Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Přenos dat na webový server (apache)

Petr Plavec

Bakalářská práce
2008

Abstrakt

Jedním z problémů při hostování stránek je přenos souborů na server. Možným řešením toho problému je zpřístupnění serveru přes FTP nebo přes SSH. Pokud je web uložen na cizím serveru, je k němu většinou jen jedno uživatelské jméno a heslo. Jestliže má na web přispívat více lidí, je tento stav nevyhovující. Cílem práce je navrhnout a vytvořit webovou aplikaci pro přenos a správu souborů na webovým serveru.

Klíčová slova

webová aplikace, přenos souborů, zabezpečení souborů, webový server

Title

Data transmission to the web server

Abstract

By one of problem at hosting pages is transfer file on server. Possible solution that problem is accessing server over FTP or over SSH. If is web stored on foreign server, is to him mostly only one user name and password. If has on web administer to more people, is this state unsatisfactory. Targets work is suggest and create application for transfer file on web server.

Keywords

webs aplication, file transfers, securities file, web server

Poděkování

Tímto bych chtěl poděkovat panu RNDr. Josefu Rakovi za náměty a připomínky ke tvorbě této práce.

Obsah

Seznam obrázků

1 Úvod

Cílem této práce je vytvořit aplikaci na přenos souborů s využitím jen webového rozhraní. Aplikace bude běžet na HTTP serveru Apache. K aplikaci budou mít přístup jen oprávněné osoby, které budou uloženy v databázi MySQL a budou mít přístup do aplikace pomocí vlastního loginu a hesla. K implementaci webové aplikace bude využit programovací jazyk PHP. Pro přenos dat bude bude uskutečněn přes protokoly FTP a HTTP. Další součástí práce je popsat zabezpečení přenášených dat a bezpečné uložení dat na serveru, popis jaké jsou možné přenosy souborů na server a jaké jsou bezpečnostní rizika jednotlivých přenosů.

1.1 Forma obsahu kapitol

Každá kapitola obsahuje několik dílčích částí; první část této práce popisuje srovnání webových serverů, možnosti přenosu souborů a zabezpečení samotných souborů. Následuje další, kde je navržen jeden z možných způsobů, jak konkrétní úkol realizovat. Konkrétní realizace je potom popsána v kapitole Aplikace na přenos souorů. V závěru je následující shrnutí celé práce.

1.2 Hlavní cíle práce

1.2.1 Webový server

Tato kapitola je zaměřena na srovnání webových serverů podle počtu používání na Internetu. Zejména pak popisuje, co to vlastně je webový server, co umí a k čemu se používá. Dále kapitola popisuje, jaké mohou být možné rozšíření webových serverů.

1.2.2 Možnosti přenosů souborů

V této kapitole je cílem popsat, jaké jsou možnosti přenosu dat na server. Zejména popis protokolů na přenos dat jako jsou FTP a SSH. Popis, jaké nabízí možnosti tyto protokoly, a jaké jsou jejich výhody a nevýhody.

1.2.3 Zabezpečení souborů

Při přenosu souborů může dojít ke ztrátě dat nebo k jejich odchycení druhou stranou. Tato kapitola popisuje jak zabezpečit data při přenosu, tedy jaké možnosti bezpečného přenosu nabízejí jednotlivé protokoly.

Další problém, který tato kapitola popisuje je bezpečnost dat na serveru. Řeší, jak správně postupovat při ukládání dat na serveru, aby nedošlo k neoprávněnému přístupu k souborům.

1.2.4 Aplikace pro přenos souborů

V této kapitole je popsán princip činnosti webové aplikace, jak je řešeno přihlašování uživatelů pomocí databáze MySQL, struktura aplikace a jak se dá aplikace vytvořit. V další části této kapitoly je ukázáno zprovoznění aplikace na serveru Apache.

2 Webový server

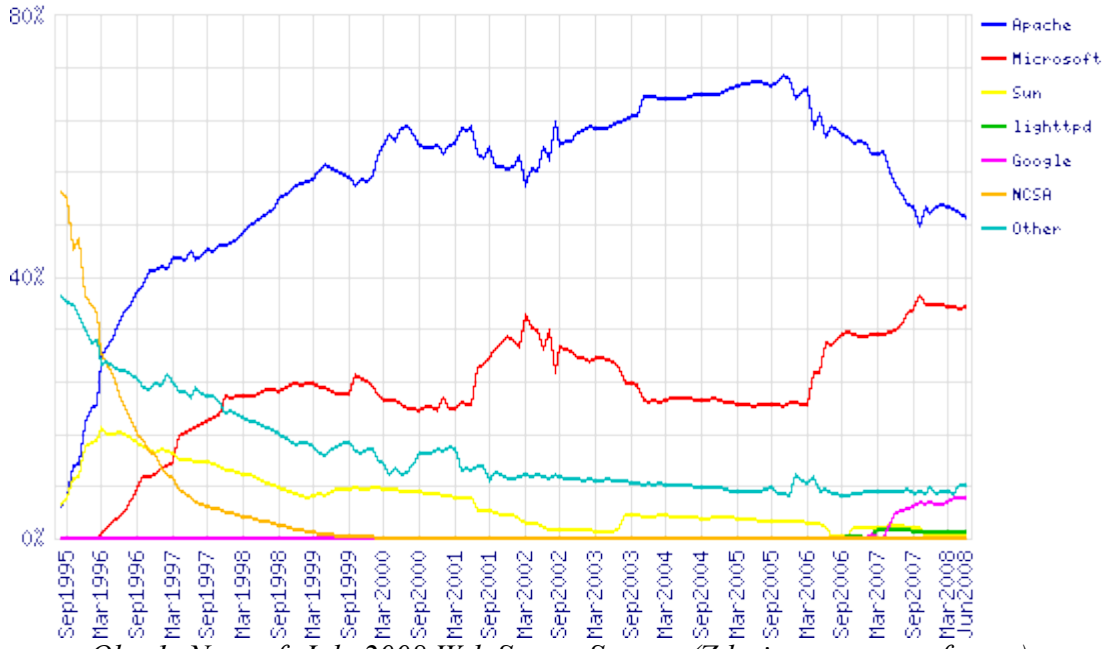
Webový server je počítačový program, který provádí komunikaci s klientem, což je vlastně webový prohlížeč, prostřednictvím HTTP protokolu. Po vyřízení požadavku pošle klientovi webovou stránku nebo stavový kód, pokud nastala nějaká chyba. Webové stránky jsou obvykle dokumenty ve formátu HTML.

Webový server používá 2 metody získávání informací. První metoda je požadavek na zobrazení HTML formátu. Server požadavek zpracuje a v nezměněné podobě zobrazí. Tomu se říká statický obsah. Druhou možností je přijetí skriptu, podle kterého se vygeneruje požadovaná HTML stránka. Tomuto způsobu zpracování přijaté informace se říká dynamický obsah. Mezi nejznámější skripty, které dokáže webový server zpracovat patří např. PHP, Perl, ASP, atd.

2.1 Srovnání webových serverů

Jednotlivé webové servery se mohou v různých funkcích značně lišit. Přesto mají několik společných vlastností. Mezi základní vlastnosti patří komunikace s webovými klienty prostřednictvím HTTP protokolu. Pokud tedy chceme prezentovat webové stránky pomocí celosvětové sítě Internet, slouží k tomu právě webový server. Webový server by tedy měl být připojený k počítačové síti, aby si vyžádaný obsah mohl prohlédnout kdokoliv. Ovšem může běžet i na vlastním počítači, např. v rámci testování nebo vývoje webových aplikací.

Podle průzkumu společnosti Netcraft, je nejpoužívanějším webovým serverem na internetu webový server Apache (viz. Obr. 1).



Obr. 1: Netcraft July 2008 Web Server Survey. (Zdroj: www.netcraft.com)

Na Obr. 1 je vidět, že rozdíl mezi nejpoužívanějšími webovými servery, kterými jsou IIS (Internet infomation services) od Microsoftu a Apache, se rychle zmenšuje. Zajímavý nárůst zaznamenává webový server od společnosti Google. Dalším webovým serverem, který se používá je SunONE Web Server od společnosti Sun Microsystems. Tento webový server však zaujímá jen malé procento z používaných serverů a jeho pozice značně oslabuje. Tyto čtyři servery si rozebereme v následující kapitole podrobněji.

2.1.1 Apache

HTTP server Apache byl vyvinut v roce 1993 Robem McCoolem v Národním centru pro superpočítačové aplikace (NCSA) na Univerzitě v Illinois. V roce 1995 byla vydána první veřejná verze 0.6.2. Po odchodu Roba McCoola z NCSA byla založena skupina Apache Groupe.

Hlavním úsilím projektu Apache HTTP server bylo vytvořit stabilní a volně dostupný zdrojový kód k implementaci HTTP (web) serveru. Projekt je řízen skupinou dobrovolníků z celého světa, pomocí internetu a webové komunikace mohou plánovat a vytvářet server a jeho související dokumentaci.

Software Apache je šířen pod licencí *Apache Software Foundation* zcela zdarma. Při použití Apache software, jak už samotného serveru, nebo jen základu kódu, bychom měli uvádět, že server patří pod licenci Apache Software Foundation, zejména pokud by byl využíván pro komerční účely.

Server se stal rozšířeným zejména díky snadné dostupnosti a modulárnosti celého serveru. Modulárnost spočívá v tom, že si můžeme sestavit server podle vlastních potřeb. Nahrajeme si moduly, které budeme na svém serveru využívat.

Nastavení serveru se provádí pomocí textových konfiguračních souborů. Hlavní konfigurační soubor se nazývá httpd.conf. Po instalaci Apache se nachází tento hlavní konfigurační soubor v adresáři *conf.* V adresáři conf jsou i pomocné konfigurační soubory, které jsou uloženy v podadresáři *extra.* Pomocí těchto souborů můžeme na serveru nastavovat služby jako SSL, virtuální hostitelské počítače, uživatelská místa atd.

Jelikož textová konfigurace může být pro některé uživatele značně nepohodlná, existuje řada nástrojů pro administraci Apache, které však nezahrnují zdaleka všechny možné nastavení Apache a používají se například jen pro nastavení serveru, který běží jako testovací stroj. Mezi takové nástroje patří Comanche nebo Mohawk. Apache je možné nainstalovat i v předkonfigurovaných balíčcích. Tyto balíčky se většinou skládají z webového serveru, skriptovacího jazyka PHP a databázového serveru MySQL. Mezi nejznámější balíčky patří Xampp, který zahrnuje jak PHP a Mysql, tak i SSL šifrování, PERL, WebDAV, aplikaci pro práci s MySQL phpMyAdmin a další služby.

Server umožňuje logování, což jsou informace o dění na serveru, které se ukládají do externích souborů. Tyto soubory jsou uloženy v adresáři *logs.* V tomto adresáři je několik souborů, kde můžeme zjistit, co se na serveru v určitou dobu dělo. Pokud nastane chyba při vykonávání určité operace, je tato chyba zapsána do souboru error.log. Apache má standardně ještě dva soubory, kde zaznamenává přístup na server a změny v instalaci. Tyto soubory se jmenují access.log a install.log. Server umožňuje také vytváření vlastních logů pomocí pravidel, které jsou popsány v dokumentaci.

Mezi základní vlastnosti, kterými server disponuje je podpora FastCGI, což je protokol na zpracování serverů. Mezi nejvyužívanější metodu hostingových společností určitě patří vytváření virtuálních hostitelských počítačů.

Bezpečnost v Apache zajišťuje protokol SSL, který se ovšem musí k serveru doinstalovat. Balíček na instalaci se dá volně stáhnout na stránkách http://www.openssl.org/. Dalším bezpečnostním prvkem v Apache je autentizace uživatelů, která je podrobněji popsaná v kapitole *Autentizace uživatelů.* Apache také umožňuje zamítnutí nebo povolení přístupu pomocí IP adres.

Nevýhodou tohoto serveru je veřejný zdrojový kód, což muže být bezpečnostní riziko. Hlavním přínosem tohoto serveru je, že pomocí přidávání modulů, které mohou tvořit programátoři na celém světě, umožňuje serveru pracovat s velkým množstvím technologií pro práci s webovým obsahem.

2.1.2 Internet Information Services (IIS)

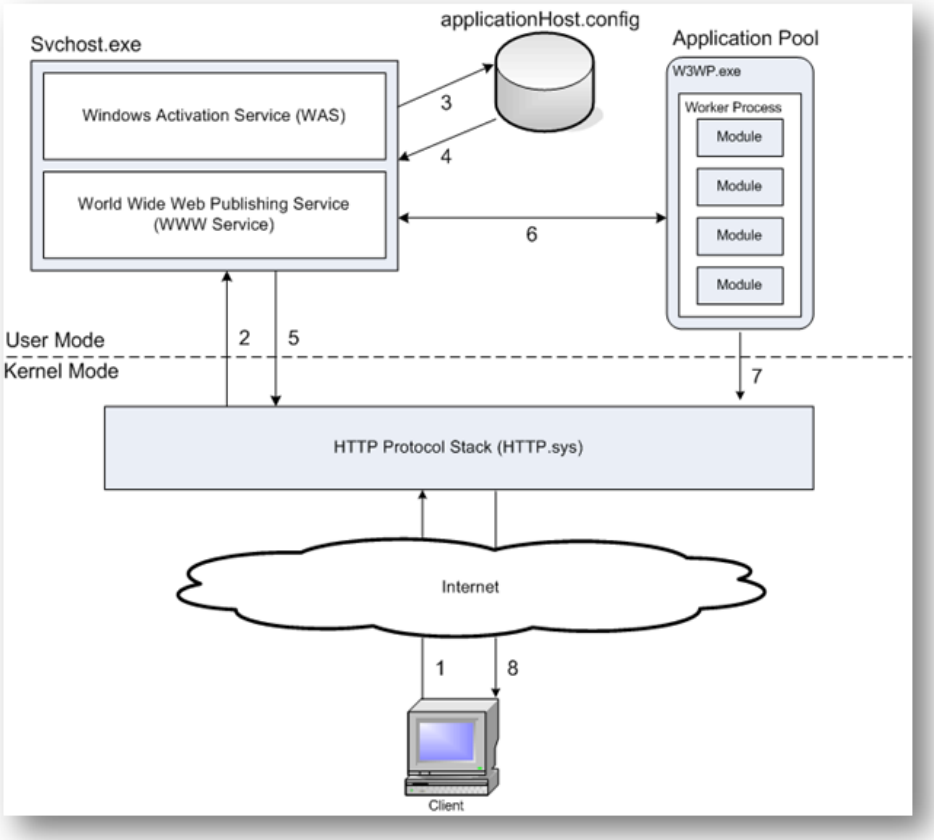
IIS (Internet Infoamtion Services) je webový server vyvinutý společností Microsoft. Jeho hlavním úkolem je usnadnění sdílení dokumentů na intranetu společností a na Internetu.

První IIS ve verzi 1.0 bylo dostupné jako volný doplněk Windows NT 3.51. Následovali verze 2.0, 3.0, 4.0 ve Win NT 4.0 a dále verze 5.0, 5.1, 6.0 pro Win Server 2000, Win XP a Win Server 2003. Poslední vydání IIS ve verzi 7.0 je volitelnou součástí Windows Vista.

V nové verzi 7.0 došlo k zásadní změně konfigurace, kde je konfigurační model ukládán jako XML. Ve starších IIS se používala pro uložení konfiguračních dat metabáze. V XML souboru bude možnost nastavit například povolení pro přístup do adresářů, různé HTTP handlery apod. Nastavení serveru ovšem může probíhat nejen pomocí tohoto XML souborů, ale také pomocí GUI rozhraní.

Novou vlastností IIS je jeho modulárnost. V předchozích verzích musely být všechny funkce provedeny na serveru samotném. Ve verzi 7.0 přichází pro IIS revoluční změna, jelikož webový server funguje jenom jako engine pro rozšiřující moduly, které si může uživatel přidávat a odebírat podle svých vlastních potřeb.

IIS 7.0 má podobnou formu zpracování HTTP jako předchozí verze 6.0. Postup zpracování požadavku je rozdělen do 8 kroků, které jsou na Obr. 1 označeny čísly od jedné do osmi. V prvním kroku pošle klient požadavek na webový server, který se zachytí v zásobníku HTTP.SYS. Tento zásobník v dalším kroku naváže spojení s Windows Activation Service (WAS). Ve 3. kroku WAS posílá žádost o informace, jaká je konfigurace úložiště uživatele. Následně WWW Service obdrží konfigurační informace. Tyto informace použije v 5. kroku pro nastavení HTTP.SYS. V 6. kroku začíná zpracování procesů, na které byla podána žádost. V 7. kroku Worker Process zpracuje požadavek a vrátí odpověď na HTTP.SYS. V posledním 8. kroku uživatel obdrží odpověď. V těchto 8 krocích probíhá komunikace pro zpracování požadavku mezi základními službami, kterými jsou HTTP.SYS, WAS, WWW Service a Worker process.



Obr. 2: Zpracování HTTP požadavku v IIS 7.0 (Zdroj: www.iis.net)

IIS umožňuje práci s různými skriptovacími jazyky jako je například PHP. PHP na IIS běží pomocí standardního modulu FastCGI. Hlavní využití pro IIS je ovšem pro skriptovací jazyk ASP.NET, který může běžet i pod Apache, ale po doinstalování a nastavení modulů. To může být celkem složitý problém. IIS spojuje práci s PHP a ASP.NET mnohem jednodušeji. Další vlastnost, kterou server umožňuje je HTTP komprese. Pomocí této komprese může posílat data na server a tím zvýšit rychlost stahování stránek. Pro IIS 7.0 byl vytvořen URL Rewrite modul, který jak už napovídá název umí přepisovat URL adresy. V uživatelském rozhraní serveru se mohou pravidla pro přepisování URL adres mazat, přepisovat a přidávat.

K zabezpečení webového serveru slouží v IIS několik metod. První metodou je autentizace. Autentizace umožňuje ověření uživatelů, kteří přistupují na web a zajišťuje, že se k obsahu dostanou jen oprávněné osoby. Podobná metoda, která zajišťuje bezpečnost je autorizace. Tato metoda se používá k nastavení práv na získávání informací. Pokud chceme například povolit přístup jednomu uživateli a ostatním přístup zamítnout včetně administrátorům, provedeme to následovně:

```
<authorization>
      <add accessType="allow" users="uzivatel" />
      <add accessType="deny" roles="administrators" />
      <add accessType="deny" users="*" />
</authorization>
```

IIS má plnou podporu SSL šifrování, které se nastavuje přímo v HTTP.SYS. Další bezpečnostní prvek, kterým server disponuje je filtrování. To se provádí přes ISAPI filtr. V IIS 7.0 je tento filtr přímo zabudovaný a nemusí se doinstalovávat jako v předchozích verzích. Tento nově zabudovaný filtr může filtrovat požadavky na základě slov, přípon souborů, velikostí atd. Omezení přístupu k serveru se může provádět i přes možnost zakázat IP adresu.

Hlavním přínosem pro vývojáře webových aplikací je v IIS podpora ASP.NET, což je asi největší plus při práci s tímto serverem.

2.1.3 SunONE Web Server

Tento webový server, který může běžet na většině dostupných operačních systémech, byl navrhnut společností Sun Microsystems. Hlavním cílem tohoto projektu bylo vytvoření bezpečného a vysoce spolehlivého webového serveru. Tento server podporuje řadu technologií. SunONE Web Server je především zaměřen na webové technologie Java Servlet a JavaServer Pages, dále pak podporuje například PHP, ColdFusion, CGI a další.

Server je vícevláknový a umožňuje paralelní běh více aplikací, které běží buď v 32-bitovým, nebo v 64-bitovým módu. Může mít více jak sto tisíc připojení najednou. Administrační rozhraní dovoluje snadnou kontrolu nad virtuálními servery, rychlý přístup k často používaným úkolům a integrovanou správu skupin buď přes webové rozhraní, nebo pomocí administrační konzole. Příkazová řádka je přístupná i na ovládání na vzdálený přístup.

Server podporuje protokol WebDAV. Díky tomu usnadňuje snadnou správu a publikaci standardních firemních dokumentů na webovém serveru. Pomocí HTTP komprese, která probíhá ještě před posláním dat k uživatelům, může uchovat větší šířku pásma a tím zlepšit výkon přenášených dat. Server dále umí automatickou korekci URL adres, které ukazují na nezajištěné spojení. Pokud existuje zajištěné spojení, je uživatel automaticky přesměrován na toto spojení. Samozřejmostí, kterou poskytuje SunOne Web

Server je podpora virtuálních serverů. Pomocí virtuálních serverů může běžet na jednom webovém serveru tisíce domén. Server také umožňuje podporu FastCGI.

Bezpečnost na serveru zajišťuje několik metod. Zdroje mohou být zabezpečený s pokročilým Access Control Listem (ACLs), Lightweight Directory Access Protokol (LDAP) s podporou flexibilního schématu, Secure Socket Layer (SSL) ve verzi 2 a ve verzi 3, Transport Layer Security (TLS) 1.0 a Elliptic Curve Cryptography (ECC).

Pokud chceme server nainstalovat, můžeme si ho po registraci zdarma stáhnout na webových stránkách http://www.sun.com/download/. Server je dostupný pouze v balíčku *Sun Java Web Infrastructure Suite*, ve kterém se nachází řada dalších aplikací pracující s webovým rozhraním. Při instalaci na operační systém Windows si musíme dát pozor, aby souborový systém byl typu *NTFS*, jinak instalace neproběhne úspěšně.

2.1.4 Google App Engine

V poslední době se do popředí dostávají webové technologie od společnosti Google. Jednou z těchto technologií je Google App Engine.

Tato technologie umožňuje běh uživatelských aplikací na systému Googlu. S App Engine nepotřebujeme provozovat žádné servery. Webovou aplikaci stačí nahrát, a je připravena k používání uživatelem. Tyto aplikace mohou být provozovány na volné doméně http://appspot.com nebo na vlastní doméně s Google Apps. Po bezplatné registraci může uživatel vyvíjet a publikovat své aplikace na Internetu. Na účtu zdarma je možné využít až 500 MB volného prostoru a až 5 milionů zobrazení stránek za měsíc.

Pomocí Google App Engine je snadné budovat aplikace, které běží spolehlivě a to i při vysokém zatížení s velkým objemem dat. Systém zahrnuje obsluhu dynamického webu s plnou podporou běžných internetových technologií, trvalé úložiště dat s dotazy, tříděním a transakcemi, automatické vyrovnávání výkonu a plnou podporu pro vývoj aplikací na lokálním počítači uživatele.

Aplikace běží v bezpečném prostředí, které poskytuje omezený přístup k základnímu operačnímu systému. K aplikaci mohou mít přístup ostatní počítače na Internetu prostřednictvím URL, emailové služby a rozhraní API. Na serveru nelze zapisovat do souborového systému, protože app umí číst pouze soubory nahrané s kódem aplikace.

App Engine poskytuje runtime prostředí, které používá programovací jazyk Python. Ostatní programovací jazyky zatím nejsou dostupné. Python runtime prostředí využívá verzi 2.5.2.

Při vývoji aplikací můžeme Google App Engine provozovat i na vlastním počítači, abychom si mohli vyzkoušet funkčnost aplikace. Balíček Google App Engine SDK se dá bezplatně stáhnout na stránkách http://code.google.com/appengine/downloads.html. Balíček je poskytován pro tyto platformy Windows, MacOS X a Linux. Pokud chceme balíček nainstalovat nesmíme, zapomenout na podporu jazyka Python, protože je to jediný programovací jazyk, který Google App Engine podporuje. Podporu jazyka Python si můžeme volně stáhnout na stránkách http://python.org/.

Ovládání celého rozhraní se provádí přes terminál, neboli příkazovou řádku. Pokud chceme například nastavit na serveru stránku, která se bude načítat v prohlížeči, uděláme to příkazem:

```
dev_appserver.py moje_aplikace
```

V tomto případě je dev_appserver.py příkaz na spuštění serveru a moje_aplikace je adresář, odkud se načítají webové stránky. V prohlížeči potom aplikaci spustíme tak, že do URL adresy napíšeme localhost:8080.

Vývoj aplikací přes Google rozhraní by mohl mít v budoucnu velký potenciál. Největší jeho nevýhoda v dnešní době je složité ovládání pomocí terminálu a podpora jenom jednoho programovacího jazyka.

2.2 Rozšíření webového serveru

Většina webových serverů se může rozšířit například o skriptovací jazyk, databázový server, atd. Tato kapitola popisuje zprovoznit na serveru skriptovací jazyk PHP, databázi MySQL a aplikaci pro práci s MySQL databází phpMyAdmin.

2.2.1 PHP

„PHP (rekurzivní akronym pro "PHP: Hypertext Preprocessor" je rozšířený univerzální skriptovací jazyk s volně dostupným zdrojovým kódem, který je obzvláště vhodný pro vývoj webových aplikací a lze jej jednoduše zapouzdřit do HTML.

Interpret jazyka PHP běží na straně serveru, tím se liší od jiných skriptovacích jazyků, jako je například JavaScript. Klient tedy obdrží pouze výstup z právě prováděného skriptu bez možnosti zjistit, či modifikovat zdrojový kód aplikace.

PHP nám poskytuje plnou podporu OOP, ovšem jeho schopnosti daleko překračují prosté generování HTML dokumentů. Máme možnost vytvářet obrázky, PDF dokumenty, FLASH animace generované za běhu, spolupracovat s databázemi atd.“[1]

PHP instalaci si můžeme volně stáhnout na oficiálních stránkách PHP www.php.net. Poté přímo na této adrese http://www.php.net/downloads.php#v5 stáhneme balík, který se jmenuje *PHP 5.2.6 zip package*.V adresáři, kde máme nahraný Apache *c:\apache* vytvoříme podadresář *php*. Do toho adresáře rozbalíme stáhnutý balíček. Po rozbalení se v adresáři nachází soubor *php.ini-recommended*. Tento soubor přejmenujeme na *php.ini*. V tomto konfiguračním souboru musíme zapnout rozšiřující moduly, které rozšiřují funkčnost php. Proto tento řádek:

```
extension_dir = "./"
```

musíme změnit na:

```
extension_dir = "C:\apache\php\ext\"
```

Toto nastavení nám určuje cestu, kde jsou uloženy rozšiřující moduly. Moduly, které chceme využívat musíme odkomentovat. Pro naši aplikaci využijeme tyto moduly:

```
extension=php_bz2.dll
extension=php_curl.dll
extension=php_gettext.dll
extension=php_mbstring.dll
extension=php_mysql.dll
extension=php_mysqli.dll
extension=php_openssl.dll
extension=php_xmlrpc.dll
extension=php_zip.dll
```

Další důležitou vlastností, kterou PHP umožňuje a měla by být součástí při vývoji aplikací, je výpis chyb přímo do prohlížeče. Pokud tedy nastane chyba v syntaxi při běhu webové aplikace, příslušný příkaz se neprovede a do prohlížeče se vypíše popis chyby, a na kterém řádku chyba nastala. Toto nastavení zajistíme tak, že v konfigu-

1 PHP DOCUMENTATION GROUP – PHP manual [online]. 2008 [cit. 2008-07-21] Dostupný z www: http://www.php.net/docs.php

račním souboru zapneme výpis chyb. Řádek s nastavením *display_errors* nastavíme takto:

```
display_errors = On
```

Neměli bychom ještě zapomenout na výpis chyb při startu PHP. To provedeme tak, že řádek s *display_startup_errors* nastavíme na:

```
display_startup_errors = On
```

Po nastavení *php.ini*, musíme ještě nastavit konfigurační soubor Apache *httpd.conf*. Aby nám webový server fungoval s rozšířením o PHP, musíme do něho doplnit tyto řádky:

```
LoadModule php5_module "C:/apache/php/php5apache2_2.dll"
PHPIniDir "C:/apache/php"
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

Poslední krok, který musíme provést je zkopírování souboru *php5ts.dll z* adresáře *c:\apache\php* do adresáře *c:\Windows\system32*. Nyní je nastavení PHP kompletní.

2.2.2 MySQL

MySQL je multiplatformní databázový server, ve kterém se komunikace zajišťuje pomocí jazyka SQL. Je poskytován jak pod bezplatnou licencí, tak i pod placenou licencí. Při instalaci ve Windows se využívá jen instalačního rozhraní, nemusí se nastavovat žádné textové konfigurační soubory.

Databázový server se dá snadno stáhnout na oficiálních stránkách MySQL http://www.mysql.com/ v sekci *Downloads*. V této sekci si vybereme, na jaký operační systém chceme instalovat instalační balíček. Pokud tedy chceme server instalovat na operační systém Windows stáhneme si balíček pro Windows, přičemž máme možnost stáhnout si jak na 32-bitový systém, tak na 64-bitový systém. Abychom mohli nainstalovat server bez přepisování konfiguračních souborů stáhneme si balíček *Windows ZIP/Setup.EXE (x86)*.

Po stáhnutí balíčku můžeme spustit instalaci. V prvním kroku se zobrazí uvítací okno, to potvrdíme tlačítkem *Next. V* dalším okně máme na výběr, jakou formu instalace chceme zvolit. Máme na výběr ze tří možností, nejvýhodnější možností je zvolit vlastní instalaci, tedy možnost *Custom*. Poté, co jsme zvolili možnost *Custom*, můžeme vybrat místo na disku, kam databázový server umístíme. Pro snadnější konfiguraci je

vhodné mít všechny rozšiřující části webového serveru pohromadě, proto zvolíme cestu následovně:

```
c:\apache\mysql
```

V dalším okně se nás instalátor ptá, jestli chceme zřídit účet na MySQL.com. Jelikož to nevyužijeme, zvolíme poslední možnost *Skip Sign-Up.* Tímto krokem je sice instalace kompletní, ale chybí nám ještě databázový server nastavit. Nastavení se provede tak, že po skončení instalace se nám v okně zobrazí možnost nastavit server pomocí průvodce. Zaškrtnutím políčka *Configure the MySQL Server now* a následném kliknutí na tlačítko F*inish* spustíme průvodce nastavením databázového serveru.

Po úvodním uvítacím okně průvodce nastavení se zobrazí, jakou konfiguraci chceme provést. Na výběr je detailní a standardní. Volíme možnost *Detailed Configuration a* klikneme na tlačítko *Next.* Další možnost, kterou máme na výběr je zvolení typu používání databázového serveru. Jedná se jen o využití CPU a paměti, pro menší náročnost zvolíme *Developer Machine.*

V dalším kroku můžeme zvolit užívání databáze. Jako nejoptimálnější je zvolit multifunkční databázi, tedy možnost *Multifunctional Database.*

V následující části nastavení serveru musíme zvolit, na jaké místo se budou ukládat tabulky na pevném disku. Zvolíme tedy pro nás nejvhodnější možnost. Po zvolení místa, kam se budou ukládat data z databáze, máme možnost nastavit kolik najednou připojení k databázi budeme mít k dispozici. Postačí první možnost 20 připojení, zaškrtneme tedy políčko s možností *Decision Support (DSS)/OLAP.*

Nyní nakonfigurujeme síťové připojení. Pro zapnutí TCP/IP spojení musíme zaškrtnout políčko *Enable TCP/IP Networting.* MySQL standardně pracuje na portu 3306, proto port měnit nemusíme. Poslední možnost, která je nabídnutá, je zapnutí striktního módu. Tato volba je doporučená proto, aby se server choval jako tradiční databázový server.

Po nastavení síťového připojení máme možnost nastavit v jakém kódovacím jazyce bude MySQL server komunikovat. Nejsnadnější možností je zvolit *Manual Selected Default Character Set / Collation.* Při této volbě vybereme požadované kódování. Mezi nejpoužívanější kódování patří latin2, UTF-8 a CP1250.

V dalším kroku nastavíme název služby pro Windows jako MySQL5 a zapneme možnost automatického spuštění po startu Windows.

Nyní nám už jen chybí nastavit heslo root pro přístup do databáze. Po nastavení hesla se nám v dalším okně zobrazí přehled změn, které se automaticky nastaví. Po stisknutí na tlačítko *Execute* se změny vykonají. Databázový server je nyní kompletně nastaven a připraven k používání.

2.2.3 phpMyAdmin

phpMyAdmin je webová aplikace, která je napsána v PHP a umožňuje kompletní správu databází běžících na databázovém serveru MySQL. Oficiální webové stránky jsou www.phpmyadmin.net, na těchto stránkách se dá aplikace volně stáhnout. Existují i české stránky o phpMyAdmin http://phpmyadmin.cz/, kde se dají najít různé informace o nastavení a používání aplikace.

Instalace aplikace není složitá. Na oficiálních stránkách v sekci *Downloads* stáhneme nejaktuálnější verzi, nejlépe s plnou podporou jazyků. Balíček s plnou podporou jazyků se jmenuje *all-languages.zip.* Tento balíček rozbalíme kamkoliv na pevný disk, nejlépe je rozbalit balíček do adresáře, kde máme webový server Apache, tedy:

```
c:\apache\
```

V tomto adresáři se nám po rozbalení vytvoří nový adresář, ten přejmenujeme na phpmyadmin. V adresáři, který jsme přejmenovali, se nachází konfigurační soubor celé aplikace *config.inc.php.* Pomocí tohoto souboru si můžeme nastavit konfiguraci aplikace podle vlastních potřeb. Nám bude stačit nastavit jméno uživatele databázového serveru *root a* heslo, které jsme si zvolili při nastavení MySQL. To provedeme následovně:

```
$cfg['Servers'][$i]['user'] = 'root'; // MySQL user
$cfg['Servers'][$i]['password'] = 'zvolené heslo';
```

Samotná aplikace je nyní nastavená, pokud ovšem chceme, aby aplikace běžela na webovém serveru Apache, musíme si vytvořit v konfiguračním souboru Apache *http.conf* alias, který spustí phpMyAdmin po jednoduchém vyplnění URL adresy. Pokud jsme použili cestu c:\apache\phpmyadmin\, vytvoříme alias z kopírováním těchto řádku do souboru *http.conf*:

```
################## MUJ ALIAS ####################
Alias /phpmyadmin "C:/apache/phpmyadmin"
<Directory "C:/apache/phpmyadmin">
   Options Indexes
```

```
    Order Deny,Allow
    Allow from all
    Deny from none
</Directory>
```

Po restartu webového serveru si funkčnost aliasu vyzkoušíme tak, že do URL adresy zadáme *localhost/phpmyadmin*. Tímto krokem je instalace a nastavení aplikace kompletní.

3 Možnosti přenosů souborů

Přenos dat zajišťuje protokol TCP/IP, který funguje jako nosič mnoho jiných protokolů. Asi nejpoužívanějším protokolem na Internetu pro přenos dat je protokol HTTP, který předává webové stránky.

Přímo pro přenos souborů byl vyvinut protokol FTP (File Transfer Protocol). Tento protokol je zejména používán pro přenos webových stránek na webový server. Jeho největší slabinou je ovšem bezpečnost. Jelikož je to jeden z nejstarších protokolů, může jméno a heslo odchytnout kdokoliv, kdo používá stejnou síť. V dnešní době se protokol FTP využívá s SSL šifrováním.

Dalším protokolem, pomocí kterého můžeme přenášet soubory mezi dvěma počítači se nazývá SSH (Secure Shell). Protokol SSH je komunikačním protokolem, který se využívá pro vzdálený přístup k jinému počítači.

V poslední době se stává čím dál více populárním protokol WebDAV pro správu souborů na vzdáleném serveru. Tento protokol spojuje práci s HTTP hlavičkou a XML soubory. Je to tedy spojení těchto dvou technologií.

3.1 FTP

FTP (File Transfer Protocol) je protokol z aplikační vrstvy TCP/IP. FTP protokol umožňuje přenos souborů mezi dvěma počítači, na kterých mohou běžet rozdílné operační systémy.

FTP protokol vznikl v roce 1971. Do transparentního přenosu TCP/IP, který získal svou dnešní podobu zhruba v roce 1977 až 1979, přešel FTP na počátku osmdesátých let.

FTP protokol je 8bitový a pracuje na principu klient-server na portech 21 a 20. Port 21 slouží k řízení přenosu a jsou jím přenášeny příkazy FTP. Port 20 slouží k vlastnímu přenosu dat.

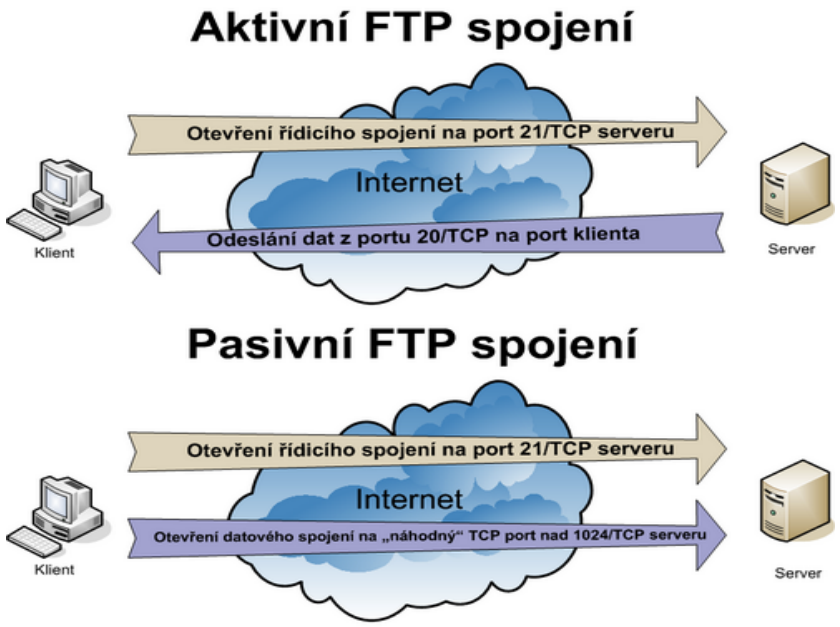
Protokol umožňuje řízený přístup pomocí přihlašování, specifikaci formátu přenášeného souboru, který může být binární nebo znakový, výpis vzdáleného adresáře atd.

Pro přenos protokolem FTP se využívají minimálně 2 počítače. Na jednom počítači musí být nakonfigurován a spuštěn FTP server. Tento počítač poskytuje data pro ostatní počítače. Další počítače se mohou pomocí FTP klienta připojovat na FTP server. Po připojení klienta se může na serveru dělat řada operací, jako odesílání dat na server, stahování dat ze serveru, přejmenování a mazání souborů nebo adresářů a řada dalších operací.

Výrobci softwaru jsou schopni vytvořit FTP server nebo klienta, protože FTP protokol je definován standardem. Tento standard je použitelný pro všechny platformy operačních systémů. To dovoluje použít pro FTP přenos jakýkoliv počítač, připojený přes TCP/IP protokol k internetu.

3.1.1 Princip činnosti

Pokud chce klient komunikovat se serverem pomocí protokolu FTP jsou mu vždy otevřena najednou dvě spojení: komunikační a datové. Pokud má klient požadavek na server, otevírá komunikační spojení, na které mu server poté odpoví. Komunikační spojení slouží také ke sdělování informací o požadavcích na změny stavu FTP spojení a posílání dalších informací. Datové spojení otevírá buď klient, nebo server, podle toho, jestli se jedná o pasivní nebo aktivní připojení.



Obr. 3: Porovnání spojení (Zdroj: www.poradna.net)

Na obrázku Obr. 3 je znázorněno, jak FTP protokol využívá dvou způsobů připojení. Aktivní připojení bylo u FTP využíváno již při jeho vzniku. Jeho princip spočívá v tom, že klient vystaví spojení ze svého portu nad 1024/TCP na port 21/TCP FTP serveru. U pasivního připojení jsou obě spojení vystaveny směrem k FTP serveru. U tohoto připojení nastává však problém, u kterého musíme odlišit jednotlivé datové porty jednotlivých klientů. FTP server to řeší tak, že na každý požadavek na datové spojení přidělí dočasný server na náhodném portu z rozsahu portů nad 1024/TCP.

3.1.2 Jak se připojit na FTP server

Pro připojení na FTP server potřebujeme mít nainstalovaného FTP klienta. FTP klient se dá většinou stáhnout zdarma. K připojení můžeme využít i zabudované FTP klienty operačních systémů. Ve Windows stačí v příkazové řádce zadat příkaz *ftp* a konzole se nám přepne do jiného režimu. Pro připojení zadáme příkaz *open hostitel.* Jako hostitele můžeme zadat IP adresu serveru, na který se chceme připojit, nebo název hostitele (např. ftp.linux.cz). Po připojení k hostiteli zadáme jméno uživatele a uživatelské heslo. Po úspěšném přihlášení můžeme posílat data na server nebo data ze serveru stahovat.

Na Internetu existuje spousta programů, které umožňují připojení k FTP serverům. Většinou jsou k dispozici na webu zdarma. Mezi takové klienty například patří Total Commander, Classic FTP a mnoho dalších.

Mezi další aplikace, které umožňují přístup na FTP server jsou webové aplikace. U webových aplikací máme výhodu, že nemusíme instalovat žádný zvláštní software. Prostřednictvím webového prohlížeče se přihlásíme přímo na FTP server. Mezi nejznámější webové FTP klienty patří www.net2ftp.com. Mnoho hostingových společností má zabudovaného webového klienta přímo v poskytované administraci ke stránkám.

3.2 SSH

SSH je síťový protokol, který umožňuje posílat data mezi dvěma síťovými zařízeními pomocí zabezpečeného kanálu. Uplatňuje se zejména na unixových systémech, tedy na systémech, založených na správu systému pomocí terminálu. SSH byl vyvinut jako náhrada protokolu TELNET, který posílal data v nezabezpečené podobě, což byl problém zejména při posílaní hesel. Díky šifrování poskytuje SSH důvěrnost a integritu dat přenášených po nezabezpečené síti, například internetu.

Protokol byl vyvinut v roce 1995 na Technologické univerzitě v Helsinkách. V roce 1996 byla vydána nová verze SSH protokolu a to SSH-2, která byla nekompatibilní z předchozí verzí. Nová verze umožňovala bezpečnější výměnu klíčů a integritu ověření pomocí zpráv ověřujících kódy. V roce 1999 byla první verze, která byla šířená jako open source, přepsána a vydána jako OpenSSH pro OpenBSD. Od verze 2.6 je OpenSSH přenositelný i na jiné operační systémy.

SSH využívá kryptografický veřejný klíč k ověření vzdáleného počítače a také umožní vzdálenému počítači ověřit uživatele. Protokol umožňuje nejen dávat příkazy vzdálenému počítači, ale může posílat i soubory. Soubory může posílat díky přidruženým protokolům jako jsou SFTP nebo SCP.

3.2.1 Architektura

Celý protokol je rozdělen do 3 vrstev, které navzájem spolu komunikují. Mezi tyto vrstvy patří:

- Transportní vrstva – tato vrstva se stará o výměnu klíčů, autentizaci serveru, šifrování a nastavuje komprese
- Ověřovací vrstva – zajišťuje autentizaci klientů a určuje jestli může ke spojení dojít.
- Vrstva spojení – tato vrstva definuje koncept kanálů, požadavků kanálů a používání globálních požadavků. Standardní typy kanálů zahrnují: *Shell* pro terminálové shelly, *Direct-tcpip* pro spojení předávaná klientem serveru a *Forwarded-tcpip* pro spojení předávaná serverem klientovi.

3.3 WebDAV

Web-based Distributed Authoring and Versioning (WebDAV) je rozšíření HTTP hlavičky, která umožňuje uživatelům společně editovat a spravovat soubory na vzdáleném serveru. Poskytuje uživatelům funkce pro vytváření, přesun a změnu na vzdáleném serveru. To může být užitečné pro upravování dokumentů, které server poskytuje, ale může fungovat i jako základní úložiště souborů, které mohou být přístupné odkudkoliv. Důležité funkce, které protokol poskytuje při práci se soubory jsou zamykání, což je prevence proti přepsání důležitých dokumentů, dále pak vlastnosti souborů, které poskytují informace o autorovi, vytvoření, změně, atd.

Protokol WebDAV byl vyvinut v roce 1996 na meetingu World Wide Web Consortium (W3C), který byl pořádán za účelem projednání problému o distribuovaném vývoji, kde byl jako host Jim Whitehead, který začal projednávat se zainteresovanými lidmi tento problém. V roce 2002 byla vydána finální verze tohoto protokolu.

Protokol se skládá z řady nových metod a hlaviček pro použití v HTTP protokolu a je to téměř první protokol, který pracuje s XML. WebDAV rozšířil HTTP o tyto metody:

- PROPFIND – používá se k získávání vlastností uložených jako XML,
- PROPPATCH – používá se ke změně a mazání vlastností o zdroji,
- MKCOL – slouží k vytvoření kolekce,
- COPY – kopíruje soubory z jednoho URI na druhý,
- MOVE – přesouvá soubory z jednoho URI na druhý,
- LOCK – používá se k uzamčení souborů,
- UNLOCK – tato metoda odemyká soubory,
- DELETE – mazání souborů.

Protokol WebDAV je podporován většinou nejpoužívanějších webových serverů, mezi které patří i dva nejpoužívanější webové servery a to Apache a Internet Infoamtion Services.

4 Zabezpečení souborů

Při přenosu souborů z jednoho počítače na druhý počítač může dojít k odchycení přenášených dat. Spojení, kterým je třeba FTP nemá žádnou ochranu, takže data přenášená po síti může zachytit kdokoliv jiný připojený k síti. Může se jednat i o heslo k FTP, ale i o velmi citlivá data. Pro bezpečnější přenos dat bylo tedy navrhnuto SSL (Secure Socket Layer) šifrování, které bylo rozšířeno i na FTP protokol. Data díky SSL procházejí sítí v zabezpečené podobě, takže je téměř nemožné zjistit jejich obsah. SSL šifrování ovšem nemá uplatnění jen u FTP protokolu, ale je velmi využíván i u protokolu HTTP, tedy u protokolu, který předává webové stránky internetovým prohlížečům z webových serverů. Protokol s tímto rozšířením se nazývá HTTPS.

Dalším problémem při uložení dat na serveru je jejich zpřístupnění jen oprávněným osobám. Autentizace uživatelů, kteří mají mít přístup k souborům, se dá na webovém serveru řešit takzvanou HTTP autentizací. Další možností, jak tento problém vyřešit, je nastavení práv pro přístup do adresářů.

4.1 SSL šifrování

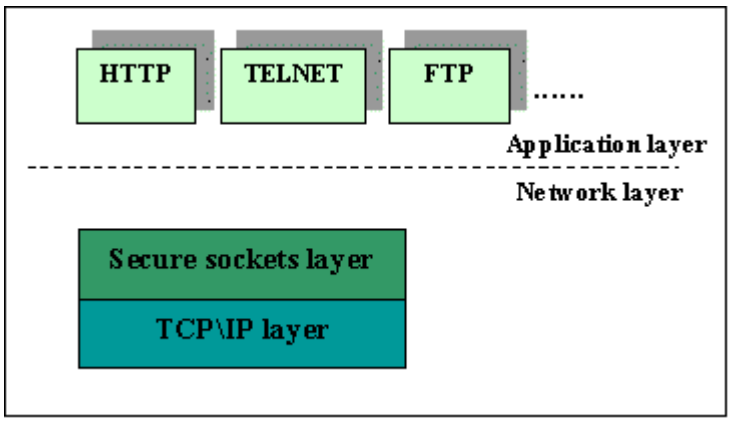
Secure Sockets Layer (SSL) technologie chrání webové stránky a nabízí návštěvníkům důvěryhodnost procházených webových stránek třemi způsoby:

- SSL certifikát umožňuje šifrování citlivých údajů během transakcí,
- každý certifikát obsahuje unikátní ověřené údaje o certifikátu majitele,
- certifikační úřad ověří pravost certifikátu.

Každý SSL certifikát se skládá z veřejného klíče a soukromého klíče. Veřejný klíč je používán pro šifrování dat a soukromý klíč je využíván pro dešifrování dat. K veřejnému klíči může mít přístup vlastně kdokoliv. Když tedy někdo zašifruje data pomocí veřejného klíče, pak tyto data může rozšifrovat pouze majitel svým soukromým klíčem. Tento postup se nazývá asymetrické šifrování.

Ověření spojení se nazývá *SSL handshake* (potřásání rukou). Toto ověření probíhá v několika krocích. Nejprve pošle klient serveru požadavek na SSL spojení. Server na jeho požadavek odpoví a pošle klientovi certifikát serveru. Podle přijatého certifikátu, který obsahuje veřejný klíč, si klient ověří autentičnost serveru. Pomocí veřejného klíče zašifruje klient svůj vygenerovaný základ šifrovacího klíče, pod kterým se bude šifrovat následná komunikace. V dalším kroku rozšifruje server pomocí svého soukromého klíče základ šifrovacího klíče. Z tohoto základu vygenerují jak server, tak klient hlavní šifrovací klíč. Server si s klientem potvrdí, že šifrované spojení bude zajištěno právě tímto vygenerovaným klíčem.

Na obrázku Obr. 4 je pozice protokolu SSL v TPC/IP modelu. Je tedy vidět, že jde o přidanou podvrstvu mezi protokol TPC/IP a protokoly z vyšší vrstvy jako je FTP, HTTP a další.



Obr. 4: Pozice SSL v síťovém protokolu TCP/IP (Zdroj: www.svetsiti.cz)

4.2 Autentizace uživatelů

Pokud chceme na webovém serveru zpřístupnit data jen oprávněným uživatelům, musíme si ověřit, jestli uživatel, který se chce dostat na náš server, má příslušná přístupová práva.

Když je na webovém serveru nastavena autentizace pro přístup k souborům, předá webový server klientovi hlavičku *401*. Tato hlavička oznamuje klientovi, že je nutné poskytnout autentizační údaje pro přístup. Jakmile klient obdrží tuto hlavičku, je vyzván, aby zadal svoje přihlašovací jméno a heslo. Pokud jsou údaje správné, je klientovi od webového serveru odeslán požadovaný zdroj.

Jelikož si HTTP protokol neumí pamatovat svůj stav, je každý požadavek pro přístup do chráněné oblasti proveden znovu. Většina dostupných prohlížečů si, ale s tímto dokáže poradit, takže uživatel nemusí znovu zadávat své ověřovací údaje. Nevýhodou ovšem je že odhlášení nastává až po ukončení prohlížeče, což může být problém pokud se na jednom počítači střídá více uživatelů.

Na webovém serveru Apache se může taktéž nastavovat přístup do adresářů jen oprávněným osobám. Existují čtyři rozšiřující moduly, které pracují s autentizací uživatelů. Základní modul je *Mod_auth*, kde jsou hesla uložená v souboru. Dalším modulem je *Mod_auth_anon*, který umožňuje přístup anonymním uživatelům. Modul *Mod_auth_dbm* umožňuje autentizaci uživatelů uložených v DBM. Posledním modulem je *Mod_auth_db*, tento modul poskytuje autentizaci uživatelů uložených v DB. Nastavení pro přístup do určitého adresáře uživatelům ze skupiny můžeme nastavit v hlavním konfiguračním souboru *httpd.conf*. Nastavení by potom mohlo vypadat následovně:

```
<Directory "C:/apache/test">
 Order Deny, Allow
```

```
 Allow from all
 Deny from none
# autentifikace přístupu (vynucení autentifikace) a nastavení
konkrétního uživatele
 Require valid-user
# nastavení typu autentifikace - ověřování uživatele pomocí jména
a hesla (Basic , Digest)
 AuthType Basic
# nastavení názvu autentifikační oblasti
 AuthName "Moje zóna"
 # nastavení souboru s uživateli
 AuthUserFile "C:/apache/pass/user"
# nastavení souboru se skupinami uživatelů
 AuthGroupFile "C:/apache/pass/group"
</Directory>
```

Pro vytvoření uživatelů s heslem slouží soubor *htpaswd.exe.* Pro vytvoření souboru hesel zadáme do příkazové řádky:

```
htpasswd –c soubor_hesel jméno_uživatele
```

Pro přidávání dalších uživatelů už potom zadáváme pouze tento příkaz:

```
htpasswd soubor_hesel jméno_uživatele
```

Přístup můžeme povolit i skupině uživatelů, kde do skupiny zařazujeme uživatele ze souboru vytvořeného pro autentizaci jednotlivých uživatelů. Pro skupinu si vytvoříme textový soubor, kam napíšeme tyto řádky:

```
Jméno_skupiny: uživatel_1 uživatel_2 … uživatel_n
```

Pokud uděláme toto nastavení a uživatel bude chtít přistoupit přes webový server do adresáře *c:/apache/test,* objeví se dialogové okno, kam zadá své heslo a jméno. Po správném zadání údajů mu bude umožněn přístup do adresáře.

5 Aplikace pro přenos souborů

Cílem je vytvoření aplikace, ke které budou mít přístup jen uživatelé uloženi v databázi MySQL. Programovací jazyk PHP má v sobě zabudované funkce, které umí komunikovat s FTP serverem pomocí protokolu FTP. Díky tomu může uživatel spravovat soubory na vzdáleném serveru jen pomocí webového rozhraní. V aplikaci má každý

uživatel svoje jméno a heslo pro přístup. Uživatele může do databáze přidávat jen administrátor, který je definován v konfiguračním souboru aplikace.

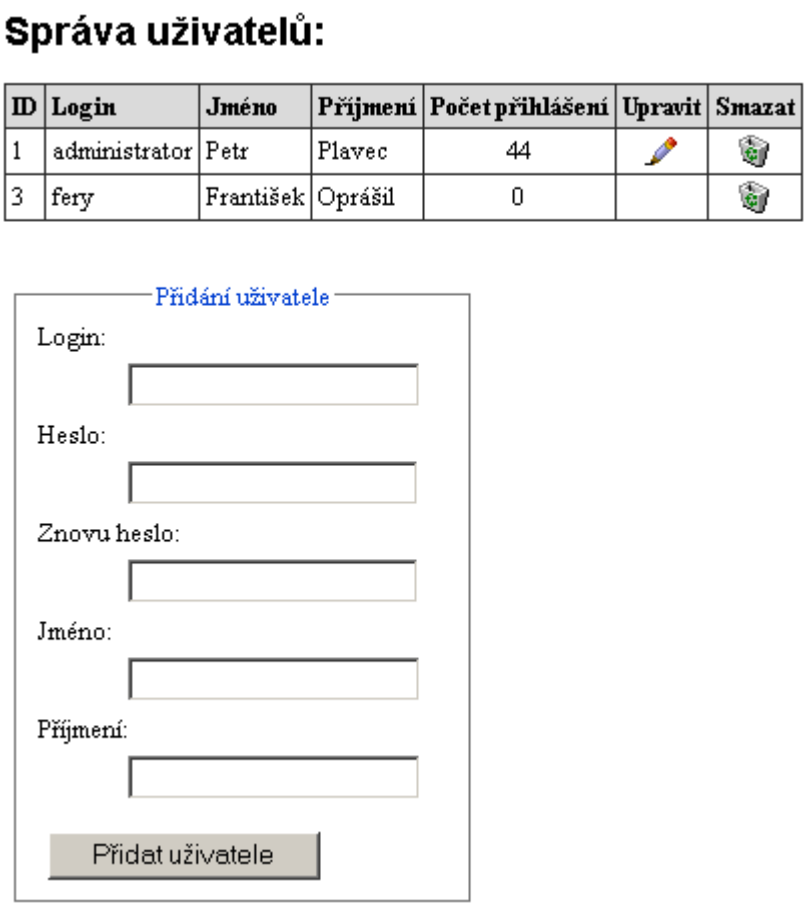
Aplikace běží na webovém serveru Apache a pro připojení k FTP serveru je využit šifrovací kanál SSL. Je to z toho důvodu, že FTP protokol není bezpečný a při přenosu citlivých souborů, kterými mohou být například zdrojové kódy, může dojít k odchycení dat nežádoucím uživatelem.

Soubory a adresáře budou mít jen jednoho vlastníka. Tento vlastník bude uživatel pro přístup k FTP serveru. Uživatel bude ovšem jen jeden, jelikož většina hostingových společností přiděluje jen jedno heslo a jméno. K FTP serveru se tedy budou připojovat uživatelé jen pod jedním účtem. Při správném nastavení práv adresářů bude moci se soubory a adresáři pracovat jen přihlášený uživatel k FTP serveru, tím budou vlastně všichni uživatelé uloženi v databázi MySQL, jelikož po jejich přihlášení do aplikace se automaticky přihlásí i na FTP server.

Při tvorbě aplikace jsem kladl zejména důraz na její jednoduchost a přehlednost. Pro vytvoření přehledné aplikace jsem použil CSS styly. K ovládání aplikace slouží navigace, která je umístěna v levé části aplikace (viz.Obr. 5).

Obr. 5: Navigace v aplikaci

Webová aplikace se skládá ze dvou částí. V první části je výpis všech uživatelů, kteří s aplikací pracují. Pokud je přihlášeným uživatelem administrátor, tak má možnost přidávat a mazat uživatele. Přidat uživatele má právo jen administrátor z toho důvodu, aby měl plnou kontrolu nad uživateli, kteří s aplikací pracují. Přihlášený uživatel má možnost opravovat svoje osobní údaje, které jsou uloženy v databázi. Hlavně si změnit heslo, které mu bylo přiděleno administrátorem.



Obr. 6: Administrační rozhraní pro správu uživatelů

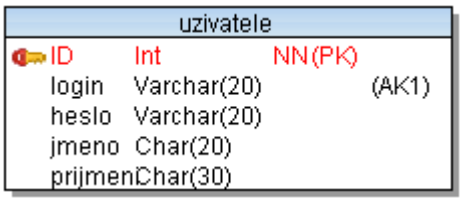
Druhá část je samotná správa souborů, které jsou uloženy na serveru. Přihlášený uživatel vidí všechny soubory a složky, se kterými může pracovat. Ve výpisu vidí, kdy byly soubory nebo adresáře naposledy změněny, velikost souboru a atributy jaké má buď adresář, nebo soubor. Procházení stromem adresářů je řešeno mocí křížových odkazů. Pokud chce uživatel stáhnout soubor, stáhne ho taktéž pomocí odkazu. Aplikace umožňuje stahování všech souborů, tedy i souborů, které by webový server interpretoval jako webové stránky. Jedná se o soubory s příponou html, php a další. Další možností, která je ve výpisu, je mazání adresářů a souborů. Pokud chce uživatel přejmenovat soubor nebo adresář je ve výpisu u daného souboru nebo adresáře ikonka na přejmenování. Po kliknutí na ní se prohlížeč přesměruje na stránku, kde je vyzván na zadání nového názvu. Po zadání názvu je opět vrácen na předchozí stránku. Pod výpisem souborů se nachází nástroje, pomocí nichž může uživatel vytvořit vlastní adresář nebo nahrát soubor pomocí dialogového okna. Při nahrávání souborů na server si musí uživatel dát pozor na to, aby nahrávaný soubor měl menší velikost jak 2 MB. Je to z toho důvodu, že PHP si nedokáže s větším souborem poradit.

5.1 Postup vytvoření aplikace

Tato kapitola popisuje, jak se dá aplikace na přenos souboru implementovat. Je zde popsán návrh tabulky, v které jsou uloženi uživatelé, kteří mají přístup do aplikace. V dalším kroku je popsána architektura aplikace, zejména pak postup při vývoji aplikace. V posledním kroku kapitoly je popis stěžejních funkcí PHP pro přenos souborů na server pomocí FTP protokolu.

5.1.1 Návrh tabulky uzivatele

Jelikož do aplikace mají mít přístup jen uživatelé uloženi v databázi MySQL, tak v prvním kroku implementace musíme vytvořit v databázovém serveru tabulku, odkud bude ověřovací funkce načítat data a zjišťovat, jestli se uživatel v databázi nachází a má přístup do aplikace. Tabulku si pojmenujeme *uzivatele.* Struktura tabulky, kde budou uloženi uživatelé s přístupem do aplikace, je na Obr. 7.

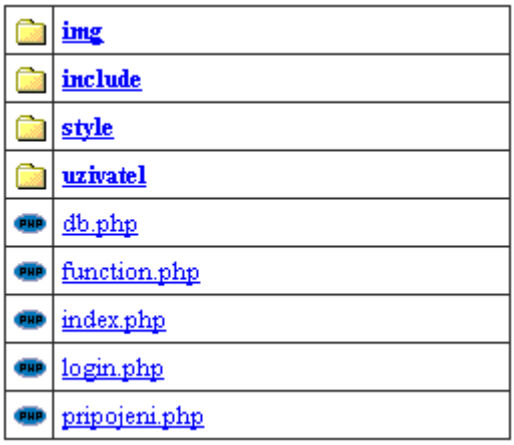


Obr. 7: Struktura tabulky uzivatele

V tabulce *uzivatele,* je primárním klíčem *ID*, což je jednoznačný identifikátor každého uživatele a je pro každého jiný. Díky tomuto identifikátoru rozlišujeme jednotlivé uživatele. V tabulce jsou také uloženy dva důležité atributy pro přihlašování a to jsou *login* a *heslo.* U atributu *login* je nastaven parametr *unique*, což nám zaručí, že každý *login* se může v tabulce objevit maximálně jednou.

5.1.2 Struktura aplikace

Pro přehlednost je aplikace rozdělena do několika podadresářů a samotných skriptů. V této kapitole je popsáno, co adresáře obsahují a jakou funkci plní skripty v programu. Samotná struktura aplikace je potom na Obr. 8.



Obr. 8: Struktura aplikace

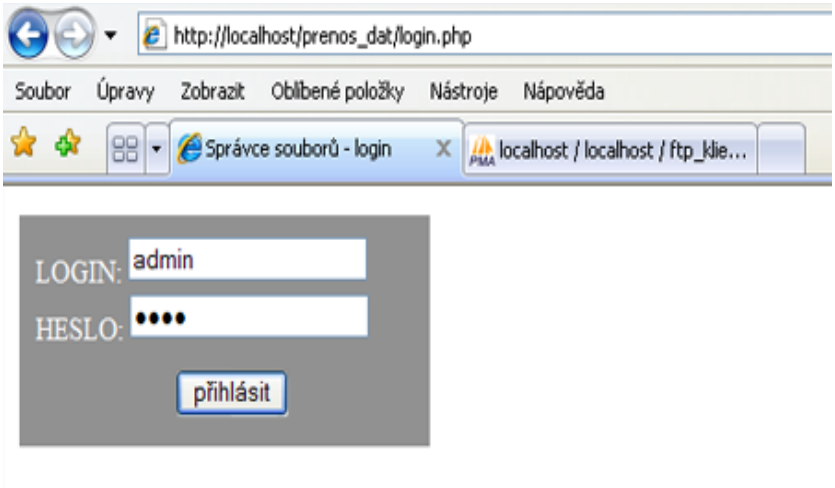
V adresáři *img* jsou uložené ikonky, které se používají při výpisu souborů a adresářů. Na Obr. 8 jsou ikonky vidět v levém sloupci tabulky. Adresář *include* obsahuje skripty, které zajišťují přepínání obsahu stránky. Pokud uživatel klikne v navigačním menu například na správu uživatelů, požadovaný obsah se mu načítá právě z tohoto adresáře. V adresáři *styl* jsou uloženy všechny grafické prvky aplikace. Mezi tyto prvky patří například pozadí, odrážky atd. Hlavním souborem v tomto adresáři je ovšem soubor *styl.css,* který zajišťuje celkové stylování stránky a určuje v jakém vzhledu se bude zobrazovat. V adresáři *uzivatel* jsou uloženy skripty pro správu uživatelů.

V kořenovém adresáři jsou také uloženy skripty. Hlavním skriptem je *index.php,* který řídí celou aplikaci. Všechny požadavky na zobrazení nebo vykonání zvolených úkonů jsou poskytovány právě tímto skriptem. Dalším důležitým skriptem je *login.php.* Na tento skript je stránka přesměrována pokud není přihlášený žádný uživatel. Po správném vyplnění přihlašovacích údajů je stránka zase zpátky přesměrovaná na *index.php.* Konfigurační soubor celé aplikace se nazývá *db.php,* kterým můžeme nastavovat připojení na ftp server, databázi MySQL atd. V souboru *function.php* jsou uloženy všechny funkce, které se používají v aplikaci. Posledním skriptem je *pripojeni.php,* díky němuž se aplikace připojuje k databázi. Obsah souboru vypadá následovně:

```
<?php
mysql_connect($hostname,$username,$password);
mysql_select_db($databaseName);
mysql_query("SET NAMES 'cp1250'");
?>
```

5.1.3 Architektura aplikace

Aplikace je rozdělaná do několika na sebe navazujících částí. V první části musíme zajistit přístup uživatelů do aplikace. Další část aplikace je správa uživatelů. Poslední část aplikace je potom samotná správa souborů na vzdáleném serveru. V prvním kroku tedy musíme vytvořit přihlašovací formulář pro uživatele (viz. Obr. 9).



Obr. 9: Ukázka přihlašovacího formuláře

Data z formuláře porovnává funkce s údaji z databáze a pokud najde shodu dojde k přihlášení uživatele do aplikace pomocí přesměrování hlavičky. Na následující ukázce je znázorněno, jak je přesměrování vytvořeno.

```
if (isset($_SESSION['login']))
{
      if ($_SESSION['login']=='ano'){
            header("Location: index.php");
            exit();
      }
}
```

Aby se do $_SESSION['login'] vložil stav *ano*, což znamená že přihlášení proběhlo úspěšně, musí ověřovací funkce zkontrolovat, jestli jsou zadaná data z formuláře obsaženy v databázi.

```
function AutorizaceUzivatele ($connection, $username, $password)
{
  if (!isset ($username) || !isset ($password))
    return false;
  $query = "SELECT * from uzivatele
             where login = '$username' AND heslo = '$password'";
  $result = @ mysql_query($query, $connection)
```

```
    or showerror();
  $row = mysql_fetch_array($result);
  $_SESSION['ID'] = $row['ID'];
  if (mysql_num_rows($result) !=1 )
    return false;
  else
    return true;
}
```

Přihlašování uživatelů je vyřešeno. Nyní se podíváme, jak se dá vytvořit rozhraní pro práci s uživateli. Správa uživatelů funguje ve dvou režimech v administračním a uživatelským. Administrační režim se liší od uživatelského tím, že administrátor může přidávat a mazat uživatelé. Administrátor přidává uživatelé přes formulář, ve kterém po vyplnění údajů a následném stisknutí tlačítka, spustí skript, který vloží pomocí následujícího dotazu uživatele do databáze.

```
$sql = "INSERT INTO uzivatele(login,heslo,jmeno,prijmeni)
      VALUES ('$login', '".md5($heslo)."', '$jmeno','$prijmeni');";
```

V předchozím dotazu je vidět, že heslo je pro větší bezpečnost šifrované metodou *md5*.

Ve výpisu uživatelů je pro jejich mazání zobrazena ikonka, na kterou, když klikneme, provede se smazání uživatele z databáze. To provádí následující dotaz:

```
$sql = "delete from uzivatele where id = $ID";
```

Uživatelé a administrátoři mohou také měnit svoje údaje v databázi. Update uživatelů v databázi provádí následující SQL dotaz:

```
$sql2 = "UPDATE uzivatele SET login = '$login',
heslo='".md5($heslo)."', jmeno = '$jmeno', prijmeni = '$prijmeni'
WHERE id ='$update';";
```

Po vyřešení přihlašování a správě uživatelů chybí ještě popsat samotnou správu souborů, které jsou pro přehlednost tvořeny tabulkou. Do tabulky zapisujeme data, které vrací funkce pro výpis obsahu serveru ftp_rawlist(). Tato funkce ovšem vrací data v této neupravené podobě:

```
-rw-r--r-- 1 mujprenos (?) 930 Jul 24 17:12 db.php
-rw-r--r-- 1 mujprenos (?) 3935 Jul 24 17:06 function.php
drwxr-xr-x 2 mujprenos (?) 4096 Aug 5 23:08 img
drwxr-xr-x 2 mujprenos (?) 42 Aug 6 12:41 include
```

```
-rw-r--r-- 1 mujprenos (?) 4140 Aug 9 23:23 index.php
-rw-r--r-- 1 mujprenos (?) 1693 Jul 24 17:17 login.php
-rw-r--r-- 1 mujprenos (?) 127 Jul 24 17:06 pripojeni.php
drwxr-xr-x 2 mujprenos (?) 122 Jul 24 17:06 style
drwxr-xr-x 2 mujprenos (?) 112 Jul 24 17:06 uzivatel
```

Pokud tedy chceme zapsat správně data do tabulky, musíme si pomocí regulárních výrazů vyselektovat příslušné údaje. První funkce, která nám zjistí, jestli daná položka je adresář, implementujeme následovně:

```
function JeToAdresar($polozka) {
   if (ereg("^d[rstwx\-]{9}", $polozka))
      return true;
   else
      return false;
}
```

Funkce *ZjistiDatum* vrací datum vzniku u adresáře nebo souboru.

```
function ZjistiDatum($polozka) {
   $shody = array();
   eregi("([a-z]{3} +[0-9]{1,2} [ 0-9\:]{4,5})", $polozka, $shody);
   return $shody[1];
}
```

Další funkce vytáhne z neutříděného seznamu název souboru nebo adresáře.

```
function ZjistiNazev($polozka) {
   $shody = array();
   eregi("[a-z]{3} +[0-9]{1,2} [ 0-9\:]{4,5} (.+)$", $polozka,
$shody);
   return $shody[1];
}
```

Funkce *ZjistiAtribut* vytáhne ze seznamu atributy souboru nebo adresáře.

```
function ZjistiAtributy($polozka) {
   $shody = array();
   eregi("^(.{10}) ", $polozka, $shody);
   return $shody[1];
}
```

Poslední funkce, která pracuje se seznamem se nazývá *ZjistiVelikost*. Tato funkce zjišťuje velikost souborů.

```
function ZjistiVelikost($polozka) {
   $shody = array();
   eregi("^.{10}[ ]+[0-9]+ [^ ]+ +[^ ]+ +([0-9]+) ", $polozka,
$shody);
   return $shody[1];
}
```

Nyní pomocí těchto pěti funkcí můžeme cyklem zapisovat data do tabulky. Tímto krokem je zobrazení obsahu serveru kompletní. Funkce, které umožňují pracovat se soubory na vzdáleném serveru jsou popsány v následující kapitole.

5.1.4 Popis funkcí pro práci s FTP

Programovací jazyk PHP má v sobě zabudované funkce, které umí pracovat s protokolem FTP a můžeme je najít v dokumentaci PHP. První takovou funkcí je *ftp_ssl_connect*. Tato funkce zajišťuje bezpečné spojení se serverem. Její použití je následovné:

```
$conn_id = @ftp_ssl_connect($ftp_server);
```

Proměnná *$ftp_server* se nastavuje v konfiguračním souboru *db.php* a může obsahovat buď název serveru, nebo jeho IP adresu.

Pokud předchozí funkce naváže spojení, musí ještě dojít k ověření uživatele pomocí loginu a hesla. Ověření provádí funkce *ftp_login*, která může být použita například takto:

```
if (@ftp_login($conn_id, $ftp_user, $ftp_pass))
   $zprava = "Server byl připojen jako $ftp_user@$ftp_server | \n";
else
   $zprava = "Server nebyl připojen jako $ftp_user | \n";
```

Funkce ftp_login má tři parametry $conn_id, což je spojení se serverem, $ftp_user a $ftp_pass. Poslední dva parametry předávají funkci jméno uživatele a heslo. Tyto parametry jsou taktéž uloženy v konfiguračním souboru.

Další funkce pracují už přímo s obsahem serveru. Požadavky na tyto funkce jsou přenášeny pomocí URL adresy a při použití větvící podmínky je příslušná funkce provedena. Mezi tyto funkce patří *ftp_mkdir*, která vytváří na serveru adresáře. Vytvoření adresáře tedy provedeme následovně:

```
case "vytvor_adresar":
```

```
          if (!@ftp_mkdir($conn_id, $cesta."/".$_POST['adresar']))
{
                $chyba = "Adresář se nepodařilo vytvořit !";
            }
```

Pokud ovšem chceme příslušný adresář opět smazat používá se k tomu funkce *ftp_rmdir.*

```
case "smaz_adresar":
     if (!@ftp_rmdir($conn_id, URLDecode($_GET['adresar']))) {
$chyba="Adresář<strong>".URLDecode($_GET['adresar'])."</strong>
se nepodařilo smazat, pravděpodobně není prázdný!";
}
```

Podobná funkce, která ovšem maže soubory se nazývá *ftp_delete.*

```
 case "smaz_soubor":
     if (!@ftp_delete($conn_id, URLDecode($_GET['soubor']))) {
           $chyba = "Soubor se nepodařilo smazat !";
       }
```

Pro přejmenování adresáře a souboru je v PHP zabudovaná funkce *ftp_rename.* Použití této funkce je znázorněno v následujícím příkladu:

```
 case "prejmenovat":
          if (!@ftp_rename($conn_id,
URLDecode($_GET['adresar']),URLDecode($_GET['cesta'])."/".$_POST['j
meno'])) {
                $chyba = "Adresář se nepodařilo přejmenovat!";
            }
```

Poslední funkce, která je v aplikaci použita, je *ftp_put.* Tato funkce umožňuje nahrání souboru na server. Soubor se nejdříve nahraje do paměti pomocí formuláře, a poté tento soubor odesílá funkce *ftp_put* na server.

```
case "uloz_soubor":
   $docasny_soubor = $_FILES["userfile"]["tmp_name"];
   $jmeno_souboru = $_FILES["userfile"]["name"];
 if (!@ftp_put($conn_id, $cesta."/".$jmeno_souboru,
     $docasny_soubor, FTP_BINARY))
     {
       $chyba = "Soubor se nepodařilo přenést !";
     }
```

V této kapitole byly popsány vybrané použité funkce pro práci s FTP. Pro doplnění chybí ještě uvést, že existuje funkce, která uzavírá FTP spojení. Pro ukončení spojení mezi serverem a klientem se využívá funkce *ftp_close()*.

5.2 Instalace a nastavení

Jestliže chceme aplikaci zprovoznit na svém počítači, musíme mít nainstalovaný webový server, který podporuje programovací jazyk PHP a databázový server MySQL. Pokud máme například nainstalovaný webový server Apache musíme si ošetřit, několik jeho základních vlastností. Nastavení serveru si můžeme zjistit pomocí jednoduchého PHP skriptu.

```
<?php
 echo phpinfo();
?>
```

Pokud spustíme ve webovém prohlížeči skript z předchozího příkladu, zjistíme z výpisu, nastavení celého serveru. Jelikož jsou v aplikaci používány funkce pro práci s FTP, je důležité, aby bylo zapnuté nastavení *FTP Support enabled.* Také je důležitá zapnutá podpora takzvaných session proměnných, do kterých se ukládají přihlašovací údaje uživatelů, při přechodu mezi stránkami. Z výpisu nastavení serveru zjistíme, jestli je *Session Support* nastaven na *enabled*. Tyto parametry by měli být nastaveny na serveru standardně, ale při hostování stránek na jiném serveru si musíme ověřit, jestli by na něm mohla být aplikace provozována. Jestliže chceme uživatelům poskytnou bezpečnou komunikace se serverem musí být na serveru nastaveno šifrování. SSL šifrování se na serveru nastaví v konfiguračním souboru Apache *httpd.conf* zapnutím modulu *mod_ssl.so. V httpd.conf* musíme ještě includovat rozšířené nastavení pro nastavování SSL, to provedeme přidáním následujícího řádku:

```
Include conf/extra/httpd-ssl.conf
```

V souboru httpd-ssl.conf potom můžeme nastavovat šifrování podle vlastních potřeb. Jelikož PHP také pracuje s šifrováním, musíme v jeho konfiguračním souboru *php.ini* odkomentovat řádek s rozšířením o SSL.

Pokud jsme vykonali všechny nastavení serveru můžeme do adresáře, který je v *httpd.conf* nadefinovaný jako DocumentRoot, nahrát aplikaci.

Aby v aplikaci fungovalo přihlašování uživatelů, musíme vytvořit v databázovém serveru MySQL databázi, která bude obsahovat tabulku *uzivatele*. Pro práci s data-

bázovým server existuje aplikace phpMyAdmin, pomocí které můžeme spravovat databáze uložené na serveru. Pokud je aplikace nastavena podle kapitoly 2.2.3, spustíme ji ve webovém prohlížeči napsáním adresy *localhost/phpmyadmin do* URL prohlížeče. Po otevření aplikace máme možnost vytvořit vlastní databázi, kterou si pojmenujeme *mujprenos*. Po vytvoření se nacházíme v samotné struktuře databáze. Nyní musíme vytvořit tabulku uzivatele. Tu vytvoříme importováním textového souboru *databaze.txt*, který je připojený k aplikaci.

Po vytvoření tabulky nám už jenom zbývá nastavit konfigurační soubor aplikace *db.php*. V tomto souboru nejprve nastavíme připojení k MySQL serveru. Pokud používáme aplikaci na lokálním počítači a při instalaci jsme nezměnili název databázového serveru a jméno uživatele mohlo by nastavení vypadat takto:

```
$hostname = "localhost";
$username = "root";
$password = "";
$databaseName = "mujprenos";
```

Pokud jsme nastavili jiné přihlašovací jméno a heslo proměnným username a password, přiřadíme tyto zvolené hodnoty.

Po nastavení databáze musíme nastavit přihlašovací údaje pro FTP server. Pro nastavení nám slouží připojení nám slouží tyto proměnné:

```
$ftp_server = "";
$ftp_user = "";
$ftp_pass = "";
$poc_cesta = "/";
```

Proměnná ftp_server určuje název serveru, na který se budeme přihlašovat. Další dvě proměnný určují jméno uživatele a jeho heslo pro pro připojení k serveru. Poslední proměnná určuje počáteční cestu po přihlášení.

Poslední údaj, který musíme nastavit je ID administrátora. Pokud například chceme aby administrátorem byl uživatel, který má ID jedna, uděláme to podle následujícího příkladu:

```
$administrator = 1;
```

Nastavením konfiguračního souboru je instalace a nastavení aplikace kompletní. Nyní můžeme prostřednictvím webového prohlížeče s aplikací pracovat.

6 Závěr

Cílem této webové aplikace bylo vyřešit problém správy souborů na vzdáleném serveru tak, aby každý uživatel se mohl do aplikace přihlásit pomocí svého vlastního jména hesla. Pro implementaci aplikace jsem použil programovací jazyk PHP, zejména pro jeho přehlednost, jednoduchost a hlavně velmi dobře popsané dokumentaci. Pro ukládání dat do databáze jsem si vybral databázový server MySQL, kvůli jeho snadné dostupnosti. Aby aplikace mohla být zpřístupněna musí být interpretována nějakým webovým server.

V práci jsem se pokusil popsat nejpoužívanější webové servery na Internetu. Mezi tyto servery určitě patří server Apache, který je na internetu šířen jako open source a server IIS od společnosti Microsoft. Server, který v poslední době zaznamenal velký vzestup v počtu používání, je server od společnosti Google. Bohužel tento server je ještě ve vývoji a nepodporuje například PHP, takže pro provoz aplikace jsem si vybral nejpoužívanější server na Internetu, server Apache. Hlavní výhoda tohoto serveru je v tom, že je modulární, takže si ho může uživatel sestavit podle vlastních potřeb, což například IIS do verze 7.0 neumožňoval.

Pro přenos dat jsem použil FTP protokol, protože má plnou podporu v PHP. Jelikož je FTP jeden z nejstarších protokolů, který se používá na Internetu, je jeho zabezpečení nedostačující. Pro lepší bezpečnost dat jsem tedy použil šifrované spojení pomocí SSL.

Aplikaci jsem si otestoval na volně dostupném serveru, který SSL šifrování podporuje. Celá aplikace tedy pracuje na adrese http://mujprenos.ic.cz.

Myslím si, že aplikace má využití zejména při přenosu malých souborů. Hlavním problémem této aplikace je přenos velkých souborů. PHP je totiž většinou nastavený tak, že délka trvání jednoho skriptu je 30 sekund. Pokud se ovšem přenáší větší soubor, je tato délka překročena, kvůli tomu přenos neproběhne úspěšně. Aplikace by se dala ještě rozšířit o statistiky, co který uživatel v danou dobu na serveru vykonal. Využil by se tím více potenciál databázového serveru.

7 Zdroje

[1] *Apache HTTP Server Documentation* [online]. c2008 [cit. 2008-08-10]. Dostupný z WWW: <http://httpd.apache.org/docs/>.

[2] *IIS Internet Information Sevices* [online]. 2008 [cit. 2008-08-18]. Dostupný z WWW: <http://learn.iis.net/>.

[3] *Google App Engine* [online]. c2008 [cit. 2008-08-25]. Dostupný z WWW: <http://code.google.com/appengine/docs/>.

[4] *Sun Java System Web Server* [online]. c2008 [cit. 2008-08-25]. Dostupný z WWW: <http://www.sun.com/software/products/web_srvr/index.xml>.

[5] PETERKA, Jiří. *FTP* [online]. [2000] [cit. 2008-07-28]. Dostupný z WWW: <http://www.earchiv.cz/a93/a333c110.php3>.

[6] *SSL Tutorial* [online]. 2001 [cit. 2008-08-25]. Dostupný z WWW: <http://www2.rad.com/networks/2001/ssl/index.htm>.

[7] *DMOZ open directory project* [online]. c2008 [cit. 2008-08-25]. Dostupný z WWW: <http://www.dmoz.org/Computers/Internet/Protocols/SSH/>.

[8] *WebDAV Resources* [online]. 2007 [cit. 2008-08-25]. Dostupný z WWW: <http://www.webdav.org/>.

[9] PHP DOCUMENTATION GROUP , *PHP manual* [online] 2008 [cit. 2008-07-21] Dostupný z WWW:<http://www.php.net/docs.php>.

[10] LUBOSLAV, Lacko. *PHP5 a MySQL5*. [s.l.] : [s.n.], 2007. 320 s. ISBN 978-80-251-1695-1.

[11] WILLIAMS, Hugh E., DAVID, Lane. *PHP a MySQL : Vytváříme webové databázové aplikace*. [s.l.] : [s.n.], 2002. 530 s. ISBN 80-7226-760-4.

[12] *Jednoduchý FTP klient v PHP* [online]. 2003 [cit. 2008-08-25]. Dostupný z WWW: <http://interval.cz/serialy/jednoduchy-ftp-klient-v-php/>.

Přílohy

Příloha č. 1 – Instalace a nastavení serveru Apache

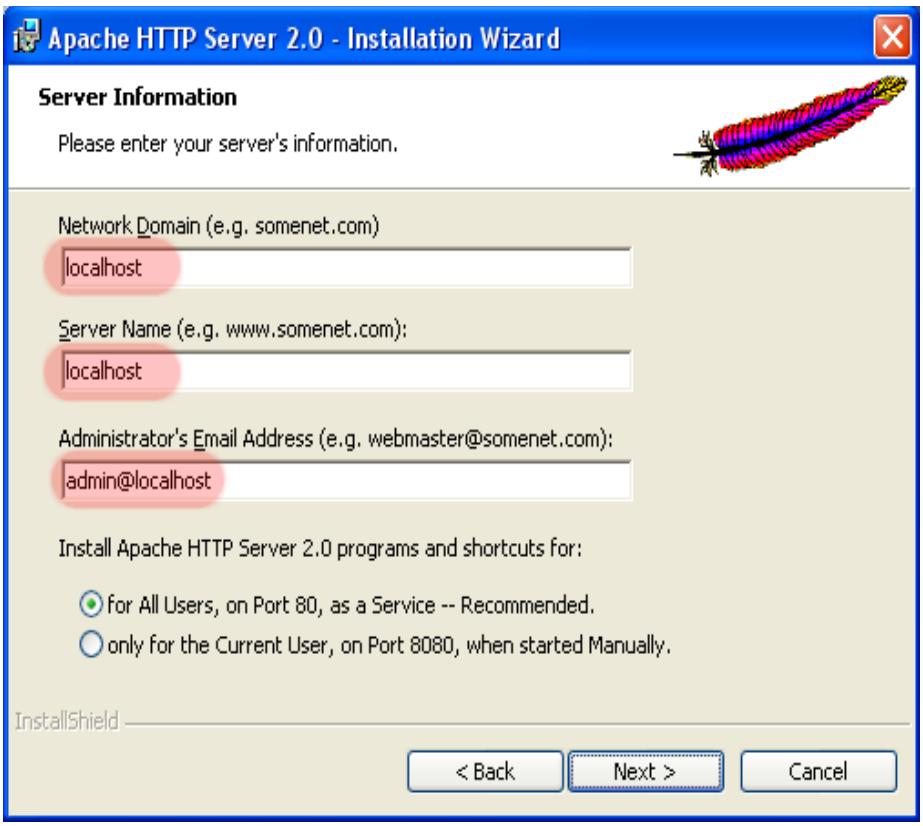
Server je multiplatformní, což znamená, že ho můžeme využít na řadě operačních systémů. Mezi nejznámější patří Microsoft Windows a OS Linux.

Pokud tedy chceme instalovat server na Microsoft Windows stáhneme si na stránkách http://httpd.apache.org/download.cgi příslušný instalační balíček. Pro Windows jsou zde dva balíčky s SSL šifrováním a bez SSL. Pro naší aplikaci využijeme balíček s SSL šifrováním, proto stáhneme tento balíček. Po stáhnutí balíčku si na disku vytvoříme adresář, kam nainstalujeme náš webový server. Nejlépe je zvolit cestu co možná nejjednodušší, kvůli pozdějšímu nastavení serveru. Cestu tedy volíme následovně:

```
c:\apache\
```

Nyní můžeme přistoupit k instalaci. Samotná instalace se skládá z několika kroků. Nejprve se nám zobrazí okno s licenčním ujednáním. Pokud chceme pokračovat musíme souhlasit. Dále se zobrazí okno o verzi Apache, toto okno potvrdíme tlačítkem *Next.*

Následuje asi nejdůležitější okno při instalaci a to nastavení doménového jména, jména serveru a e-mailu správce. Dále můžeme určit, jestli chceme server využívat pro všechny uživatele na portu 80 nebo jen pro aktuálně přihlášené uživatele s manuálním spouštěním na portu 8080. Pro nás je výhodnější první možnost. Ostatní údaje vyplníme podle Obr. 10.



Obr. 10: Ukázka nastavení serveru (Zdroj: www.programujte.com).

Dalším krokem je jaký typ instalace chceme zvolit. Na výběr je instalace *Typical* (typická) nebo *Custom* (vlastní). Jelikož bude potřeba změnit umístění cílového adresáře instalace zvolíme *Custom* (vlastní). V dalším okně máme na výběr, jaký adresář vybrat pro instalaci. Stiskneme tlačítko *Change* a vybereme adresář, který jsme si vytvořili před začátkem instalace. Nyní je všechno připraveno a stačí už jen spustit instalaci tlačítkem *Install*. Po instalaci by se měla objevit zpráva, že instalace proběhla úspěšně. Po stisknutí tlačítka *Finish se* automaticky webový server spustí. Toho si můžeme všimnout, že v trayi (oznamovací oblast u hodin) naběhne ikonka. Po kliknutí na tuto ikonku se zobrazí nabídka co s webovým serverem můžeme nyní udělat. Momentálně není server ještě nastaven, takže server zatím můžeme vypnout. To uděláme výběrem položky *Stop*.

Nyní máme vše připraveno pro nastavení serveru a pro doinstalování dalších služeb.

Nastavení serveru Apache se provádí pomocí textových souborů. Hlavní konfigurační soubor se nazývá *httpd.conf*. Tento soubor nalezneme v adresáři:

```
c:\apache\conf\
```

V tomto souboru je nutné udělat několik změn, abychom zajistili správnou funkčnost serveru. Nejdříve se musíme podívat na nastavení *ServerRoot*. Toto nastavení se nachází na *36* řádku. Pokud jsme tedy použili cestu *c:\apache\,* tak tento řádek by měl vypadat následovně:

```
ServerRoot "C:/apache"
```

Na řádku *47* můžeme nastavit na jakém portu server bude naslouchat. Standardně je to port 80, ale může se stát, že port 80 je obsazený jiným programem např. Skype. Tato chyba bývá nejčastější při neúspěšném nastavení serveru. Jestli je port obsazený zjistíme tak, že do prohlížeče napíšeme *localhost.* Když se v prohlížeči nezobrazí žádná stránka, port obsazený není a můžeme tento řádek ponechat takto:

```
Listen 80
```

Jestliže je tento port již obsazený, můžeme buď příslušnému programu, který tento port využívá přiřadit jiný port, nebo server necháme naslouchat na jiném portu např. 90. Řádek by potom vypadal následovně:

```
Listen 90
```

Poté, co jsme nastavili port, musíme nastavit jméno serveru, které budeme používat. Toto nastavení nalezneme na řádku *171.* Tento řádek by měl mít tuto podobu:

```
ServerName localhost:80
```

Číslo, které se nachází za dvojtečkou, určuje číslo portu, takže pokud jsme v předchozím kroku změnili port, na kterém bude server naslouchat, zadáme tento port např. 90.

V dalším kroku v nastavení musíme určit z jakého adresáře se budou načítat webové stránky. Můžeme zvolit standardní cestu, ale můžeme také zvolit kterékoliv místo na disku. Řádek může vypadat např. takto:

```
DocumentRoot "C:/apache/htdocs"
```

Pro funkční nastavení webového serveru výše popsané kroky stačí. V další kapitole webový server rozšíříme o skriptovací jazyk PHP, databázový server MySQL a o aplikaci na správu MySQL, phpMyAdmin.

Příloha č. 2 – Obsah přiloženého CD

Na přiloženém CD jsou vytvořeny složky Aplikace, Bakalářská práce a Obrázky.

- **Složka Aplikace**

Tato složka obsahuje kompletní aplikaci pro správu souborů.

- **Složka Bakalářská práce**

V této složce je uložená bakalářská práce ve formátu pdf.

- **Složka Obrázky**

Složka obrázky obsahuje všechny obrázky, které byly použity v bakalářské práci.