

Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií



Diplomová práce

Webový server Apache – možnosti nastavení

Jan Šusta

© 2012 ČZU v Praze

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Katedra informačních technologií

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Šusta Jan

Informatika

Název práce

Webový server Apache - možnosti nastavení

Anglický název

Web server Apache - setting up

Cíle práce

Diplomová práce je zaměřena na webový server Apache. Práce má za cíl vysvětlit princip fungování webového serveru a jaké jsou možnosti jeho využití.

Metodika

Metodika je založena na studiu odborné literatury. Praktická část je zaměřena na návrh konfigurace serveru Apache a spuštění aplikace. Na základě studia literatury a výsledků praktické části budou zformulovány závěry.

Harmonogram zpracování

- 1) Studium literatury 7-8/2011
- 2) Vypracování přehledu daného problému 9/2011
- 3) Literární rešerše 10-12/2011
- 4) Vypracování praktické části 1-2/2012
- 5) Odevzdání 3/2012

Rozsah textové části

60 - 80 stran

Klíčová slova

Webový server, Linux, prvky Apache, moduly Apache, konfigurace Apache, konfigurační soubor

Doporučené zdroje informací

BLUM, Richard; BRESNAHAN, Christine. Linux Command Line and Shell Scripting Bible. Second Edition. Indianapolis : Wiley Publishing Inc., 2011. 812 s. ISBN 978-1-118-00442-5.

NEGUS, Christopher. Linux bible : boot up to Ubuntu, Fedora, KNOPPIX, Debian, SUSE, and 13 other distributions. Indianapolis : Wiley Publishing Inc., 2009. 851 s. ISBN 978-0-470-37367-5.

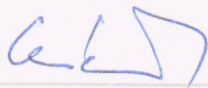
SOBELL, Mark G. Practical Guide to Linux Commands, Editors, and Shell Programming. Second Edition . [s.l.] : Prentice Hall, 2009. 1080 s. ISBN 978-0131367364.

Vedoucí práce

Vasilenko Alexandr, Ing.

Termín odevzdání

březen 2012



doc. Ing. Zdeněk Havlíček, CSc.

Vedoucí katedry



prof. Ing. Jan Hron, DrSc., dr.h.c.

Děkan fakulty

V Praze dne 21.11.2011

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Webový server Apache – možnosti nastavení" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 5. dubna 2012 _____

Poděkování

Rád bych touto cestou poděkoval Ing. Alexandru Vasilenkovi za odbornou pomoc a jeho cenné rady, které mi pomohly při zpracování této diplomové práce.

Webový server Apache – možnosti nastavení

Web server Apache – setting up

Souhrn

Předmětem této diplomové práce je charakterizovat webový server Apache a jeho možnosti nastavení pro poskytování statických HTML stránek i stránek generovaných dynamicky pomocí PHP. Pro ochránění obsahu na serveru jsou popsány možnosti nastavení omezeného přístupu k prostředkům pomocí přihlašovacího jména a hesla, které byly porovnány se souborem, který je pro autentizaci určen.

Webový server Apache byl nainstalován do dvou virtuálních strojů, ve kterých běží operační systém Linux Debian 5. V tomto prostředí proběhlo měření výkonu webového serveru Apache a byla provedena komparace výkonu pro jednotlivé virtualizační nástroje. Pro ukázkou možnosti jednoduššího nastavení byl nainstalován nástroj pro ovládání serveru pomocí grafického rozhraní. Tento nástroj umožňuje komplexní správu serveru.

V práci je uvedeno velké množství grafických a číselných údajů, které se vztahují k problematice měření výkonu, změn nastavení serveru a statistikám přístupů na webový server.

Klíčová slova:

Webový server, klient, konfigurační soubor, modul, direktiva, internetový prohlížeč, požadavek, protokol, virtualizace, operační systém, Linux, příkazová řádka.

Summary

The subject of this thesis is to characterize the Apache Web server and its configuration options for serving static HTML pages and pages generated dynamically using PHP. There is description of options using limited access to resources using login and password. They are compared to a file that is used for authentication.

Apache is installed on two virtual machines, which run on the Linux operating system Debian 5. In this operating system is measured the Apache web server performance and it is compared to individual performance in virtualization tools. To demonstrate the possibility to set configuration easily is installed control of the server using a graphical interface. This tool provides comprehensive server management.

In this thesis is shown a large number of graphical and numerical data, which applies to the issue of performance measurement, change server settings and statistics of visitors to your web server.

Keywords:

Web server, client, configuration file, module, directive, internet browser, request, protocol, virtualization, operating system, Linux, command line

Obsah

Obsah	8
1. Úvod.....	10
2. Cíl práce a metodika	11
2.1. Cíl práce	11
2.2. Metodika	11
3. Teoretická východiska	12
3.1. Protokol HTTP.....	12
3.1.1. Metody dotazování HTTP/1.1	12
3.1.2. Hlavičky protokolu HTTP	13
3.1.3. Testování protokolu HTTP	14
3.1.4. Stavové kódy protokolu HTTP	15
3.2. Architektura Apache	16
3.2.1. Jádro.....	16
3.2.2. Moduly Apache 2.0.....	16
3.3. PHP	18
3.4. Instalace serveru Apache.....	18
3.4.1. Kompilace Apache.....	18
3.4.2. Instalace binárních souborů Apache	22
3.4.3. Spuštění serveru Apache.....	23
3.5. Základní konfigurace Apache	24
3.5.1. Soubor httpd.conf.....	25
3.5.2. Platnost direktiv	26
3.5.3. Prostředí hlavního serveru	27
3.5.4. Obecné direktivy severu Apache	28
3.5.5. Kontejnerové direktivy	28
3.5.6. Soubor .htaccess.....	30
3.6. Zabezpečení serveru Apache.....	31
3.6.1. Základní nástroje pro zabezpečení serveru	31

3.6.2.	Omezení dle původu klienta	32
3.6.3.	Omezení dle identifikace uživatele	32
3.7.	Zjišťování výkonu serveru	33
3.8.	Virtualizace	34
3.8.1.	Virtualizační software	35
4.	Praktická část práce	37
4.1.	Instalace virtuálního stroje	37
4.2.	Spuštění webového sídla	41
4.3.	Výpis adresářů	42
4.4.	Omezení přístupu k prostředkům	44
4.5.	Grafický nástroj pro nastavení Apache	47
4.5.1.	Instalace nástroje Webmin	47
4.5.2.	Spuštění nástroje Webmin	47
4.6.	Měření výkonu webového serveru Apache	50
4.6.1.	Měření pro plný výkon serveru	51
4.6.2.	Měření pro snížený výkon serveru	59
4.7.	Sledování serveru Apache	62
4.7.1.	Program ApacheTop	64
5.	Zhodnocení výsledků a doporučení	66
6.	Závěr	70
7.	Seznam použitých zdrojů	73
7.1.	Seznam tabulek	73
7.2.	Seznam obrázků	74
7.3.	Seznam grafů	75
7.4.	Literatura	76
8.	Přílohy	78

1. Úvod

Webový server Apache je dlouhodobě nejrozšířenější server, který má jednoduchou konfiguraci a klade malé nároky na operační systém. Tento server je možné provozovat nejen na platformě Unix/Linux, ale i na systémech Microsoft Windows, Novell NetWare nebo Mac OS. V březnu 2012 byl celkový podíl serveru Apache 65 procent. Mezi hlavní konkurenty patří komerční server Microsoft IIS, který je nainstalován na 14 procentech serverů, a open-sourcový webový server Nginx, který běží na 10 procentech serverů. Tato situace ukazuje jasnou převahu nad konkurenty a dokazuje, že oproti komerčním serverům od konkurence, na funkčnosti nic neztrácí. Jediné, co by mohlo být serveru Apache vytýkáno je, že nepatří mezi nejrychlejší.

Hlavní výhodou Apache získává tím, že je distribuován zdarma. Distribuce je označena jako open-source software, který je dostupný s neomezenou licencí a možností zkoumat a upravovat zdrojový kód. Pokud se provozuje na některé z verzí operačního systému Linux, je možné získat velmi dobré řešení s minimálními náklady. Distribuce web serveru jsou určeny pro více typů operačních systémů od každé platformy. Při použití zdrojového kódu, který se přeloží v prostředí operačního systému, se Apache stává nezávislým na hardware. Tzn.: může být provozován na strojích s procesory Intel a AMD nebo na strojích s procesory od ostatních výrobců.

Mezi další pozitivní vlastnost web serveru Apache je možné zařadit, že je rozšiřitelný pomocí modulů, které zajišťují velké množství funkcí. Do poledních verzí serveru Apache byla implementována podpora IP protokolu IPv6. IP adresy v tomto formátu se budou v brzké době používat stále častěji, jako reakce na malý počet volných IP adres typu IPv4.

2. Cíl práce a metodika

2.1. Cíl práce

Hlavním cílem práce je charakterizovat webový server Apache. Nastavit tento server pro poskytování statických HTML stránek i stránek generovaných pomocí PHP. Součástí této části bude i nastavení serveru pro omezení přístupu uživatelů k prostředkům.

Dílčími cíli jsou nainstalování dvou virtuálních strojů, na kterých poběží webový server Apache ve stejné konfiguraci. Na těchto stojích bude změřen výkon, jaký poskytuje server. Provede se porovnání naměřených hodnot a určení, který virtuální stroj je vhodnější z hlediska menší zátěže hardware. Dále bude uvedena charakteristika nástrojů pro sledování webového serveru.

Dalším dílčím cílem práce je instalace a spuštění nástroje pro ovládání serverů pomocí grafického rozhraní, které bude pro uživatele přívětivější než příkazová řádka.

2.2. Metodika

Pro napsání této práce byla prostudována odborná literatura a internetové zdroje zabývající se problematikou. Následovalo rozhodnutí, jakou podobu by práce měla mít a co vše by v práci mělo být zahrnuto. Informace získané na základě literární rešerše byly analyzovány a následně, pomocí metod syntézy a deskripce, byl utvořen přehled o možnostech instalace, nastavení a následného použití webového serveru.

Webový server Apache byl nainstalován do virtuálních počítačů, na kterých běžel operační systém Linux Debian 5. Změny nastavení serveru byly prováděny použitím modulů a úpravou konfiguračního souboru `httpd.conf` pomocí direktiv.

Výkon webových serverů Apache byl měřen pomocí aplikace Apache Benchmarking pro statické HTML stránky a pro stránky generované pomocí PHP. Metoda komparace byla aplikována v rozhodovací fázi pro určení vhodného virtuálního stroje.

Pro některé uživatele je ovládání webového serveru pomocí příkazového řádku příliš složité, proto byl aplikován grafický nástroj pro ovládání serverů Webmin 1.5. Tento grafický nástroj umožňuje provádět veškeré nastavení serveru pomocí internetového prohlížeče a jeho ovládání je tak snazší.

3. Teoretická východiska

3.1. Protokol HTTP

Jedná se o bezstavový protokol, který funguje na principu dotaz a odpověď. Komunikace se serverem probíhá přes TCP, nad kterým je http postaven [24]. Je důležité, aby dotaz/odpověď měl specifikovanou metodu, URI – je dané absolutní nebo relativní cestou k souboru nebo pomocí URL dokumentu, verzi a hlavičky.

3.1.1. Metody dotazování HTTP/1.1

- Get

Toto je nejpoužívanější metoda, která slouží k načítání prostředku, který je identifikovaný z URL požadavku. Tzn.: používá se pro vyzvednutí html souboru nebo obrázku ze serveru. [19, s. 53]

- Head

Tato metoda je rovnocenná s GET, ale nevrací tělo zprávy. Používá se pro zjištění, jestli objekt na serveru existuje, nebo nikoliv. [19, s. 54]

- Post

Zasílá serveru informace, že má od klienta přijmout data. Nejčastější použití bývá přijetí informací zadaných do webových formulářů. [24]

- Put

Touto metodou se říká serveru, aby otevřel soubor, do kterého bude ukládat informace, které bude přijímat od klienta. [19, s. 55]

- Options

Provádí požadavek na informace o možnostech komunikace poskytované serverem. [19, s. 52]

- Trace

Používá se pro sledování cesty celého dotazu. [24] Většinou se jedná o testovací účely, kdy klient dostává odpovědi seřazené podle systémů, kterými požadavek procházel.

- Connect

Používá se s proxy a může být dynamicky přepnuta na tunelové propojení od klienta na server např. pomocí SSL.[19, s. 57]

3.1.2. Hlavičky protokolu HTTP

Protokol http definuje ve verzi 1.1 velké množství hlaviček jak pro požadavky, tak pro odpovědi. Zde budou uvedeny jen nejdůležitější z nich.

3.1.2.1. Dotazové hlavičky

Dotazové hlavičky se používají pro předávání informací na server. Umožňují vložit informace, které určují odpověď a specifikují přijatelnou odpověď.

Accept

Tyto hlavičky ukazují, jaký typ dat je schopen klient zpracovat. Na serveru je vybrána nejvhodnější varianta. Jsou zde obsaženy hlavičky Accept (MIME typ dokumentů), Accept-Charset – určuje znakovou sadu, Accept-Encoding a Accept-Language. [24]

Host

Definuje internetový název hostitele a číslo portu, přes který bude komunikovat. [1, s. 42] Na jedné IP adrese je možno provozovat více virtuálních serverů. Avšak pomocí hlavičky musí být definováno jméno každého serveru.

User-Agent

Touto hlavičkou se klientský program identifikuje, je možné např. posílat různé informace různým prohlížečům. [24]

3.1.2.2. Hlavičky odpovědí

Tyto hlavičky se používají na posílání dalších informací, které nejsou přímou odpovědí na dotaz klienta.

Content

Hlavičky, které popisují obsah odpovědi. Může v nich být obsažena např. délka odpovědi, jazyk, typ dokumentu atd. [24]

Server

Hlavička, která obsahuje identifikační údaje serveru, jako jsou jméno a verze. Apache v této hlavičce identifikuje sám sebe. [1, s. 43]

Expires

Touto hlavičkou je určena doba platnosti dokumentu. Pokud platnost dokumentu vyprší, musí si server stáhnout aktuální verzi dokumentu. [19, s. 127]

3.1.3. Testování protokolu HTTP

Pro testování hlaviček odpovědí byl použit program Telnet. Z výpisu je vidět, jaké hlavičky protokol HTTP posílá před odesláním požadovaného prostředku. Byl vznesen požadavek metodou GET na verzi protokolu HTTP/1.0, což je vidět na čtvrtém řádku obrázku 1.

```
$telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
GET / HTTP/1.0

HTTP/1.1 200 OK
Date: Thu, 08 Mar 2012 18:58:31 GMT
Server: Apache/2.2.9 (Debian)
Last-Modified: Tue, 28 Feb 2012 18:11:13 GMT
ETag: "416f7-33-4ba0a26f6f240"
Accept-Ranges: bytes
Content-Length: 51
Vary: Accept-Encoding
Connection: close
Content-Type: text/html

<html><body><h1>It works!Jupiii</h1></body></html>
```

Obrázek 1: Výpis programu telnet pro požadavek GET / HTTP/1.0.

Zdroj: Vlastní zpracování. 1.3.2012

Z obrázku 1 je také patrné, že server, na který byl požadavek odeslán, podporuje verzi HTTP/1.1. Proto vypsál kód 200, který znamená vše v pořádku a vypsál soubor *index.html*. Dále je ukázáno, jaká odpověď přijde na požadavek GET / HTTP/1.1. Oproti předchozímu požadavku je vidět značný rozdíl.

```
$telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
GET / HTTP/1.1

HTTP/1.1 400 Bad Request
Date: Thu, 08 Mar 2012 18:58:07 GMT
Server: Apache/2.2.9 (Debian)
Vary: Accept-Encoding
Content-Length: 300
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<hr>
<address>Apache/2.2.9 (Debian) Server at localhost Port 80</address>
</body></html>
```

Obrázek 2: Výpis programu telnet pro požadavek GET / HTTP/1.1

Zdroj: Vlastní zpracování. 1.3.2012

V první hlavičce je uvedena podporovaná verze protokolu HTTP a kód chyby 400. Tato hlavička znamená, že požadavek od klienta nebyl kompletní a selhal. To z důvodu, že protokol HTTP/1.1 ještě vyžaduje, aby byl zadán název hostitele, který se ve většině případů shoduje s názvem webového serveru.

3.1.4. Stavové kódy protokolu HTTP

Jako součást hlavičky odpovědi je také stavový kód, který udává, jak byla odpověď serverem zpracována. Klient stavový kód odpovědi interpretuje a podle toho udělá potřebné kroky. Součástí odpovědi je také stavové hlášení, což znázorňuje slovní popis stavového kódu [15]. Stavové kódy jsou dle charakteru rozděleny do pěti kategorií.

Informační	100-199
Požadavek byl úspěšně zpracován	200-299
Požadavek byl přesměrován	300-399
Požadavek byl nekompletní	400-499
Chyby serveru	500-599

3.2. Architektura Apache

Architektura webového serveru je tvořena několika základními vrstvami. První vrstvu představuje operační systém, kde může pracovat buď operační systém Linux a Unix, nebo nějaký operační systém od firmy Microsoft např. Windows Server 2003, MacOS a jiné.

Druhá vrstva je tvořena programem Apache a zajišťuje základní funkcionalitu. Skládá se z jádra, modulů a standardních knihoven. Jádro Apache spolu s modulem *http_core* implementuje základní funkcionalitu a zajišťuje propojení API (Application Programming Interface) se třetí vrstvou.

Třetí vrstva je tvořena moduly. Ty rozšiřují funkcionalitu o další možnosti. Pro základní funkci ovšem nejsou moduly vyžadovány. Moduly mohou být pevnou součástí nebo mohou být dynamicky přidány za běhu serveru. Moduly obsahují direktivy, které nastavují vlastnosti serveru.

Čtvrtá vrstva může být buď prázdná, nebo je možné sem umístit moduly dalších výrobců. Je sem možné umístit např. *mod_perl* nebo *mod_php*. [6, s. 5]

3.2.1. Jádro

Jádro má dvě základní funkce:

- poskytuje základní funkčnost serveru
- vytváří aplikační rozhraní (API) pro moduly

Když se řekne, že jádro zajišťuje základní funkcionalitu, znamená to, že zabezpečuje výměnu zdrojů pomocí souborových deskriptorů a segmentů paměti, podporuje model procesů, naslouchá na komunikačních portech protokolu TCP/IP a přenáší požadavky protokolu http. Další funkcí jádra je programové propojení modulů a zabezpečení výměny informací mezi nimi. [6, s. 6]

3.2.2. Moduly Apache 2.0

Apache je poskytován s modulem jádra a velkým množstvím dalších modulů. Výsledná funkce je určena tím, jaké moduly jsou v okamžiku spuštění Apache načteny. Moduly mohou být vestavěny do programu Apache nebo mohou být přidány jako externí soubory. Moduly jsou dostupné ve dvou typech souborů. Když jsou určeny pro překlad, tak mají

příponu `.c` a jsou součástí instalačního souboru. [4, s 148] Druhým typem souborů jsou dynamicky sdílené objekty DSO (Dynamic Shared Object), ty mají příponu `.so` a jsou umístěny v adresáři `/modulem` v kořenovém adresáři serveru. Zavádějí se při startu Apache.

Další možností je použít nástroj `apxs` (Apache Extension), který se nachází v adresáři `/bin` v kořenovém adresáři serveru. Tímto nástrojem je možné vytvořit další moduly DSO [1, s. 139].

Moduly jsou řízeny pomocí Apache a fungují na principu registrování funkcí, které poskytují. Funkce, které jsou zaregistrovány, může server spouštět. Požadavky jsou prováděny pomocí cyklu zpracování. Cyklus zpracování začíná přijetím požadavku na prostředek od klienta a končí odesláním odpovědi. Síťové spojení se většinou uzavírá na konci cyklu zpracování požadavku, často je však při jediném spojení vyřízeno více požadavků. Většina modulů Apache registruje funkce, které se provádí ve víc než 1 kroku zpracování. Cyklus zpracování se dělí na 4 základní fáze:

[1, s. 135]

- **Fáze analýzy požadavku**

V první fázi je přijat požadavek na prostředek od klienta, který posílá informace přes hlavičky požadavku protokolu HTTP. Požadavek je umístěn v URL, které je vyhodnoceno jako prostředek nebo akce. Ve většině případů se jedná o soubor, který je uložen na serveru. V této fázi se také rozhoduje, jestli nebude požadavek přeměrován.

- **Fáze zabezpečení**

Ve druhé fázi může být uživatel, který žádá o prostředek, indentifikován. Pomocí ověřovacího formuláře se od něj vyžádá přihlašovací jméno a heslo a použijí se prostředky pro ověření, jestli je uživatel autorizován pro obdržení požadavku.

- **Fáze přípravy**

Ve třetí fázi se určuje druh prostředku, který bude odeslán.

- **Fáze zpracování**

V této fázi je klientovi předán prostředek, o který žádal. Může se jednat o statický HTML soubor nebo výsledek skriptu.

3.3. PHP

Jedná se o hypertextový preprocesor, který interpretuje HTML stránky před odesláním na klientský počítač. PHP umožňuje vkládat skripty nebo celé programy do HTML stránek. Programy do stránek je možné také vkládat pomocí JavaScriptu. Oproti JavaScriptu je zásadní rozdíl v tom, že PHP se interpretuje na serveru. Jazyk PHP je interpretovaný, což znamená, že do té doby, dokud není použit, je uložen ve zdrojovém tvaru. Jazyky, které používají kompilátory, musí nejdříve kód přeložit, poté je teprve možné kód použít [2, s. 15]. Provádění kódu na serveru přináší výhody i nevýhody. Mezi výhody patří snadná interakce s aplikacemi na serveru a nenáročnost na hardware klientského počítače, jelikož výstupem ve většině případů bývá kód HTML, který zobrazí i starší prohlížeče. Nejsou kladeny velké nároky na přenos dat. Zdrojový kód se provede na serveru a na výstupu se již neobjeví. Za nevýhodu lze označit větší zátěž serveru.

Serverové technologie jsou založeny na principu, že když klient požaduje webovou stránku, server tuto stránku nejdříve sestaví a odešle na klienta. Podle toho, jaké požadavky klient vznesl, může být pokaždé sestavena jiná stránka. Při zpracování skriptu server vezme HTML kód tak jak je, provede kód PHP, zkombinuje výsledek a odešle prohlížeči. [22]

3.4. Instalace serveru Apache

Jsou tři základní možnosti, jak server Apache nainstalovat. První možností je stáhnout zdrojový kód a na vlastním počítači zkompileovat. Druhou možností je stáhnout binární soubory, které jsou již zkompileovány pro danou verzi operačního systému. Třetí možností je použít balíčkovací systém, který stáhne balíček s programem a automaticky nainstaluje.

3.4.1. Kompilace Apache

Hlavní výhodou kompilace Apache ze zdrojového kódu je, že můžeme přidávat vlastní úpravy kódu např. pomocí záplat. Záplaty často umožňují lidem, kteří nejsou zbyhlí v programování provádět sofistikované úpravy systému. Problém může spočívat v nalezení správné záplaty pro konkrétní operační systém.

Dalším důvodem, proč kompilovat ze zdrojového kódu je optimalizace kompilátoru pro hardwarovou platformu a operační systém na kterém by měl být webový server

spuštěn. Tato možnost není tak důležitá, protože nalezení binárních souborů pro daný systém není tak složité.

3.4.1.1. Stažení zdrojového kódu Apache

Zdrojový kód Apache je možné stáhnout na stránce <http://archive.apache.org/dist/httpd/>. Poslední verze je `httpd-2.4.1.tar.gz`, tento soubor je komprimován pomocí unixového formátu *tar*. [8, s. 193]

	httpd-2.3.8-deps.tar.bz2.asc	2010-08-31 13:00	187
	httpd-2.3.8-deps.tar.bz2.md5	2010-08-31 13:00	59
	httpd-2.3.8-deps.tar.bz2.sha1	2010-08-31 13:00	67
	httpd-2.3.8-deps.tar.gz	2010-08-31 13:00	1.6M
	httpd-2.3.8-deps.tar.gz.asc	2010-08-31 13:00	187
	httpd-2.3.8-deps.tar.gz.md5	2010-08-31 13:00	58
	httpd-2.3.8-deps.tar.gz.sha1	2010-08-31 13:00	66
	httpd-2.3.8.tar.bz2	2010-08-31 13:00	3.6M
	httpd-2.3.8.tar.bz2.asc	2010-08-31 13:00	187
	httpd-2.3.8.tar.bz2.md5	2010-08-31 13:00	54
	httpd-2.3.8.tar.bz2.sha1	2010-08-31 13:00	62
	httpd-2.3.8.tar.gz	2010-08-31 13:00	4.9M
	httpd-2.3.8.tar.gz.asc	2010-08-31 13:00	187
	httpd-2.3.8.tar.gz.md5	2010-08-31 13:00	53
	httpd-2.3.8.tar.gz.sha1	2010-08-31 13:00	61
	httpd-2.4.1.tar.bz2	2012-02-19 13:59	3.9M
	httpd-2.4.1.tar.bz2.asc	2012-02-19 13:59	825
	httpd-2.4.1.tar.bz2.md5	2012-02-19 13:59	54
	httpd-2.4.1.tar.bz2.sha1	2012-02-19 13:59	62
	httpd-2.4.1.tar.gz	2012-02-19 13:59	5.3M
	httpd-2.4.1.tar.gz.asc	2012-02-19 13:59	825
	httpd-2.4.1.tar.gz.md5	2012-02-19 13:59	53
	httpd-2.4.1.tar.gz.sha1	2012-02-19 13:59	61

Obrázek 3: Distribuce zdrojového kódu Apache.

Zdroj: www.apache.org/dist/httpd 1.3.2012

Důležité je změnit adresář na místo, kam by se měl rozbalit kód Apache a kde se bude kompilovat [1, s 65]. Nejčastější umístění v systému Linux je adresář `/usr/local/src`. Toto se provede pomocí příkazu `cd /usr/local/src`, z toho adresáře se spustí nástroj pro dekomprimaci a extrahování souborů *tar*. Ten automaticky vytvoří potřebné adresáře.

3.4.1.2. Nástroj Autoconf

Tento nástroj umožňuje konfiguraci zdroje Apache před kompilací za účelem zadání určitých konfiguračních možností a zahrnutí nebo vyloučení některých modulů. Autoconf ještě provádí různé testy, které mají za úkol zjistit podrobnosti o hardwaru a operačním systému. Autoconf vytváří soubory, které určují, jak se má kompilace provést. Nejdůležitější úkol je vytvořit soubory, které používá linuxový nástroj *make* pro řízení kompilátoru C. Dále se používá pro generování skriptu *configure* pro aplikaci, která může být použita pro generování souborů, které zkompilují aktuální zdrojový kód C. Při kompilování Apache se bude používat konfigurační skript, vytvořený pomocí nástroje Autoconf místo přímého použití Autoconf. [20]

3.4.1.3. Skript Configure

Skript *configure* se nachází v adresáři nejvyšší úrovně zdroje Apache. Jeho funkcí není kompilovat sever Apache, ale prohledat systém, zjistit možnosti instalace a najít podpůrné soubory, které jsou zapotřebí. Další funkcí tohoto skriptu je informovat, že není možné Apache sestavit a předávat informace, jak problémy, které odhalil, opravit. Na současných distribucích systému Linux však k těmto problémům již většinou nedochází. Když skript *configure* zjistí, že je možné Apache sestavit, nastaví nejlepší možnou kombinaci pro daný systém. Hodnoty nastavení serveru Apache a informace o operačním systému jsou uloženy do souboru *config.status*. [1, s. 66]

Skript *configure* se většinou spouští s dalšími parametry příkazového řádku. Pokud se tento skript spustí bez jakýchkoliv dalších parametrů, systém nahlásí chybu, že se bude konfigurovat Apache s implicitním nastavením, což nemusí být to, co bylo zamýšleno. Tento postup se bude opakovat při každé další dodatečné instalaci nějakých nových modulů nebo změnách nastavení serveru.

Skript *configure* je možné spustit s parametrem *--with-layout*, který předává serveru Apache informaci, aby se podíval do souboru *config.layout*, který je umístěn v kořenovém adresáři stromu zdroje Apache. Tato část souboru bude použita k nastavení implicitních hodnot pro adresáře, ve kterých jsou soubory Apache uloženy. To je důležité, při požadavku instalace více verzí serveru Apache na jednom stroji, aby bylo možné změnit, do jakého adresáře bude další verze Apache nainstalována. [1, s. 66]

Pro zapnutí podpory dynamického přidávání modulů je potřeba spustit skript *configure* s parametrem *--enable-shared=max*. To způsobí, že všechny moduly, které jsou v Apache obsaženy, budou sestaveny jako DSO s výjimkou *http_core* a *mod_so*, které musí být do jádra připojeny staticky. V modulu *http_core* jsou obsaženy základní direktivy serveru Apache a modul *mod_so* zajišťuje podporu dynamických modulů. [10]

Pro zahrnutí všech standardních modulů se používá parametr *--enable-mods-shared=most*. Tento parametr zajistí nainstalování nejpoužívanějších modulů. Pokud se jako parametr příkazového řádku zadá *--enable-mods-shared=all*, budou nainstalovány všechny moduly, které server Apache obsahuje.

Skript *configure* vytváří sadu instrukcí pro kompilátor, aby bylo možné vytvořit fungující systém. Používá informace, které jsou zadány z příkazového řádku a o možnostech operačního systému např. dostupnost knihoven. Výsledkem je vytvoření souborů *Makefile*, které instruuje nástroj *make*, jak zkompileovat zdrojový kód a nainstalovat soubory do správných míst. Pokaždé, když se spustí skript *configure*, vytvoří se soubor *config.log*, který je umístěn v adresáři se zdrojem Apache [1, s. 67]. Pokud v tomto umístění již soubor existuje, je přepsán. Tento soubor obsahuje výstup *autoconf* a používá se pro odstraňování problémů, které vzniknou při použití skriptu *configure*. V prvním řádku toho souboru je obsažen příkaz, který byl použit pro spuštění *configure*.

Pokud se spustí skript *configure* s parametrem *--help*, je zobrazen kompletní seznam všech parametrů, s jakými je možné skript spustit

3.4.1.4. Soubor *config.layout*

Tento soubor obsahuje cesty, které se používají k umístění souborů v průběhu kompilace a pro zjištění, kam soubory přesunout v průběhu instalace. Soubor obsahuje implicitní nastavení sady cest pro různé systémy. Každá sada je identifikována názvem operačního systému, proto se jim říká pojmenovaná rozmístění. Po spuštění skriptu *configure* se Apache pokusí pomocí nástroje *GuessOS* zjistit, o jaký operační systém se jedná. Pokud toto zjištění odpovídá názvu některého z pojmenovaného rozmístění, použije toto rozmístění pro správné stanovení informací o cestách. Pokud se mu nepodaří zjistit název operačního systému, použije implicitní nastavení Apache, které je definováno v souboru *config.layout*. Každý řádek v souboru *config.layout* obsahuje název cesty

k adresáři. Je možné vytvořit vlastní rozmístění, mnoho lidí však tuto cestu považuje za riskantní a radši používá implicitní nastavení cest v serveru Apache.[1, s. 73]

3.4.1.5. Vytvoření Apache

Když je dokončena fáze konfigurace, je sestavena sada souborů *makefile* v různých místech stromu se zdrojovým kódem Apache. Pro započítí kompilace se používá příkaz *make*. Tento příkaz vytvoří soubor *httpd*, jehož velikost bude různá v závislosti na tom, jestli bylo rozhodnuto o sestavení serveru Apache se statickými nebo dynamickými moduly. Pro zmenšení velikosti souboru *httpd* se používá příkaz *strip*. Ten odstraňuje informace, které používají ladící programy a vývojářské nástroje. Tyto informace je důležité odstranit, aby se snížily nároky jádra na paměť. Pokud jsou odstraněny tyto informace, již není možné používat ladící programy.

Poslední fází je opětovné spuštění *make* s parametrem *install*. Tento příkaz přemístí binární a podpůrné soubory do jejich implicitního umístění. [1, s. 77]

3.4.2. Instalace binárních souborů Apache

Pokud není v plánu přidávat do serveru moduly třetích stran, které nejsou kompilované jako DSO a provádět takové změny v konfiguraci, aby server podporoval například moduly Perl nebo PHP není potřeba, aby byl server Apache kompilován ručně. Pokud bude dostačovat server, který podporuje jednoduché skripty CGI, je jednodušší nainstalovat Apache pomocí zkompilovaných binárních souborů pomocí správce balíčků Linuxu nebo stažení binární distribuce. Tento způsob je nejjednodušší cestou instalace. Jeho nevýhodou je, že soubory, které náleží serveru Apache, jsou rozmístěny do různých složek operačního systému. Uživatel tyto soubory jen těžko hledá, když nemá předchozí představu o jejich umístění. Nainstalovaný server pomocí balíčků má moduly typu DSO, proto není problém doinstalovat např. *mod_php*.

3.4.2.1. Binární distribuce

Tyto distribuce jsou kompilovány pro velké množství operačních systémů a hardwarových platform. Při stahování binární distribuce je potřeba dát si pozor, pro jakou rodinu procesorů distribuci stahujeme. Když stáhneme verzi pro platformu Intel, nebude s největší pravděpodobností korektně fungovat na platformě AMD, pokud tedy vůbec fungovat bude.

V operačním systému Linux existuje nástroj, který je umístěn `/bin/uname -m`, který vrátí výsledek, do jaké rodiny procesorů ten, který pracuje na daném počítači, spadá.

- Nejdříve je nutné stáhnout binární distribuci Apache pro Linux, která bude komprimována pomocí archivu `tar`. Je nutné vybrat správnou verzi podle rodiny procesorů.
- Zkomprimovaný soubor se přesune do místa, kde se bude muset rozbalit a po skončení instalace odstranit.
- pomocí příkazu `tar -xvzf` se archiv s distribucí rozbalí a zobrazí se všechny soubory, které archiv obsahoval.
- Přejde se do vytvořeného adresáře `httpd`, který byl vytvořen a spustí se skript `./install-bindist.sh`
- Po obdržení hlášení, že instalace proběhla úspěšně, může být proveden pokus o spuštění serveru Apache. [7, s. 286]

To, že je server Apache funkční se ověří spuštěním příkazu `./httpd -v` z adresáře `bin`. Tento příkaz vrátí verzi nainstalované verze Apache a skončí.

3.4.3. Spuštění serveru Apache

Apache server se spouští jedním spustitelným souborem `httpd`, který je podporován s velkým množstvím modulů, které jsou zavedeny po načtení konfiguračních souborů. Spouštět je možné buď ručním voláním z příkazového řádku, nebo automaticky po zavedení operačního systému pomocí spouštěcího skriptu. Za normálních okolností se `httpd` spustí jako systémový daemon, který naslouchá nejčastěji na portu 80 nebo čeká na připojení klientů přes http na jednom nebo více soketech. Soubor `httpd` je možné spustit s několika parametry. Ty ho mohou spustit, zobrazit nějaké informace nebo ukončit, aniž by přešel do režimu daemonu.

Je možné spustit s parametrem `httpd-V`, tento způsob zobrazuje všechny implicitní hodnoty zkompilevané do `httpd`. Nejužitečnější informací jsou implicitní cesty, kde může server najít své podpůrné soubory a kam zapisuje seznamy.

Pokud se server spustí pomocí `httpd -l`, jsou zobrazeny všechny moduly, které jsou zkompileovány do `httpd`. K `httpd` je staticky připojen pouze modul `http_core` a modul `mod_so`, který je zaveden, pokud je povoleno dynamické zavádění modulů.

S parametrem `httpd -t` je možné spustit test syntaxe konfiguračních souborů, ale nespustí server. Tento test upozorňuje na číslo řádku, ve kterém je v souboru chyba. [1, s. 86]

3.4.3.1. Apachectl

Tento nástroj se používá pro provádění základních operací pro řízení serveru. Pomocí tohoto nástroje je možné spouštět a zastavovat Apache server. Server se spustí pomocí příkazu `apachectl start`. Spuštění příkazu `apachectl` je výhodnější, než pouhé spuštění `httpd`, jelikož `apachectl` nejprve zjišťuje, jestli už nejsou spuštěny nějaké procesy `httpd`. Pro zastavení Apache je možné použít příkaz `apachectl stop`, který vyhledá soubor `httpd.pid`, extrahuje PID hlavního procesu a použije `kill` pro zastavení procesu. Při spuštění `apachectl configtest` se ověří správnost syntaxe konfiguračního souboru. [8, s. 201]

3.4.3.2. Spuštění více serverů Apache

Někteří lidé potřebují na jednom fyzickém stroji spustit více webových serverů najednou. Tento způsob provozu mohou zvolit např. kvůli oddělení důvěrných dokumentů, ke kterým by bylo potřeba přistupovat zabezpečeným připojením a ostatní dokumenty poskytovat klasickým nezabezpečeným spojením. Jiným důvodem může být testování.

Webový server Apache umožňuje nainstalovat více verzí na jeden fyzický stroj, důležité však je, aby každá verze měla svůj vlastní konfigurační soubor. Při instalaci každé verze je nutné zadat různé hodnoty `--prefix`, což je cesta ke kořenovému adresáři, kde je Apache umístěn. Tím je zajištěno, že každá verze má svůj vlastní konfigurační soubor, vlastní spustitelný soubor `httpd` a vlastní sadu DSO modulů. Když je potřeba spustit právě jednu verzi Apache, provede se to spuštěním `httpd` s parametrem `-f` a zadáním cesty k danému souboru. [1, s. 91]

3.5. Základní konfigurace Apache

Po instalaci je server Apache připraven k základnímu použití. V tomto stavu však umožňuje jen velmi málo funkcí, jako např. předávání jednoduchých webových stránek

ze zvolených adresářů. Jelikož na server budou s největší pravděpodobností kladeny mnohem sofistikovanější požadavky, musí se provést ještě změny konfigurace.

Chování serveru Apache a jeho možnosti konfigurace se řídí pomocí direktiv. Direktivy mají stanovený přesný účel, místo a syntaxi. Používají se k přizpůsobení chování vybraného webového sídla. Pomocí direktiv se Apache neřídí, ale říká se mu, kde má hledat prostředky. Direktivy se dělí na základní, které jsou dostupné vždy a ty, které jsou přístupné pouze s volitelnými moduly. Pokud volitelné moduly nejsou zapojeny, Apache jejich direktivy ani nerozpozná. Základní direktivy jsou zkompileovány přímo do jádra Apache a nevyžadují pro používání žádnou speciální konfiguraci.

3.5.1. Soubor `httpd.conf`

Dříve používal server Apache tři konfigurační soubory:

- `httpd.conf`

Toto byl hlavní konfigurační soubor serveru.

- `access.conf`

V tomto souboru byla uložena přístupová práva.

- `srm.conf`

Soubor, ve kterém bylo uloženo nastavení prostředků.

V současných verzích Apache už existuje pouze jeden konfigurační soubor `httpd.conf`. Tento soubor je rozdělen do tří částí. Pro lepší orientaci a čitelnost je lepší tyto části udržovat.

- nastavení globálního prostředí

Do této části se umísťují direktivy, které řídí Apache jako celek. Tyto direktivy řídí, jak bude server vyřizovat požadavky uživatelů.

- hlavní nebo implicitní nastavení serveru

Zde jsou umístěny direktivy, které definují parametry hlavního serveru, který odpovídá na požadavky virtuálních hostitelů.

- sekce virtuálních hostitelů

Tato část obsahuje nastavení pro virtuální hostitele, umožňuje zpracovávat jediným procesem požadavky na různé IP adresy nebo názvy hostitelů. [1, s. 95]

3.5.2. Platnost direktiv

U direktiv můžeme hovořit o kontextu, ve kterém se používají. Ten určuje rozsah platnosti direktivy a rozsah použitelnosti, to znamená, že je určeno, kde mohou být direktivy umístěny. Existují čtyři kontexty direktiv. [1, s. 95]

3.5.2.1. Obecný kontext serveru

Direktivy, které zde pracují, se používají pro celý server. Jsou zde direktivy, které mají smysl pouze v tomto kontextu. Direktivy, které jsou použity v tomto kontextu, většinou mají nastaveny implicitní hodnoty, které si správce serveru může nastavit, jak potřebuje.

3.5.2.2. Kontext kontejneru

Zde jsou definovány direktivy, které jsou platné pouze, když jsou uzavřeny v jednom ze tří kontejnerů `<Directory>`, `<Files>` nebo `<Location>`. Tyto direktivy mají použitelnost danou rozsahem kontejneru.

3.5.2.3. Kontext virtuálního hostitele

Virtuální hostitel je definován pomocí kontejneru `<VirtualHost>`, ale pro definování kontextu se bere odděleně, jelikož virtuální hostitelé dost často přenastavují direktivy v obecném kontextu. Tito hostitelé běží na jednom stroji a pro připojující se uživatele se tváří jako právě jeden server.

3.5.2.4. Kontext `.htaccess`

S direktivami v tomto souboru se pracuje podobně, jako s direktivami v kontejneru `<Directory>` v `httpd.conf`. Rozdíl je v tom, že direktivy objevující se v tomto souboru mohou být zakázány pomocí direktivy `AllowOverrides`. To umožňuje uživatelům, kteří nemají dostatečná přístupová práva, aby mohli upravovat konfigurační soubor `httpd.conf`, čímž mohou omezit přístup k prostředkům pomocí souboru `.htaccess`.

3.5.3. Prostředí hlavního serveru

Pro nastavení serveru a jeho naslouchacích procesů se používají obecné direktivy. Tyto direktivy nemohou být použity v jiném, než obecném kontextu serveru.

3.5.3.1. Direktiva `ServerAdmin`

Tato direktiva zobrazí na chybové stránce e-mailovou adresu na správce serveru. Direktiva se nastavuje, aby uživatel v případě problému měl komu napsat. [6, s. 72]

3.5.3.2. Direktiva `ServerName`

Pro Apache je důležité, aby byl schopen určit název hostitele, na kterém běží. Tento název je používán, aby Apache mohl vytvořit na sebe odkazující URL. Pokud je na serveru spuštěn více jak jeden virtuální hostitel, každý se identifikuje pomocí direktivy `ServerName`. [4, s. 35]

3.5.3.3. Direktiva `ServerRoot`

Direktivou `ServerRoot` definuje adresář, ve kterém je server umístěn. Tento adresář je nastaven při instalaci předvolbou `--prefix`. K ostatním konfiguračním souborům jsou cesty vztaženy relativně k hodnotě prefix. Dá se tedy předpokládat, že v adresáři, kam ukazuje prefix, budou ostatní konfigurační soubory. [1, s. 97]

3.5.3.4. Direktiva `DocumentRoot`

Tato direktiva se používá k definování adresáře nejvyšší úrovně. Z tohoto adresáře se budou předávat soubory, na které obdrží server požadavky s URL. Je možné tento adresář nastavit i mimo kořenovou strukturu serveru Apache, s tím ale souvisí i změna kontejnerové direktivy `<Directory>`, do které se také musí přidat cesta k adresáři, která je stejná jako u direktivy `DocumentRoot`. [1, s 98]

3.5.3.5. Direktiva `ScriptAlias`

`ScriptRoot` definuje adresář, ve kterém jsou umístěny spustitelné skripty. Jedná se např. o skripty CGI, které je možné spouštět z webového prohlížeče. Apache implicitně vytváří `ScriptAlias` pro všechny požadavky URL na prostředky do adresáře `/cgi-bin/`. Adresář je možné opět libovolně změnit přidáním příslušné cesty do direktivy. [1, s 98]

3.5.4. Obecné direktivy severu Apache

3.5.4.1. Direktiva `ErrorDocument`

Server Apache zobrazuje standardní kódy chyb a hlášení, pokud při zpracování požadavku narazí na chybu. Direktiva `ErrorDocument` slouží k vytvoření vlastní odpovědi na chybu. Toto hlášení může být realizováno dvěma způsoby. Prvním z nich je nastavení vlastní hlášky, např. pro chybu, kdy stránka nebyla nalezena. `ErrorDocument 404 "Stránka nebyla nalezena.` Druhou možností je přesměrování na jiné URL. Apache odešle ke klientovi adresu, kde má požadovaný dokument hledat. [6, s. 68]

3.5.4.2. Direktiva `DefaultType`

Direktiva používaná pro nastavení implicitního obsahu souboru typu MIME pro dokumenty, které jsou požadovány ze serveru. Pokud se tato direktiva nepoužije, jsou všechny soubory na serveru brány jako typ MIME `text/html`.

3.5.5. Kontejnerové direktivy

Tyto direktivy se používají k omezení rozsahu platnosti direktiv serveru Apache, které uzavírá do ostrých závorek. [1, s. 105]

3.5.5.1. `<Directory>` a `<DirectoryMatch>`

Pomocí této kontejnerové direktivy lze každému neprázdnému adresáři, ke kterému má sever Apache přístup, definovat vlastnosti souborů a nastavit pravidla pro přístup konkrétních uživatelů. Adresář, u kterého mají být definovány vlastnosti, musí být zadán plným názvem cesty. Direktiva `<Directory>` je párová, tzn.: že musí být vždy ukončena `</Directory>`. Pravidla použití jsou stejná jako u značkovacích jako např. HTML.

Direktiva `<DirectoryMatch>` je navržena pro používání regulárních výrazů. Tato direktiva je totožná s direktivou `<Directory>` s tím rozdílem, že adresáře, na které se aplikuje, jsou porovnávány s regulárními výrazy.

3.5.5.2. <Files> a <FilesMatch>

Předchozí direktiva umožňovala nastavit vlastnosti adresářů. Direktiva <Files> slouží k nastavení přístupu k jednotlivým souborům. Jméno souboru musí být zadáno pomocí celého názvu nebo pomocí zástupných znaků.

Direktiva <FilesMatch> je přímo navržena pro regulární výrazy, je identická s direktivou <Files> s tím rozdílem, že soubory jsou definovány regulárními výrazy.

3.5.5.3. <Location> a <LocationMatch>

Direktiva <Location> uzavírá do svého těla konkrétní URL. Tato direktiva je velmi podobná direktivě <Directory>, protože URL obsahuje odkaz na adresář vztahující se k *DocumentRoot*, avšak nepřistupuje k souborovému systému, pouze bere URL požadavku. U direktivy <LocationMatch> se adresa URL zadává pomocí regulárních výrazů.

3.5.5.4. Způsoby vyhodnocení kontejnerových direktiv

Jako první se vyhodnocuje kontejner s direktivou <Directory>, pokud se nepoužívají regulární výrazy a sloučí soubory .htaccess, které se týkají požadavku. Podle rozsahu platnosti se vyhodnotí kontejnery <Directory> od nejširšího po nejužší. Direktivy, které se nacházejí v souboru .htaccess, přenastaví ty, které se v <Directory> týkají stejného adresáře.

Dále se vyhodnocují direktivy <DirectoryMatch>, které jsou určeny pro používání regulárních výrazů. Postup je stejný jako u direktiv, které regulární výrazy nepoužívají.

Následují direktivy, které se týkají souborů, které jsou umístěny v adresářích. Soubory, které jsou umístěny v kontejnerech <Files> a <FilesMatch> mají přednost před soubory, které jsou umístěny v <Directory>.

Jako poslední se použijí direktivy v kontejnerech <Location> a <LocationMatch>. Tyto direktivy mají vztah k požadovanému URL a mají přednost před direktivami ze všech ostatních kontejnerů.

3.5.6. Soubor `.htaccess`

Server Apache se nastavuje pomocí souboru `httpd.conf`. Změna tohoto souboru nemusí být nejlepším způsobem, jak nakonfigurovat Apache. Pokud je potřeba např. omezit práva pro řízení přístupu nebo seskupit několik direktiv pro jeden konkrétní adresář, je lepší použít soubory `.htaccess`. Tento soubor umožňuje mít direktivy, které se týkají skupiny souborů uvnitř adresáře na jednom místě. Dále je možné přidělit práva pro úpravu souborů `.htaccess` podle adresářů. To přináší výhodu v tom, že uživatel může měnit přístupová práva k jednotlivým souborům, aniž by měl nastavena superuživatelská privilegia. Výhodou je také to, že je možné měnit soubor `.htaccess` bez potřeby restartovat server Apache. [4, s. 47]

Server Apache hledá soubory `.htaccess` v každém adresáři, ze kterého předává prostředky. V případě, že je nalezen, jsou v něm obsažené konfigurační direktivy sloučeny s ostatními direktivami, které se týkají daného adresáře.

3.5.6.1. Direktiva `AllowOverrides`

Implicitní chování serveru Apache vzhledem k souborům `.htaccess` je možné změnit pomocí direktivy `AllowOverrides`. Tato direktiva určuje, jak budou soubory `.htaccess` zpracovány. Pokud server Apache přijme požadavek na prostředek, přejde do adresáře, kde je prostředek umístěn a tam hledá soubor `.htaccess`. Pokračuje v hledání dalších souborů `.htaccess` v hierarchii systému souborů až do kořenového adresáře. Všechny direktivy, které byly nalezeny ve všech souborech `.htaccess` v systému souborů, jsou sloučeny s dalšími direktivami platnými pro adresář. Po sloučení direktiv ze souborů `.htaccess` se všemi direktivami `<Directory>` server Apache uplatňuje pořadí vyhodnocení od nejširšího po nejužší kontext.

Direktiva `AllowOverrides` určuje, které direktivy ze souborů `.htaccess` dostanou přednost před konfliktními direktivami, které již platí. Tato direktiva se nepoužívá pro zakázání nebo povolení direktiv, ale pro možnost přenastavení direktiv v souborech `.htaccess`.

U direktivy `AllowOverrides` je možné použít velké množství voleb. Zde budou uvedeny jen dvě nejdůležitější, a to volba `All` a `None`. Volba `All` umožňuje přenastavení všech direktiv, které obsahuje soubor `.htaccess`. Všechny direktivy, které obsahuje tento soubor a jsou povolené, mohou být povoleny pro použití nastavení ze souboru `httpd.conf`.

Volba *None* zakazuje přenastavování direktiv pomocí souborů *.htaccess*. Pokud je u direktivy nastavena tato volba, nebudou se za žádných okolností načítat soubory *.htaccess*, i když jsou v cílovém adresáři umístěny. Pokud je tato volba nastavena pro kořenový adresář, nebudou se načítat žádné soubory *.htaccess*. [1, s. 112]

3.6. Zabezpečení serveru Apache

Pro zajištění bezpečnosti webového serveru je třeba chránit privátní prostředky uživatelů pomocí nástrojů, které uživatele nutí, aby se identifikoval tím, že při každém přihlášení zadával přihlašovací údaje. Z toho vyplývá, že správce serveru by měl vědět, kdo se k serveru přihlašuje a jaká má práva.

3.6.1. Základní nástroje pro zabezpečení serveru

Pro základní zabezpečení serveru Apache se používají moduly, které slouží pro řízení přístupu k prostředkům. Tyto moduly fungují tak, že vrací hodnotu OK, pokud je prostředek povolen nebo hodnotu FORBIDDEN, pokud by měl být prostředek zakázán. Jsou aplikovány tři fáze zpracování požadavku, u kterého se musí rozhodnout, jestli se bude přistupovat k chráněným prostředkům serveru Apache.

První fáze se nazývá řízení přístupu. V průběhu této fáze zpracování požadavku může modul zamítnout přístup k prostředku na základě informací uvedených v požadavku HTTP. Za tuto fázi je zodpovědný modul *mod_access*, který má na starosti zamítat nebo povolovat přístup k prostředkům na základě adresy, ze které požadavek dorazil.

Druhá fáze se nazývá ověření. V průběhu této fáze se zavolá modul, který má za úkol ověřit přihlašovací údaje a heslo, které bylo zadáno uživatelem.

Posledním krokem je fáze autorizace. Když je modul, který je zodpovědný za autorizační fázi zavolán, považuje se identita uživatele za ověřenou a kontrolují se informace o řízení přístupu. Jsou zkontrolována práva přístupu k požadovanému prostředku. Když uživatel práva vlastní, modul odpoví OK, pokud uživatel práva nevlastní, modul odpoví FORBIDDEN.[1, s 397]

3.6.2. Omezení dle původu klienta

Pro odepření přístupu klientům podle IP adresy se využívá modul *mod_access*, který zjišťuje, jestli IP adresa, ze které klient přistoupil, není zakázána nebo jestli nechybí na seznamu IP adres, které jsou povoleny. Pro definování IP adres, které mají povolený přístup, se používá direktiva *allow*. Pro definování IP adres, které mají přístup zakázán, se používá direktiva *deny*. Omezení klientů je možné zadat různými způsoby. Je možné povolit nebo odepřít přístup všem zapsáním direktivy *allow/deny from all*. Pro povolení přístupu jedné konkrétní IP adresy lze zapsat *allow from 192.168.0.1*. [5, s. 29]

3.6.3. Omezení dle identifikace uživatele

Omezení dle původu klienta není použitelné vždy, protože správce serveru nemůže přesně vědět, z jaké IP adresy se klient připojí. Tento způsob je spíše vhodný pro adresy, které máme pod kontrolou. Pro ověření uživatele pomocí HTTP se používá nešifrované spojení pro přenos uživatelského jména a hesla. Druhým způsobem ověření pomocí HTTP je za použití ověřování, které se nazývá *digest*. To umožňuje přenášet data pomocí šifrovaného spojení. Data se zašifrují ještě před odesláním, což zmenšuje hrozbu zachycení hesla a přihlašovacího jména při přenosu.

3.6.3.1. Autorizovaný přístup

Na webovém serveru Apache je možné zabezpečit přístup k prostředkům pomocí vynuceného ověření uživatele. Internetový prohlížeč zobrazí formulář pro vyplnění přihlašovacího jména a hesla. Zadané údaje server poté porovná s údaji, které jsou uvedeny v textovém souboru. O vynucené zadání přihlašovacího jména hesla se stará direktiva *require*. Avšak pouhé použití této direktivy nestačí. Aby bylo možné plně využívat této funkce, musí se ještě nastavit typ autentifikace pomocí nastavení direktiv *AuthName* a *AuthType*. Dále se ještě musí vybrat modul, který má být této operaci přiřazen. Nejjednodušším způsobem je svěřit tento krok modulu *mod_auth*, pokud není vyžadován šifrovaný přenos přihlašovacích údajů.

Pro vytvoření soubory s hesly se použije příkaz *htpasswd -c jmeno_souboru jmeno_uzivatele*. Tento soubor není vhodné uložit do adresářové struktury serveru Apache, to není bezpečné místo. Soubor s hesly osahuje na každém řádku dvě položky oddělené dvojtečkou. První položka je uživatelské jméno, druhá položka je heslo. Pro přidání

dalšího uživatele do souboru s hesly se použije stejný příkaz, jako pro vytvoření souboru, ale bez parametru `-c`. Pokud je nutné nějakého uživatele z tohoto souboru vymazat, není jiná cesta, než ručně pomocí textového editoru. Tento postup je vhodný pouze pro vymazání uživatelů, textový editor by se nikdy neměl používat pro úpravu záznamů v souboru nebo přidání uživatelů.

Pro ověřování přístupu se dále používá soubor se skupinami uživatelů. Soubor obsahuje název skupiny a jména všech uživatelů, kteří mají být do této skupiny přiřazeni. Struktura je podobná jako u souboru s hesly, první je uveden název skupiny, za dvojtečkou následují jména uživatelů. Jména uživatelů jsou oddělena mezerou. Soubor se skupinami uživatelů se vytváří ručně, opět na nějakém bezpečném místě mimo adresářovou strukturu serveru Apache. Pro zpracování se používá direktiva *require*. Uživatel bude vyzván k vyplnění uživatelského jména a hesla. Nejdříve se provede kontrola, jestli uživatel je členem dané skupiny, pokud ano, provede se kontrola hesla. [6, s. 201]

3.6.3.2. Ověřování pomocí databáze

Základní ověřování pomocí souborů může fungovat jen do určitého počtu uživatelů. Pokud počet uživatelů překročí 1000, dochází ke zpomalení ověření. Musí se vždy otevřít soubor s údaji o uživateli, který není indexovaný a každý řádek souboru porovnat se zadanými údaji. Řešením toho problému je přesunout seznam uživatelů do databáze. Databáze obsahují indexovaná řešení pro rychlé vyhledávání dvojice klíč-hodnota. Ověřování uživatelů pomocí databáze DBM je poskytováno modulem *mod_auth_dbm*. [1, s. 407]

3.7. Zjišťování výkonu serveru

Pro testování výkonu serveru Apache se používá utilita Apache Benchmark, která je součástí instalačního balíku serveru. Tento nástroj se používá pro zjištění, jak velkou zátěž server zvládne [18]. Test výkonu se spouští pomocí příkazu *ab* s dalšími různými parametry[9].

-A *auth-username:password* – používá jako základní autentizace serveru. Pokud není ověřeno uživatelské jméno a heslo, je zaslána chyba s kódem 401.

-c *concurrency* – je parametr, který udává, kolik požadavků se má vykonat v jeden okamžik. Pokud se tento parametr nezadá, je automaticky nastavena hodnota 1.

-k – umožňuje použít speciální funkci, při které se provede vícenásobný požadavek během jedné HTTP relace.

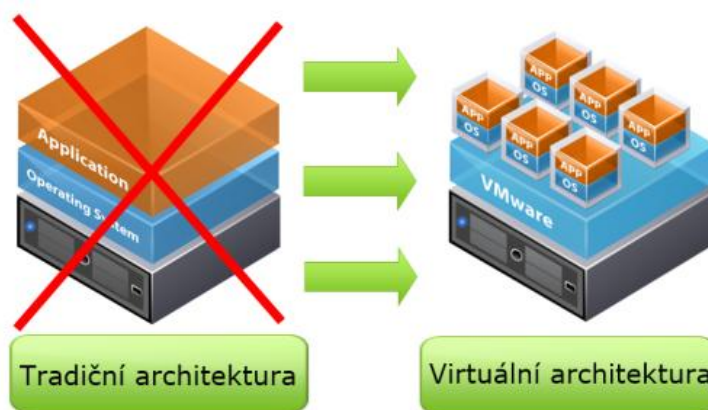
-n requests – určuje počet požadavků, které se budou provádět během jedné testovací relace. Základní hodnota je také nastavena na 1. Jeden požadavek ovšem nemůže přinést žádné relevantní výsledky.

-t timeout – stanovuje maximální časové rozpětí, během kterého je možné provádět testování. Jako základní hodnota je nastaveno: žádný časový limit.

Testování je možné provádět pomocí dvou základních způsobů. Prvním je testování pomocí localhost, kdy se testuje přímo na stroji, na kterém je spuštěn webový server. Tento způsob je vhodný pro otestování výkonu serveru bez omezení síťovou vrstvou. Druhý způsob je otestovat výkon serveru pomocí jiného stroje přes síť. Výsledek tohoto testu bude spíše zajímat uživatele než správce serveru [13].

3.8. Virtualizace

Virtualizace představuje uspořádání, při kterém se k systémovým zdrojům přistupuje jako k množině výkonu, kde se nebere ohled na fyzické parametry, pomocí kterých se přistupuje normálně. Pod pojmem server se tedy ve světě virtualizace nemyslí fyzický počítač, ale skupina zdrojů, ke které je možné přistoupit. Virtualizace umožňuje na jednom fyzickém počítači spustit více počítačů virtuálních. Na každém počítači může běžet jiný operační systém. Všechny najednou běží v reálném čase.



Obrázek 4: Rozdíl mezi tradiční architekturou a virtuální architekturou.

Zdroj: <http://www.oldanygroup.cz/virtualizace-vmware-zakladni-informace-9/>. [17]5.3.2012.

Existuje několik druhů virtualizace. V současné době nejrozšířenější je hardwarová virtualizace, které umožňuje správu a přiřazování hardwarových prostředků jednotlivým virtuálním počítačům. Tento typ virtualizace umožňuje spuštění více operačních systémů na jednom fyzickém stroji. Druhým druhem je para-virtualice, která je svým chováním velmi podobná hardwarové. Para-virtualizace ovšem počítá s optimalizací zátěže mezi virtualizační vrstvy. Také podporuje více operačních systémů, avšak operační systém musí být na tento způsob běhu připraven. Poslední je virtualizace na úrovni operačního systému, kde je virtualizační vrstva umístěna mezi operačním systémem serveru a virtuálními servery. Na jednom fyzickém stroji je možné provozovat pouze jeden virtuální stroj [14].

Výhody virtualizace spočívají v plném využití hardwarového výkonu, umožňuje provozovat více různých typů operačních systému najednou, zjednodušuje zálohování a umožňuje dynamické přidělování výkonu tzn.: je možné zvyšovat nebo snižovat výkon procesoru a měnit velikost operační paměti.

3.8.1. Virtualizační software

VMware Workstation

U tohoto produktu je již vyřešena otázka hardwarové a softwarové podpory. Tomuto produktu nechybí prakticky nic k dokonalému emulování hardwarového prostředí. Jako jediný podporuje grafickou 3D akceleraci pomocí technologie Direct 3D 8. Tento produkt však není zdarma a s cenou kolem 4000 Kč není vhodnou volbou pro domácí použití nebo menší firmy. Dalším menším nedostatkem může být, že prostředí není nejrychlejší.[12]

Firma VMware nabízí také jednodušší produkt zdarma pod názvem VMware Player. U této verze je však možné nainstalovat pouze operační systémy, které jsou v nabídce pro instalaci.

Virtual PC

Tento virtuální stroj je produkt od firmy Microsoft, čímž je zaručeno, že s operačními systémy od této firmy nebudou problémy jak s rychlostí, tak s podporou. Pokud je potřeba nainstalovat jiný operační systém, mohou nastat problémy. Při běhu některých distribucí systému Linux mohou nastat problémy se zobrazováním. Oproti VMware je tento virtualizační nástroj méně univerzální. Výhodou je, že Virtual PC je distribuován zdarma. [12]

Parallels Workstation

Tento nástroj má velmi rychlé prostředí bez nutnosti instalování jakýchkoliv doplňků nebo ovladačů. Parallels Workstation má několik hardwarových nedostatků. Hlavními nedostatky jsou, že virtualizační prostředí podporuje běh pouze 32 bitových operačních systémů a nedostatečnou podporu USB 2.0. Vyskytují se zde také problémy s vícajádrovými procesory.[12]

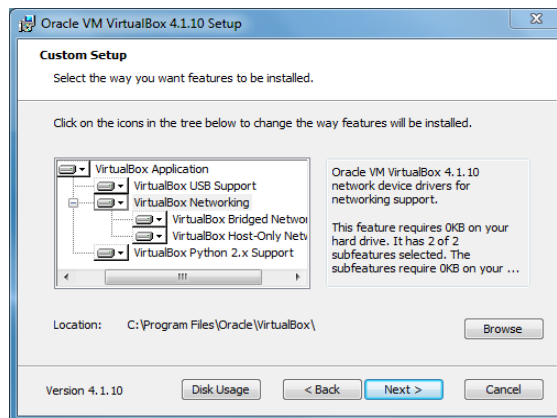
Virtual Box

Virtual Box má zásadní výhodu v distribuci zdarma. Nástroj nepodporuje méně obvyklé operační systémy, avšak při provozování běžných operačních systémů funguje bezproblémově. U některých operačních systémů je nutné doinstalovat ovladače. Rychlost prostředí je srovnatelná s produktem VMware. [12]

4. Praktická část práce

4.1. Instalace virtuálního stroje

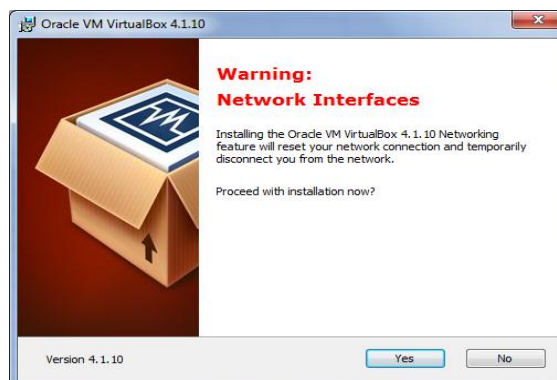
Pro porovnání výkonu byly vybrány dva virtualizační nástroje. Byly vybrány VMware Player 4 a Oracle VM Virtual Box. Produkty musely splňovat dvě podmínky, a to že musely být k použití zdarma s možností instalace 64 bitového operačního systému Linux Debian 5. Samotná instalace je velmi jednoduchá. Dále je popsán postup instalace Virtual Boxu 4.1.10. Instalace produktu VMware je velmi podobná, proto zde uvedena nebude.



Obrázek 5: Možnosti instalace Virtual Boxu.

Zdroj: Vlastní zpracování 5.3.2012.

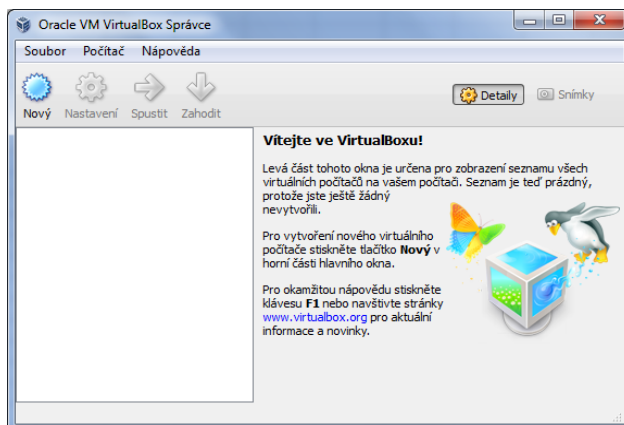
Po odkliknutí úvodní obrazovky je možné vybrat, jaké všechny součásti bude virtualizační nástroj obsahovat. Na obrázku 5 je vidět, že je možné zvolit, jestli bude nástroj podporovat připojení disků USB a připojení k síti.



Obrázek 6: Varování při instalaci Virtual Boxu.

Zdroj: Vlastní zpracování 5.3.2012.

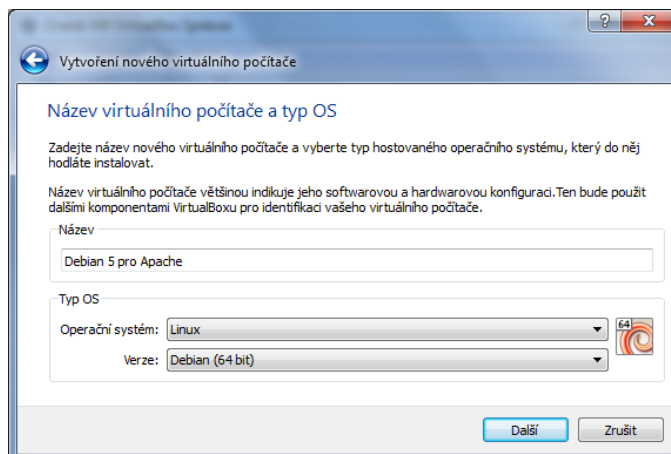
Na obrázku 6 je uvedeno varování, že instalace Virtual Boxu resetuje síťové připojení, a proto síť bude dočasně nedostupná. V průběhu instalace vznesl instalační program ještě několik požadavků, jestli mají být v systému nainstalovány síťové karty a jestli mohou být přidány do povolených zařízení. Po skončení instalace se objeví okno, které vyzývá ke spuštění aplikace.



Obrázek 7: Program Virtual Box bez nainstalovaného OS.

Zdroj: Vlastní zpracování 5.3.2012.

Po spuštění aplikace se objeví okno, kde je po kliknutí na ikonu Nový, možnost nainstalovat operační systém.

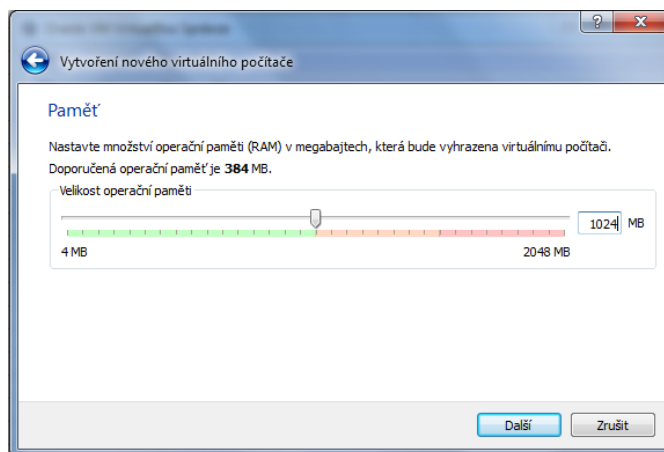


Obrázek 8: Okno pro výběr typu operačního systému.

Zdroj: Vlastní zpracování 5.3.2012.

Jelikož je Virtual Box zdarma, musí se uživatel spokojit s instalací předpřipravených operačních systémů. Na obranu toho řešení je nutné říci, že nabídka obsahuje v podstatě všechny současně používané operační systémy. Je možné nainstalovat všechny verze systému Microsoft Windows, nejpoužívanější distribuce Linuxu, jako jsou Debian,

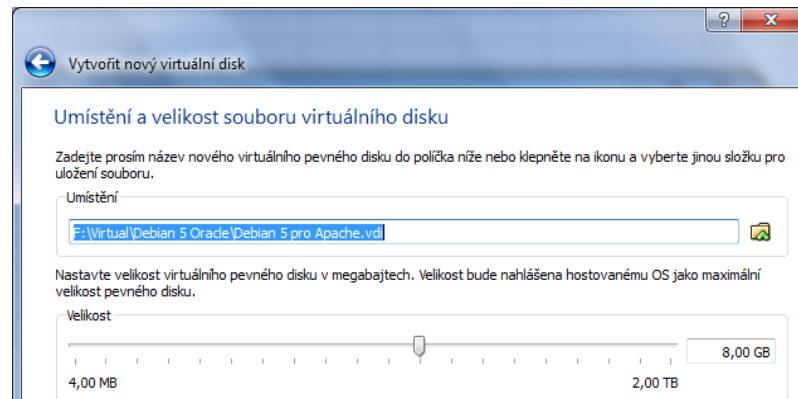
Mandriva, Gentoo, FreeBSD, OpenSuse, Fedora, Ubuntu a další. Podporované platformy jsou Linux, Microsoft, Mac OS, BSD, IBM/OS2 a pokud by si někdo nevybral, je ještě možné vybrat záložku Other, pod kterou se skrývá DOS, Netware nebo možnost Neznámý operační systém.



Obrázek 9: Nastavení velikosti operační paměti.

Zdroj: Vlastní zpracování 5.3.2012.

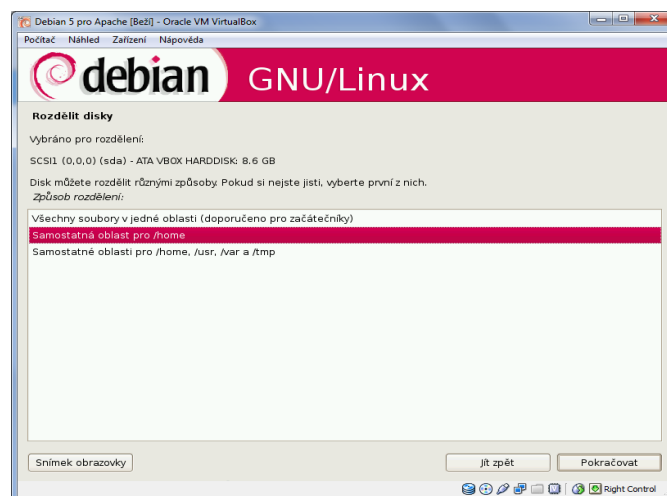
V tomto kroku se nastavuje velikost operační paměti. Bylo nastaveno 1024 MB, což je přesná polovina operační paměti fyzického stroje. Velikost paměti bude v průběhu měření výkonu serveru měněna. Ihned po nastavení velikosti operační paměti se vytváří bootovací pevný disk, který má nastavenou defaultní velikost 8 GB. Na obrázku 10 je vidět určení velikosti disku a řádek, kde je nutné zadat, kde se bude virtuální pevný disk nacházet. Nutné je také zvolit formát, ve kterém bude pevný disk vytvořen. Ten byl ponechán na defaultní hodnotě Virtual Box Disk Image. Pevný disk byl nastaven tak, aby automaticky alokoval místo na fyzickém pevném disku. V této konfiguraci se může pevný disk automaticky zvětšit, pokud je potřeba. Mohla by např. nastat situace, kdy bude mít operační systém nastavenou malou velikost paměti RAM, nebude mít možnost zvětšit swapovací prostor a ukončí se. Jako poslední krok se objeví okno, kde je uvedeno shrnutí, jaké nastavení virtualizačního nástroje bylo zvoleno, jaký operační systém se instaluje a kolik bylo přiděleno operační paměti. Ještě je zde upozornění, kde je možné najít nástroj pro změnu parametrů stroje.



Obrázek 10: Určení umístění virtuálního pevného disku.

Zdroj: Vlastní zpracování 5.3.2012.

Po skončení instalace virtuálního prostředí, je nutné nainstalovat operační systém. Tlačítkem přidat nový systém se spustí průvodce instalací. V prvním kroku se určí, z jakého média se operační systém bude instalovat. Použit byl obraz ISO DVD Debian 5 64 bit. ISO obrazy virtuálního prostředí podporuje bez nutnosti instalace doplňků. Nainstalovat operační systém z klasického optického média je také možné.

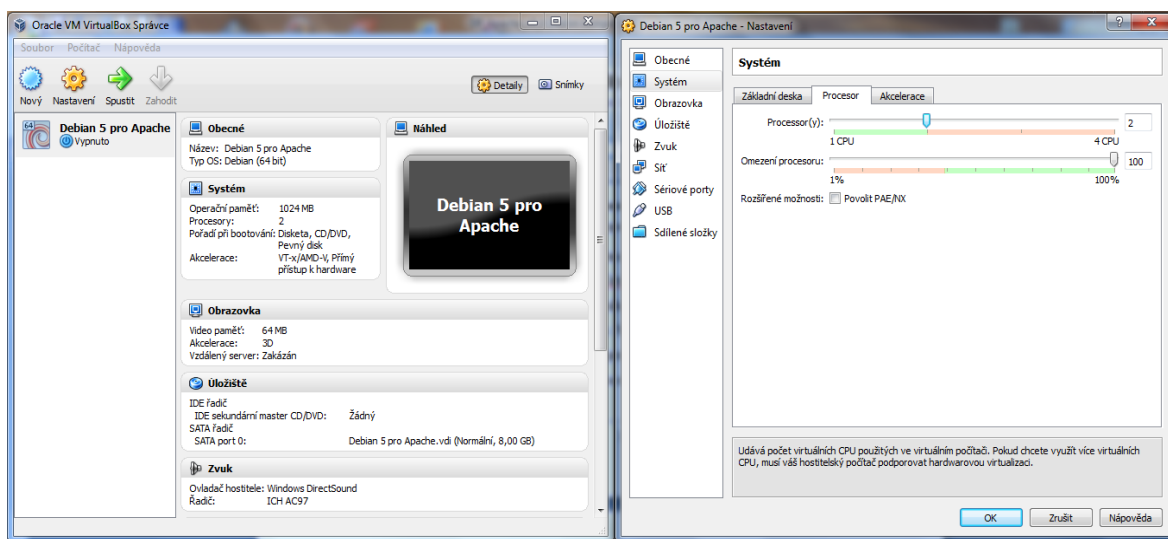


Obrázek 11: Instalace OS Debian 5.

Zdroj: Vlastní zpracování 5.3.2012.

Po dokončení instalace operačního systému je vše připraveno k použití. Na obrázku je zachyceno virtuální prostředí programu Virtual Box. Je zde vidět, jaký operační systém je nainstalován a výpis parametrů virtuálního stroje. Po stisku tlačítka Spustit dojde k bootování operačního systému. Tlačítko Nastavení slouží ke změně parametrů virtuálního stroje. Je zde možné nastavit velikost operační paměti, kolik jader procesoru má být použito a na kolik procent výkonu se procesor má omezit. Dále se zde nastavuje výkon

grafiky, kde se určí velikost grafické paměti a jestli bude povolena nebo zakázána 2D a 3D akcelerace. Tyto akce je však možné provádět pouze, když je operační systém vypnutý.



Obrázek 12: Prostředí virtuálního stroje Virtual Box a možnosti nastavení.

Zdroj: Vlastní zpracování 5.3.2012.

Byly nainstalovány dvě virtuální prostředí zastoupené programy VMware Player 4 a Oracle Virtual Box 4. V těchto prostředích byl nainstalován shodný operační systém Linux Debian 5. V obou prostředích byl nainstalován webový server Apache. Spuštění a nastavení webového serveru je popsáno v další kapitole.

4.2. Spuštění webového sídla

Když je webový server nainstalován, bude dalším krokem pokus o jeho spuštění. Překvapením může být, že ihned po zadání příkazu *apachectl start* se objeví chyba, že server není možné spustit. Tato chyba je způsobena tím, že v souboru *httpd.conf* se musí zadat název serveru a port na kterém se bude naslouchat. V tomto případě bylo zadáno pomocí direktivy takto:

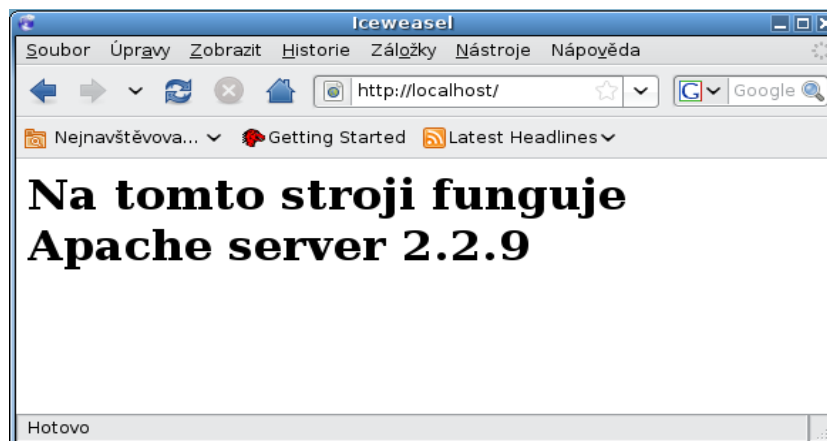
```
ServerName localhost:80
```

Důležitou direktivou, která také musí být v souboru *httpd.conf* je *ServerRoot* a *DocumentRoot*. Tyto dvě direktivy nastavují umístění serveru Apache a umístění dokumentů, které bude server poskytovat.

```
ServerRoot /etc/apache2/
```

```
DocumentRoot /var/www/
```

Pokud soubor *httpd.conf* obsahuje tyto tři řádky, je již možné vyzkoušet funkčnost webového serveru. Do složky *www* se umístí soubor s názvem *index.html* a do adresního řádku webového prohlížeče se napíše adresa <http://localhost> nebo je možné adresu zadat pomocí IP adresy <http://127.0.0.1>, oba způsoby povedou ke stejnému výsledku, kterým je zobrazení souboru *index.html*.

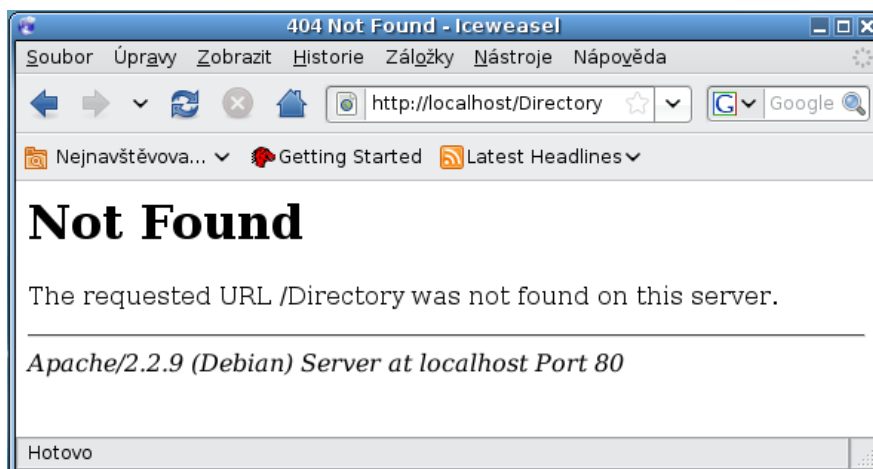


Obrázek 13: Soubor *index.html*

Zdroj: Vlastní zpracování 10.3.2012.

4.3. Výpis adresářů

V základním nastavení je server Apache nastaven tak, aby zobrazoval webové stránky. Při požadavku, aby webový prohlížeč zobrazil adresářovou strukturu nějaké složky, webový prohlížeč zobrazí chybu s oznámením, že požadovaná složka se na daném umístění nenachází.



Obrázek 14: Chyba při zobrazení obsahu složky.

Zdroj: Vlastní zpracování. 10.3.2012

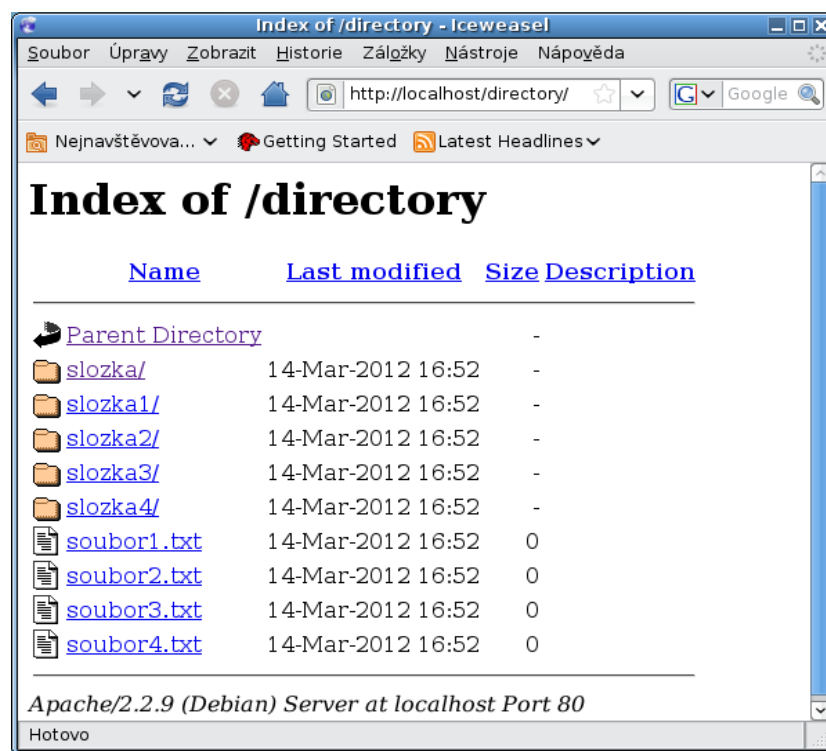
Aby server Apache byl schopen zobrazovat obsah složek, musí se použít direktiva *FancyIndexing*, která využívá modul *mod_autoindex* pro zobrazování obsahu složek. Pro zobrazení obsahu složky Directory v umístění *DocumentRoot* byl použit tento zápis:

IndexOptions FancyIndexing

Ukázka, jakou změnu přinesla tato direktiva, je zobrazena na obrázku číslo 15. Je zde uveden výpis souborů a složek, které jsou uloženy ve složce directory. Dalšími možnostmi výpisů jsou možnosti nastavit, aby se soubory a složky zobrazovaly podle jejich verzí nebo nastavit, aby ikony před soubory byly součástí odkazu. Jinou možností je také změnit velikost ikon. Na obrázku 16 je zobrazeno nastavení výpisu, při kterém jsou ikony o velikosti 30 px a jsou součástí odkazu. Zároveň je nastaveno, aby se nezobrazoval sloupec s velikostí souborů.

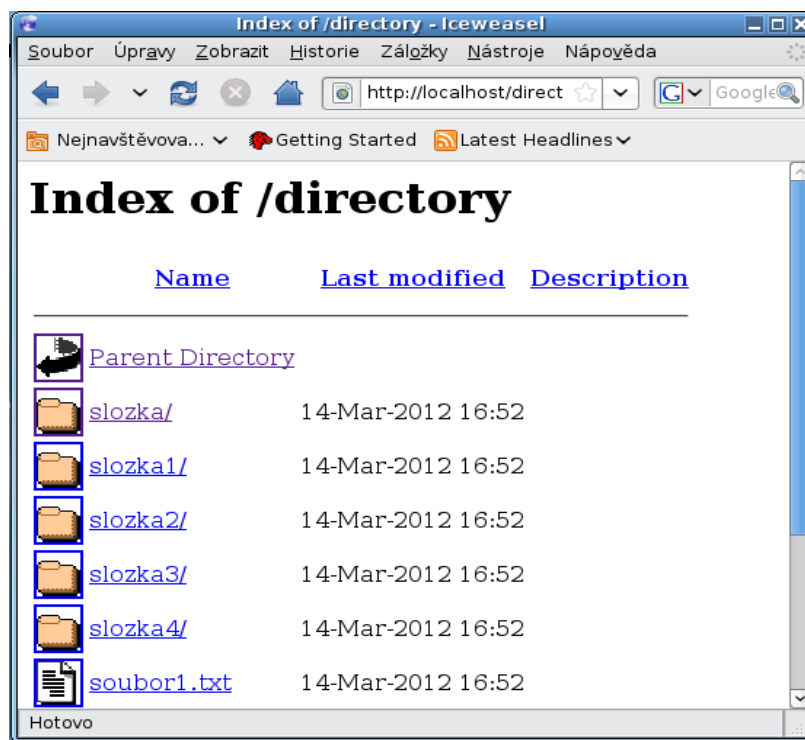
FancyIndexing VersionSort IconAreLinks IconWidth=30 SuppressSize

Tento zápis v souboru *httpd.conf* zajistí, aby se výpis složky directory zobrazil ve výše popsané struktuře.



Obrázek 15: Výpis obsahu složky - základní nastavení.

Zdroj: Vlastní zpracování. 10.3.2012

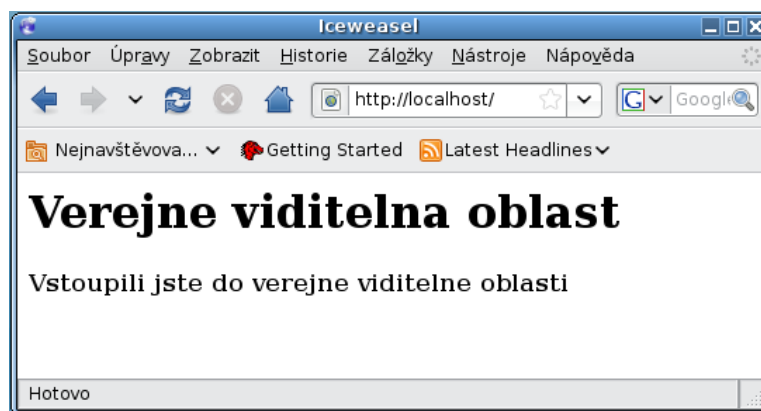


Obrázek 16: Výpis obsahu složky pomocí speciálních parametrů.

Zdroj: Vlastní zpracování. 10.3.2012

4.4. Omezení přístupu k prostředkům

Pro omezení přístupu k prostředkům byla zvolena metoda ověření uživatelského jména a hesla. Uživatelské jméno a heslo bylo uloženo do souboru *apachepasswd.txt* pomocí příkazu *htpasswd*. Pomocí direktivy bylo nastaveno umístění tohoto souboru. Na obrázku 17 je vidět situace, které nastane po přístupu k souboru *index.html*. V prohlížeči se normálně zobrazí obsah tohoto souboru.



Obrázek 17: Zobrazení index.html před aplikací omezení.

Zdroj: Vlastní zpracování. 10.3.2012

Pomocí příkazu `htpasswd -c` byl vytvořen soubor `apachepasswd.txt` s uživatelskými jmény a hesly. Do toho souboru bylo vloženo 5 uživatelů.

Jméno uživatele:heslo
susta:ua787amD1uQ.o

uzivatel1:0EuPWBI34.Xso

uzivatel2:0IKQuameFEFws

uzivatel3:uVTem0gwPPrlI

uzivatel4:6YiQGqyEiFIYg

uzivatel5:2cBg3JyZ2Piko

Tabulka 1: Výpis souboru `apachepasswd.txt`.

Zdroj: Vlastní zpracování. 10.3.2012

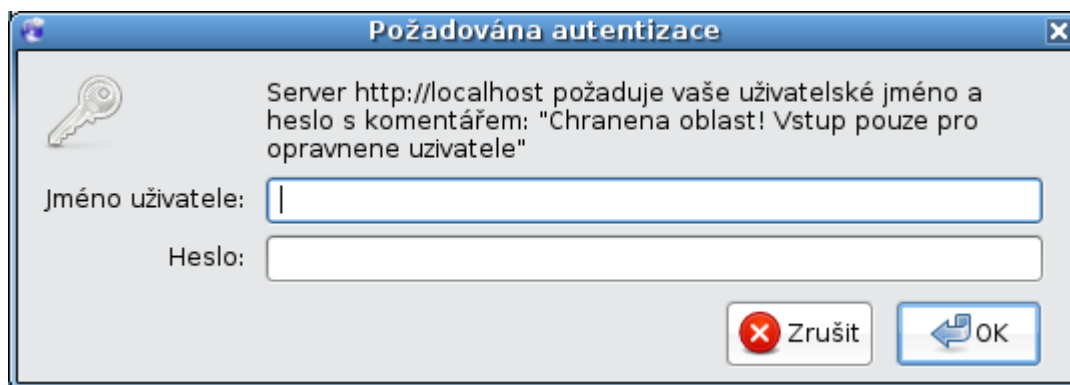
Ze souboru s hesly lze snadno vyčíst uživatelská jména, avšak hesla jsou zadaná pomocí hash funkce. Pro omezení přístupu k prostředkům se používá direktiva *Require*. Pomocí této direktivy je možné zadat uživatele, kteří budou mít ke chráněným prostředkům serveru přístup.

Zde je uveden výpis části souboru `httpd.conf`, který obsahuje omezení přístupu k prostředkům.

```
<Directory "/var/www/">
    Options ExecCGI
    DirectoryIndex index.html
    AllowOverride All
    Allow from All
    Deny from none
    Require User susta uzivatel1
    AuthType Basic
    AuthName "Chranena oblast! Vstup pouze pro opravnene uzivatele"
    AuthUserFile "/home/hans/dokumenty/apachepasswd"
</Directory>
```

Z toho výpisu vyplývá, že soubor `index.html` se nachází na umístění `/var/www/`. Direktiva *Require* umožňuje přístup dvěma uživatelům, kteří mají přihlašovací jména `susta` a `uzivatel1`. Pomocí direktivy *AuthType Basic* je určeno, pomocí kterého modulu se bude provádět autentifikace. Tento příkaz říká, že byl použit modul `mod_auth`. Direktiva *AuthName* udává důvod, proč se musí uživatel přihlásit. V tomto případě je důvodem,

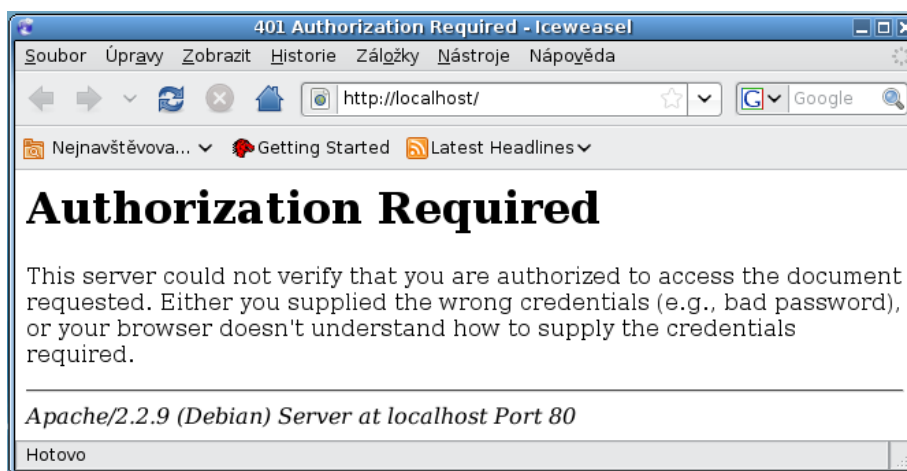
že uživatel vstupuje do chráněné oblasti. *AuthUserFile* udává umístění souboru s přihlašovacími jmény a hesly.



Obrázek 18: Ověřovací formulář.

Zdroj: Vlastní zpracování. 10.3.2012

Na obrázku 18 je znázorněn ověřovací formulář, který musí uživatel vyplnit. Pokud vyplní špatné uživatelské jméno a heslo, bude se stále dokola objevovat formulář pro ověření uživatelského jména a hesla. Pokud uživatel ověření zruší, zobrazí se mu stránka s upozorněním, že je vyžadována autorizace. Tato stránka je zobrazena na obrázku 19.



Obrázek 19: Odmítnutí přístupu ke stránce.

Zdroj: Vlastní zpracování. 10.3.2012

Když uživatel zadá správné jméno a heslo, jsou mu požadované stránky normálně zobrazeny.

4.5. Grafický nástroj pro nastavení Apache

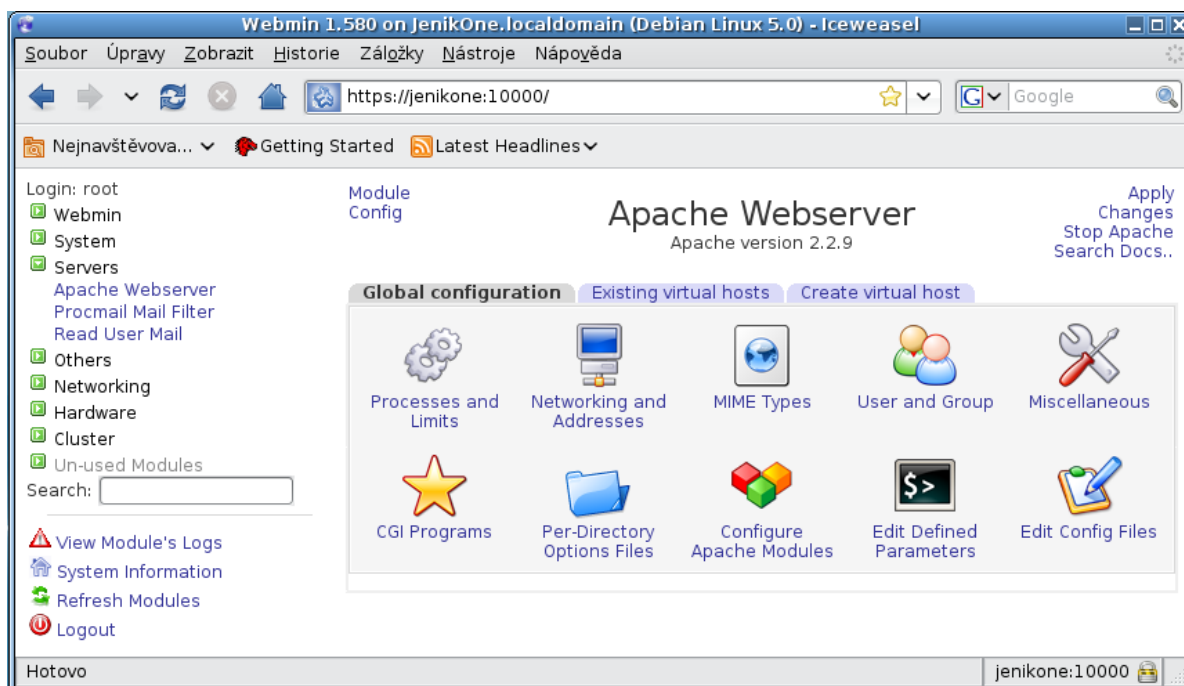
Pro nastavení serveru pomocí grafického rozhraní, byl nainstalován nástroj Webmin, který umožňuje správu celého serveru. Jedna z variant správy serveru je nastavení a správa webového serveru Apache.

4.5.1. Instalace nástroje Webmin

Z internetových stránek www.webmin.com byl stažen instalační balíček .DEB pro operační systém Debian. První pokus o instalaci skončil neúspěšně. Po spuštění příkazu `dpkg -i webmin` proběhla instalace, která se nezdařila, kvůli 6 chybějícím balíčků operačního systému. Tyto balíčky musely být doinstalovány pomocí `aptitude install`. Po opětovném spuštění instalační sekvence `dpkg -i webmin` již vše proběhlo v pořádku. V příkazové řádce bylo vypsáno, jakým způsobem se k správci serveru Webmin přihlásit.

4.5.2. Spuštění nástroje Webmin

Spuštění se provádí pomocí internetového prohlížeče, ve kterém se do adresního řádku napíše jméno hostitele - v tomto případě <https://jenikone:10000/>. Po načtení této adresy je uživatel vyzván k zadání uživatelské jména a hesla.



Obrázek 20: Webmin - grafický nástroj pro ovládání serveru.

Zdroj: Vlastní zpracování. 20.3.2012

Aby bylo možné využívat všechny funkce nástroje Webmin, je důležité, aby se uživatel přihlásil s právy root. Pokud tak neučiní a přihlásí se do systému jako klasický uživatel, většina konfigurací nebude umožněna.

Po přihlášení se zobrazí podrobný přehled o běžícím operačním systému. Jsou zde uvedeny běžící procesy, vytížení CPU, počet pevných disků a další informace. V levém menu je možné vybrat, jaká část serveru se bude spravovat. Je zde možné vybrat některé zajímavé funkce jako nastavení zálohování systému na určitý čas a datum, změna nastavení samotného grafického nástroje, správa uživatelů systému, je možné přes tento nástroj server restartovat nebo vypnout. Zajímavou funkcí je možnost stahovat ze serveru nebo na server nahrávat soubory. Pro tyto účely je zde speciální manager.

Pro účely poskytování webových stránek je ovšem nejdůležitější záložka Servers, kde se vybere Apache webserver. Globální možnosti nastavení jsou zobrazeny na obrázku 20. První možnost nastavení je Processes and Limits, kde je možné nastavit maximální počet hlaviček v požadavku, maximální počet požadavků na jeden proces serveru nebo minimální počet záložních procesů serveru. Vše je možné zakliknout nebo vepsat do textového pole. Pro provedení zmíněného nastavení pomocí příkazového řádku, by bylo potřeba do souboru *httpd.conf* zapsat direktivy *MinSpareServers*, *MaxSpareServes* a *StartServers*, pokaždé s příslušným číslem, podle toho kolik nastartovaných procesů je požadováno.

Další záložka je Networking and Addresses, kde se nastavuje, na jaké IP adrese a portu bude server naslouchat, je možné zapnout vícenásobné požadavky na jedno připojení a nakonfigurovat virtuální server.

MIME Types obsahuje soupis podporovaných MIME typů. Je zde také možné změnit umístění souboru s těmito typy nebo vybrat soubor jiný. Hned vedle se nachází záložka Users and Groups, kde se určí, jako jaký Unixový uživatel se Apache server spustí. V tomto případě je vytvořen speciální uživatel *www-data*.

Per Directory Options File slouží pro použití souboru *.htaccess*. Je zde možné tyto soubory vytvářet, nebo nastavovat umístění, kde se nacházejí. Je možné vybrat, jestli se soubory *.htaccess* budou hledat automaticky nebo ve zde zadaném umístění.

Záložka Configure Apache Modules slouží k povolení nebo zakázání vybraných modulů. Jsou zde uvedeny všechny moduly, které jsou na webovém serveru Apache nainstalovány.

Moduly, které jsou na obrázku 21 označeny zeleně a zaškrtnuty, jsou ve webovém serveru poveleny k používání. Ty moduly, které zaškrtnuty nejsou a mají bílou barvu, jsou zakázány. Pokud se modul povoluje, jednoduše se zaškrtně a zmáčkne se dole pod soupisem tlačítko Enable Selected Modules. Po znovunačtení stránky je modul povolen a připraven k použití. Aby bylo možné využívat funkce modulu, musí se ještě do konfiguračního souboru zapsat direktivy, které mu náleží. V příkazovém řádku se povolení a zakázání modulů provádí pomocí příkazů *a2enmod* a *a2dismod*.

Module	Current state	Module	Current state
<input type="checkbox"/> actions	Disabled	<input type="checkbox"/> ext_filter	Disabled
<input checked="" type="checkbox"/> alias	Enabled	<input type="checkbox"/> file_cache	Disabled
<input type="checkbox"/> asis	Disabled	<input type="checkbox"/> filter	Disabled
<input checked="" type="checkbox"/> auth_basic	Enabled	<input type="checkbox"/> headers	Disabled
<input type="checkbox"/> auth_digest	Disabled	<input type="checkbox"/> ident	Disabled
<input type="checkbox"/> authn_alias	Disabled	<input type="checkbox"/> imagemap	Disabled
<input type="checkbox"/> authn_anon	Disabled	<input type="checkbox"/> include	Disabled
<input type="checkbox"/> authn_dbd	Disabled	<input type="checkbox"/> info	Disabled
<input type="checkbox"/> authn_dbm	Disabled	<input type="checkbox"/> ldap	Disabled
<input type="checkbox"/> authn_default	Disabled	<input type="checkbox"/> log_forensic	Disabled
<input checked="" type="checkbox"/> authn_file	Enabled	<input type="checkbox"/> mem_cache	Disabled
<input type="checkbox"/> authnz_ldap	Disabled	<input checked="" type="checkbox"/> mime	Enabled
<input type="checkbox"/> authz_dbm	Disabled	<input type="checkbox"/> mime_magic	Disabled
<input checked="" type="checkbox"/> authz_default	Enabled	<input checked="" type="checkbox"/> negotiation	Enabled
<input checked="" type="checkbox"/> authz_groupfile	Enabled	<input checked="" type="checkbox"/> php5	Enabled
<input checked="" type="checkbox"/> authz_host	Enabled	<input type="checkbox"/> proxy	Disabled
<input type="checkbox"/> authz_owner	Disabled	<input type="checkbox"/> proxy_ajp	Disabled
<input checked="" type="checkbox"/> authz_user	Enabled	<input type="checkbox"/> proxy_balancer	Disabled
<input checked="" type="checkbox"/> autoindex	Enabled	<input type="checkbox"/> proxy_connect	Disabled
<input checked="" type="checkbox"/> cache	Enabled	<input type="checkbox"/> proxy_ftp	Disabled
<input type="checkbox"/> cern_meta	Disabled	<input type="checkbox"/> proxy_http	Disabled
<input checked="" type="checkbox"/> cgi	Enabled	<input type="checkbox"/> rewrite	Disabled

Obrázek 21: Soupis nainstalovaných modulů v prostředí Webmin.

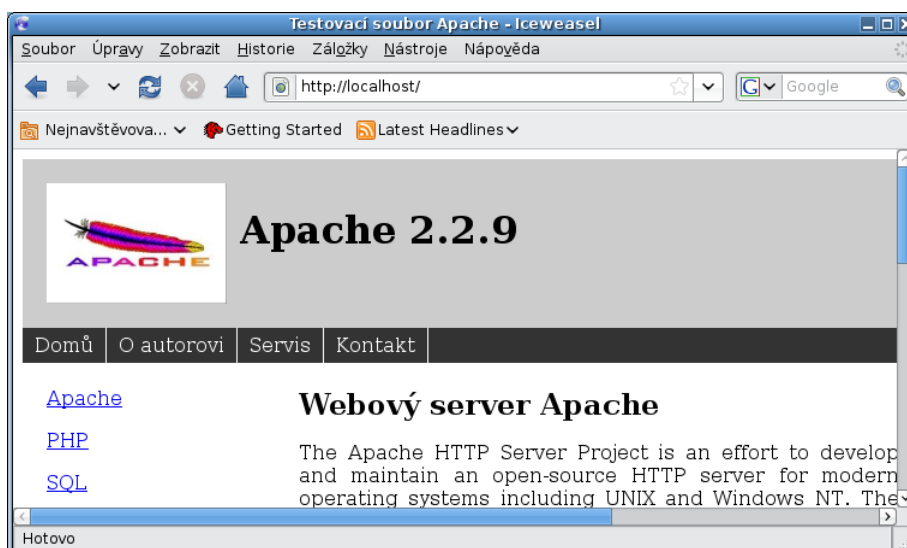
Zdroj: Vlastní zpracování. 20.3.2012

V záložce Edit Defined Parametrs se nastavuje, jaké parametry bude mít Apache po spuštění. Nastavuje se zde, jako jaký uživatel a jaká skupina bude server spuštěn. V tomto případě je nastaveno již zmiňované *www-data* pro obě položky. Je zde také možné nastavit, pod jakým PID Apache poběží.

Poslední záložkou globálního nastavení serveru Apache je Edit Config Files. Zde je možné vybrat, jaký konfigurační soubor chceme upravit. Zmáčkne se tlačítko Edit Directives in Files a tento soubor je načten v poli pod tímto tlačítkem. Ten je možné libovolně upravovat, po dokončení úprav se zmáčkne pod textovým polem tlačítko *Save*. Tento způsob je velmi pohodlný, v příkazové řádce je možné konfigurační soubory upravovat pouze pomocí textových editorů, jako jsou *vi* nebo *vim*. Práce s těmito editory vyžaduje jistou zkušenost a trpělivost. Pro uživatele, kteří uvidí tento editor poprvé, se může zdát nelogický až matoucí.

4.6. Měření výkonu webového serveru Apache

Měření výkonu webového serveru Apache bylo prováděno na virtuálních strojích VMware Player 4 a Oracle Virtual Box 4. Byl měřen výkon pro poskytování 2 typů souborů. V prvním kroku byly měřeny 2 statické soubory HTML, kde každý ze souborů měl různý obsah. První soubor obsahoval pouze krátký text, druhý soubor obsahoval 3 odstavce textu, obrázky, menu a byl formátován pomocí kaskádových stylů. Ve druhém kroku byl měřen výkon pro poskytování souborů, ze kterých se generují stránky pomocí PHP. První soubor opět obsahoval pouze krátký text. Druhý soubor obsahoval program, který po spuštění skriptu vypsal v prohlížeči kalendář. Měření dále probíhalo ve 2 fázích – virtuální stroj měl nastaveny plné parametry, tzn.: mohl používat obě jádra procesoru a 1 GB operační paměti. Ve druhé fázi byly virtuální stroje omezeny tak, že mohly používat pouze jedno jádro procesoru a 512 MB operační paměti.



Obrázek 22: Ukázka HTML souboru s použitím CSS stylů.

Zdroj: Vlastní zpracování. 14.3.2012

Pro získání naměřených hodnot bylo použito programu Apache Benchmark. V příkazu byly vždy nastaveny stejné parametry pro každé měření.

```
ab -n 10000 -c 500 http://127.0.0.1:80/ > /Desktop/vypis.txt
```

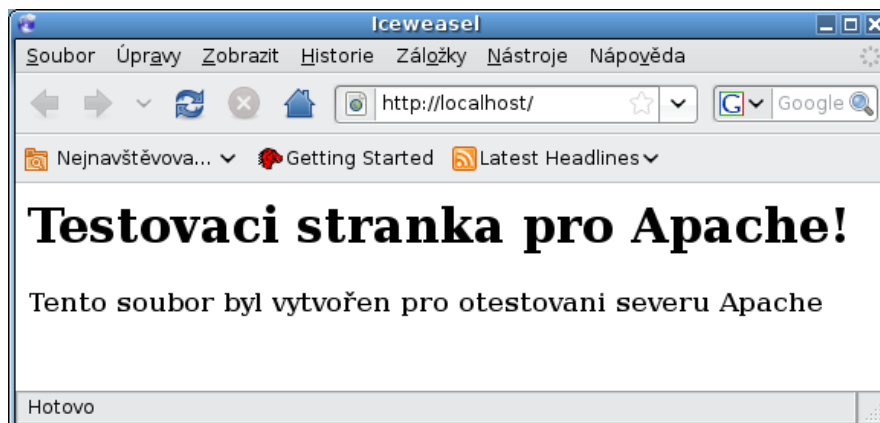
Tento zápis říká, že server obslouží 10000 požadavků, kdy se bude připojovat 500 uživatelů ve stejný okamžik. V příkazu je uvedena adresa, na kterou se přistupuje a přeměření výstupu do souboru *vypis.txt*.

4.6.1. Měření pro plný výkon serveru

VMware Player 4

- **Statický HTML soubor**

Jako *index.html* byl na server nakopírován html soubor, který obsahoval pouze krátké oznámení.



Obrázek 23: Zobrazení souboru index.html.

Zdroj: Vlastní zpracování. 14.3.2012

```

Concurrency Level:      500
Time taken for tests:   6.479 seconds
Complete requests:     10000
Failed requests:       0
write errors:          0
Total transferred:     4464906 bytes
HTML transferred:     1321452 bytes
Requests per second:   1543.44 [#/sec] (mean)
Time per request:      323.951 [ms] (mean)
Time per request:      0.648 [ms] (mean, across all concurrent requests)
Transfer rate:         672.98 [kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:    0   73  447.8     3   3009
Processing: 0  124  533.2    50   6394
waiting:    0  122  533.3    48   6390
Total:      41  197  791.3    52   6465

Percentage of the requests served within a certain time (ms)
 50%    52
 66%    54
 75%    55
 80%    57
 90%    70
 95%   123
 98%   3065
 99%   6005
100%   6465 (longest request)
    
```

Tabulka 2: Výpis hodnot programu ab pro HTML soubor; VMware.

Zdroj: Vlastní zpracování. 14.3.2012

Z tabulky 2 je patrné, že celkový čas, který server potřeboval na zpracování všech požadavků je 6,479 vteřiny, za jednu sekundu byl tedy schopen zpracovat 1543 požadavků. Průměrný čas na zpracování jednoho požadavku činil 0,648 ms. V části Connection Times je uvedeno, kolik času bylo potřeba na jednotlivé fáze zpracování. Fáze connect určuje, kolik času zabralo připojení na požadovanou adresu, processing určuje, jak dlouho trvalo zpracování, waiting udává čas, než se přenese první byte a total je součet connect a processing.

V části Percentage of the requests served within certain time je uvedeno, kolik procent požadavků bylo vyřízeno v určitém čase.

- **Statický HTML soubor formátovaný pomocí CSS stylů**

Soubor *index.html*, který byl nakopírován na server je zobrazen na obrázku 22. Tento soubor obsahuje 3 odstavce textu, menu a obrázky.

```

Concurrency Level:      500
Time taken for tests:   6.067 seconds
Complete requests:     10000
Failed requests:       0
Write errors:          0
Total transferred:     33089842 bytes
HTML transferred:      29927946 bytes
Requests per second:   1648.29 [#/sec] (mean)
Time per request:      303.345 [ms] (mean)
Time per request:      0.607 [ms] (mean, across all concurrent requests)
Transfer rate:         5326.33 [Kbytes/sec] received

Connection Times (ms)
      min  mean[+/-sd] median  max
Connect:    3    77 431.1     13   3025
Processing:  9    76 148.8     60   1794
Waiting:    2    65 149.7     49   1775
Total:      45   153 530.6     73   4449

Percentage of the requests served within a certain time (ms)
 50%    73
 66%    75
 75%    76
 80%    77
 90%    82
 95%    92
 98%   3017
 99%   3098
100%   4449 (longest request)

```

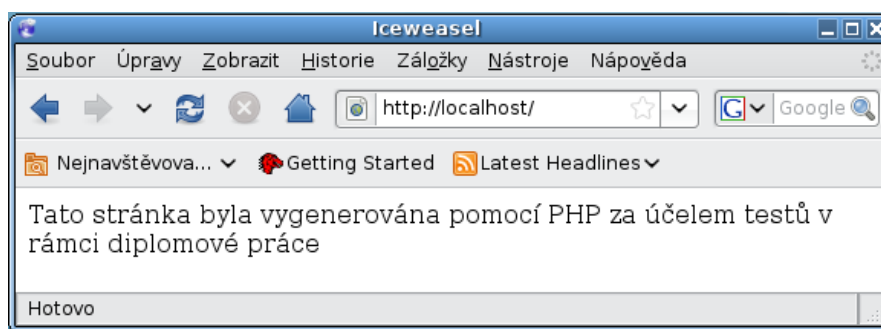
Tabulka 3: Výpis hodnot programu ab pro HTML soubor s CSS styly; VMware.

Zdroj: Vlastní zpracování. 14.3.2012

Tabulka 3 říká, že celkový čas potřebný na vyřízení všech požadavků byl 6,067 vteřiny. Tento údaj je v podstatě shodný s předchozím měřením souboru bez CSS stylů. To potvrdilo předpoklady měření, že na stylpis by program *ab* neměl brát zřetel. Za vteřinu bylo zpracováno 1648 požadavků. Zpracovat jeden požadavek trvalo průměrně 0,607 ms. Z části Connection times vyplývá, že otevření portu a připojení trvalo 3025 ms a zpracování trvalo 1794 ms. V Percentage of the requests je uvedeno, jaký čas je potřeba na zpracování 50 až 100 procent požadavků. Rozdíl ve velikosti souborů je vidět na množství přenesených dat. U tohoto souboru byl objem přenesených dat 5326 Kbytu, zatímco u předchozího pouze 672 Kbytu.

- **Jednoduchý PHP soubor**

Z PHP souboru je dynamicky generována stránka, která je zobrazena na obrázku 24. Pro tento test byl nainstalován a povolen modul třetích stran *mod_php*.



Obrázek 24: Dynamicky vygenerovaná stránka pomocí PHP.

Zdroj: Vlastní zpracování. 14.3.2012

Celkový čas, který server strávil obsluhou všech požadavků, byl 10,892 vteřiny. Na první pohled je patrný rozdíl v času zpracování HTML a PHP souborů. Rozdíl je způsoben tím, že server musí ze souboru načíst skript, následně provést a klientovi odeslat výsledek zpracování. Za vteřinu tedy bylo zpracováno 918 požadavků na prostředek. Průměrný čas na zpracování jednoho požadavku byl 1,089 ms. Objem přenesených dat činil 295 Kbytu. To bylo nejméně ze všech testovaných souborů. V části Percentage of the requests je uvedeno, že 50 procent požadavků bylo obslouženo za 97 ms, 90 procent požadavků bylo obslouženo za 151 ms a 100 procent požadavků bylo obslouženo za 9054 ms.

```

Concurrency Level:      500
Time taken for tests:   10.892 seconds
Complete requests:     10000
Failed requests:       0
Write errors:          0
Total transferred:     3290000 bytes
HTML transferred:     810000 bytes
Requests per second:   918.14 [#/sec] (mean)
Time per request:      544.581 [ms] (mean)
Time per request:      1.089 [ms] (mean, across all concurrent requests)
Transfer rate:         294.99 [kbytes/sec] received

Connection Times (ms)
      min      mean[+/-sd] median    max
Connect:    0    155 892.9      4    9023
Processing:  1    197 642.1     90    6585
Waiting:    0    194 641.4     86    6585
Total:      67    352 1187.4    97    9054

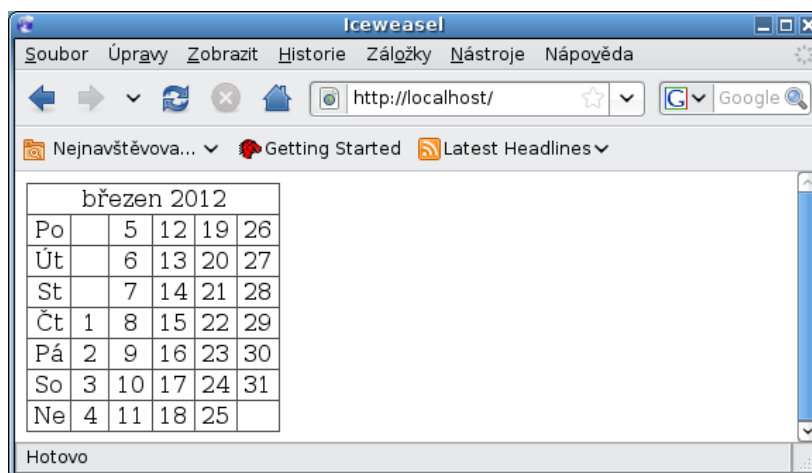
Percentage of the requests served within a certain time (ms)
 50%    97
 66%   115
 75%   125
 80%   130
 90%   151
 95%  1636
 98%  5907
 99%  6840
100%  9054 (longest request)
    
```

Tabulka 4: Výpis hodnot programu ab pro PHP soubor; VMware.

Zdroj: Vlastní zpracování. 14.3.2012

- **PHP soubor zobrazující na stránkách kalendář**

Soubor byl vytvořen v prostředí PS Pad za použití návodu na internetových stránkách [23]. Zdrojový kód tohoto souboru je uveden jako příloha 2.



Obrázek 25: Dynamicky vygenerovaná stránka s kalendářem pomocí PHP.

Zdroj: Vlastní zpracování. 14.3.2012

```

Concurrency Level:      500
Time taken for tests:   19.353 seconds
Complete requests:     10000
Failed requests:       0
Write errors:          0
Total transferred:     15014503 bytes
HTML transferred:     12513753 bytes
Requests per second:   516.71 [#/sec] (mean)
Time per request:      967.667 [ms] (mean)
Time per request:      1.935 [ms] (mean, across all concurrent requests)
Transfer rate:         757.63 [Kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:    0  193 980.8      6   9074
Processing:  1  504 1508.4    146  14764
Waiting:    0  494 1505.5    142  14763
Total:      93  697 2004.4    157  19300

Percentage of the requests served within a certain time (ms)
 50%    157
 66%    190
 75%    255
 80%    306
 90%    711
 95%   3538
 98%   9030
 99%  10210
100%  19300 (longest request)

```

Tabulka 5: Výpis hodnot programu ab pro PHP soubor s kalendářem; VMware.

Zdroj: Vlastní zpracování. 14.3.2012

Z tabulky 5 vyplývá, že ke zpracování všech požadavků byl potřeba dvakrát delší čas než u souboru, který generoval pouze jednoduchý text. Čas potřebný pro zpracování požadavků byl 19,3 vteřiny. Nebyly zaznamenány žádné chybné požadavky ani chyby zápisu. Průměrný čas na zpracování jednoho požadavky byl 1,935 ms. Za jednu vteřinu byl server schopen zpracovat 516 požadavků. Objem přenesených dat činil 757 Kbytů. 50 procent požadavků bylo zpracováno za 157 ms, 90 procent požadavků bylo zpracováno za 711 ms a 100 procent za 19,3 vteřiny.

Oracle Virtual Box 4

Pro tento virtuální stroj byla provedena stejná měření jako pro VMware Player 4.

- **Statický HTML soubor**

Soubor *index.html* je stejný jako při testování VMware. Jak se soubor zobrazí, je ukázáno na obrázku 23.

```

Concurrency Level:      500
Time taken for tests:   35.529 seconds
Complete requests:     10000
Failed requests:       0
Write errors:           0
Total transferred:     5090000 bytes
HTML transferred:      1380000 bytes
Requests per second:   281.46 [#/sec] (mean)
Time per request:      1776.458 [ms] (mean)
Time per request:      3.553 [ms] (mean, across all concurrent requests)
Transfer rate:         139.90 [Kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:    0   508 1521.5   190  21277
Processing:  1  1077 1705.9   580  15768
Waiting:    0   937 1710.9   416  15729
Total:      156 1585 2351.5   827  21996

Percentage of the requests served within a certain time (ms)
 50%    827
 66%    905
 75%    965
 80%   1026
 90%   4015
 95%   6797
 98%   9953
 99%  11418
100%  21996 (longest request)

```

Tabulka 6: Výpis hodnot programu ab pro HTML soubor; Virtual Box.

Zdroj: Vlastní zpracování. 14.3.2012

Oba dva virtuální stroje měly nastaveny shodné parametry výkonu. Webový server Apache měl shodný konfigurační soubor. Rozdíl ve výkonu serveru ve Virtual Boxu je však markantní. Z tabulky 6 je vidět, že celkový čas na zpracování všech požadavků byl 35,5 vteřiny. To je hodnota téměř šestkrát větší než v případě produktu VMware. Průměrný čas potřebný na zpracování jednoho požadavku je zde 3,5 ms. Tato hodnota je opět téměř šestkrát větší než v případě konkurenčního produktu. V tomto případě velkou roli sehrála hodnota waiting, která byla 15 vteřin. 50 procent požadavků bylo zpracováno za 827 ms, 90 procent požadavků bylo zpracováno za 4,015 vteřiny a 100 procent požadavků za 21,996 vteřiny.

- **Statický HTML soubor formátovaný pomocí CSS stylů**

Jedná se opět o stejný soubor *index.html*, který je zobrazen na obrázku 22.

```

Concurrency Level:      500
Time taken for tests:   36.784 seconds
Complete requests:     10000
Failed requests:       0
write errors:          0
Total transferred:     33640000 bytes
HTML transferred:     29910000 bytes
Requests per second:   271.86 [#/sec] (mean)
Time per request:      1839.183 [ms] (mean)
Time per request:      3.678 [ms] (mean, across all concurrent requests)
Transfer rate:         893.10 [Kbytes/sec] received

Connection Times (ms)
      min  mean[+/-sd] median  max
Connect:    0   323 1522.2    62   21130
Processing: 12 1277 2384.9   489   19518
Waiting:    0 1223 2373.1   424   19478
Total:      65 1600 3148.1   620   32258

Percentage of the requests served within a certain time (ms)
 50%    620
 66%    743
 75%    808
 80%    904
 90%   5280
 95%   7693
 98%  11081
 99%  15651
100% 32258 (longest request)

```

Tabulka 7: Výpis hodnot programu ab pro HTML soubor s CSS styly; Virtual Box.

Zdroj: Vlastní zpracování. 14.3.2012

Tabulka 7 říká, že celkový čas na obslužení všech požadavků, který server potřeboval, byl 36,78 vteřiny. Server zpracoval zaokrouhleně 272 požadavků za vteřinu s průměrným časem na jeden požadavek 3,678 ms. Tyto hodnoty jsou opět přibližně šestkrát větší než u produktu od VMware. Při měření se potvrdil stejný předpoklad o měření, že čas potřebný na zpracování HTML souboru, který je formátován pomocí kaskádových stylů a obsahuje obrázky je stejný, jako u souboru, který formátování neobsahuje.

- **Jednoduchý PHP soubor**

Dynamicky vygenerovaná stránka ze souboru *index.php* je zobrazena na obrázku 24. Jedná se o soubor, který generuje pouze jednoduché oznámení.

```

Concurrency Level:      500
Time taken for tests:   45.243 seconds
Complete requests:     10000
Failed requests:       0
Write errors:          0
Total transferred:     3886641 bytes
HTML transferred:     813483 bytes
Requests per second:   221.03 [#/sec] (mean)
Time per request:      2262.154 [ms] (mean)
Time per request:      4.524 [ms] (mean, across all concurrent requests)
Transfer rate:         83.89 [kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:    0  352 1846.9    89  45009
Processing:  4 1561 3179.2   662  33165
Waiting:    0 1471 3169.2   540  32955
Total:      98 1913 3891.5   884  45183

Percentage of the requests served within a certain time (ms)
 50%    884
 66%   1047
 75%   1161
 80%   1220
 90%   5223
 95%   9835
 98%  12879
 99%  13700
100%  45183 (longest request)

```

Tabulka 8: Výpis hodnot programu ab pro PHP soubor; Virtual Box.

Zdroj: Vlastní zpracování. 14.3.2012

Při zpracování všech požadavků na generování stránky pomocí skriptu PHP server potřeboval celkový čas 45 vteřin. Průměrný čas potřebný na zpracování jednoho požadavku činí 4,524 ms. Server byl schopen zpracovat 221 požadavků za vteřinu. Zvýšení celkového času na zpracování požadavků pomocí modulu PHP je způsobeno tím, že server musí skript zpracovat a následně odeslat. Avšak oproti produktu VMware je zpracování 4,5 krát pomalejší.

- **PHP soubor zobrazující na stránkách kalendář**

Vygenerovaný kalendář je zobrazen na obrázku 25. Z tabulky 9 je patrné, že celkový čas na zpracování všech požadavků činil 64,232 vteřiny. Tento čas je třikrát delší, než v případě použití VMware Player 4. Za vteřinu tedy bylo vyřízeno 155 požadavků. Průměrný čas na zpracování jednoho požadavku je 6,423 ms. Při zatěžování serveru nebyl zjištěn žádný chybný požadavek, ani chyba zápisu. Padesát procent požadavků bylo vyřízeno za 1095 ms, devadesát procent požadavků bylo vyřízeno za 9,991 vteřiny a sto procent požadavků za 43,984 vteřiny. Podle předpokladu byl nejsložitější skript prováděn nejdelší dobu.

```

Concurrency Level:      500
Time taken for tests:   64.232 seconds
Complete requests:     10000
Failed requests:       0
Write errors:          0
Total transferred:     15590000 bytes
HTML transferred:     12510000 bytes
Requests per second:   155.68 [#/sec] (mean)
Time per request:      3211.615 [ms] (mean)
Time per request:      6.423 [ms] (mean, across all concurrent requests)
Transfer rate:         237.02 [kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:    0  497 1271.6   167  21008
Processing: 159 2359 4141.5   799  43960
Waiting:    14 2251 4141.8   645  43960
Total:      159 2856 4482.0  1095  43984

Percentage of the requests served within a certain time (ms)
 50%    1095
 66%    1263
 75%    1825
 80%    4050
 90%    9991
 95%   12731
 98%   15724
 99%   20528
100%   43984 (longest request)

```

Tabulka 9: Výpis hodnot programu ab pro PHP soubor s kalendářem; Virtual Box.

Zdroj: Vlastní zpracování. 14.3.2012

4.6.2. Měření pro snížený výkon serveru

Na obou virtuálních strojích byla nastavena možnost používání pouze jednoho jádra procesoru a byla snížena operační paměť z 1024 MB na 512 MB. Vzhledem k naměřeným hodnotám všech souborů, které odpovídaly předpokladům, že výkon serveru bude nižší, budouv další části uvedeny výsledky pouze pro soubor PHP, který na stránce generuje kalendář.

VMware Player 4

- **PHP soubor zobrazující na stránkách kalendář**

V tabulce 10 je uvedeno, že celkový čas potřebný na zpracování všech požadavků na zobrazení kalendáře na stránce byl 21,242 vteřiny. Tato hodnota je nižší, než bylo očekáváno. Při využití všech zdrojů serveru bylo dosaženo času 19,353 vteřiny. Rozdíl činí pouze 2 vteřiny, při snížení systémových prostředků na polovinu. Z toho plyne, že velikost operační paměti 512 MB a pouze jedno jádro procesoru není pro webový server Apache,

nainstalovaný na virtuálním stroji VMware Player 4, zásadním omezením. Server celkem zpracoval 470 požadavků za vteřinu.

```

Concurrency Level:      500
Time taken for tests:   21.242 seconds
Complete requests:     10000
Failed requests:       0
Write errors:          0
Total transferred:     15074543 bytes
HTML transferred:     12563793 bytes
Requests per second:   470.76 [#/sec] (mean)
Time per request:      1062.107 [ms] (mean)
Time per request:      2.124 [ms] (mean, across all concurrent requests)
Transfer rate:         693.02 [Kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:    0   216 1353.1    16  21034
Processing: 3   600 1542.0   247  13488
Waiting:    0   576 1538.2   221  13396
Total:      170  816 2106.9   303  21106

Percentage of the requests served within a certain time (ms)
 50%    303
 66%    331
 75%    378
 80%    442
 90%    619
 95%   4128
 98%   8330
 99%  13159
100%  21106 (longest request)

```

Tabulka 10: Výpis programu ab pro zobrazení PHP souboru s kalendářem; VMware.

Zdroj: Vlastní zpracování. 14.3.2012

Průměrný čas na zpracování jednoho požadavku činil 2,124 ms. Při testování tohoto souboru nebyl zjištěn žádný chybný požadavek a nedošlo k žádné chybě při zápisu. Celkový objem přenesených dat byl 693 Kbytu.

U ostatních souborů, které byly testovány, byl naměřen podobný pokles výkonnosti jako v tomto případě. Tento pokles byl v řádu 10 – 20 % oproti předchozímu nastavení virtuálního stroje.

Oracle Virtual Box 4

- **PHP soubor zobrazující na stránkách kalendář**

Jak je uvedeno v tabulce 11, celkový čas potřebný na zpracování všech požadavků činil 71,094 vteřiny. Tento čas je o 7 vteřin delší, než při plném výkonu serveru. Tato ztráta odpovídá přibližně stejnému procentu jako u VMwaru, avšak tento virtuální stroj byl schopen všechny požadavky zpracovat tři a půl krát rychleji. Tento rozdíl může být

způsoben nároky na výkon fyzického stroje nebo virtuální stroj není plně přizpůsoben pro používání operačního systému Linux Debian 5. Toto je však daň za možnost používání Virtual Boxu zdarma. Při sledování využití prostředků na hostitelském fyzickém počítači nebylo pozorováno, při spuštění Virtual Boxu, nadměrné přetěžování procesoru nebo nedostatek operační paměti. Proto je pravděpodobnější varianta špatných výsledků testů výkonu webového serveru špatná podpora operačního systému Linux Debian 5 virtuálním strojem.

```

Concurrency Level:      500
Time taken for tests:   71.094 seconds
Complete requests:     10000
Failed requests:       0
write errors:          0
Total transferred:     15591559 bytes
HTML transferred:     12511251 bytes
Requests per second:   140.66 [#/sec] (mean)
Time per request:      3554.713 [ms] (mean)
Time per request:      7.109 [ms] (mean, across all concurrent requests)
Transfer rate:         214.17 [Kbytes/sec] received

Connection Times (ms)
      min    mean[+/-sd] median    max
Connect:    0   470 1950.5    124   21286
Processing:  3  2724 4017.2    968   49764
Waiting:    0  2647 4005.1    886   49764
Total:      277 3194 4789.8   1241  71049

Percentage of the requests served within a certain time (ms)
 50%    1241
 66%    1613
 75%    2336
 80%    5027
 90%    9855
 95%   12558
 98%   15850
 99%   22622
100%   71049 (longest request)

```

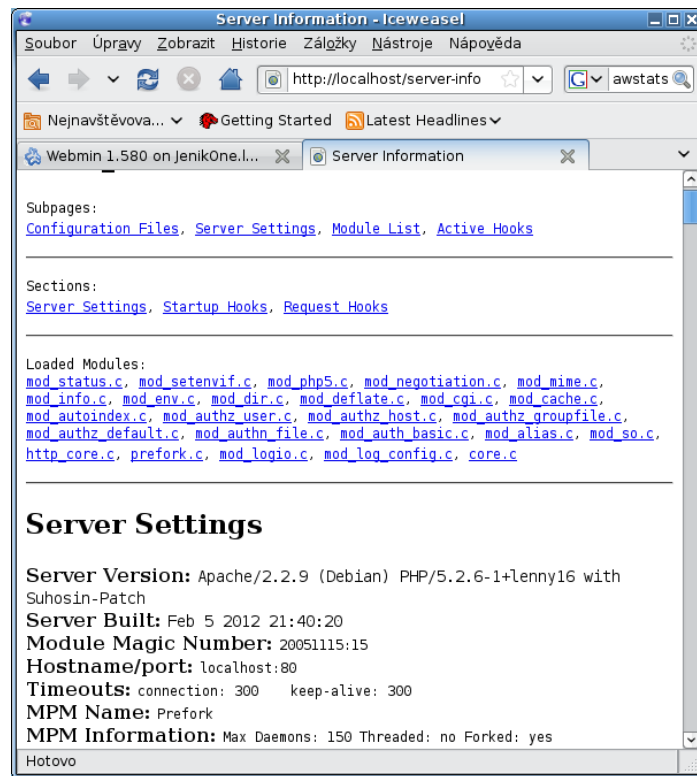
Tabulka 11: Výpis programu ab pro zobrazení PHP souboru s kalendářem; Virtual Box.

Zdroj: Vlastní zpracování. 14.3.2012

Server byl schopný zpracovat 140 požadavků za vteřinu a průměrný čas na zpracování jednoho požadavku se pohyboval kolem 7 ms. Dále byly testovány dva statické HTML soubory a jeden jednoduchý PHP skript. Výsledky u těchto souborů měly pokles v rozsahu 10 – 20 % stejně jako VMwaru, avšak v přímém porovnání byly opět 4 – 6 krát horší.

4.7. Sledování serveru Apache

Pro sledování serveru Apache se používají vestavěné statistiky. Tyto statistiky se zobrazí, pokud se do adresního řádku napíše za doménu příkaz `/server-status` nebo `/server-info`. Tato funkce je však přístupná pouze pokud jsou povoleny moduly `status` a `info`.



Obrázek 26: Zobrazení nastavení pomocí Server-info.

Zdroj: Vlastní zpracování. 20.3.2012

Výpis `server-info` je uveden na obrázku 26. Je zde vidět, jaké jsou zavedeny moduly a je možné odsud přecházet do různých sekcí. Je možné si zobrazit konfigurační soubory, nastavení serveru a celkový soupis modulů.

Pokud se použije příkaz `server-status`, objeví se výpis, který je uveden na obrázku 27. Z tohoto výpisu je vidět, jaká verze serveru Apache je nainstalována a kdy byl server sestaven. Níže jsou uvedeny statistiky provozu, jako od kdy se statistika počítá, celkový počet přístupů, objem přenesených dat a jak byl využit procesor. Při požadavku na zobrazování úplných statistik, musí být ještě do konfiguračního souboru `httpd.conf` zapsána direktiva `ExtendedStatus On`, která zobrazuje podrobnější výpis obsahující ještě informace o jednotlivých procesech. Statistika jsou ztvárněny i textově. V místech, kde je zobrazen symbol „_“, čekal server na spojení, kde je zobrazen symbol „W“, zasílal server odpověď.



Obrázek 27: Statistika serveru.

Zdroj: Vlastní zpracování. 20.3.2012

Pokud se mají statistiky pravidelně obnovovat, zapíše se do adresního řádku <http://localhost/server-status?refresh=20>. Tento příkaz znamená, že se statistiky obnoví každých 20 vteřin.

Pro sledování systému je také důležité kontrolovat soubor *error.log*. Do tohoto souboru se ukládají chyby, které nastaly při běhu serveru Apache. Umístění tohoto souboru je uvedeno v konfiguračním souboru pod direktivou *ErrorLog /var/log/apache2/errorlog*. Pokaždé, když nastanou nějaké chyby nebo server nefunguje korektně, jsou v tomto souboru k nalezení odpovědi, se kterými komponentami problémy jsou. Na uvedeném umístění se také nachází soubor *access.log*, do kterého se zapisují všechny přístupy k serveru.

Výpis souboru *error.log* je uveden jako příloha 3.

4.7.1. Program ApacheTop

Pro sledování provozu webového serveru Apache je také možné použít program ApacheTop. Tento nástroj sleduje požadavky na prostředky v reálném čase a zobrazuje je na displeji. Hlavní výhodou spočívá v použití příkazového řádku, v předchozím případě bylo nutné použít internetový prohlížeč. Tento nástroj je v operačním systému Linux Debian 5 možné nainstalovat pomocí správce balíčků. Použije se příkaz `aptitude install apachetop`. Jako zdrojový soubor tento nástroj bere `acces.log`.

Pro zatížení serveru, aby byly vidět změny v reálném čase, je možné použít nástroj Apache Benchmarking, který byl používán pro zjištění výkonnosti serveru. Ve výpisu programu ApacheTop je vidět doba běhu programu, celkový souhrn všech zpracovaných požadavků, průměrný počet požadavků za vteřinu a průměrný datový tok za vteřinu.

```

last hit: 18:21:28      atop runtime: 0 days, 00:23:56      18:21:29
All:      55604 reqs ( 40.3/sec)      11.9M ( 9041.2B/sec)      224.4B/req
2xx:    35578 (64.0%) 3xx:         0 ( 0.0%) 4xx: 20026 (36.0%) 5xx:         0 ( 0.0%)
R ( 29s): 7328 reqs ( 252.7/sec)      986.1K ( 34.0K/sec)      137.8B/req
2xx:    7328 (100%) 3xx:         0 ( 0.0%) 4xx:         0 ( 0.0%) 5xx:         0 ( 0.0%)

REQS REQ/S   KB KB/S URL
7317 252.3 986.1 34.0*/
 11  0.48   0.0  0.0 *

```

Obrázek 28: Výpis programu ApacheTop.

Zdroj: Vlastní zpracování. 24.3.2012

Hodnoty zobrazené jako 2xx, 3xx, 4xx, 5xx označují, pod jakým stavovým kódem webový server Apache na požadavek odpověděl. Testován byl soubor `index.php`, který se na serveru nachází. Z tohoto testu vyplývá zaznamenaná hodnota z obrázku 28, která má hodnotu 35578 požadavků. Je také vidět, že toto číslo bylo 64 procent ze všech požadavků, které server přijal. V druhém kroku byl testován soubor `index.html`, který se na serveru nenachází. Z tohoto testu jsou zaznamenány všechny požadavky, které se nachází za kódem 4xx, který znamená nenalezeno. Bylo změřeno 20026 těchto požadavků, což bylo 36 procent ze všech serverem přijatých požadavků na prostředek. Ovládání programu se provádí pomocí písmen zadaných z klávesnice. Po zadání znaku `h` nebo `?`, se vyvolá nápověda, kde jsou popsány všechny symboly pro ovládání. Ovládání ovšem funguje pouze, pokud server zpracovává nějaké požadavky. Pokud je ve stavu, kdy server

nedělá nic, ani program ApacheTop nereaguje na ovládání přes klávesnici. ApacheTop je možné spustit s různými parametry, podle statistik, které se budou zobrazovat.

- -H se používá pro zobrazení statistik za zadaný počet kliknutí
- -T zobrazuje statistiku za zadaný počet vteřin
- -r udává čas, za který se statistika na displeji bude obnovovat

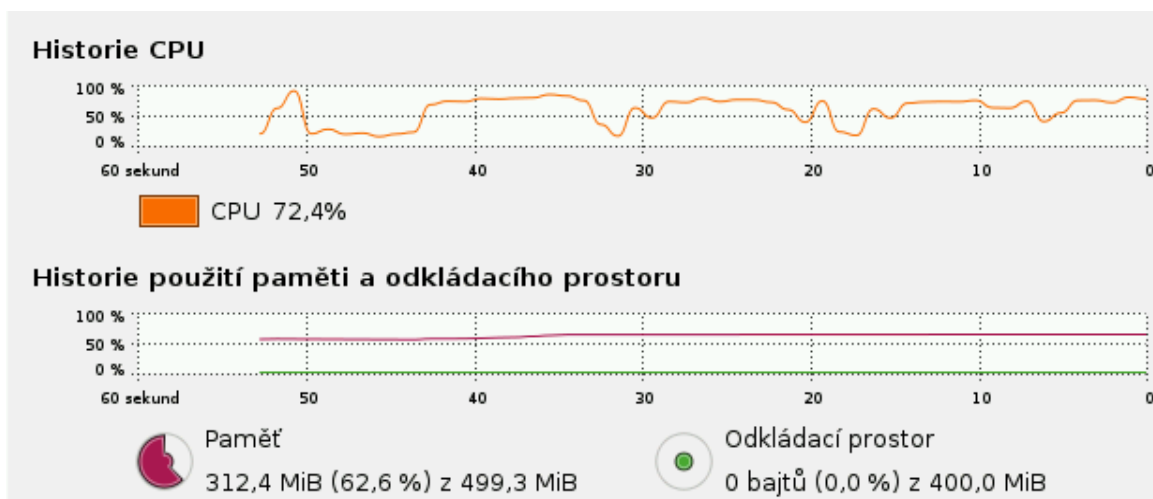
Pro ovládání programu ApacheTop se používají tyto zkratky:

- q pro ukončení činnosti
- h nebo ? pro zobrazení nápovědy
- s pro určení řazení
- f pro určení filtru

Pro použití filtru se stiskne klávesa f, na displeji se zobrazí nabídka, kde je možné přidat nebo editovat stávající filtry, všechny filtry vymazat a zobrazit aktivní filtry. Pokud se přidává nový filtr, je možné vybrat, jestli se bude filtrovat dle URL, *refferers* nebo *hosts*.

5. Zhodnocení výsledků a doporučení

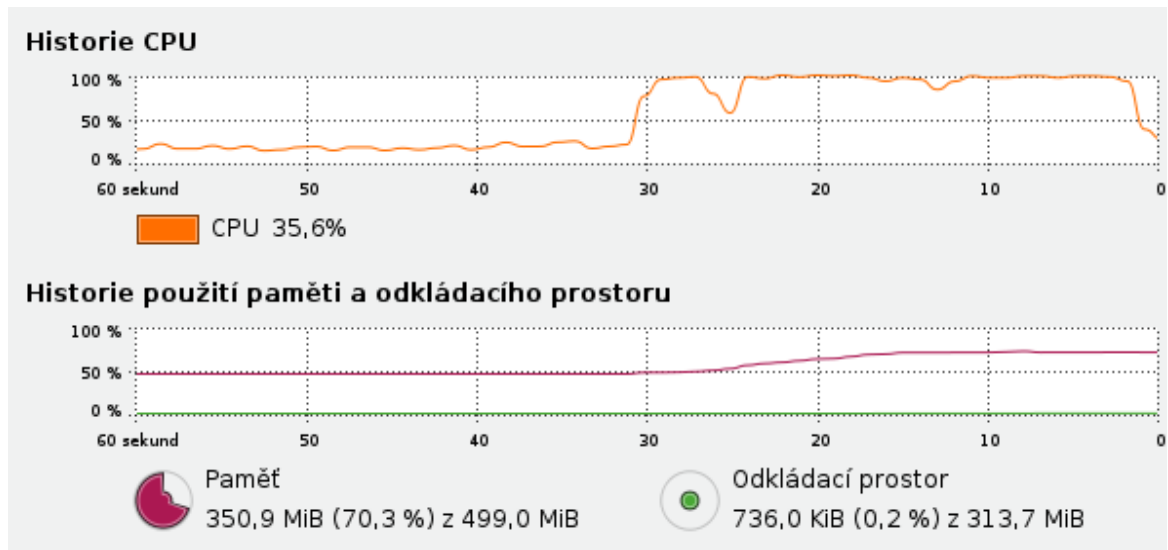
Vzhledem k tomu, že oba dva virtuální stroje byly nastaveny na stejné parametry výkonu, bylo očekáváno, že rozdíly ve výkonu webového serveru Apache se budou pohybovat v řádu jednotek až desítek procent. Z provedených měření však vyplývá, že virtuální stroj Virtual Box od firmy Oracle není plně optimalizovaný pro použitý operační systém. Ačkoliv v nastavení výkonu pro Virtual Box bylo zaškrtnuto pole využívat 100 výkonu fyzického stroje. Fyzický počítač nemohl poskytnout stejný výkon jako pro produkt VMware Player 4. Toto tvrzení je založeno na obrázku 29, na kterém je vidět, že při zpracování požadavků nebyl naplno využit procesor ani operační paměť.



Obrázek 29: Zátěž operačního systému při použití Apache Benchmark pro Virtual Box.

Zdroj: Vlastní zpracování. 24.3.2012

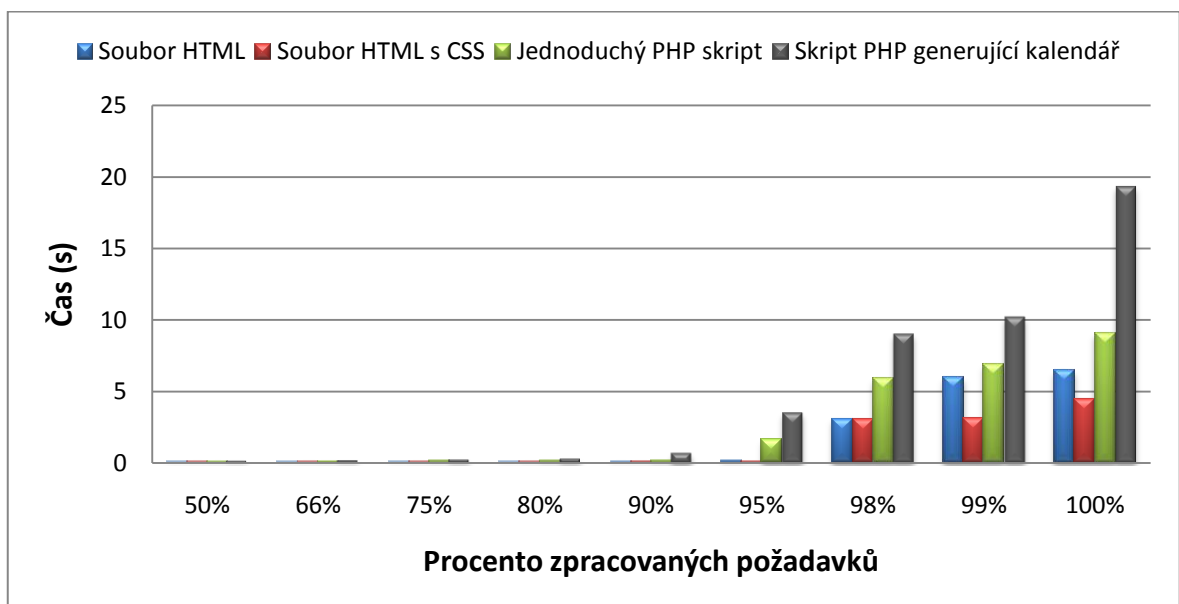
Na obrázku 30 je uvedena zátěž procesoru na operačním systému Linux Debian 5, nainstalovaném ve VMwaru. Na první pohled je vidět, že při spuštění programu Apache Benchmark systém využívá procesor na 100 procent. Operační paměť je také využita více. Zde se používá 350 MB oproti 312 MB na Virtual Boxu. Důvodem toho chování Virtual Boxu by také mohla být špatná práce s pevným diskem při ukládání a čtení dat.



Obrázek 30: Zátěž operačního systému při použití Apache Benchmark pro VMware.

Zdroj: Vlastní zpracování. 24.3.2012

Výsledky měření výkonu jasně vyzněly pro VMware Player 4. Webový server Apache, který byl nainstalován na operačním systému Debian 5, který běžel v tomto virtuálním stroji, poskytoval 4krát až 6krát vyšší výkon v závislosti na tom, jaký prostředek poskytoval. U obou virtuálních strojů byl nejhorší výsledek naměřen při zpracování složitějšího PHP skriptu.



Graf 1: Percentage of the requests served within a certain time (s) pro VMware.

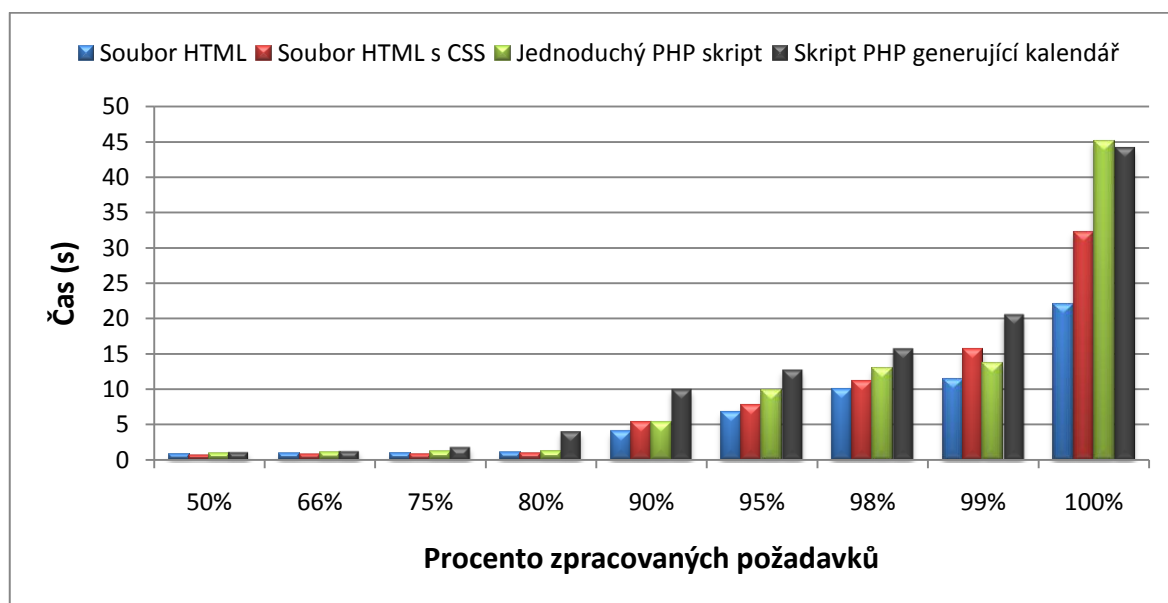
Zdroj: Vlastní zpracování. 24.3.2012

Na grafu 1 jsou uvedeny celkové časy, jaké potřeboval server Apache běžící na virtuálním stroji VMware Player 4 pro zpracování určitého procenta požadavků. V grafu jsou uvedeny časy zpracování všech měřených souborů. Měřeny byly dva HTML soubory a dva soubory PHP.

Na grafu 2 jsou uvedeny stejné charakteristiky jako u grafu 1, ale měřené na serveru Apache, který běžel ve virtuálním stroji Virtual Box.

Pro virtuální stroj VMware bylo nejsložitější zpracovat soubor PHP, který na stránkách zobrazoval kalendář. Toto zpracování, jak vyplývá z grafu 1, trvalo zaokrouhleně 20 vteřin. Zatímco na Virtual Boxu byl tento čas zaokrouhleně 45 vteřin. Důvody toho rozdílu jsou uvedeny v textu výše a vysvětleny pomocí obrázků 29 a 30.

Zajímavým ukazatelem je také vyjádření času zpracování požadavků v procentech. Z grafů je vidět, že 90 procent požadavků je ještě zpracováno ve velmi krátkém čase a zbylých 10 procent znamená velký nárůst času zpracování.

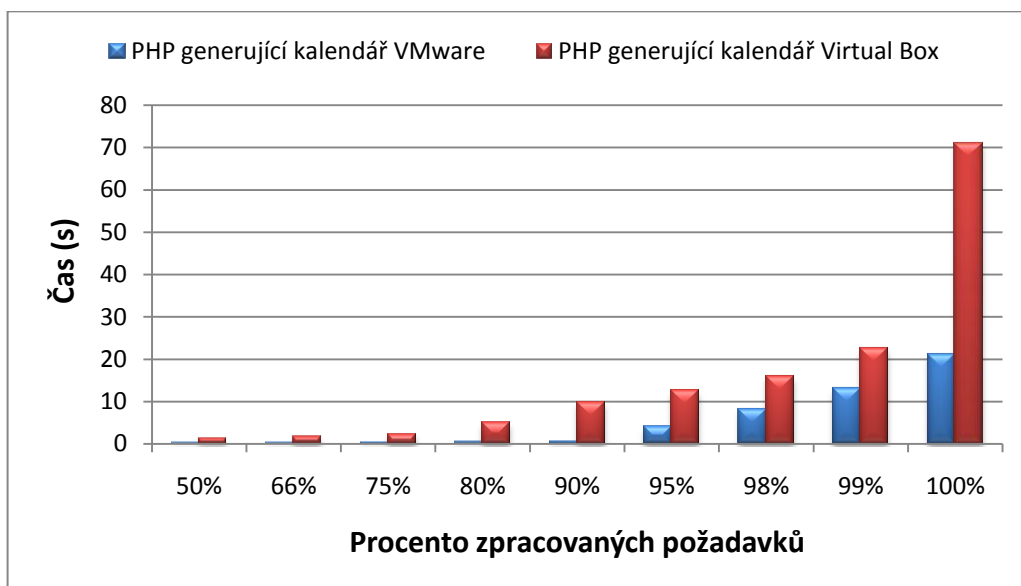


Graf 2: Percentage of the requests served within a certain time (ms) pro Virtual Box.

Zdroj: Vlastní zpracování. 24.3.2012

Na grafu 3 je znázorněn čas ve vteřinách, jaký byl potřeba na zpracování požadavků na PHP soubor, který zobrazuje na stránkách kalendář. Tyto časy byly měřeny při sníženém výkonu serveru na poloviční hodnoty. V tomto nastavení byly rozdíly ještě markantnější, kde hodnota pro VMware byla lehce nad 20 vteřin, což je velmi malý rozdíl

oproti předchozí plné specifikaci. U Virtual Boxu hodnota činila 70 vteřin, kde v předchozí specifikaci bylo toto číslo 45 vteřin. Toto znamenalo velké zhoršení.



Graf 3: Percentage of the requests served within a certain time (ms)VMware vs. Virtual Box pro PHP.

Zdroj: Vlastní zpracování. 24.3.2012

Z výsledků testů vyplývá, že při provozování webového serveru Apache ve virtuálním stroji s operačním systémem Linux Debian 5, bude lepší volbou VMware Virtual Player 4. V tomto virtuálním stroji dosahoval webový server Apache výrazně lepších výsledků při poskytování webových stránek.

Pokud uživatelé nechtějí používat příkazovou řádku, je možné Apache ovládat pomocí grafického nástroje Webmin. Tento nástroj je velmi přehledný a jednoduchý na ovládání.

6. Závěr

V teoretických východiscích byl charakterizován webový server Apache jako celek. Důraz byl kladen na možnosti, jakými lze webový server získat. Je možné server stáhnout jako zdrojový kód, který je určen k ruční kompilaci, jako zkomprimovaný linuxový balíček nebo pomocí balíků systému Debian, za použití funkce `aptitude install`, která balíček stáhne a automaticky nainstaluje.

Pokud se budou poskytovat statické HTML stránky, je server Apache možné použít ihned po instalaci jak byl nainstalován. Do konfiguračního souboru se запиše direktiva *DocumentRoot*, která říká, kde se nacházejí soubory, které se budou uživatelům poskytovat. Na toto místo se nakopíruje soubor, který má název *index.html*. Poskytování statických HTML stránek je ve většině případů nedostačující. Pokud je potřeba poskytovat dynamicky generované stránky pomocí PHP, je nezbytně nutné doinstalovat modul třetích stran *mod_php*, který není implicitně součástí webového serveru Apache. Po instalaci tohoto modulu je již webový server schopen zpracovat skripty PHP. Postup je stejný jako u statických souborů – na místo kam ukazuje direktiva *DocumentRoot* se nakopíruje soubor s názvem *index.php*.

Pro omezení přístupu k prostředkům je možné využít několik způsobů, např.: ověření uživatele pomocí jména a hesla, pomocí certifikátu nebo pomocí databáze. V práci bylo použito ověření uživatelů pomocí jména a hesla. Do části serveru, která není přístupná přes internet, byl umístěn soubor s uživatelskými jmény a hesly, který byl vytvořen pomocí příkazu *htpasswd*. V konfiguračním souboru následně musí být použita direktiva *Require*, pomocí které se určí uživatelé ze souboru s uživatelskými jmény a hesly, kteří budou mít k prostředkům přístup. Aby vše fungovalo, je ještě nutné do konfiguračního souboru zapsat, pomocí jakého modulu se bude ověřovat. Zde byl použit modul *mod_auth*. Omezení libovolného přístupu k prostředkům zajišťuje vyšší bezpečnost dat, které jsou na serveru umístěna výrazně omezuje možnost zneužití těchto dat.

Na fyzický stroj byly úspěšně nainstalovány a spuštěny dva virtuální stroje. Byly použity produkty VMware Player 4 a Oracle Virtual Box 4. Do těchto virtuálních počítačů byl nainstalován operační systém Linux Debian 5 a webový server Apache. Aby byla zajištěna stejná konfigurace webového serveru, byl zkopírován konfigurační soubor z prvního webového serveru a přenesen na druhý. Z měření výkonu vyplynulo, že mnohem

lepší výsledky je schopen poskytovat webový server Apache, který je spuštěn na operačním systému běžícím ve VMware Playeru. Rozdíl v poskytovaném výkonu je velmi markantní, kdy zpracování všech požadavků trvalo ve VMware v průměru čtyřikrát kratší dobu, než v případě Virtual Boxu. Při sníženém výkonu virtuálního stroje opět poskytoval lepší výsledky Apache nainstalovaný ve VMware Playeru. Zajímavé bylo, že VMware reflektoval omezení výkonu ve velmi malém měřítku, kdy se zpracování všech požadavků prodloužilo o 2 vteřiny. Virtual Box zaznamenal zhoršení o 25 vteřin. Toto ukazuje, že ve virtuálních strojích a jejich činnosti existují značné rozdíly. Podle očekávání se výsledné výkony neměly zásadně lišit. Tento rozdíl může mít několik příčin. Např. špatná kompatibilita virtuálního stroje pro použitý operační systém, i když použitý operační systém figuruje mezi podporovanými systémy pro instalaci, špatná práce s operační pamětí nebo pevnými disky. Toto je nevýhoda virtuálních řešení, které jsou zdarma, jelikož není možné nainstalovat jakýkoliv operační systém, ale je nutné vybrat z předdefinované nabídky. Pro měření výkonu byl používán speciální program Apache Benchmarking, který je schopen nasimulovat požadovaný počet požadavků a počet naráz přistupujících uživatelů.

Pro sledování činnosti webového serveru Apache, byly použity vestavěné statistiky, které se zobrazují pomocí webového prohlížeče po aktivaci modulů *mod_info* a *mod_status*. První jmenovaný modul zobrazuje nastavení webového serveru, druhý je zodpovědný za zobrazování statistik provozu. Pokud mají být ve statistikách sledovány i procesy, musí se v konfiguračním souboru uvést direktiva *ExtendedStatus On*, která tuto funkcionalitu zajistí. Tento způsob má nevýhodu v použití přes webový prohlížeč, pokud je k dispozici pouze příkazová řádka, je nutné použít jiné řešení. Pomocí balíčkovacího systému byl nainstalován program ApacheTop, který po spuštění přistupuje k souboru *access.log* a zobrazuje charakteristiky zpracovávaných požadavků v reálném čase. Tento program je schopen roztrždit požadavky podle stavového kódu odpovědi. Statistika provozu jsou velmi důležitými informacemi pro správce serveru. Přinášejí mu povědomí o počtu přístupů na webové stránky. Pokud správce zaznamená přetěžování serveru, má možnost změnit konfiguraci nebo se ujistit, že bude nutné posílit hardware nebo změnit výkonostní parametry virtuálního stroje.

Instalace grafického nástroje pro ovládání serveru byla provedena ze staženého balíčku *.deb* ručně. Na první pokus instalace selhala kvůli chybějícím balíkům vytvářejícím

závislostí. Po doinstalování šesti chybějících balíků vše proběhlo, jak mělo. Nástroj Webmin se spustil pomocí webového prohlížeče, zadáním do adresního řádku jména počítače a portu. Tento nástroj je velmi přehledný a umožňuje konfiguraci serveru bez použití příkazové řádky. Je možné upravovat konfigurační soubory serveru Apache, aktivovat a deaktivovat moduly nebo vytvářet virtuální hostitele. Další důležitou funkcí je správa uživatelů webového serveru. Je zde možné nastavit, jako jaký uživatel se webový sever spustí a s jakými právy. Mezi obecné vlastnosti tohoto nástroje patří propracované funkce zálohování podle data a času, možnost server vypnout nebo restartovat na dálku a nahrávat na server nebo z něj stahovat soubory. Jedná se o komplexní správu serveru, při které nemusí být správce u serveru fyzicky přítomen a může vše udělat na dálku.

7. Seznam použitých zdrojů

7.1. Seznam tabulek

Tabulka 1: Výpis souboru apachepasswd.txt.....	45
Tabulka 2: Výpis hodnot programu ab pro HTML soubor; VMware.	51
Tabulka 3: Výpis hodnot programu ab pro HTML soubor s CSS styly; VMware.	52
Tabulka 4: Výpis hodnot programu ab pro PHP soubor; VMware.	54
Tabulka 5: Výpis hodnot programu ab pro PHP soubor s kalendářem; VMware.	55
Tabulka 6: Výpis hodnot programu ab pro HTML soubor; Virtual Box.....	56
Tabulka 7: Výpis hodnot programu ab pro HTML soubor s CSS styly; Virtual Box.	57
Tabulka 8: Výpis hodnot programu ab pro PHP soubor; Virtual Box.....	58
Tabulka 9: Výpis hodnot programu ab pro PHP soubor s kalendářem; Virtual Box.	59
Tabulka 10: Výpis programu ab pro zobrazení PHP souboru s kalendářem; VMware.....	60
Tabulka 11: Výpis programu ab pro zobrazení PHP souboru s kalendářem; Virtual Box. .	61

7.2. Seznam obrázků

Obrázek 1: Výpis programu telnet pro požadavek GET / HTTP/1.0.	14
Obrázek 2: Výpis programu telnet pro požadavek GET / HTTP/1.1	15
Obrázek 3: Distribuce zdrojového kódu Apache.	19
Obrázek 4: Rozdíl mezi tradiční architekturou a virtuální architekturou.	34
Obrázek 5: Možnosti instalace Virtual Boxu.	37
Obrázek 6: Varování při instalaci Virtual Boxu.	37
Obrázek 7: Program Virtual Box bez nainstalovaného OS.	38
Obrázek 8: Okno pro výběr typu operačního systému.	38
Obrázek 9: Nastavení velikosti operační paměti.	39
Obrázek 10: Určení umístění virtuálního pevného disku.	40
Obrázek 11: Instalace OS Debian 5.	40
Obrázek 12: Prostředí virtuálního stroje Virtual Box a možnosti nastavení.	41
Obrázek 13: Soubor index.html	42
Obrázek 14: Chyba při zobrazení obsahu složky.	42
Obrázek 15: Výpis obsahu složky - základní nastavení.	43
Obrázek 16: Výpis obsahu složky pomocí speciálních parametrů.	44
Obrázek 17: Zobrazení index.html před aplikací omezení.	44
Obrázek 18: Ověřovací formulář.	46
Obrázek 19: Odmítnutí přístupu ke stránce.	46
Obrázek 20: Webmin - grafický nástroj pro ovládání serveru.	47
Obrázek 21: Soupis nainstalovaných modulů v prostředí Webmin.	49
Obrázek 22: Ukázka HTML souboru s použitím CSS stylů.	50
Obrázek 23: Zobrazení souboru index.html.	51
Obrázek 24: Dynamicky vygenerovaná stránka pomocí PHP.	53
Obrázek 25: Dynamicky vygenerovaná stránka s kalendářem pomocí PHP.	54
Obrázek 26: Zobrazení nastavení pomocí Server-info.	62
Obrázek 27: Statistika serveru.	63
Obrázek 28: Výpis programu ApacheTop.	64
Obrázek 29: Zátěž operačního systému při použití Apache Benchmark pro Virtual Box. .	66
Obrázek 30: Zátěž operačního systému při použití Apache Benchmark pro VMware.	67

7.3. Seznam grafů

Graf 1: Percentage of the requests served within a certain time (s) pro VMware.....	67
Graf 2: Percentage of the requests served within a certain time (ms) pro Virtual Box.....	68
Graf 3: Percentage of the requests served within a certain time (ms)VMware vs. Virtual Box pro PHP.....	69

7.4. Literatura

Tištěné zdroje

1. AULDS, Charles. *Linux: Administrace serveru Apache*. první. Praha: Grada, 2003, 536 s. ISBN 80-247-0640-7.
2. BRÁZA, Jiří. *PHP 5: Začínáme programovat*. První. Praha: Grada, 2005, 244 s. ISBN 80-247-1146-X.
3. KABIR, Mohammed J. *Apache server 2: Kompletní příručka administrátora*. První. Brno: Computer Press, 2004. ISBN 80-251-0319-6.
4. LAURIE, Ben a Peter LAURIE. *Apache: Správa webového serveru*. První. Praha: Computer Press, 1997, 247 s. ISBN 80-7226-043-X.
5. MOBILY, Tony. *Hardening Apache*. Berkley: Apress, 2004, 270 s. ISBN 1-59059-378-2.
6. POŠMURA, Vlastimil. *Apache: Příručka správce www serveru*. první. Praha: Cpress, 2002, 311 s. ISBN 80-7226-696-9.
7. RAY, Deborah S. a RAY, Eric J. *Unix: Podrobný průvodce*. První. Praha: Grada, 2009, 416 s. ISBN 978-80-247-2125-5.
8. ROSEBROCK, Eric a Eric FILSON. *Linux, Apache, MySQL a PHP: Instalace a konfigurace prostředí pro pokročilé webové aplikace*. první. Praha: Grada, 2005, 344 s. ISBN 80-247-1260-1.

Elektronické zdroje

9. APACHE SOFTWARE FOUNDATION. *Apache HTTP Server Version 2.0: ab - Apache HTTP server benchmarking tool* [online]. 2011 [cit. 2012-03-14]. Dostupné z: <http://httpd.apache.org/docs/2.0/programs/ab.html>
10. APACHE SOFTWARE FOUNDATION. *Apache HTTP Server Version 2.0: configure - Configure the source tree* [online]. 2011 [cit. 2012-03-11]. Dostupné z: <http://httpd.apache.org/docs/2.0/programs/configure.html#installationdirectories>
11. APACHE SOFTWARE FOUNDATION. *Apache License, Version 2.0* [online]. leden 2004 [cit. 2012-03-30]. Dostupné z: <http://www.apache.org/licenses/LICENSE-2.0.html>
12. ČECH, Nikola. *VMware, Virtual PC, Parallels nebo VirtualBox?* [online]. 15.6.2007 [cit. 2012-03-16]. Dostupné z: <http://www.emag.cz/vmware-virtual-pc-parallels-nebo-virtualbox/>

13. DOČEKAL, Michal. *Správa linuxového serveru: Thttpd a benchmark webového serveru* [online]. 3.5.2011 [cit. 2012-03-14]. Dostupné z: <http://www.linuxexpres.cz/praxe/sprava-linuxoveho-serveru-thttpd-a-benchmark-weboveho>
14. HÁJEK, Jan. *Virtualizace: mýtus, kouzlo, hype nebo realita?* [online]. 28.11.2007 [cit. 2012-03-16]. Dostupné z: <http://interval.cz/clanky/virtualizace-mytus-kouzlo-hype-nebo-realita/>
15. JAKEL, Milan. Stavové kódy a hlášení v odpovědi protokolu HTTP. *Interval.cz* [online]. 29.5.2002 [cit. 2012-03-11]. Dostupné z: <http://interval.cz/clanky/stavove-kody-a-hlaseni-v-odpovedi-protokolu-http/>
16. NETCRAFT. *March 2012 Web Server Survey* [online]. 2012 [cit. 2012-03-27]. Dostupné z: <http://news.netcraft.com/archives/category/web-server-survey/>
17. OLDANY GROUP. *Co je to virtualizace* [online]. 2012 [cit. 2012-03-16]. Dostupné z: <http://www.oldanygroup.cz/virtualizace-vmware-zakladni-informace-9/>
18. POLZER, Jan. *Jak otestovat rychlost a výkon hostingu* [online]. 23.2.2012 [cit. 2012-03-14]. Dostupné z: <http://www.maxiorel.cz/comment/8296>
19. RFC 2616. *Hypertext Transfer Protocol -- HTTP/1.1*. Internet Engineering Task Force (IETF), 1999. Dostupné z: <http://tools.ietf.org/html/rfc2616>
20. SEMLER, Miloslav. *Apache HTTP server verze 2.0: Kompilace a instalace* [online]. 2005 [cit. 2012-03-11]. Dostupné z: <http://majkls.pretel.cz/index.php?tm=apache2&cl=instal>
21. VAŠEK, Jiří. *Virtualizace aneb Jak vytvořit systém v systému* [online]. 24.9.2007 [cit. 2012-03-16]. Dostupné z: <http://pctuning.tyden.cz/software/ladeni-windows/9429-virtualizace-aneb-jak-vytvorit-system-v-systemu>
22. ZAJÍC, Petr. *PHP: Jak to funguje* [online]. 28.5.2004 [cit. 2012-03-14]. Dostupné z: http://www.linuxsoft.cz/article.php?id_article=172
23. ZAJÍC, Petr. *PHP (16): Vyrobme si kalendář* [online]. 25.6.2004 [cit. 2012-03-21]. Dostupné z: http://www.linuxsoft.cz/article.php?id_article=206
24. ZAPLETAL, Lukáš. *Protokol-http-1-1-pod-lupou*. In: *Root.cz* [online]. 27.3.2001 [cit. 2012-03-11]. Dostupné z: <http://www.root.cz/clanky/protokol-http-1-1-pod-lupou/>

8. Přílohy

Příloha I: Seznam modulů

Příloha II: Zdrojový kód kalendáře v PHP

Příloha III: Výpis souboru error.log

Příloha IV: Podíl jednotlivých webových serverů

Příloha V: Licence Apache 2.0

Příloha I: Seznam modulů

Základní moduly a moduly MPM

- **core**

Základní modul Apache. Obsahuje direktivy, které jsou vždy k dispozici a zabezpečují základní funkčnost serveru.

- **worker**

Modul MPM implementující víceprocesový a vícevláknový webový server

- **mpm_winnt**

Modul MPM implementující víceprocesový a vícevláknový webový server pro platformy operačních systémů Windows NT a Windows 2000.

- **perchild**

Modul MPM implementující víceprocesový vícevláknový server pro virtuální hosty s rozdílnou identifikací uživatelů.

- **prefork**

Modul MPM implementující nevláknový model podobný Apache 1.3.

- **mpm_netware**

Modul MPM implementující vícevláknový model pro platformu operačního systému Novell NetWare.

Moduly pro vytvoření prostředí

- **mod_env**

Předává proměnné prostředí do skriptů CGI a SSI.

- **mod_setenvif**

Nastavuje proměnné na základě informací od klienta.

- **mod_unique_id**

Vytvoří jednoznačný identifikátor pro každý požadavek.

Moduly pro vyhodnocení obsahu

- **mod_charset_lite**

Nastavení znakové sady.

- **mod_negotiation**

Umožňuje vyhodnotit obsah dokumentu na základě nastavení prohlížeče.

- **mod_mime_magic**

Nastavení MIME typu souboru na základě obsahu souboru. Tzn. že informace o typu souboru se nacházejí mimo soubor.

- **mod_mime**

Nastavení MIME typu souboru pomocí přípony souboru.

Moduly pro mapování URL

- **mod_alias**

Slouží pro přeměrování URL k mapování rozdílných částí souborového systému do struktury dokumentů.

- **mod_rewrite**

Mapování adresy URL přepsáním na jinou URL adresu. Provádí se použitím kombinací direktiv a regulárních výrazů.

- **mod_userdir**

Používá se pro nastavení a publikaci domovských adresářů uživatelů operačního systému.

- **mod_speling**

Oprava chybně zadaných URL adres.

- **mod_vhost_alias**

Dynamická konfigurace více virtuálních hostů.

Moduly pro nastavení vlastností adresářů

- **mod_dir**

Nastavení výchozích souborů v adresáři.

- **mod_autoindex**

Zajišťuje automatické zobrazení obsahu adresářů.

Moduly pro řízení přístupu

- **mod_access**

Nastavení omezeného přístupu pomocí jména nebo IP adresy počítače.

- **mod_auth**

Autorizace uživatelů pomocí textových souborů.

- **mod_auth_dbm**

Autorizace uživatelů pomocí souborů DBM.

- **mod_auth_anon**

Autorizace anonymních uživatelů.

- **mod_auth_digest**

Autorizace uživatelů pomocí autentifikace MD5.

- **mod_auth_ldap**

Autorizace uživatelů pomocí LDAP.

Moduly pro správu HTTP zpráv

- **mod_headers**

Správa hlaviček http.

- **mod_cern_meta**

Podpora metasouborů.

- **mod_expires**

Vytvoření hlavičky http Expires v odpovědi.

- **mod_asis**

Odesílání souborů s vlastními hlavičkami http.

Moduly zobrazující interní informace

- **mod_status**

Zobrazení statistiky Apache

- **mod_info**

Zobrazení konfigurace Apache.

Moduly pro tvorbu dynamického obsahu

- **mod_include**

Použití serverem vkládaných vsuvek SSI.

- **mod_cgi**

Používání skriptů CGI.

- **mod_cgid**

Používání skriptů CGI pomocí externích procesů.

- **mod_actions**

Vykonává skript CGI buď na základě typu souboru, nebo pomocí požadované metody.

- **mod_isapi**

Podpora Windows ISAPI.

- **mod_ext_filter**

Filtrování obsahu externím programem.

- **mod_suexec**

Spouštění CGI skriptů specifickým uživatelem.

Moduly pro protokolování činnosti

- **mod_log_config**

Uživatelsky nastavitelné protokolování činnosti.

- **mod_usertrack**

Protokolování činnosti uživatele pomocí cookies.

Moduly pro vývoj

- **mod_example**

Slouží pro ukázkou Apache API.

Další různé moduly

- **mod_imap**

Poskytuje podporu obrazových map.

- **mod_proxy**

Nastavuje vlastnosti proxy serveru.

- **mod_so**

Umožňuje podporu pro moduly DSO.

- **mod_file_cache**

Zajišťuje ukládání souborů do mezipaměti, čímž je zajištěn rychlejší přístup.

- **mod_deflate**

Komprimuje soubory před odesláním klientovi.

- **mod_ssl**

Modul, který umožňuje šifrování použitím SSL (Secure Socket Layer) a TLS (Transport Layer Security)

- **mod_ldap**

Slouží pro vytvoření spojení se serverem LDAP (Lightweight Directory Access Protocol) a výsledky uloží do vyrovnávací paměti, aby bylo možné je použít pro další moduly.

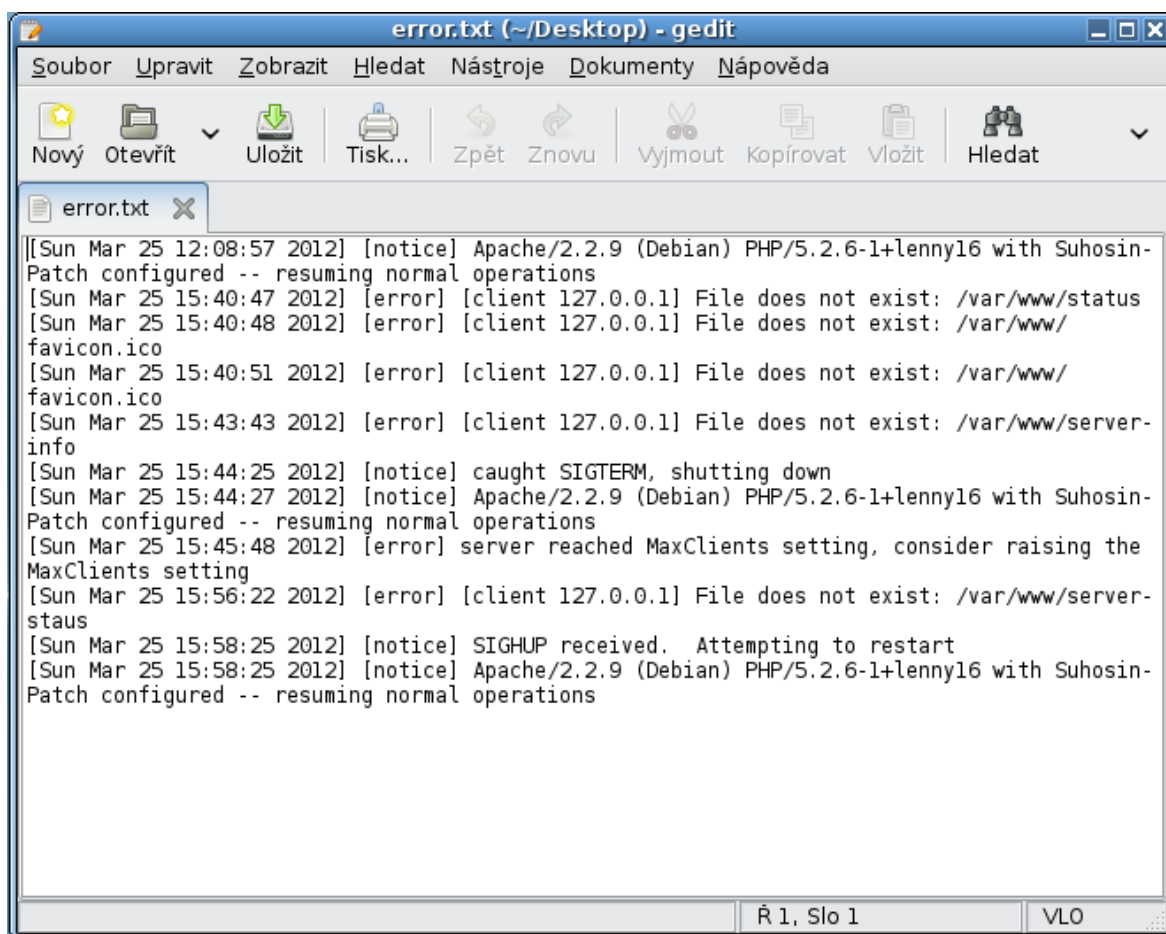
Příloha II: Zdrojový kód kalendáře v PHP

```

<?
function JePrechodnyRok ($rok)
{
    return (boolean) date("L", mktime(0,0,0,1,1,$rok));
}
function PocetDnu ($mesic, $rok)
{
    return cal_days_in_month(CAL_GREGORIAN, $mesic, $rok);
}
function PrvniDen ($mesic, $rok)
{
    $anglickeporadi = date("w", mktime(0, 0, 0, $mesic, 1, $rok));
    return ($anglickeporadi==0) ? 7 : $anglickeporadi;
}
function Bunka ($radek, $sloupec, $PrvniDen, $PocetDnu)
{
    $dny=Array(1=>"Po", "Út", "St", "Čt", "Pá", "So", "Ne");
    if ($sloupec==1) return $dny[$radek];
    $chcivratit = ($sloupec-2)*7 + $radek - $PrvniDen+1;
    if ($chcivratit<1 || $chcivratit>$PocetDnu) return "&nbsp;"; else return $chcivratit;
}
function Kalendar ($mesic, $rok)
{
    $mesice=Array(1=>"leden", "únor", "březen", "duben", "květen", "červen", "červenec", "srpen", "září", "říjen", "listopad", "prosinec");
    if (!is_numeric($mesic)) return "Měsíc musí být číslo!";
    if (!is_numeric($rok)) return "Rok musí být číslo!";
    if ($mesic<1 || $mesic>12) return "Měsíc musí být číslo od 1 do 12";
    if ($rok<1980 || $rok>2050) return "Rok musí být číslo od 1980 do 2050";
    // zjištění počtu sloupců
    $PocetDnu = PocetDnu ($mesic, $rok); $PrvniDen = PrvniDen($mesic,$rok);
    $sloupcu = date("W", mktime(0, 0, 0, $mesic, $PocetDnu-7, $rok)) - date("W", mktime(0, 0, 0, $mesic, 1+7, $rok))+4;
    // a tabulka
    echo "<TABLE border=\"1\" style=\"border-collapse: collapse\" width=\"\",$sloupcu*30,\">";
    echo "<TR><TD colspan=$sloupcu width=\"\",$sloupcu*30,\" align=\"center\">".$mesice[$mesic]."&nbsp;";
    for ($radek=1;$radek<=7;$radek++)
    {
        echo "<TR align=\"center\">";
        for ($sloupec=1; $sloupec<=$sloupcu; $sloupec++) echo "<TD width=\"30\">".Bunka($radek, $sloupec, $PrvniDen, $PocetDnu)."</TD>";
        echo "</TR>\n";
    }
    echo "</TABLE>";
}
Kalendar (3,2012);
?>

```

Příloha III: Výpis souboru error.log

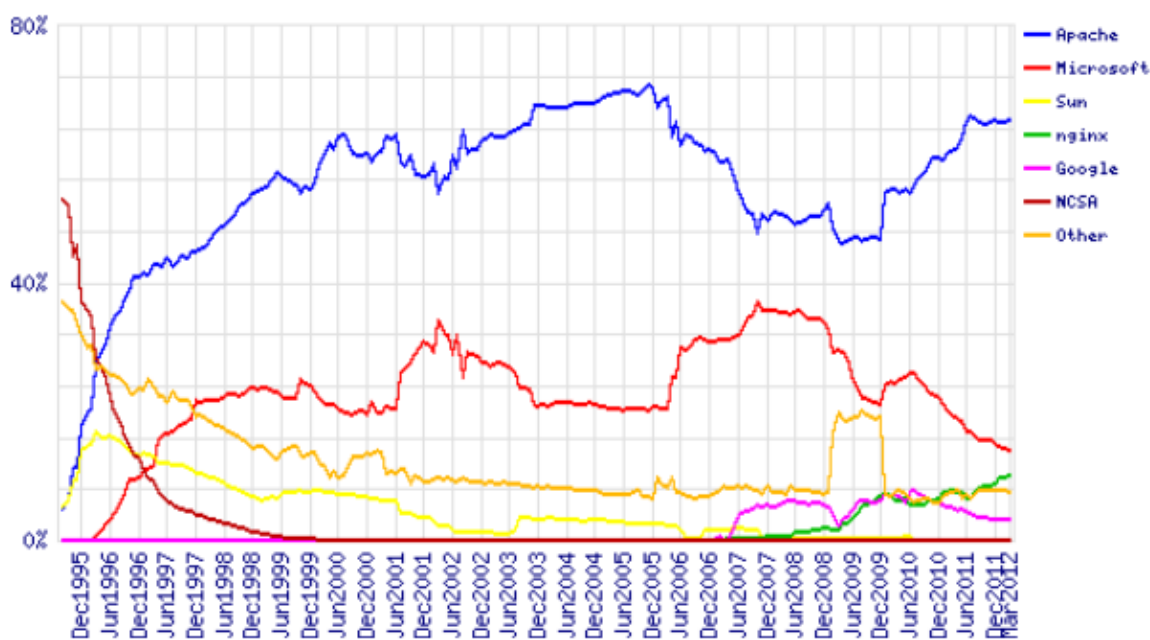


```
error.txt x
[[Sun Mar 25 12:08:57 2012] [notice] Apache/2.2.9 (Debian) PHP/5.2.6-1+lenny16 with Suhosin-
Patch configured -- resuming normal operations
[Sun Mar 25 15:40:47 2012] [error] [client 127.0.0.1] File does not exist: /var/www/status
[Sun Mar 25 15:40:48 2012] [error] [client 127.0.0.1] File does not exist: /var/www/
favicon.ico
[Sun Mar 25 15:40:51 2012] [error] [client 127.0.0.1] File does not exist: /var/www/
favicon.ico
[Sun Mar 25 15:43:43 2012] [error] [client 127.0.0.1] File does not exist: /var/www/server-
info
[Sun Mar 25 15:44:25 2012] [notice] caught SIGTERM, shutting down
[Sun Mar 25 15:44:27 2012] [notice] Apache/2.2.9 (Debian) PHP/5.2.6-1+lenny16 with Suhosin-
Patch configured -- resuming normal operations
[Sun Mar 25 15:45:48 2012] [error] server reached MaxClients setting, consider raising the
MaxClients setting
[Sun Mar 25 15:56:22 2012] [error] [client 127.0.0.1] File does not exist: /var/www/server-
staus
[Sun Mar 25 15:58:25 2012] [notice] SIGHUP received. Attempting to restart
[Sun Mar 25 15:58:25 2012] [notice] Apache/2.2.9 (Debian) PHP/5.2.6-1+lenny16 with Suhosin-
Patch configured -- resuming normal operations

Ř 1, Slo 1 VLO
```

Zdroj: Vlastní zpracování 25.3.2012

Příloha IV: Podíl jednotlivých webových serverů



Zdroj: NETCRAFT. *March 2012 Web Server Survey* [online]. 2012 [cit. 2012-03-27]. Dostupné z:

<http://news.netcraft.com/archives/category/web-server-survey/>[14]

Příloha V: Licence Apache 2.0

Apache License
Version 2.0, January 2004
<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally

submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.
4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
 - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
 - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
 - (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work,

excluding those notices that do not pertain to any part of the Derivative Works; and

- (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.
6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill,

work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.

Zdroj: APACHE SOFTWARE FOUNDATION. *Apache License, Version 2.0* [online]. leden 2004 [cit. 2012-03-30]. Dostupné z: <http://www.apache.org/licenses/LICENSE-2.0.html>[11]