casper netboot=nfs
nfsroot=192.168.1.1:/srv/boot/iso/ubuntu-12.04.2-desktop-i386
```

Vidíme, že pro zobrazení bootovacího rozhraní bude náš server využívat defaultně ovladač **menu.c32**, který jsme získali z balíčku syslinux. Položkou **TIMEOUT** nastavujeme dobu po kterou bude naše menu k dispozici. Po ulynutí této doby se nám spustí příkaz **ONTIMEOUT**, který nám říká, co se má stát po uplynutí daného časového intervalu. Nadefinoval jsem si akci **localboot**. V případě, že si nic nevyberu, dojde k přepnutí na menu, které mi dá nabídku spouštění z lokálních nainstalovaných systémů. Další položkou je **MENU TITLE PXE server**, kde si zadáme název našeho serveru. Po této hlavní konfiguraci následují jednotlivé položky se systémy, které poskytujeme k bootování. Jako první jsem si nastavil bootovaní z **lokálního harddisku**, který jsem si zvolil jako defaultní volbu. Druhý nastavený systém je systém bootovaný přes síť, který jsem si stáhl v ISO obrazu. V souboru můžeme vidět cesty ke spouštěcím souborům v ISO obrazech, a dále pak cestu k operačnímu systému uloženému na NFS serveru s IP adresou 192.168.1.1.

6. Teď přikročíme k připojení obrazu systému do jednotlivých složek serverů. První příkaz nám připojí systémové soubory z ISO obrazu do **složky TFTP boot**. Je to z toho důvodu, aby TFTP server měl přístup ke spouštěcím souborům, které musí poskytnout pro spuštění systému. Tuto akci provedeme příkazem:

   ```
   mount -t iso9660 -o loop /home/petr/iso/ubuntu-12.04.2
   -desktop-i386.iso /var/lib/tftpboot/iso/ubuntu-12.04.2
   -desktop-i386
   ```

7. Dále musíme připojit systémové soubory na **NFS server**, aby nám došlo ke sdílení těchto souborů v naší síti, abychom s nimi mohli pracovat a nabootovat tak náš systém. Příkaz kterým připojíme sdílené soubory je:

   ```
   mount --rbind /var/lib/tftpboot/iso /srv/boot/iso/
   ```

8. Posledním příkazem si nadefinujeme **práva** k systémovým souborům na NFS serveru pro spouštění systému ze sítě. Tento příkaz můžeme použít jako náhradu, kdybychom nechtěli upravovat data v souboru **exportfs** od NFS. Práva jinak nastavíme například:

   ```
   exportfs -i -o async,no_root_squash,no_subtree_check,ro
   0.0.0.0/0.0.0.0:/srv/boot/iso/ubuntu-12.04.2-desktop-i386
   ```

9. Výše uvedené příkazy, pokud je nemáme definovány v souboru **/etc/rc.local** [13], musíme zadávat po každém restartovaní serveru. Pokud tyto kroky nechceme spouštět při každém restartu, tak si práci můžeme ulehčit tím, že si vytvoříme **spouštěcí skrypt**. Aby nám tento skript fungoval, tak jej musíme umístit do složky **/root** s příponou **.sh**. Vytvoříme si jej příkazem:

   ```
   gedit /root/obrazy.sh
   ```

   Poté do tohoto souboru vložíme všechny připojovací příkazy:

   ```
   mount -t iso9660 -o loop /home/petr/iso/ubuntu-12.04.2-desktop
   -i386.iso /var/lib/tftpboot/iso/ubuntu-12.04.2-desktop-i386
   sleep 1
   mount --rbind /var/lib/tftpboot/iso /srv/boot/iso/
   sleep 1
   exportfs -i -o async,no_root_squash,no_subtree_check,
   ro 0.0.0.0/0.0.0.0:/srv/boot/iso/ubuntu-12.04.2-desktop-i386
   echo "Obrazy pripojeny."
   exit 0
   ```

   Příkaz echo nám oznámí, jestli akce proběhla a jestli tyto příkazy byly provedeny při spuštění skriptu.

10. Nyní tomuto **skriptu povolíme práva**, abychom jej mohli používat a spouštět. To provedeme pomocí příkazu **chmod**. Příkaz povolení práv je:

    ```
    chmod 777 obrazy.sh
    ```

11. Když máme vše nastaveno, tak si skript **spustíme** příkazem:

    ```
    ./obrazy.sh
    ```

Pro kontrolu můžeme navšívit složky **/var/lib/tftpboot/iso/ubuntu-12.04.2-desktop-i386** a **/srv/boot/iso/**. V těchto složkách bychom měli vidět systémové soubory z ISO obrazu systému. V případě, že by tyto soubory nebyly vidět, musíme aplikovat celý postup znovu a zkontrolovat, zda jsme něco chybně nezadali. Druhou možností je zkontrolovat práva přístupu k daným složkám, které pro systém používáme.

Pomocí těchto postupů, které jsme prošli, bychom měli mít fungující PXE server, který nám stačí zapojit do dané sítě. Po připojení do sítě, by nám server měl nastavit adresy jednotlivých klientů přes DHCP protokol. Při spuštění klienta, který má povoleno spouštění ze sítě, by měl PXE server nabízet menu se síťovými systémy.

3.5 Nastavení klienta

Nastavení klientského počítače je jedna z nejsnažších částí bakalářské práce. První podmínkou je mít síťovou kartu s podporou PXE bootování. Pokud tuto kartu máme, tak druhou podmínkou je připojení k síti, ve které máme spuštěný a funkční PXE server. Splněním těchto podmínek a při správně nakonfigurovaném serveru, bychom neměli mít žadný problém se síťovým spouštěním.

1. Nejdříve si **spustíme klientský počítač**. Při spouštění stiskneme klávesovou zkratku, kterou se dostaneme do BIOSu. Standardně se jedná o klávesu:

   ```
   delete nebo F2
   ```

2. Když se nacházíme v konfiguračním prostředí BIOS, tak přejdeme do záložky s názvem boot. Zde si najdeme položku BOOT configuration, ve které provedeme dvě akce. První bude vyhledání položky, která nám povolí bootování ze sítě. Mělo by se jednat o položku:

   ```
   BOOT ROM [enable]
   nebo
   PXE [enable]
   případně
   NETWORK BOOT [enable]
   ```

3. Poté, když máme bootování po síti povoleno, **musíme nastavit pořadí zařízení**, ze kterých se má spouštět systém. Já si nastavil toto pořadí:

   ```
   PXE boot
   Flash HDD
   CD/DVD - ROM
   Local HDD
   ```

   Primárně mám nastaveno, aby nám klient nejdříve prošel síť a zjistil, zda není dostupný PXE server, a teprve potom se zaměřil na bootování z lokálních médií.

4. Přepneme se do složky exit, kde vybereme a potvrdíme volbu:

   ```
   exit with save
   ```

   Obdobným způbobem nastavíme všechny klienty. Musíme si uvědomit, že každý systém BIOS má své rozhraní a jinou konfigurační topologii, ale zpravidla by si tato rozhraní měla být podobná. Tento příklad nastavení klienta slouží pouze jako orientační příklad, který má nastínit postup konfigurace klienta. Nemá podávat detailní nastavení daného klienta.

3.6 Grafické bootovací menu a jeho úprava

Kromě klasického textového menu, které použijeme při nakopírování souboru menu.c32, máme možnost použít také grafické menu. Oproti textovému, které má jednoduchou základní podobu podobnou systémům BIOS v základních deskách a nelze v něm provádět úpravy vzhledu, má grafické menu možnosti této vizuální úpravy. Je proto vhodnější pro naše idividuální potřeby z hlediska vizuálního dojmu, který má dát náš systém uživateli. Toto grafické menu můžeme použít jako náhradu za obyčejné textové menu tak, že nahradíme soubor menu.c32 souborem vesamenu.c32, který se nachází ve stejné složce v syslinux. V následujících několika řádcích ukážu, jak si takové spouštěcí menu můžeme upravit, a popíšu jednotlivé příkazy a jejich vliv na vzhled menu. [4, 18, 19]

1. Než budeme moci začít upravovat naše menu, tak budeme potřebovat soubor **vesamenu.c32**, kterým nahradíme textové menu za grafické. Nakopírujeme ho tedy ze složky syslinux do tftpboot, kterou využíváme pro bootování.

   ```
   cp /ust/lib/syslinux/vesamenu.c32 /var/lib/tftpboot
   ```

2. Když máme potřebný soubor vesamenu.c32 v bootovácí složce, tak můžeme přejít k úpravě tohoto menu. Tuto úpravu můžeme provést tak, že si vytvoříme nějaký **textový soubor**, kde provedeme jednotlivé úpravy. V souboru **default** zapíšeme cestu k tomuto souboru. Druhá možnost je, že provedeme úpravy rovnou v default souboru, kde také připojujeme jednotlivé obrazy se systémem. Já jsem úpravy provedl v default souboru, kde mám všechny konfigurace ohledně menu a jednotlivých systémech, které spouštím.

   ```
   gedit /var/lib/tftpboot/pxelinux.cfg/default
   ```

3. Nejdříve si přepíšeme řádek DEFAULT podle toho, jaké menu budeme pro spouštění používat tak, že v řádku default **nahradíme menu.c32 za vesamenu.c32**.

   ```
   DEFAULT vesamenu.c32
   TIMEOUT 600
   ONTIMEOUT localboot
   MENU TITLE PXE server Petr Antoncik
   prompt 0
   ```

4. První věcí, kterou můžeme nastavit je **pozadí plochy** při bootovaní. Můžeme vybrat jakýkoli obrázek ve formátu **jpeg nebo npg**, ale musíme si pamatovat, že při bootování se nám zobrazí plocha v základním rozlišení **640x480** pixelů. Při nastavování musíme dbát na to, aby nám rozlišení nepřesahovalo tuto hodnotu, jinak se nám místo obrázku načte černé pozadí. Obrázek musíme opět uložit do složky **/vat/lib/tftpboot**. Můžeme jej dát do nějaké složky, kterou umístíme do tftpboot, ale pak nesmíme zapomenout zadat cestu v souboru default pro toto pozadí.

```
menu background abc.jpg
```

5. Když máme nastavené pozadí, tak můžeme nastavit **barvy jednotlivých textů a tabulek**. Při psaní musíme dodržet tento typ formátu **menu color (prvek) (vlastnosti prvku) (barva)**, kde jednotlivé prvky znamenají toto: screen - plocha obrazovky, title - vlastnosti nadpisu menu, border - okraje menu, tabmsg - vlastnosti oznamovací zprávy v menu, sel a unsel - vlastnosti vybrané a nevybrané položky v menu, timeout_msg - vlastnosti zprávy odpočítavající čas do konce výběru položky, timeout - vlastnosti samotné časové veličiny a hotsel spolu s hotkey - vlastnosti rychlé klávesy pro vybrání položky z menu. Zpravidla se jedná o první písmeno položky, díky kterému můžeme rychle vybrat a spustit daný systém.

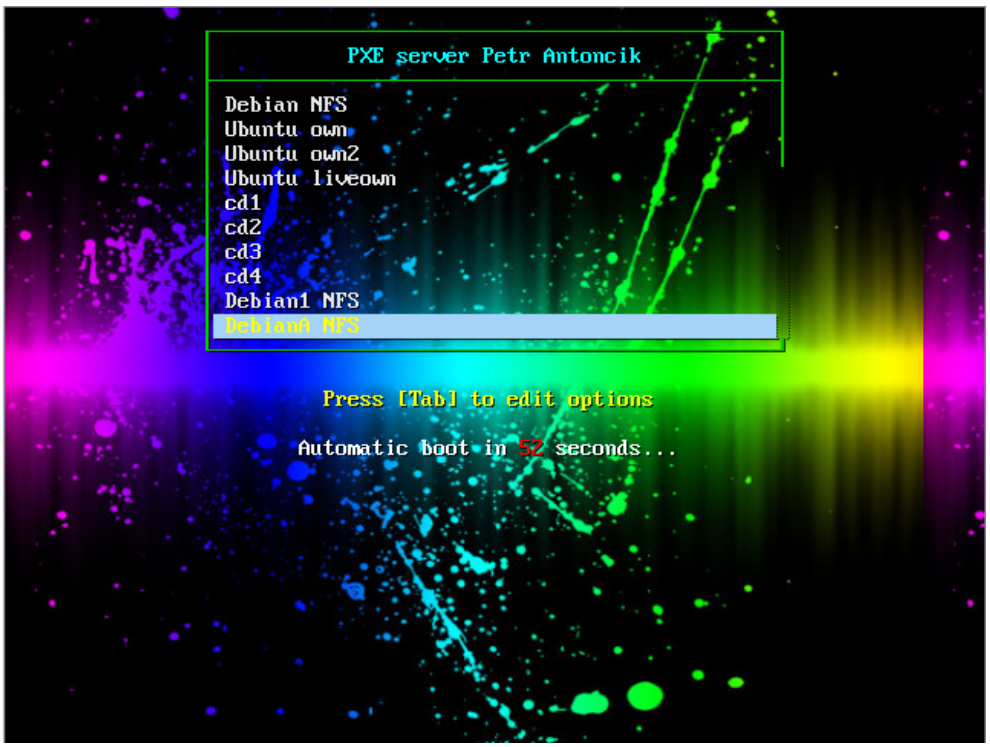
```
menu color screen 37;40 #80ffff #00000000 std
menu color title 1,31,40 #00ffff #00000000 std
menu color border 1;37;40 #00cc00 #00000000 std
menu color tabmsg 1;31;40 #ffff00 #00000000 std

menu color sel 7;37;40 #ffff00 #ffa3d3f8 all
menu color unsel 1;31;40 #e6e6e6 #00000000 std

menu color timeout_msg 37;40 #ffffff #00000000 std
menu color timeout 1;31;40 #ff0000 #00000000 std

#menu color hotsel 7;37;40 #000000 #ffa3d3f8 all
#menu color hotkey 1;31;40 #009933 #00000000 std
```

Zde je výpis jednotlivých znaků s vlastnostmi pro prvek:

```
0     reset vlastností do výchozích hodnot
1     nastavení tučného písma
4     podtržení
5     blikání
7     obrácené zobrazení
22    běžná ostrost
24    vyplé podtržení
25    vyplé blikání
27    vyplé obrácené zobrazení
30    černé popředí
31    červené popředí
32    zelené popředí
33    hnědé popředí
34    modré popředí
35    purpurové popředí
```

```
36    tyrkysové popředí
37    bílé popředí
38    zaplé podtržení, výchozí popředí
39    vyplé podtržení, výchozí popředí
40    černé pozadí
41    červené pozadí
42    zelené pozadí
43    hnědé pozadí
44    modré pozadí
45    purpurové pozadí
46    tyrkysové pozadí
47    bílé pozadí
49    výchozí pozadí
```

Také si můžeme všimnout, že barvy zadáváme v **osmi hexadecimálních místech**. To proto, že první dvě místa slouží pro sytost barvy a zbylá jsou pro RGB hodnoty. Hodnota **std nám udává standartní stínování, kdy jsou zvýrazněny pixely v popředí. Hodnota all značí, že jsou zvýrazněny pixely z popředí i z pozadí a hodnota rev značí zvýraznění pixelů, které se nacházejí na pozadí**. Na obrázku 3.1 můžeme vidět upravené menu.



Obrázek 3.1: Ukázka upraveného bootmenu.

3.7 Nastavení serveru pro připojení stanic k internetu

Jelikož jsem si při vytvoření sítě vybral achitekturu, kde jsou všichni klienti připojeni ke switchi, který je připojen k serveru a server samotný je připojen do vnější internetové sítě, tak potřebuji skrz tento server zpřístupnit internetové připojení k síti pro jednotlivé klienty. Protože klienti mají jen jednu síťovou kartu, tak provoz do vnější sítě bude probíhat skrz tento server. Proto budu muset nastavit server pro přeposílání paketů do vnější a vnitřní sítě, aby nám klienti přes server komunikovali a mohli se připojit k vnější síti. [17]

1. Aby nám server přeposílal pakety a klienti komunikovali s internetovou sítí, musíme vytvořit pravidla v **ip tables**, která nám tuto komunikaci zajistí. V následujících řádcích jsou sepsána pravidla, která musíme pro přeposílání aplikovat. Prvním příkazem nastavíme nové pravidlo, které nám povolí **přeposílání paketů** z první síťové karty, která je připojena ve vnější síti, do síťové karty připojené do vnitřní sítě.

   ```
   sudo iptables -A FORWARD -i wlan0 -o eth0 -s 192.168.1.1/24
   -m state --state NEW -j ACCEPT
   ```

2. Další pravidlo, které musíme přidat je, aby nám byly **pakety přeposílány i opačným směrem** z vnitřní do vnější sítě. Tento příkaz nám říká o přeposílání zpětných paketů u již vytvořeného spojení z vnější do vnitřní sítě.

   ```
   sudo iptables -A FORWARD -m state --state ESTABLISHED,RELATED
   -j ACCEPT
   ```

3. Když máme nastavené přeposílání oběma směry, tak nesmíme zapomenout, že se jedná o přeposílání mezi dvěmi různými sítěmi, a tak musíme přidat pravidlo, které nám zajistí, aby se nám překládali adresy u paketů z vnější sítě na adresy sítě vnitřní a naopak.

   ```
   sudo iptables -A POSTROUTING -t nat -j MASQUERADE
   ```

4. Posledním příkazem tato pravidla aktivujeme a zpřístupníme připojení klientů na internet.

   ```
   sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip\_forward"
   ```

5. Pro bezchybné fungování internetu nesmíme zapomenout nastavit u systému, který spouštíme, adresu **DNS serveru v souboru /etc/resolv.conf**, kde jsem zadával adresu školního DNS.

   ```
   sudo gedit /etc/resolv.conf
   nameserver 158.196.149.9
   ```

4 Vytvoření vlastního ISO CD

Jak jistě každý z nás ví, tak s každou nově příchozí verzí Linuxu vycházejí nové aplikace, které Linux v nové verzi obsahuje. Ať už se jedná o různé druhy programů, které usnadňují práci v tomto systému nebo různé aktualizace již používaných programů a aplikací, které zvyšují efektivitu svých operací. Toto vše nové verze obsahují. Ovšem existují i různé aplikace, které předinstalovány nejsou nebo nejsou oficiálně podporovány systémem, ale my víme, že daná aplikace na tomto systému funguje. Ať už se jedná o první nebo druhou věc, tak tyto programy, které nejsou s nově nainstalovaným systémem přidány, musíme po instalaci systému nainstalovat. To se může zdát velmi neefektivní v případě, že máme systém v ISO obraze spouštěný ze sítě. Po každém spuštění tohoto systému bychom museli dané programy opět nainstalovat a počítat s tím, že při příštím spuštění tam nebudou. Abychom tuto problematiku obešli, tak si můžeme vytvořit vlastní systémové CD, kde si můžeme předem rozmyslet, jaké programy budeme používat, jaké pro nás budou zbytečné a případně jak se má systém chovat. V následující části bakalařské práce se zaměřím na to, jak si můžeme vytvořit vlastní CD z již existující LIVE distribuce, na co si při tvorbě budeme muset dávat pozor a co nesmíme zapomenout. Uvidíme zde, jak bychom měli postupovat krok za krokem, až k vytvoření vlastního funkčního CD, které pak budeme moci použít jako bootovací a spouštět ze sítě. [15]

4.1 Manuálně

1. První věc, kterou musíme udělat, než začneme upravovat a vytvářet vlastí LIVE CD je, že si nainstalujeme potřebný software, abychom mohli s danými soubory pracovat, a poté je zabalit do výsledného ISO souboru, který nahrajeme na PXE server. Nejprve si nainstalujeme program s ovladači, který se nazývá **squashfs-tools**. Díky tomuto programu budeme moci číst, zasahovat a upravovat **squashfs soubory systému Linux**. Tyto soubory jsou vysoce komprimované, určeny pouze pro čtení a uzamknuté běžným uživatelům, aby se zamezilo jakýmkoliv změnám, které by mohly způsobit pády systému a jiné nežádoucí chování systému. Ten nainstalujeme pomocí apt-get příkazu.

   ```
   sudo apt-get install squashfs-tools
   ```

2. Druhým programem, který určitě využijeme, je aplikační **balíček mkisofs**, který je v nových distribucích linuxu nahrazen balíčkem genisoimage a bývá již předinstalován. Tento balíček nainstalujeme stejným způsobem jako squashfs-tools příkazem:

   ```
   sudo apt-get install mkisofs
   ```

   Aplikační balíček nám umožní zabalit naše upravené systémové soubory do jediného souboru, který bude ve formátu ISO 9660, a který pak nakopírujeme na PXE server nebo si jej vypálíme na CD.

3. Když máme potřebné programy, tak si vytvoříme složky, ve kterých budeme se soubory pracovat. Doporučil bych si vytvořit čtyři složky. Dvě, ke kterým si připojíme soubory z již existujícího CD, kde se v jedné budou nacházet běžná data systému a druhou pro připojení squashfs dat, která zatím nemůžeme přepisovat. Zbývající dvě složky budou sesterské složky datové a squashfs složky, do kterých si překopírujeme data z připojených složek, která pak budeme upravovat. Tyto složky si vytvoříme v adresáři **/home/petr/iso**. Zde je příklad složek:

```
mkdir cd-mnt
mkdir cd-data
mkdir sq-mnt
mkdir sq-data
```

4. Následně si připojíme všechny adresáře systému z ISO obrazu do složky **cd-mnt**, abychom viděli, co vše se v ISO obrazu nachází. Zase se nacházíme v účtu superuživatele, takže příkazy jsou napsány bez předpony sudo. Připojení provedeme příkazem:

```
mount ubuntu-12.04.2-desktop-i386.iso cd-mnt/ -o loop
-t iso9660
```

   Po provedení bychom měli v této složce spatřit všechny adresáře a soubory, které Linux Ubuntu obsahuje.

5. Teď překopírujeme všechny systémové soubory, kromě zmíněných komprimovaných, nepřepisovatelných squashfs souborů s názvem **filesystem.squashfs**, který se nachází v připojené složce v adresáři **/casper**. Překopírování bez komprimovaných souborů provedeme příkazem:

```
rsync --exclude=/casper/filesystem.squashfs -a cd-mnt/
cd-data
```

   Těchto souborů je poměrně málo, a tak by kopírování nemělo zabrat příliš mnoho času.

6. Když máme data překopírována, tak se pustíme do **práce se squashfs soubory**, které jsou chráněny proti přepsání. Ty obdobně jako datové připojíme do složky **sq-mnt**. Připojení pouze squashfs souboru z ISO obrazu do složky provedeme pomocí:

```
mount cd-mnt/casper/filesystem.squashfs sq-mnt -o loop
-t squashfs
```

   Po provedení bychom měli vidět jednotlivé otevřené squashfs soubory v náší sq-mnt složce.

7. Když máme připravenou složku se squashfs soubory, tak tyto soubory **rozzipujeme a překopírujeme do složky sq-data**, kde už je budeme moci upravovat. Překopírování uděláme standardním cp příkazem:

   ```
   cp -a sq-mnt/* sq-data/
   ```

8. Pro celou práci a tvorbu **LIVE CD budeme pracovat pouze s překopírovanými soubory ve složkách sq-data a cd-data**, proto odpojíme soubory z ISO obrazu ve složkách sq-mnt a cd-mnt a složky smažeme. Odpojení a smazání složky provedeme v jednom příkazu odděleným středníkem:

   ```
   umount sq-mnt/; rmdir sq-mnt
   umount cd-mnt/; rmdir cd-mnt
   ```

9. Předtím, než zasáhneme do systémových souborů, si vytvoříme přídavný soubor **lang**, který bude obsahovat odkaz na jazykovou sadu, kterou použijeme při mimo systémovém spuštění a úpravě. Přidání provedeme příkazem:

   ```
   echo "cs" > cd-data/isolinux/lang
   ```

10. Poslední věcí, kterou provedeme před úpravou systému je, že z našeho spuštěného systému překopírujeme konfigurační soubor **resolv.conf**, který obsahuje konfiguraci sítí, přes které se připojujeme na internet. Spolu s ním také překopírujeme soubor **hosts**, který obsahuje práva uživatelů využívající systém. Tyto soubory nám zajistí přístup k internetu a práva k manipulaci. Soubory získáme ze složky **/etc** našeho systému pomocí:

    ```
    cp /etc/resolv.conf sq-data/etc/
    cp /etc/hosts sq-data/etc/
    ```

11. Jakmile máme vše připravené, tak se **dostaneme do adresáře sq-data, který si spustíme jako podsystém**. Připojíme k němu systémové oddíly a nastavíme si důležité proměnné:

    ```
    sudo chroot sq-data
    mount -t proc none /proc
    mount -t sysfs none /sys
    export HOME=/root
    export LC_ALL=C
    ```

    Příkazy nám zajistí, že s adresářem budeme moci pracovat jako by jsme pracovali s právě spuštěným systémem. **Tímto způsobem můžeme stahovat a instalovat aplikace a měnit chování systému**.

12. Teď se nacházíme v systému, který upravujeme. Pro ulehčení vyhledávání programů a jejich instalací si můžeme nainstalovat program **aptitude**, který nám může usnadnit vyhledávání programů v textovém editoru. Před instalací je vhodné provést aktualizaci seznamu s programy, které si můžeme stáhnout. Aktualizaci a instalaci provedeme příkazy:

```
apt-get update
apt-get install aptitude
aptitude update
```

13. Zde je příklad pro instalaci jazykových sad do systému a k různým programům:

```
sudo apt-get install language-pack-cs language-pack-cs-base
(mozilla-firefox-locale-cs-cz) myspell-cs-cz (aspell-cs)
openoffice.org-help-cs openoffice.org-l10n-cs
openoffice.org-thesaurus-cs thunderbird-locale-cs
gimp-help-cs
sudo apt-get install build-essential
sudo apt-get install language-support-cs
```

    Upravovaný systém měl již předinstalované jazykové sady v závorkách.

14. Zde je zase příklad, kde si můžeme stáhnout tapetu na plochu. Tu stáhneme pomocí **wget a příslušného internetového odkazu do složky backgrounds**, a pak si ji nastavíme jako defaultní při spouštění systému. Obdobným způsobem můžeme stahovat a instalovat jakékoliv programy, které chceme v našem systému mít.

```
cd /usr/share/backgrounds
wget (adresa)
sudo gedit /usr/share/gnome-background-properties/
ubuntu-wallpapers.xml
```

15. Většina spouštěčů aplikací se nachází v adresáři **/usr/share/applications/**. Pokud chceme mít zástupce těchto aplikací na ploše, pak musíme provést tři příkazy:

```
ln -s /usr/share/applications/(aplikace).desktop
/home/petr/Plocha/
chmod 777  /home/petr/Plocha/(aplikace).desktop
chmod +x  /home/petr/Plocha/(aplikace).desktop
```

    První příkaz nám vytvoří odkaz na plochu. Tomuto odkazu pak povolíme všechna práva a nakonec přiřadíme právo spouštění jako aplikace.

16. Když skončíme všechny naše úpravy, tak smažeme naše soubory, pomocí kterých jsme byli připojeni k internetu, odpojíme data od systému a odejdeme z podsystému.

```
apt-get clean
rm -rf /tmp/*
rm /etc/resolv.conf  /etc/hosts
umount /proc
umount /sys
exit
```

17. S hotovými úpravami si **připravíme soubory squashfs a soubory pro instalaci** tohoto systému:

    ```
    chmod +w cd-data/casper/filesystem.manifest
    chroot sq-data dpkg-query -W --showformat='${Package}
    ${Version}\n' > cd-data/casper/filesystem.manifest
    cp cd-data/casper/filesystem.manifest cd-data/casper/
    filesystem.manifest-desktop
    sed -ie '/ubiquity/d' cd-data/casper/
    filesystem.manifest-desktop
    ```

    První příkaz **povolí zápis do systémového manifestu**, souboru s informacemi o systému ve složce cd-data. Dále pak **do tohoto manifestu zapíšeme informace z námi upraveného squashfs** souboru. Další příkaz vytvoří **kopii souborů na ploše** budoucího systému a poslední příkaz nám z tohoto systému **vytvoří instalační soubor**.

18. Z datové složky **vymažeme nepřepisovatelný filesystem.squashfs** soubor a **nahradíme ho naším upraveným** a vytvořeným ze složky sq-data.

    ```
    rm -f cd-data/casper/filesystem.squashfs
    mksquashfs sq-data/ cd-data/casper/filesystem.squashfs
    ```

19. Můžeme taky změnit jmenovku svazku. To provedeme pomocí:

    ```
    gedit cd-data/README.diskdefines
    ```

    Kde nastavíme vlastní název svazku.

20. Vytvoříme kontrolní součet souborů:

    ```
    cd cd-data
    sudo bash -c "find . \( -path './isolinux/isolinux.bin'
    -or -path './md5sum.txt' -or -path './cd-data/isolinux/
    boot.cat' -prune \) -or -type f -print0 | xargs -0
    md5sum > md5sum.txt"
    cd ..
    ```

21. A vytvoříme ISO obraz se systémem pomocí miksofs aplikace, který pak můžeme spouštět:

```
cd cd-data
sudo mkisofs -r -V "$IMAGE_NAME" -cache-inodes -J -l -b
isolinux/isolinux.bin -c isolinux/boot.cat -no-emul-boot
-boot-load-size 4 -boot-info-table -o
../ubuntu-7.04-desktop-i386-custom.iso
```

ISO obraz si opět vložíme do složky, kde máme obrazy systémů a stejným způsobem ho nastavíme pro bootování, jako by se jednalo o originální systém.

4.2 Pomocí programu UCK

Druhý způsob, jak můžeme vytvořit vlastní ISO obraz, je pomocí programu **Ubuntu customization kit**, který jsem objevil v distribucích linuxu. Tento program nás postupně vede krok po kroku a my pouze vybíráme z dostupných možností, které nám program nabídne. Díky tomu se nemusíme zabývat, které složky z poskytovaného ISO obrazu systému máme kopírovat a jaké musíme připojit, abychom mohli použít příkaz chroot a upravovat náš poskytovaný systém. To vše udělá program za nás. [16]

1. Nejprve si nainstalujeme balíček s programem Ubuntu customization kit. Opět se nacházíme pod rootem.

```
apt-get install uck
```

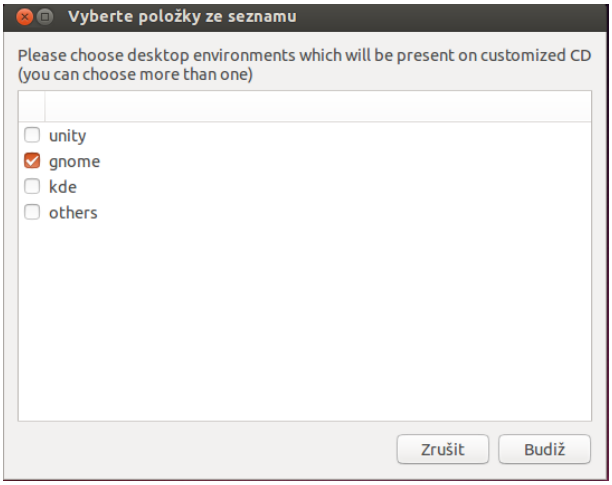
2. Když máme program nainstalovaný, tak jej spustíme. Spuštění programu provedeme pod **uživatelským účtem** nikoli pod rootem. Při spuštění nás uvítá úvodní obrazovka.

```
uck-gui
```

3. V dalším kroku se nás program zeptá, jaké jazykové balíčky se mají nainstalovat, jaké budou dostupné když ISO spustíme jako Live CD a jaký jazyk bude nastaven jako výchozí při spuštění.

4. Vybereme si grafické prostředí, zvolíme cestu k ISO obrazu, který chceme upravit, a nastavíme název novému obrazu. Výběr grafického prostředí znázorňuje obrázek 4.1.

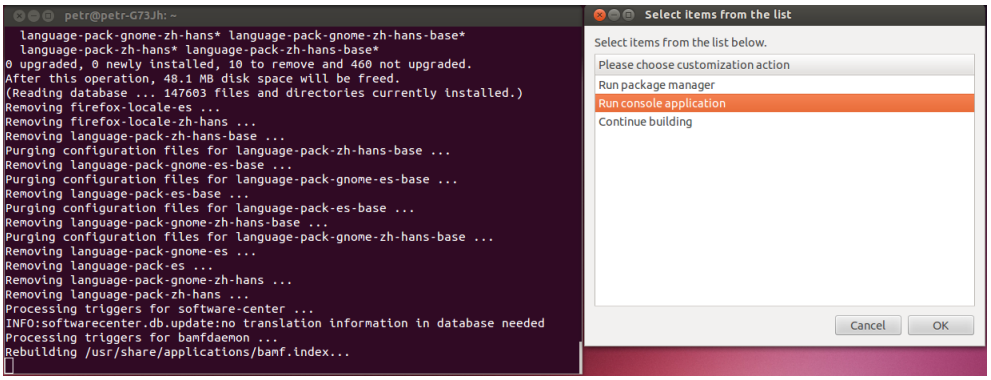
5. Program se nás zeptá, zda budeme chtít upravit obraz manuálně. Jelikož jej chceme upravit, tak vybereme možnost yes. Také se zeptá, jestli se mají zachovat soubory pro automatické spouštění obrazu ve windows, a jestli má být obraz hybridní, tedy vhodný ke spouštění z USB zařízení.



Obrázek 4.1: Výběr grafického prostředí.

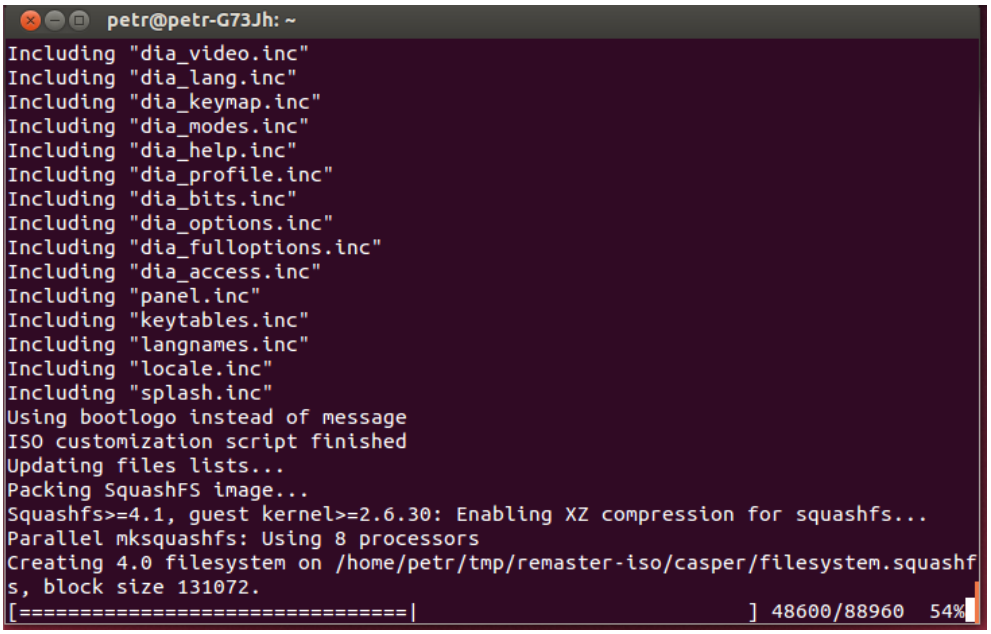
6. Před spuštěním samotné úpravy program oznámí, že byly vyplněny potřebné informace a ještě si vyžádá heslo než začne připravovat soubory pro editaci.

7. Když máme vše připravené, vybereme možnost run console aplication a program nám spustí upravovaný systém pod chrootem. My si systém upravíme podle našich představ a doinstalujeme aplikace, které chceme využívat hned v základu. Výběr můžeme vidět na obrázku 4.2.



Obrázek 4.2: Výběr editace systému.

8. Jsme-li s úpravami hotovi, odejdeme z chroota příkazem exit a vybereme možnost continue building. Program nám sestaví nový obraz v domovském adresáři ve složce tmp. Sestavení je ukázano na obrázku 4.3.



Obrázek 4.3: Sestavování systému.

5 Vytvoření plnohodnotného bootovatelného systému

Můžeme si všimnout, že bootování systému z ISO obrazu má jednu nevýhodu a to tu, že všechny změny provedené v tomto systému nám po restartu zanikají. To je způsobeno tím, že je tento systém ze sítě načten pouze do paměti RAM počítače, ve které k těmto změnám dochází, ale samotný obraz systému zůstává neměnný. Pokud bychom potřebovali nainstalovat nový program v tomto systému, tak musíme vytvořit zcela nový námi upravený obraz systému. Také bychom měli pamatovat na to, že pro uchování změn v našich dokumentech a jiných potřebných souborech potřebujeme přístup k některému NFS serveru v síti, kde tyto soubory uložíme. Takže je vhodné náš systém nastavit tak, aby nám automaticky připojil naší složku se soubory uloženými na NFS při spuštění klientské stanice. Naštěstí, kromě bootování systému z ISO obrazu, máme možnost vytvořit si takzvaný plnohodnotný systém, který nám zachovává všechny změny, nejen pouze změny v dokumentech, jako tomu bylo u NFS serveru, kde jsme naše soubory ukládali při použití systému spouštěného z našeho ISO obrazu, ale také změny v samotném systému. Díky tomu se námi spuštěný systém chová zcela stejně, jako by byl nainstalován na našem lokálním disku, i když všechny změny jsou provedeny na vzdáleném PXE serveru. [10, 11]

5.1 Příprava systému

1. Abychom si mohli vytvořit systém, který by byl podle našich představ, tak budeme nejprve potřebovat nainstalovat program, který nám pomůže s instalací základního systému do určitého adresáře, který využijeme pro bootovaní systému v síti. Tento program se jmenuje **Debootstrap** a je založen na deb balíčcích. Nejprve si tento program nainstalujeme.

   ```
   apt-get install debootstrap
   ```

2. Nyní budeme potřebovat systém, který se rozhodneme používat pro bootování ze sítě. Jako příklad jsem použil systém **debian ve verzi 6**, jelikož je založen na deb balíčku, který program debootstrap dokáže použít pro instalaci. Pro přehlednost si systém stáhnu a nainstaluji do složky **/mnt** na svém linuxovém serveru. Nejprve se přesunu do složky **mnt, a poté spustím instalaci debianu squeeze** (základní ořezané verze) pomocí debootstrapu. Opět všechny příkazy provádím pod root uživatelem.

   ```
   cd /mnt
   debootstrap --arch=i386 squeeze debian
   ```

3. Všimněme si, že se nám ve složce **mnt/** objevil adresář debian, který obsahuje základní systémové soubory nainstalované linuxové distribuce debian. Abychom si mohli tento systém upravit, musíme do jeho složek **/proc, /sys a /dev připojit naše adresáře z právě běžícího systému**. Jedná se o podobnou věc, kterou jsme dělali při manuální úpravě ISO obrazu. Připojení provedeme standartním příkazem mount.

```
mount -o bind /dev debian/dev/
mount -t proc none debian/proc
mount -t sysfs none debian/sys
```

4. Když máme potřebné adresáře připojeny, začneme se samotnou úpravou našeho nainstalovaného systému v adresáři debian tak, že se do něj dostaneme a spustíme si jej jako podsystém pomocí **chtoot** příkazu.

   ```
   chroot debian
   ```

5. Abychom mohli náš systém používat, budeme potřebovat nastavit jednotlivé účty pro uživatele, kteří budou se systémem pracovat. Účty přidáme příkazem **adduser a název uživatele**. V praktické části jsem pro demonstraci vytvořil účet uživatel s heslem abcd.

   ```
   adduser uzivatel
   ```

   Systém bude po nás chtít zadat 2x heslo k danému účtu, doplnit případné informace o účtu a potvrdit, zda námi zapsané informace souhlasí.

6. Nesmíme zapomenout **nastavit heslo pro root uživatele**. Bez nastavení tohoto hesla se nedostaneme do roota a nebudeme moci provádět změny v systému. Změnu root hesla provedeme pomocí příkazu **passwd**.

   ```
   passwd
   ```

7. Teď si upravíme tabulku přípojných bodů v souboru **/etc/fstab**. Pro tento účel využijeme textový editor **nano**.

   ```
   nano /etc/fstab
   ```

   Uvnitř souboru nastavíme tyto hodnoty:

```
proc            /proc           proc    defaults        0       0
/dev/nfs        /               nfs     _netdev,rsize=32768,
wsize=32768,hard,async,noatime,udp,nolock,
nfsvers=4       1       1
none            /tmp            tmpfs   defaults        0       0
none            /var/run        tmpfs   defaults        0       0
none            /var/lock       tmpfs   defaults        0       0
none            /var/tmp        tmpfs   defaults        0       0
```

Zde nám **netdev** říká, že připojení musí být ve funkční síti, **rsize a wsize** udává velikost datových rámců při komunikaci se serverem, **hard** udává, že nebudou hlášeny chyby při výpadku se serverem, **async** říká, že se bude jednat o asynchronní zápis do souborového systému, **noatime** zase znamená, že nebude měněn čas přístupu u souborů, **udp** značí, že bude použit UDP protokol, **nolock** znamená, že nebudou použity NFS zámky a **nfsvers=4**, který nám oznámí, že se bude používat NFS verze 4.

8. V dalším kroku nainstalujeme balíček s **jádrem systému**.

   ```
   apt-get install linux-image-2.6.32-5-686
   ```

9. Když máme jádro nainstalované, otevřeme **/etc/initramfs-tools/initramfs.conf** soubor, ve kterém upravíme položku **modules a definujeme u ní boot přes síť, a také položku boot, kde definujeme, že systém bude spouštěn z NFS serveru**. Znovu použijeme editor nano, který budeme mít k dispozici.

   ```
   nano /etc/initramfs-tools/initramfs.conf
   ```

   Nastavíme toto:

   ```
   MODULES=netboot
   BOOT=nfs
   ```

10. Ještě potřebujeme **aktualizovat nové initrd pomocí příkazu update**. Při aktualizaci se může stát, že nám aktualizace selže, protože systém neumí přečíst tabulku připojených souborových systémů, jelikož je prázdná. V takovém případě vytvoříme novou tabulku mtab z našeho adresáře **/proc/mounts**, a pak provedeme aktualizaci znovu. [20]

    ```
    grep -v rootfs /proc/mounts > /etc/mtab
    update-initramfs -u
    ```

11. Aby byl schopen obyčejný uživatel využívat systém, tak si v posledním kroku nainstalujeme **grafické rozhraní** [12] a programy, které chceme mít po spuštění systému k dispozici.

    ```
    apt-get install xorg gnome-terminal gnome-core gdm3
    ```

5.2 Dodatečná úprava NFS exports a syslinux default souboru

Když máme celý systém připraven, tak musíme, stejně jako u systému spouštěného z ISO obrazu, NFS deamonovi povolit adresář s nainstalovaným systémem. TFTP deamonovi musíme poskytnout jádro tohoto systému, initrd soubor a přidat položku v samotném bootovacím menu.

1. Stejně jako u ISO obrazu musíme nejdříve povolit adresář se systémem pro NFS.

   ```
   gedit /etc/exports

   /mnt/debian 192.168.1.1/255.255.255.0(rw,no_root_squash,
   async)
   ```

   Aktualizaci provedeme pomocí:

   ```
   exportfs -a
   ```

2. Překopírujeme **jádro systému a initrd** do složky od TFTP deamona, která je defaultně nastavena na **/var/lib/tftpboot**. Jedná se o složku, kde máme také syslinux soubory.

   ```
   cp debian/boot/vmlinuz-2.6.32-5-686 /var/lib/tftpboot/debian
   cp debian/boot/initrd.img-2.6.32-5-686 /var/lib/tftpboot/
   debian-initrd
   ```

3. Ještě doplníme systém do bootovacího menu úpravou souboru **/var/lib/tftpboot/-pxelinux.cfg/default**.

   ```
   gedit /var/lib/tftpboot/pxelinux.cfg/default

   LABEL Debian
       MENU label Debian
       KERNEL debian
       APPEND root=/dev/nfs initrd=debian-initrd
   nfsroot=192.168.1.1:/mnt/debian ip=dhcp rw selinux=0
   ```

6 Testování

V poslední části své bakalářské práce jsem se zaměřil na samotné otestování vytvořené sítě a systému spouštěného přes tuto síť. Jednalo se o 1 Gbit síť skládající se z jednoho serveru, kde byl nastaven DHCP daemon, NFS daemon, Syslinux a uložen systém, který byl bootován v klientských stanicích. Tento server byl připojen jednou síťovou kartou do vnější sítě (internetu) a druhou síťovou kartou do vnitřní sítě ke klientským stanicím. Druhá karta sloužila také pro nastavení IP adres pro klienty skrz DHCP a pro spuštění systému do vnitřní sítě. Za tímto serverem se nacházel switch, na který byly připojeny všechny klientské stanice. Testování jsem provedl ve dvou fázích. V první fázi jsem se rozhodl otestovat samotný systém pomocí běžných testů, které se využívají pro zjištění výkonu počítače. Tyto testy jsem provedl nejdříve na počítači, na který jsem nainstaloval systém na lokální disk. Výsledky testů jsem si zaznamenal, a pak jsem tytéž testy provedl na stejné hardwarové konfiguraci počítače, ale systém jsem nechal spustit ze serveru. V druhé fázi jsem se zaměřil na průběžné vytížení serveru při spuštění různého množství stanic. Tyto výsledky vytížení serveru jsem si zaznamenal spolu s jednotlivými časy spouštění.

6.1 Porovnání výkonu systému lokálního se systémem spouštěným ze sítě

V rámci testování systému lokálního a spouštěného ze sítě jsem si zvolil testovací nástroj Phoronix. Jedná se o utilitu určenou především pro linuxové distribuce systému, která v sobě obsahuje množství různých testů, kterými si můžeme otestovat náš systém na nějaké hardwarové konfiguraci. Z této utility jsem si vybral pár běžně používaných testů, abych porovnal výkon mezi lokálně nainstalovaným systémem a systémem spouštěným ze sítě. [7]

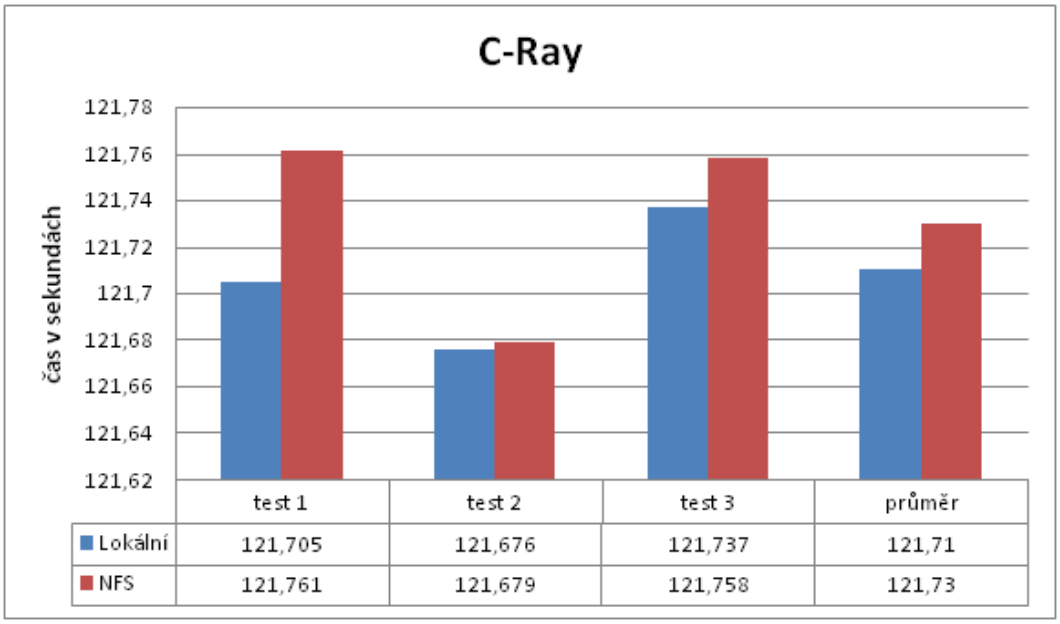
6.1.1 Hardwarová konfigurace PC

Procesor: Intel Core i7 920 @ 2.66GHz (8 Jader), **Základní deska**: Intel DX58SO, **Chipset**: Intel 5520/5500/X58 + ICH10R, **Paměť**: 3,5GB a 4GB 1067MHz, **Disk**: 750GB Western Digital WD7500AADS-0, **GPU**: NVIDIA Quadro FX 1800, **Zvuková karta**: Realtek ALC889, **Monitor**: AL2223W, **Síťová karta**: Intel 82567LM-2 Gigabit Connection OS:, **Systém**: lokální - Debian 6.0.7, přes nfs - Debian 6.0.9, **Kernel**: 2.6.32-5-686 (i686), **Prostředí**: GNOME 2.30.2, **Display Server**: X Server 1.7.7, **Display Driver**: nouveau 0.0.15, **OpenGL**: 2.1 Mesa 7.7.1, **Compiler**: GCC 4.4.5, **Rozlišení**: 1680x1050.

6.1.2 C-Ray test

Tento test nám slouží k ověření si výkonu, jak rychle nám dokáže počítač pracovat s plovoucí desetinnou čárkou. Jedná se o multivláknový test, který dokáže při výpočtech využít až 16 vláken pro jádro a 8 paprsků pro pixel při antialiasingu s vygenerováním obrazu v rozlišení 1600 na 1200 pixelů. Hlavní komponentu, kterou test zatěžuje, je pro-

cesor. Výsledkem je časová hodnota, za kterou nám počítač spočítá testovací výpočty. Nižší hodnota znamená rychlejší výpočet, a tedy vyšší výkon sestavy.



	test 1	test 2	test 3	průměr
Lokální	121,705	121,676	121,737	121,71
NFS	121,761	121,679	121,758	121,73

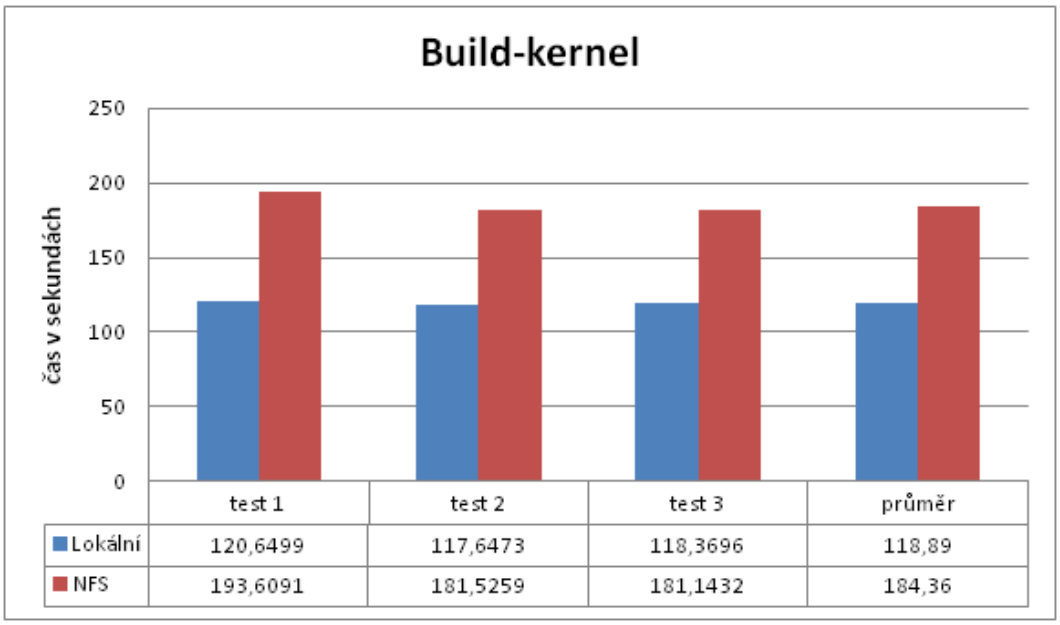
Obrázek 6.1: Test s plovoucí desetinnou čárkou.

Z obrázku 6.1 můžeme vidět, že oba systémy dosahovaly podobných výsledků. Ačkoli ve všech provedených testech dosahoval lokální systém lepších výsledů, tak tato převaha činila pouze 0,006 až 0,056 sekund. Můžeme konstatovat, že se výkon příliš nelišil při výpočtech na lokálním systému vůči výpočtům na systému spuštěném přes síť. Na obou systémech trval čas něco málo přes 121,7 sekund. Z toho se dá usuzovat, že v rámci výpočtů s plovoucí desetinnou čárkou, systém spouštěný ze sítě a nabootovaný do RAM paměti nijak nezpomaloval výpočty procesoru.

6.1.3 Build-kernel test

Tímto testem jsem se rozhodl ověřit výkon počítače při sestavování kernelového jádra systému. Jde o test, který nám změří časy potřebné k sestavení a kompilaci kernelu, a tyto časy nám vypíše k porovnání. Kratší čas opět znamená vyšší výkon.

Z obrázku 6.2 vidíme, že při sestavování kernel jádra a jeho kompilaci, je systém lokální výrazně efektivnější než systém spouštěný ze sítě. Je to dáno především prodlevou v síti, kdy systém spouštěný ze sítě musí přistupovat k jádru systému na vzdáleném síťovém disku, a částečně jiným výkonem disku. Rozdíl v sestavení a kompilaci činil kolem 66 sekund, což je až minutová prodleva ve výkonu.



	test 1	test 2	test 3	průměr
Lokální	120,6499	117,6473	118,3696	118,89
NFS	193,6091	181,5259	181,1432	184,36

Obrázek 6.2: Test rychlosti sestavení jádra.

6.1.4 Diskstress test

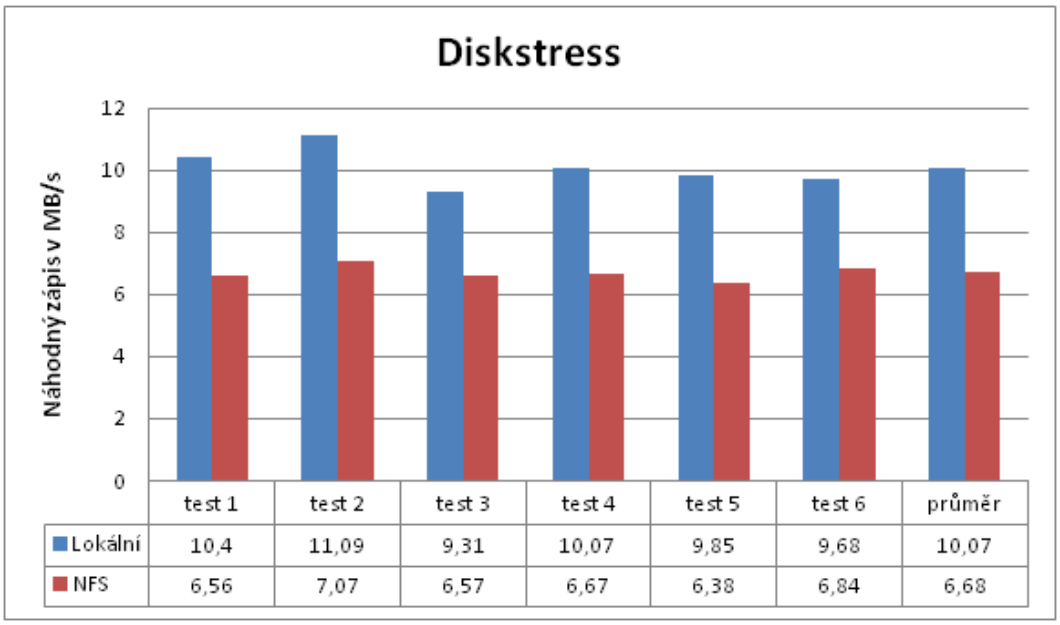
Následujícím testem jsem otestoval výkon disku. Tento test nám vytíží harddisk na samotné hranici a vypíše nám maximální rychlost náhodného zápisu, kterou dokáže použít při zaznamenávání dat. Vyšší hodnota zápisu znamená vyšší výkon.

Z obrázku 6.3 je patrné, že rychlost náhodného zápisu pro systém lokální je výrazně vyšší než pro systém spouštěný ze sítě. Je to způsobeno tím, že při testování nebyl testován harddisk, který se nacházel v počítači, i přestože testovací program tento místní harddisk vypsal. Testován byl harddisk, na kterém se nacházel systém, tedy harddisk serveru. I když se jednalo o prakticky totožné disky s podobnými parametry, tak rychlost náhodného zápisu na síťový disk byla výrazně snížena prodlevami v síti. Při přenášení dat byla rychlost o 3,4 MB/s nižší než u lokálního harddisku.

6.1.5 FSmark test

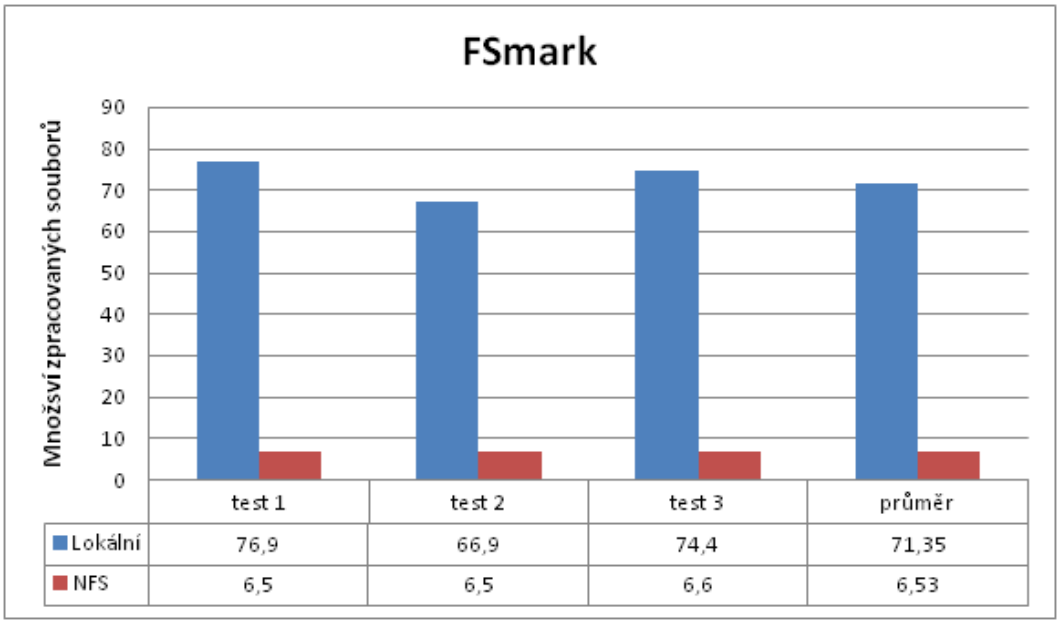
FSmark test neboli Filesystem mark je test, který slouží k ohodnocení rychlosti systému. Test nám zjistí, jak rychle nám dokáže systém pracovat se systémovými soubory. Při testování jsem si vybral možnost, kdy jsem měl 1000 souborů každý o velikosti 1MB. Výsledkem testu byla hodnota, která popisovala kolik souborů je schopen systém najednou zpracovat za daný časový interval. Test u lokálního systému proběhl dokonce 6x, avšak v grafu jsem nechal zobrazit pouze první 3 hodnoty spolu s 3 otestovanými hodnotami v síťovém systému. Vyšší počet zpracovaných souborů znamená vyšší výkon systému.

Na obrázku 6.4 vidíme rozdíl, který nám nastává při práci se systémovými soubory na lokálním a síťovém disku. Když byl systém nainstalován na lokálním disku, tak nám



	test 1	test 2	test 3	test 4	test 5	test 6	průměr
Lokální	10,4	11,09	9,31	10,07	9,85	9,68	10,07
NFS	6,56	7,07	6,57	6,67	6,38	6,84	6,68

Obrázek 6.3: Test rychlosti náhodného zápisu.



	test 1	test 2	test 3	průměr
Lokální	76,9	66,9	74,4	71,35
NFS	6,5	6,5	6,6	6,53

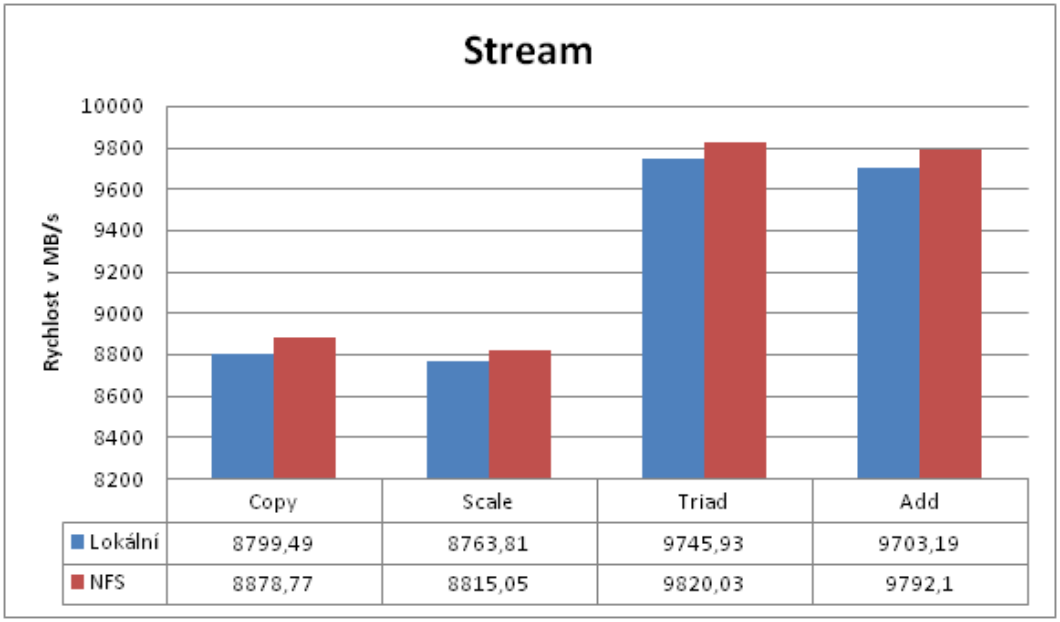
Obrázek 6.4: Test rychlosti procházení systémových souborů.

počítač dokázal projít a zpracovat 71,35 systémových souborů za daný časový interval v testu. Když byl počítač spuštěn ze sítě, tak přistupoval k souborům na vzdaleném disku a stačil zpracovat pouze 6,53 souborů, což značilo až skoro 11x nižší výkon vůči systému lokálnímu. Pozitivum pro síťový systém bylo, že zpracování souborů bylo konstantní a

nedocházelo k výrazým odchylkám jako u lokálního systému.

6.1.6 Stream test

Pomocí Stream testu jsem si otestoval rychlost pracování RAM paměti. Tento test provedl čtyři různé testování: test copy, scale, add a triad. První test byl zaměřen na kopírování souborů, druhý na setřídění, třetí na srovnávání hodnot a poslední pro přidávání souborů do paměti. Jelikož se jednalo o rychlost práce s daty v MB/s, tak vyšší rychlost znamenala vyšší výkon.



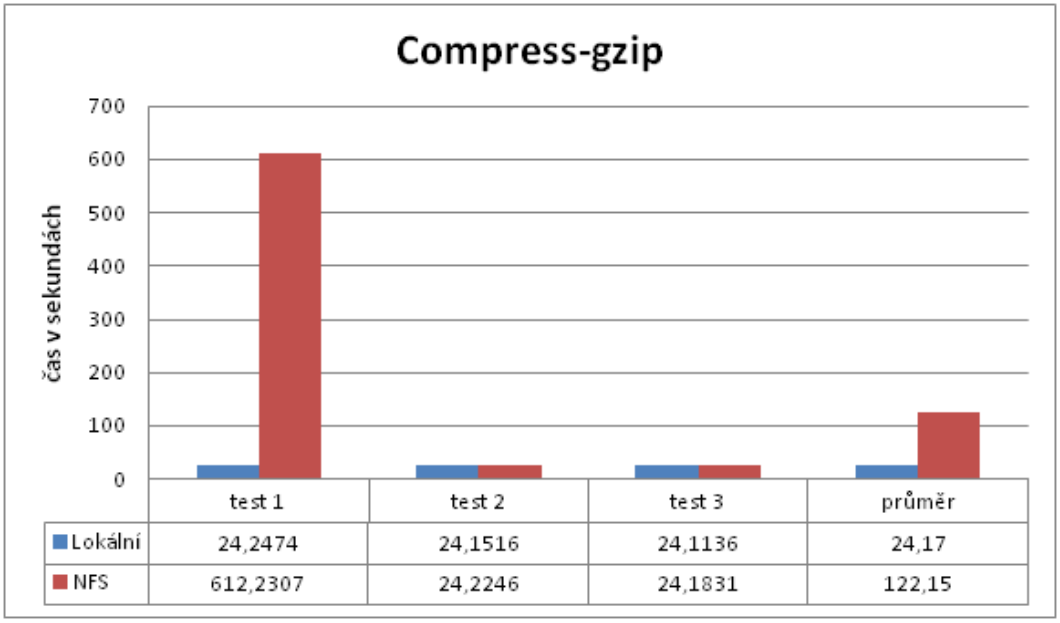
	Copy	Scale	Triad	Add
Lokální	8799,49	8763,81	9745,93	9703,19
NFS	8878,77	8815,05	9820,03	9792,1

Obrázek 6.5: Test rychlosti paměti.

Protože se mi na dřívějším počítači, na kterém jsem testoval lokální systém, poškodila síťová karta, tak jsem test musel provést na totožné hardwarové konfiguraci. Vyjímkou bylo, že druhá sestava obsahovala 4GB RAM paměti a předchozí pouze 3,5GB. Paměti však pracovaly na stejných frekvencích. Na obrázku 6.5 můžeme vidět, že hodnoty rychlostí u testu copy a scale se pro lokální systém pohybují pod hranicí 8800 MB/s. V případě systému síťového se tato hodnota naopak pohybuje mírně nad tuto hranici. U testu triad a add hodnoty lokálního systému přesahují hranici 9700 MB/s a u síťového je přesažena hodnota 9790 MB/s. V rámci porovnání systémů vidíme, že v síťovém systému byly hodnoty ve všech testech přibližně o 80 MB/s vyšší než u lokálního systému. Dle mého předpokladu je to způsobeno tím, že při testování systému ze sítě jsem měl k dispozici o 512MB paměti více, takže ve výsledku by testy nabývaly vyrovnaných hodnot bez těchto 512MB. Lze usuzovat, že na úrovni paměti a při práci s ní nedochází ke snížení výkonu tím, že je systém spouštěn ze vzdáleného úložiště.

6.1.7 Compress-gzip test

Posledním testem, který jsem se rozhodl provést, je test komprese dat. Tento test nám měří průměrnou dobu potřebnou pro kompresi daného množství dat a vypíše výsledný čas, po který komprese trvala. Kratší čas znamená vyšší výkon systému.



Obrázek 6.6: Test komprese dat.

Na obrázku 6.6 vidíme, že nejdéle tvala prvotní komprese u síťového systému. Pravděpodobně tato dlouhá doba, která trvala přes 10 minut, byla způsobena buď kvůli datům, která se musela stáhnout ze síťového disku, a nebo kvůli nějakému výpadku v síti. Když se podíváme na zbylé testy, tak nám hodnoty dosahovaly pouze něco málo přes 24 sekund a byly totožné s hodnotami naměřenými na lokálním systému. Proto ve výsledku průměrný čas potřebný na kompresi byl u síťového systému o 98 sekund delší než u systému lokálního.

6.2 Vytížení serveru při bootování různého počtu stanic

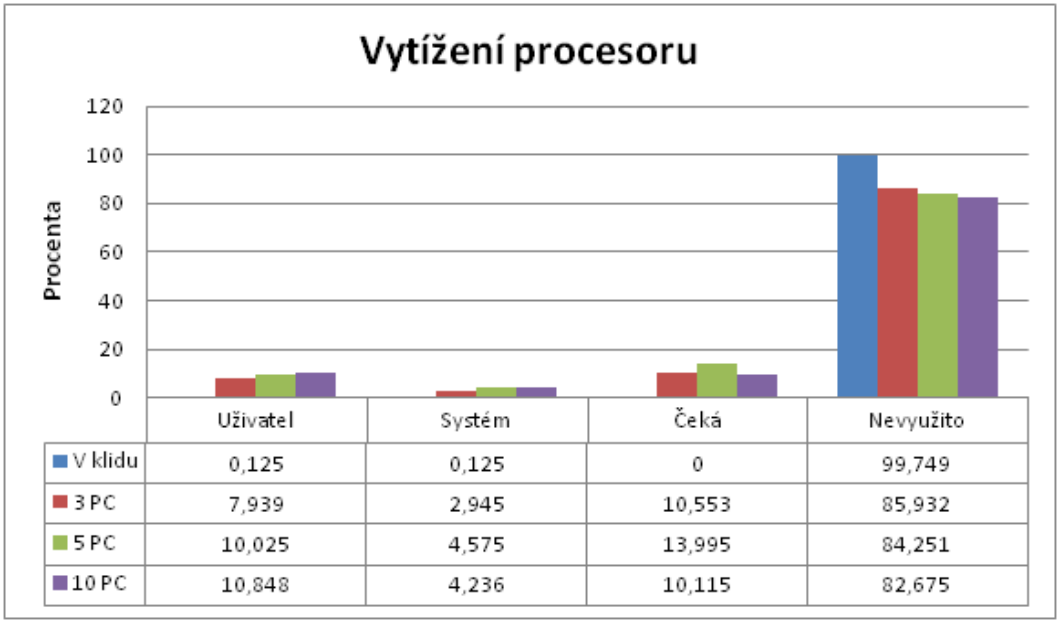
Druhým nástrojem, který jsem si vybral, je utilita Dstat. Ta slouží k průběžnému monitorování vytížení serveru nebo počítače. Zde jsem provedl testování při nabootování tří, pěti a deseti počítačů ze sítě a zaznamenával jsem vytížení serveru. Zaměřil jsem se na vytížení procesoru při bootování, využití paměti RAM a jednotlivé časy po které trvalo spuštění všech stanic. Tato naměřená data jsem zaznamenal a porovnal mezi sebou.

6.2.1 Hardwarová konfigurace serveru

Procesor: Intel Core i7 720QM @ 1.6GHz (4 Jádra) s Turboboost na 2.8GHz, **Paměť**: 8GB 1066MHz, **Disk**: 500GB Seagate ST9500420AS ATA 7200 ot/min, **GPU**: ATI Mobility Radeon 5870 1GB, **Síťová karta**: Atheros AR8131 PCI-E Gigabit Ethernet Controler, Atheros AR9285 **Systém**: Ubuntu 13.04.

6.2.2 Vytížení procesoru

V obrázku 6.7 uvidíme porovnání nejvyšších hodnot, kterých dosáhl prosecor serveru při spouštění různého počtu stanic ze sítě.



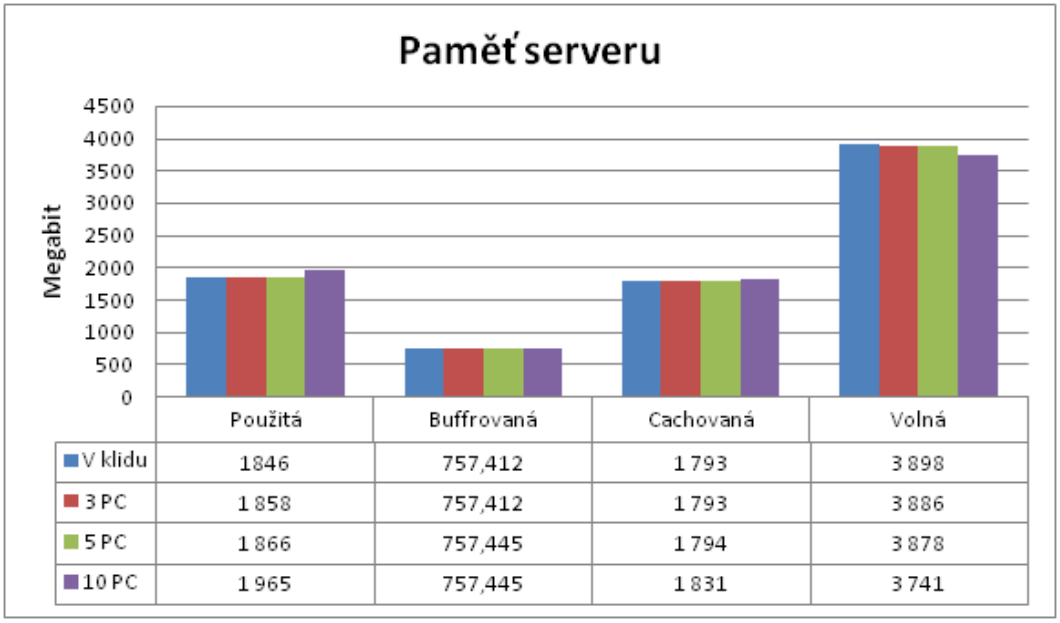
	Uživatel	Systém	Čeká	Nevyužito
V klidu	0,125	0,125	0	99,749
3 PC	7,939	2,945	10,553	85,932
5 PC	10,025	4,575	13,995	84,251
10 PC	10,848	4,236	10,115	82,675

Obrázek 6.7: Vytížení procesoru.

Z obrázku 6.7 vidíme, že při spouštění tří počítačů bylo dosaženo vytížení procesoru pro uživatelské úlohy maximálně 7,9%. Při spuštění pěti počítačů se nám vytížení zvýšilo maximálně na 10% a u deseti počítačů bylo vytížení 10,8%. V rámci systémových úloh se tato hodnota při spuštění tří počítačů zvedla na 2,9%, při pěti počítačích na 4,6% a u deseti počítačů na 4,2%. V rámci procesů čekajících na vyřízení se hodnota vytížení pohybovala kolem 10,5% u všech testů. Výjimkou bylo, že u testování pěti počítačů se mi podařila naměřit také hodnota 14%. Můžeme říct, že vytížení procesoru se v rámci jednotlivých úloh pohybovalo ve stejných hodnotách, které byly velmi nízké. Znatelného rozdílu si lze povšimnout až v rámci nevyužitého výkonu procesoru, který se nám snížil z 99,7% na 85,9% u tří spuštěných počítačů, na 84,3% u pěti počítačů a 82,7% u deseti počítačů. Takže celkové vytížení proceroru při deseti počítačích činilo pouze 18% z celkového výkonu. Výsledkem je, že procesor při spouštění počítačů ze serveru je vytížen jen minimálně.

6.2.3 Vytížení RAM paměti

Tímto testem jsem si zjistil, jak se mění množství dat v RAM paměti, a jak je tato paměť vytížená pro různý počet klientů.



	Použitá	Buffrovaná	Cachovaná	Volná
V klidu	1846	757,412	1 793	3 898
3 PC	1 858	757,412	1 793	3 886
5 PC	1 866	757,445	1 794	3 878
10 PC	1 965	757,445	1 831	3 741

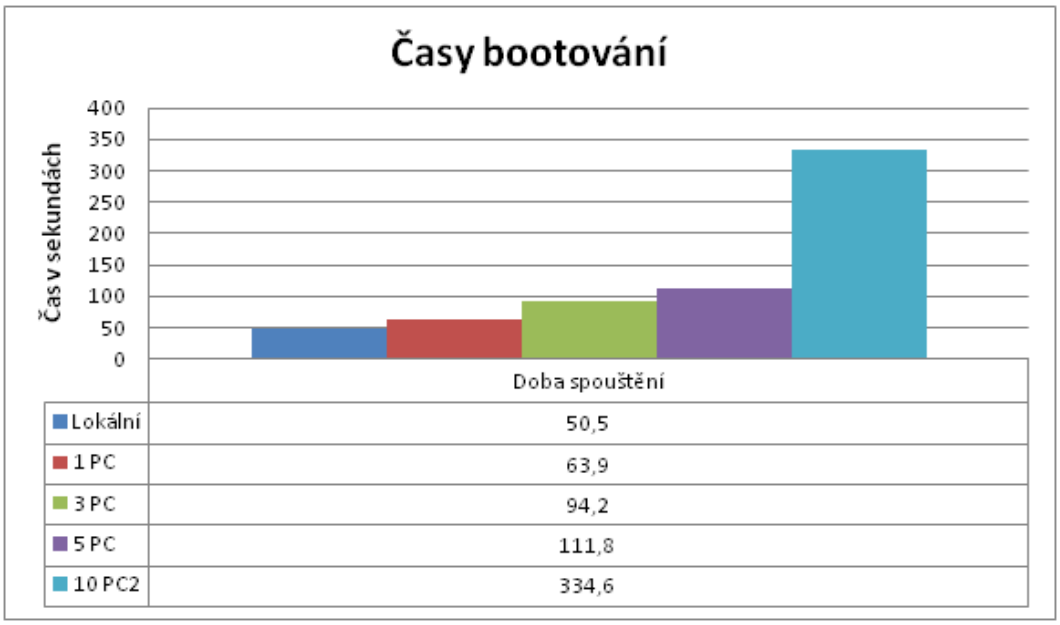
Obrázek 6.8: Vytížení paměti.

V rámci využití RAM paměti si můžeme povšimnout v obrázku 6.8, že běžné využití paměti spuštěného serveru bylo 1846 Mbit. Tato hodnota se nám pro tři počítače zvedla o 12 Mbit a o dalších 8 Mbit pro pět počítačů. K výraznému zvýšení však došlo až při spouštění deseti počítačů, kdy se vytížení zvedlo z původní hodnoty na hodnotu až o 119 Mbit vyšší. V rámci buffrované paměti se hodnota držela na 757,412 Mbit a při pěti počítačích a více se změnila pouze nepatrně na 757,445 Mbit. Pamět v cache byla vytížena na 1793 Mbit. Při pěti počítačích se zvýšila o 1 Mbit. Ke znatelnému rozdílu došlo při spuštění deseti počítačů, kdy tato hodnota vzrostla o 38 Mbit. V rámci rozdílu zbývající volné paměti se hodnota snížila o 157 Mbit oproti volné paměti u tří počítačů.

6.2.4 Délka spouštění různého počtu stanic

Posledním testem jsem si chtěl ověřit běžnou funkčnost systému v praxi, který by byl spouštěn ze sítě. Proto jsem se rozhodl změřit jednotlivé časy spouštění stanic ze serveru. Čas jsem měřil od spuštění prvního počítače po nabootování posledního počítače do uživatelského menu pro spuštění účtu uživatelem.

V obrázku 6.9 jsem znázornil jednotlivé časy, po které spouštění trvalo, a porovnal jsem je s časem naměřeným při spouštění systému na lokálním disku. Spouštění běžného počítače s tímto systémem trvalo 50,5 s. V rámci spouštění jednoho počítače ze sítě se tato hodnota zvedla o necelých 14 s. Tento čas se dá přičíst k času potřebnému pro přidělení



Obrázek 6.9: Časy spouštění.

IP adresy DHCP serverem a pro vyhledání serveru se systémem. Při spuštění tří počítačů se nám tento čas prodloužil o 30 s a s dalšíma dvěma o dalších 17,6 s. S pěti počítači se nám zvýšil čas tedy o minutu oproti lokálnímu času spouštění. Můžeme říct, že v rámci pěti počítačů byl tento čas velmi přijatelný. U deseti počítačů začalo ovšem docházet k výraznému prodloužení času potřebného ke spuštění. Tento čas činil 5 minut a 34,6 sekund.

7 Závěr

Při testování jsem si mohl ověřit funkčnost systému spuštěného ze sítě a otestovat jeho dopady na různé hardwarové komponenty. Zjistil jsem, že v rámci serveru je jeho procesor vytížen jen minimálně, a to i při spuštění deseti počítačů. Větší nároky jsou spíše kladeny na paměť RAM. Zcela jistě můžu říct, že největší nároky byly kladeny na harddisk serveru, který byl v nepřetržitém čtecím režimu, a také na síťové karty a síť takovou, kterou protékalo velké množství dat. Ověřil jsem si, že systém spouštěný ze sítě se v mnoha ohledech vyrovná systému lokálnímu, pokud jde o výkon, který zpracovává již spuštěný počítač. Z testů jsem zjistil, že ke snížení výkonu dochází teprve, až když systém potřebuje přistupovat k datům, která jsou uložena na síťovém disku. Tedy při úlohách, které ke zpracování potřebují operace čtení a zápis na disk. Aby nedocházelo ke snižování výkonu spuštěných počítačů, měly by být tyto počítače vybaveny dostatečně velkou RAM pamětí, aby se minimalizovala potřeba neustále číst a zapisovat data na síťovém disku. Také síťový disk by měl být velmi rychlý, aby stačil obstarávat velký počet počítačů. Poslední věc, kterou bych zmínil, by byla dobře sestavená bezkolizní síť nejlépe gigabitová. Sám jsem systém testoval na 1 Gbit síti, ve které se také při spouštění většího množství počítačů objevovaly podstatné časové odchylky. Při použití 100 Mbit sítě by tyto časové odchylky jistě vzrostly. U serveru bych pak doporučil, aby obsahoval alespoň dvě 1 Gbit karty pro síť, ve které by byly spouštěny počítače, aby se tak maximálně zefektivnily přístupy počítačů na server pro data.

V rámci porovnání systému předinstalovaného pomocí programu debootstrap a systému spouštěného z ISO obrazu musím říct, že předinstalovaný systém byl ve všech ohledech lepší než systém spouštěný z ISO obrazu. Ať už se jednalo o rychlost s jakou systém pracoval nebo o rychlost spouštění. Při použití ISO obrazu u deseti počítačů docházelo k zamrzávání počítačů během spouštění a pouze pár z těchto počítačů bylo schopno se spustit. Můžu říci, že systém spouštěný z ISO obrazu přes PXE je vhodný spíše pro síťovou instalaci nebo pro spouštění pouze pár počítačů. Při větším počtu klientů se stává tento systém neefektivní a pomalý. V takovém případě bych doporučil použít předinstalovaný systém ve složce vytvořený debootstrapem.

Při sepisovaní této práce jsem si rozšířil znalosti ohledně počítačových sítí a jednotlivých daemonů, jako je DHCP, díky kterému nyní vím, jak funguje automatická konfigurace sítě, jaké zprávy probíhají při komunikaci a jaký je výsledek této komunikace. Nahlédl jsem do NFS daemona, kde jsem se naučil nastavovat sdílení složek a souborů pro jednotlivé sítě a distribuci těchto souborů do této sítě pomocí FTP a TFTP protokolů. Kromě síťových prvků a zařízení jsem měl možnost nahlédnout hlouběji do linuxových distribucí systému a prozkoumat jejich strukturu, a jak tyto systémy fungují v praxi.

Měl jsem také možnost poznat nové testovací nástroje a na základě výstupů porovnat, jak se systémy chovají v různých podmínkách a porovnat jejich efektivitu pro běžného uživatele.

Literatura

NEMETH, Evi, Garth SNYDER a Trent R HEIN. *Linux: kompletní příručka administrátora : 2. aktulizované vydání.* Vyd. 1. Brno: Computer Press, 2008. ISBN 978-80-251-2410-9.

DROMS, Ralph a Ted LEMON. *The DHCP handbook. 2nd ed.* Indianapolis, Ind.: Sams, c2003, xxvii, 588 p. ISBN 06-723-2327-3.

STERN, Hal, Mike EISLER a Ricardo LABIAGA. *Managing NFS and NIS. 2nd ed.* Sebastopol, CA: O'Reilly, c2001, xviii, 490 p. ISBN 15-659-2510-6.

Syslinux [online]. 3. červen 2012 [cit. 2014-04-07]. Dostupné z: http://www.syslinux.org/wiki/index.php/Menu

UBUNTU *Wiki-ubuntu [online].* 10. srpen 2012 [cit. 2014-04-07]. Dostupné z: http://wiki.ubuntu.cz/syst%C3%A9m/datov%C3%A1_%C3%BAlo%C5%BEi%C5%A1t%C4%9B/adres%C3%A1%C5%99ov%C3%A1_struktura

Syslinux [online]. 10. leden 2014 [cit. 2014-04-07]. Dostupné z: http://www.syslinux.org/wiki/index.php/PXELINUX

Askubuntu [online]. 2. leden 2012 [cit. 2014-04-07]. Dostupné z: http://askubuntu.com/questions/92292/what-software-can-i-use-test-the-performance-of-my-system

LUBOŠ DOLEŽEL. *Abclinuxu [online].* 21. listopad 2012 [cit. 2014-04-07]. Dostupné z: http://www.abclinuxu.cz/clanky/bootovani-ze-site-pxelinux-a-korenovy-adresar-na-nfs

Cyberciti [online]. 15. listopad 2013 [cit. 2014-04-07]. Dostupné z: http://www.cyberciti.biz/faq/install-configure-tftp-server-ubuntu-debian-howto/

ADAM ŠTRAUCH. *Root [online].* 27. květen 2011 [cit. 2014-04-07]. Dostupné z: http://www.root.cz/clanky/bootujte-ze-site-plnohodnotny-system/

ROTTER MARTIN. *Wiki [online].* 21. březen 2011 [cit. 2014-04-07]. Dostupné z: http://wiki.inf.upol.cz/index.php/Instalujeme_distribuci_Debian_Squeeze

ANDREW. *Webupd8 [online].* 26. září 2013 [cit. 2014-04-07]. Dostupné z: http://www.webupd8.org/2013/09/how-to-install-gnome-310-in-ubuntu-1310.html

Forum-ubuntu [online]. 22. červen 2009 [cit. 2014-04-07]. Dostupné z: http://forum.ubuntu.cz/index.php?topic=36881.0

UBUNTU. *Wiki-ubuntu [online].* 26. červenec 2012 [cit. 2014-04-07]. Dostupné z: http://wiki.ubuntu.cz/nfs

TADEÁŠ PAŘÍK. *Wiki-ubuntu [online].* 27. prosinec 2013 [cit. 2014-04-07]. Dostupné z: http://wiki.ubuntu.cz/vytvo%C5%99en%C3%AD_vlastn%C3%ADho_livecd

CHRIS HOFFMAN. *Howtogeek [online].* 27. březen 2012 [cit. 2014-04-07]. Dostupné z: http://www.howtogeek.com/109736/how-to-create-a-custom-ubuntu-live-cd-or-usb/

TADEÁŠ PAŘÍK. *Wiki-ubuntu [online].* 6. prosinec 2013 [cit. 2014-04-07]. Dostupné z: http://wiki.ubuntu.cz/sd%C3%ADlen%C3%AD_internetov%C3%A9ho_p%C5%99ipojen%C3%AD_1

Fogproject [online]. 30. březen 2011 [cit. 2014-04-07]. Dostupné z: http://www.fogproject.org/wiki/index.php/Graphical_Menu_Configruation _Advanced

Syslinux [online]. 24. leden 2014 [cit. 2014-04-07]. Dostupné z: http://www.syslinux.org/wiki/index.php/Comboot/menu.c32

Insanelabs [online]. 20. duben 2009 [cit. 2014-04-07]. Dostupné z: http://insanelabs.com/linux/linux-df-cannot-read-table-of-mounted-file-systems/

8 Obsah CD

Součástí této bakalářské práce je CD s elektronickými přílohami. CD obsahuje elektronickou podobu práce v PDF, zdrojové config soubory použité při zkoušení systému, elektronickou literaturu a výsledky z provedených testů. Samozřejmostí jsou také PNG obrázky použité v práci a nasnímané v průběhu testování.

Adresářová struktura:

print - elektronická verze bakalářské práce určená pro tisk
- **src** - zdrojový kód bakalářské práce pro Latex

lit - elektronická literatura

conf - konfigurační soubory použité při sestavení a testování PXE

test - výsledky testů a měření
- **xml** - výsledky testů ve formátu *.xml
  - **local** - lokální systém
    - **pngloc** - screen obazovky testů z lokálně nainstalovaného systému ve formátu *.png
  - **pxe** - systém ze sítě
    - **pngpxe** - screen obazovky testů ze systému spuštěného ze sítě ve formátu *.png
- **csv** - soubory obsahující vytížení serveru ve formátu *.csv

A Elektronické přílohy

A.1 Zdrojové konfigurační soubory

Zdrojové konfigurační soubory, které byly použity pro sestavení PXE rozhraní. Soubory jsou uloženy na přiloženém CD v adresáři *\CD\conf*.

A.2 Zdrojové soubory testů z lokálního systému

Zdrojové soubory testů pořízené programem Phoronix z testování lokálně nainstalovaného systému ve formátu *.xml. Soubory jsou uloženy na přiloženém CD v adresáři *\CD\test\xml\local*. V adresáři *\CD\test\xml\local\pngloc* jsou pro zpřehlednění uloženy snímky obrazovky pořízené během testů ve formátu *.png.

A.3 Zdrojové soubory testů ze síťového systému

Zdrojové soubory testů pořízené programem Phoronix z testování systému spuštěného ze sítě přes PXE rozhraní ve formátu *.xml. Soubory jsou uloženy na přiloženém CD v adresáři *\CD\test\xml\pxe*. V adresáři *\CD\test\xml\pxe\pngpxe* jsou pro zpřehlednění uloženy snímky obrazovky pořízené během testů ve formátu *.png.

A.4 Zdrojové soubory z monitorování vytížení serveru

Zdrojové soubory z monitorování vytížení serveru pořízené programem Dstat ve formátu *.csv. Soubory jsou uloženy na přiloženém CD v adresáři *\CD\test\csv*.