TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: B2612 – Elektrotechnika a informatika

Studijní obor: 1802R022 – Informatika a logistika

Vizualizace DNS transakcí

DNS transactions visualisation

Bakalářská práce

Autor: **Ondřej Slavík**

Vedoucí práce: doc. RNDr. Pavel Satrapa, Ph.D.

Konzultant: Mgr. Martin Slavík

V Liberci 19. 5. 2011

Originální zadání

Prohlášení

Byl(a) jsem seznámen(a) s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 o právu autorském, zejména § 60 (školní dílo).

Beru na vědomí, že TUL má právo na uzavření licenční smlouvy o užití mé bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom(a) toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Bakalářskou práci jsem vypracoval(a) samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

Datum

Podpis

Abstrakt a klíčová slova (v českém jazyce)

Bakalářská práce popisuje tvorbu programu pro vizualizaci komunikace mezi DNS servery a získávání dalších informací distribuovaných pomocí systému DNS a možnosti jejich využití. Práce vznikala v roce 2011 s přihlédnutím k aktuálním tématům a potřebám na TUL.

V úvodní části autor vysvětluje volbu tématu, rozebírá zadání a cíl práce a soustředí se na teoretické aspekty v oblasti výzkumu DNS transakcí např. na vazbu mezi „jménem počítače“ a IP adresou.

V druhé části již popisuje samotné DNS transakce, hierarchii domén a syntaxi pro domény. Dále se věnuje resolveru, jmenným serverům, zdrojovým záznamům a použitým programovacím jazykům.

Autor se pokouší o rozčlenění programovací jazyků dle relevance a vlastního užití v rámci projektu, který probíhal před samotnou tvorbou bakalářské práce. Popisuje vývojová prostředí jak pro HTML, JavaScript a PHP, tak pro CSS. V této části se bakalářské práce rovněž věnuje grafickému rozhraní celé aplikace, systému souborů v programu a přikládá ukázky kódu pro „form.php, uvod.php, index.php a pozadi.css“. Dále je ověřena vlastní funkčnost celé aplikace na příkladu několika IP adres.

V závěrečné části autor popisuje možnosti dalšího využití či případného rozšíření programu o možnosti interaktivní volby a sumarizuje výsledky celého projektu.

Klíčová slova: DNS, vizualizace, transakce, jmenný server, doména

Abstrakt a klíčová slova (v anglickém jazyce)

This bachelor's thesis describes the development of programs for visualizing the communication between DNS servers and obtaining other information distributed via DNS and its potential applications. The thesis was written in 2011 with regard to current issues and needs at TUL.

In the introductory part the author explains the choice of topic, and discusses the assignment and its goal, focusing on the theoretical aspects of DNS transactions, such as the relation between a computer's "hostname" and IP address.

The second part describes the DNS transaction as such, domain hierarchy and domain syntax. It also includes discussion of the resolver, name servers, source records and programming languages in use.

The author attempts to sort programming languages according to their relevance and use in the project, which preceded the actual creation of this thesis. It describes development environments for both HTML, JavaScript and PHP, and CSS. In this part of the thesis he also deals with the graphical interface of the applications and its file system.Attached is a code sample for "Form, Introduction, Index and Background." The application functionality is verified using several IP addresses as an example.

In the final part the author describes the possibilities of further use of the program or its expansion with interactive options. The results of the whole project are summarized.

Key words: DNS, visualisation, transactions, name server, domain

Obsah

Seznam obrázků

Seznam použitých zkratek

DNS - Domain Name System (systém doménových jmen)

HTML - HyperText Markup Language (značkovací jazyk pro hypertext)

HTTP - HyperText Transfer Protocol (protokol pro výměnu dokumentů v HTML)

CSS - Cascading Style Sheets (kaskádové styly)

PHP - Hypertext Preprocessor (původně Personal Home Page)

RR - Resource Records (zdrojové záznamy, věty)

TTL - Time To Live

TLD - Top Level Domain (doména nejvyššího řádu)

SW – Software

HW - Hardware

NS - Name Server (jmenný server)

ROOT NS - Root Name Server (kořenový jmenný server)

TCP/IP - Transmission Control Protocol/Internet Protocol (primární transportní protokol - TCP/protokol síťové vrstvy - IP).

FTP - File Transfer Protocol (protokol pro přenos souborů)

1 Úvod

1.1 Co je to DNS

V úvodu mé práce bych rád popsal co je to DNS, k čemu slouží a jak funguje, abych alespoň trochu přiblížil o čem tato práce je. Všechny aplikace, které v Internetu zajišťují komunikaci mezi počítači, používají k identifikaci komunikujících uzlů IP adresu. Pro člověka jako uživatele jsou však IP adresy těžko zapamatovatelné. Proto se používá místo IP adresy název síťového rozhraní. Pro každou IP adresu máme zavedeno jméno přesněji řečeno doménové jméno. Toto doménové jméno můžeme používat ve všech příkazech, kde je možné použít IP adresu. (Výjimkou, kdy se musí jedině použít IP adresa, je specifikace samotného jmenného serveru.)

Jedna IP adresa může mít přiřazeno i několik doménových jmen.

Vazba mezi jménem počítače a IP adresou je definována v DNS databázi. DNS (Domain Name System) je celosvětově distribuovaná databáze. Databáze DNS obsahuje jednotlivé záznamy, které se také nazývají „DNS věty“ (Resource Records – RR). Jednotlivé části této databáze jsou umístěny na tzv. jmenných serverech.[1]

1.2 Cíl práce

Cílem práce bylo vytvořit program na vizualizaci komunikace mezi DNS servery, tedy zobrazit hierarchie serverů a zjistit tak, v jaké posloupnosti je třeba se ptát, abych dostal správnou odpověď na mnou položený dotaz. Pro splnění tohoto úkolu jsem si vybral prostředí webových aplikací a s ním spojené programovací jazyky HTML, CSS, JavaScript, PHP atd. Samotné výsledky práce jsou poté prezentovány ve webovém rozhraní, kde vkládáme dotazy a poté se nám zobrazí vizuálně zprostředkované výsledky.

1.3 Proč jsem zvolil toto téma?

Problematika DNS mne zajímala již delší dobu, ale nikdy jsem neměl příležitost se jí hlouběji věnovat, proto mne toto téma zaujalo na první pohled. Dalším důvodem proč jsem si vybral práci na toto téma, bylo to, že jsem chtěl vylepšit své znalosti protokolů

[1] **Dostálek, Libor a Kabelová, Alena.** *Velký průvodce protokoly TCP/IP a systémem DNS.* Brno : Books, a.s., 2005. (str. 245) , ISBN 80-7226-675-6.

TCP/IP, systému DNS a také využít a následně dále prohloubit své předchozí zkušenosti v programování webových aplikací a vylepšit své schopnosti při použití jazyka PHP, JavaScript a dalších.

2 Rozbor zadání, tvorba programu a popis jeho funkce

2.1 DNS transakce

2.1.1 Hierarchie domén

K tomu abychom pochopili co to je DNS transakce (dotaz), potřebujeme nejdříve vědět, jak Internet funguje a jak se v něm komunikuje.

Celý jmenný prostor DNS je rozdělen do tzv. domén, tj. skupin jmen, která k sobě logicky patří. Domény specifikují, patří-li jména jedné firmě, jedné zemi apod. V rámci domény můžeme vytvářet podskupiny, tzv. subdomény, např. doméně firmy lze vytvořit subdomény pro oddělení. Z jednotlivých „jmenovek (label)“ je pak složeno doménové jméno uzlu. Např. uzel se jménem jakub.firma.cz je uzel se jménem jakub v subdoméně firma domény.cz.

Doménové jméno se skládá z řetězců vzájemně oddělených tečkou. Jméno se zkoumá zprava doleva. Nejvyšší instancí je tzv. kořenová (root) doména, která se vyjadřuje tečkou zcela vpravo (tato tečka bývá často vypouštěna). V kořenové doméně jsou definované domény nejvyšší (první) úrovně (Top Level Domains – TLD). Pro Českou republiku je vyhrazena doména .cz [2]

TLD se dále dělí na tyto:

- **Národní TLD** (*country-code TLD, ccTLD*) seskupující domény jednoho státu. Mají dvoupísmenný název, až na výjimky odpovídající kódu země podle normy ISO 3166-1
- **Generické TLD** (*generic TLD, gTLD*) seskupující obecné domény (např. `org` pro *neziskové organizace*), nejsou spojené s jedním určitým státem (až na výjimku TLD *mil*, *gov* a *edu* které jsou z historických důvodů vyhrazeny pro vojenské, resp. vládní počítačové sítě v USA)
- **Infrastrukturní TLD** používané pro vnitřní mechanismy Internetu. V současné době existuje pouze jediné takové TLD: `arpa` a root používané systémem DNS.

[2] **Dostálek, Libor a Kabelová, Alena.** *Velký průvodce protokoly TCP/IP a systémem DNS.* Brno : CCP Books, a.s., 2005. (str. 246), ISBN 80-7226-675-6.

Generická doména nejvyššího řádu (anglicky generic top-level domain, gTLD) je doménou nejvyššího řádu (alespoň teoreticky), která je společná pro daný typ objektů. Její název je nejméně třípísmenný.

Dělení gTLD:

- Neomezené - jejich použití není nijak omezeno
  .com .info .net .org .eu
- Vyhrazené - jsou určeny pro specifický účel, ale není to vyžadováno
  .name
- Vymezené - mohou být použity pouze pro daný účel
  .biz .int .pro .xxx
- Garantované - mohou být využity pouze pro daný účel, navíc mají garanta – organizaci stanovující podmínky registrace a dohlížející na provoz. Označují se sTLD (podle sponsored TLD, kde sponsor znamená garant)
  .aero .cat .coop .jobs .mobi .museum .travel
- Exkluzivní - tyto domény smějí využívat pouze servery v USA, navíc s určitými restrikcemi
  .edu .gov .mil
- Infrastrukturní - neveřejné
  .arpa .root
- Zrušené - .nato
- Rezervované - .example .invalid .localhost .test
- Připravované - .tel .post
- Navržené - .asia .geo .kid .mail .sco .web
- Pseudodomény - .bitnet .csnet .onion .uucp

2.1.2 Syntaxe pro domény

Celé jméno domény může obsahovat maximálně 255 znaků, jednotlivá „jmenovka“ poté maximálně 63 znaky. Řetězec může být složen z písmen, číslic a pomlčky. Pomlčka se nesmí vyskytovat na začátku ani na konci řetězce. Existují i rozšíření určující bohatší repertoár znaků možných pro tvorbu jmen. Ze zásady se však těmto dalším znakům snažíme vyhnout, protože toto rozšíření podporují jen některé aplikace a prohlížeče.

Mohou se použít velká i malá písmena, ale není to zase tak jednoduché. Z hlediska uložení a zpracování v databázi jmen (databázi DNS) se velká a malá písmena nerozlišují. Tj. jméno Newark.com bude uloženo v databázi na stejné místo jako NewYork.com nebo NEWYORK.com atp. Tedy při překladu jména na IP adresu je jedno, kde uživatel zadá velká a kde malá písmena. Avšak v databázi je jméno uloženo s velkými a malými písmeny, tj. bylo-li tam uloženo např. NewYork.com, pak při dotazu databáze vrátí NewYork.com. Poslední tečka je součástí jména.[3]

2.1.3 Reverzní domény - stromová struktura IP - adres

Reverzními doménami jsou nazvány ty domény, které jsou vytvářeny IP adresou. Stejně jako jmenné domény vytvářejí stromovou strukturu. Pro potřeby reverzního překladu byla utvořena pseudodoména „in-addr.arpa“, pro IPverze6 pak „IP6.arpa“. Tato pseudodoména má jméno historického původu, jde o zkratku „inverse addresses in the Arpanet“.

Pod doménou in-addr.arpa jsou domény, které se jmenují jako první číslo z IP adresy sítě. Tj. doména in-addr.arpa má subdomény 0 až 255. Každá z těchto subdomén obsahuje nižší subdomény opět 0 až 255 atd. Např. síť 195.47.37.0/24 patří do domény 37.47.195.in-addr-arpa. Ta sama patří do domény 47.195.in-addr.arpa atd. Všimněte si, že domény jsou zde tvořeny jakoby IP-adresami sítí psanými ale pozpátku.[4]

Rezervované domény a pseudodomény

Později se ukázalo, že jako TLD je možné využít i jiné domény. Některé další TLD byly rezervovány RFC-2606:

doména .test pro testování,

doména .example pro vytváření dokumentace a příkladů,

doména .invalid pro navozování chybových stavů,

doména .localhost pro softwarovou smyčku.

Obdobně byla rezervována doména .local pro intranety. Význam této domény je obdobný jako význam sítě 10.0.0.0/8.[5]

[3] **Dostálek, Libor a Kabelová, Alena.** *Velký průvodce protokoly TCP/IP a systémem DNS.* Brno : CCP Books, a.s., 2005. (str. 247) , ISBN 80-7226-675-6.
[4] Tamtéž (str. 248)
[5] Tamtéž (str. 250)

2.1.4 Resolver

A nyní, když je popsána datová struktura DNS, tak se můžeme dostat k popisu samotné funkčnosti a pokládání dotazů (překladů). Přeložení jména na IP adresu zprostředkovává tzv. resolver. Resolver není samostatná aplikace, ale je to klient (obvykle součást operačního systému), který se dotazuje jmenného serveru. Jelikož je databáze celosvětově distribuována, nemusí nejbližší jmenný server znát konečnou odpověď a může požádat o pomoc další jmenné servery. Získaný překlad pak jmenný server vrátí jako odpověď resolveru. Veškerá komunikace se skládá z dotazů a odpovědí.

Jmenný server během svého startu načte do paměti data pro zónu, kterou spravuje. Primární jmenný server načte data z lokálního disku, sekundární jmenný server získá pro spravované zóny dotazem zone transfer data z primárního jmenného serveru a rovněž je uloží do paměti. Tato data primárního a sekundárního serveru se označují jako autoritativní (nezvratná). Dále jmenný server načte z lokálního disku do paměti data zóny cache/hint, která nejsou součástí dat jeho spravované zóny, ale umožní mu spojení s kořenovými (root) jmennými servery. Tato data se označují jako neautoritativní. [6]

Jak jsem již dříve zmínil resolver není konkrétní program, jedná se o skupinu knihovních funkcí, která se přizpůsobuje (linkuje) konkrétní aplikaci, která si vyžádala tuto službu. Může se jednat o aplikace jako je například ftp, telnet, www prohlížeč a další. Každá z těchto aplikací si v případě potřeby vybere z knihovních funkcí ty potřebné a s jejich pomocí vytvoří dotaz, který je poté odeslán na DNS server. Dalším aspektem při překladu jsou časová omezení. Je totiž možnost, že na dotaz nedostane resolver odpověď, ale při opakování toho stejného dotazu po nějaké, relativně krátké době, již dostaneme korektní odpověď. Tato situace je způsobena tím, že server v mezičase získal odpověď, ale protože ta z jiného jmenného serveru dlouho nepřicházela, tak první dotaz nebyl zodpovězen.

Ještě bych se rád stručně zmínil o vlastnostech resolveru a jeho konfiguraci. Možnosti konfigurace závisí na operačním systému, na kterém pracujeme. Nejdůležitější atribut, jak při konfiguraci na Unixu, Windows a dalších systémech, je korektní nastavení adresy (IP adresy, ne doménové jméno!) jmenného serveru, ke kterému bude resolver přistupovat

[6] **Dostálek, Libor a Kabelová, Alena.** *Velký průvodce protokoly TCP/IP a systémem DNS.* Brno : CCP Books, a.s., 2005. (str. 251) , ISBN 80-7226-675-6.

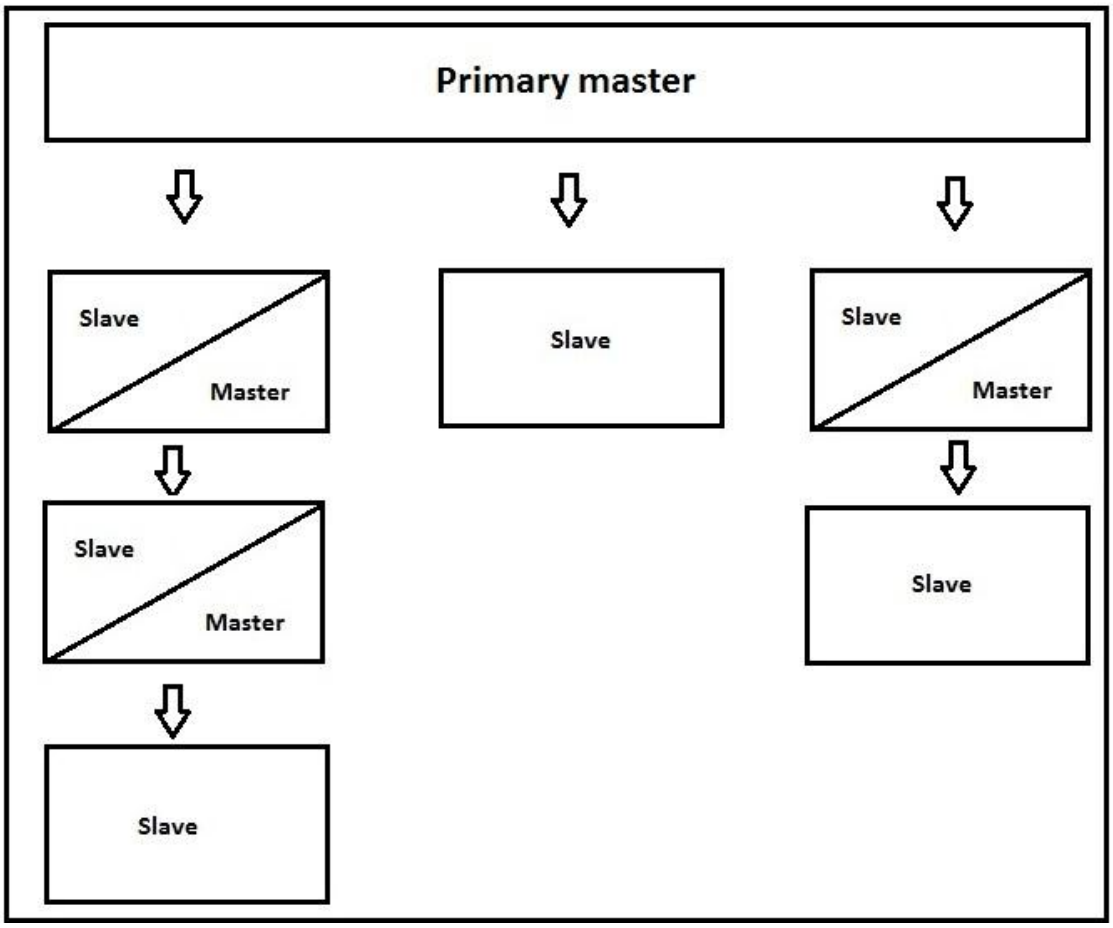
2.1.5 Jmenný server

Na jmenném serveru jsou uloženy informace pro překlad názvů počítačů na IP adresy (resp. i naopak pro reverzní překlad). Jmenný server se stará o určitou část z prostoru jmen všech počítačů. Tento prostor, o který se stará, nazýváme zóna. Tato zóna je tvořena celou doménou nebo pouze její částí. Pomocí věty NS (v konfiguraci jmenného serveru) je totiž možné delegovat udržování subdomény na jmenný server nižší úrovně. Tím je subdoména vyjmuta ze zóny serveru a je delegována na server „nižší úrovně “, který jí nyní bude spravovat. Jmenné servery můžeme rozčlenit podle typu uložení dat. Máme v podstatě 6 typů serverů (viz Tab. 1)

Tab. 1: Rozdělení jmenných serverů podle typu uložených dat

Typ serveru	Typ uložení dat
Primární jmenný server	Je to pro zónu primární zdroj, autoritativní server zóny. Data o své zóně čerpá z databází, které jsou uloženy na lokálních discích. Podle verze Bindu (Berkeley Internet Name Domain) se označuje buď jako primary master nebo jako primární jmenný server. Databáze tohoto serveru jsou ručně vytvářené správcem. Tento primární server musí být uveden v NS záznamu jako autoritativní jmenný server (pro každou zónu je jen jeden).
Sekundární jmenný server/slave server	Data o doméně extrahuje z primárního jmenného serveru. Sekundární server má svou kopii dat (databáze), kterou aktualizuje při změně; na tu je buď aktivně upozorněn primárním serverem (NOTIFY) nebo se sám dotáže na SOA záznam a pokud došlo ke změně sériového čísla, stáhne si aktuální verzi (AXFR/IXFR) A když jsou tyto databáze uložené na disku, tak se již dále needitují, protože se v další intervalu vše stáhne z primárního serveru znovu. Tyto servery jsou autoritativní pro svou zónu.
Caching only server	Pro danou doménu neplní funkci ani primárního ani sekundárního jmenného serveru. Používá obecnou vlastnost jmenných serverů, tzn., že data, která jím putují, ukládá ve své paměti (existuje ale i řada autoritativních serverů bez cache - např. software NSD žádnou cache nemá) Tato data se poté označují jako neautoritativní. Server může vykonávat i více funkcí najednou. Může být pro některé domény primární, sekundární pro další a caching only pro zbytek domén.
Root name server	Jmenný server starající se o root doménu. Všechny tyto root servery jsou zároveň i primární servery a právě tím se od ostatních jmenných serveru odlišuji.
Stealth server	Tajný server, typ serveru, který není nikde zveřejňován. Vědí o něm pouze ty servery, které mají jeho IP adresu zanesenu v konfiguraci. Je to autoritativní server a data pro jemu přidělenou zónu získává pomocí zónového přenosu. Používá se z důvodu bezpečnosti (není na očích, tudíž se snižuje riziko útoku) nebo proto, že mezi ním a veřejnými servery probíhá nějaké další zpracování DNS dat, typicky DNSSEC podpis.

Jeden jmenný server může být pro určitou zónu master server a pro nějaké je slave serverem. Primární master je však pro každou zónu jen jeden. Tuto architekturu nám více přiblíží diagram (obr. 1).



Obr. 1 – Master/slave architektura

Z hlediska uživatelů je jedno zda komunikuje s master či slave serverem, pro určenou zónu jsou oba dva autoritativními servery, mají data stejné váhy. Klient ani neví (z DNS odpovědi se ani nedozví), s jakým typem serveru komunikuje. Oproti tomu caching server poskytuje neautoritativní odpovědi a pokud odpověď nezná, kontaktuje pro danou zónu autoritativní server. Jak již bylo řečeno, data jsou uložena na lokálním disku, ale databáze kořenový serverů, pro které není autoritou, se zavádí do sekundární paměti příkazem cache. Jmenný server (označovaný též DNS server) naplňuje svou paměť několika způsoby. Autoritativní data načte ze souborů na disku, nebo je získá pomocí dotazu *zone transfer* z paměti jiného serveru. Neautoritativní data jmenný

server získává postupně z paměti jiných serverů, tak jak vyřizuje jednotlivé DNS dotazy.[7]

2.1.6 Zdrojové záznamy (věty) - RR

Informace o doménových jménech a jim příslušných IP adresách, stejně tak jako všechny ostatní informace distribuované pomocí DNS, jsou uloženy v paměti jmenných serverů ve tvaru zdrojových vět (Resource Records – RR).

V případě, že DNS klient potřebuje získat informace z DNS, pak požaduje po jmenném serveru věty RR podle zadaných požadavků. [8]

Klientem, o kterém mluvíme, bývá vždy resolver, buď na straně uživatele, případně resolver použitý jiným jmenným serverem, který na daný dotaz nezná odpověď. V protokolu DNS mají všechny zdrojové věty stejnou strukturu a skládají se z následujících polí:

- NAME – jméno domény
- TYPE – typ věty
- CLASS – třída věty
- TTL – Time to live. Doba, po kterou bude tento záznam v cache serveru uváděn jako platící. Po vypršení této doby je záznam neplatný
- RDLENGHT – určuje délku pole RDATA
- RDATA – řetězec měnící se délky. Ta závisí na typu a třídě RR

Pro větší přehlednost je níže uveden přehled RR vět podle typu v Tab. 2.

[7] **Dostálek, Libor a Kabelová, Alena.** *Velký průvodce protokoly TCP/IP a systémem DNS.* Brno : CCP Books, a.s., 2005. (str. 263) , ISBN 80-7226-675-6.
[8] Tamtéž

Tab. 2: Typy RR vět

Typ	**Název (anglický)**	**Popis pole**
A	address	Obsahuje IPv4 adresu přiřazenou danému jménu (32 bitů)
AAAA	IPv6 address	Obsahuje IPv6 adresu (128 bitů)
NS	name server	Obsahuje název autoritativního DNS serveru, který je autoritou pro danou doménu
CNAME	canonical name	Alias - jiné jméno pro již existující a známé jméno, využívá se pro servery zavedených služeb, jako je například WWW. Jeho určení pomocí přezdívky jej později dovoluje jednoduše přemístit na jiný počítač
MX	mail exchange	Ukazuje adresu a prioritu serveru pro příjem elektronické pošty pro danou doménu. Tentokrát obsahuje dvě pole - preference (přirozené číslo, menší znamená vyšší prioritu) a doménové jméno e-mailového serveru
SOA	**s**tart **o**f **a**uthority	Je uvozující záznam zónového souboru. Obsahuje název primárního serveru, adresu elektronické pošty jejího správce (zavináč nahrazen tečkou) a dále: *Serial* - sériové číslo (potřeba zvýšit s každou změnou v záznamu), podle něj sekundární server pozná, že je doména změněna; *Refresh* - interval, v kterém se má sekundární server ptát na novou verzi zóny (v sekundách); *Retry* - jak často má sekundární server opakovat své pokusy, pokud se mu nedaří spojit s primárním *Expire* - čas po kterém jsou záznamy na sekundárním serveru označeny za neaktuální (nedaří se kontaktovat primární server) *TTL* - implicitní doba platnosti záznamů
PTR	pointer	Zvláštní typ záznamu pro reverzní zóny (použití pro reverzní překlad)
SRV		specifikace umístění konkrétní služby (podle čísla portu) na dané doméně
TXT	TeXT	Textová poznámka
NXT	Next domain	Další doménové jméno
AXFR		Požadavek na získání transferu celé zóny
IXFR	Incremental Zone Transfer	Požadavek na získání inkrementálního zone transferu
*		Získání všech vět

2.2 Použité programové vybavení (programovací jazyky, nástroje pro programování)

2.2.1 HTML

HyperText Markup Language, označovaný zkratkou HTML, je značkovací jazyk pro hypertext. S jeho pomocí se vytvářejí stránky v systému World Wide Web, který umožňuje prezentaci na Internetu. Jazyk je aplikací dříve vytvořeného rozsáhlého univerzálního značkovacího jazyka SGML (Standard Generalized Markup Language). Vývoj HTML zpětně ovlivňovaly webové prohlížeče.

V roce 1990 byl navržen jazyk HTML a protokol pro jeho přenos v počítačové síti – HTTP (HyperText Transfer Protocol – přenosový protokol hypertextu). Zároveň také Tim Berners-Lee vytvořil první webový prohlížeč, který nazval WorldWideWeb.

HTML je charakterizován množinou značek (tzv. *tagů*) a jejich atributů lišících se podle verze. Mezi tagy se uzavírají části textu dokumentu a tím se určuje jejich význam (*sémantika*). Názvy jednotlivých značek se umisťují mezi úhlové závorky < a >. Část dokumentu skládající se z otevírací značky, nějakého obsahu a odpovídající ukončovací značky tvoří tzv. *element* (prvek) dokumentu.

Dokument v jazyku HTML má předepsanou strukturu:

- Kořenový element – element `html` (značky `<html>` a `</html>`) uvozuje a končí celý dokument.
- Hlavička elementu – obsahuje metadata, vztahující se k celému dokumentu. Definují např. název dokumentu, jazyk, kódování, klíčová slova, popis, použitý styl zobrazení. Hlavička je uzavřena mezi tagy `<head>` a `</head>`.
- Tělo dokumentu – obsahuje vlastní text dokumentu. Tvoří se značkami `<body>` a `</body>`.

2.2.2 Cascading Style Sheets (CSS) - Kaskádové styly

CSS popisuje vzhled webů, odděluje obsah a prezentaci webové stránky. Styly jsou kaskádové, protože je možno je aplikovat z externího souboru, zevnitř sekce stylů webové stránky nebo z řádky a každá z nižších úrovní stylů přepisuje jakoukoliv dříve definovanou stylovou charakteristiku.[9]

[9] **Lavin, Peter.** *PHP objektově orientované.* Praha : Grada Publishing, a.s., 2009. (str. 192), ISBN 978-80-247-2137-8.

Soubor kaskádových stylů se sestává z několika pravidel. Každé pravidlo zahrnuje selektor a blok deklarací. Každý blok deklarací pak má seznam deklarací oddělených středníky ; a každá deklarace sestává z identifikátoru vlastnosti, následuje dvojtečka : a hodnota vlastnosti. Nepovinně ještě může následovat označení !important, které zvýší sílu deklarace.

CSS definuje mnoho různých selektorů, které obvykle můžeme kombinovat. Mezi základní patří:

- `body` – Tyto deklarace platí pro všechny výskyty elementu `body`.
- `body p` – Tyto parametry budou určující pro všechny elementy `p`, které se nachází v elementu `body`, v jakékoliv hloubce.
- `body>div` – Tyto deklarace budou určovat všechny elementy `div`, které jsou potomky elementu `body`. Tudíž pokud bychom měli element `div`, který se nachází v `<body><blockquote><div>…`, tyto deklarace by pro něj neplatily, protože tento `div` není přímým dítětem elementu `body`.
- `.trida` – Tyto deklarace budou platit pro všechny elementy, které mají v HTML nastavenou třídu `trida`. Toho se využívá pomocí tagu atributem `class`.

2.2.3 JavaScript

Je to multiplatformní, objektově orientovaný skriptovací jazyk. Autorem je Brendan Eich z tehdejší společnosti Netscape.

JavaScript je určen především k programování aktivních částí WWW stránek. V omezené míře jej lze použít i k vytváření skriptů na straně serveru či dokonce pro normální aplikace.[10]

Jsou jím obvykle řízeny různé aktivní prvky GUI (tlačítka, textová políčka) nebo tvořeny animace a efekty obrázků. Jeho syntaxe poněkud připomíná C/C++/Java Slovo Java je v jeho názvu pouze z marketingových důvodů a s Javou jako programovacím jazykem jej vedle názvu spojuje jen místy podobná syntaxe. JavaScriptu byl ustaven standart asociací ECMA (Europen Computer Manufacturers Association) v červenci 1997 a v srpnu 1998 organizací ISO (International Organization for Standardization).

[10] **Satrapa, Pavel.** *Web design.* Havlíčkův Brod : Neokortex spol. s.r.o., 1997.(str. 269), ISBN 80-902230-1-X.

Program v JavaScriptu obvykle nabíhá až po stažení WWW stránky z Internetu (tzv. na straně klienta), na rozdíl od ostatních jiných interpretovaných programovacích jazyků (např. PHP a ASP), které pracují na straně serveru ještě před stažením z Internetu. Z toho plynou jistá bezpečností omezení, JavaScript např. nemůže pracovat se soubory, aby tím nenarušil soukromí uživatele

JavaScript je jednoduchý skriptovací jazyk, využívající objekty, určený pro snadné webové programování. Ačkoliv se i v JavaScriptu objevují prvky objektově orientovaných jazyků, má blíže ke skriptovacím jazykům jako je Perl. Hodí se k provádění jednoduchých úkolů, například ke kontrole údajů ve formuláři, dynamickému generování HTML kódu a k základním výpočtům, týkajícím se data a času nebo typu prohlížeče.[11]

2.2.4 PHP

Ačkoliv se na vývoji PHP podílí několik autorů, jeho skutečným duchovním otcem je Rasmus Lerdorf. První parser pro PHP napsal v roce 1995 jako CGI skript v Perlu, kterému říkal „Personal Home Page“, nebo jen PHP. Původně jej používal pro webovou knihu návštěv.

PHP je vedle Active Server Pages (ASP), a Perlu jedním z nejrozšířenějších serverových skriptovacích jazyků. PHP se vyznačuje neuvěřitelnou rozmanitostí a lze jej použít i pro samostatné, s webem nesouvisející, aplikace. Nejčastěji ale bývá používán pro dynamické předzpracování stránek na unixových serverech (především u Apache; viz http://www.apache.org). Jedná se o nejpoužívanější rozšíření pro Apache.[12]

Syntaxe jazyka čerpá z několika programovacích jazyků (Perl, C, Pascal a Java). PHP je nezávislý na platformě, rozdíly v různých operačních systémech jsou zanedbatelné a omezují se na několik OS-závislých funkcí a skripty lze většinou mezi nimi přenášet bez jakýchkoli změn.

PHP podporuje mnoho knihoven pro různé účely - např. zpracování textu, grafiky, práci se soubory, přístup k většině databázových systémů (mj. MySQL, ODBC, Oracle, PostgreSQL, MSSQL), podporu celé řady internetových protokolů (HTTP, SMTP, SNMP, FTP, IMAP, POP3, LDAP, …).

---

[11] **McClure, Stuart, Shah, Saumil a Shah, Shreeraj.** *Web Hacking: Útoky a Obrana.* Praha : SoftPress s.r.o., 2003. (str. 59), ISBN 80-86497-53-4.

[12] Tamtéž (str. 43)

PHP je vedle ASP, jedním ze dvou nejrozšířenějších skriptovacích jazyků pro web na světě. Oblíbeným se stal hlavě díky jednoduchosti použití, bohaté zásobě funkcí, a tomu, že kombinuje vlastnosti více programovacích jazyků a je tak tolerantnější k vývojáři, který má částečnou svobodu v syntaxi. V kombinaci s operačním systémem Linux, databázovým systémem (obvykle MySQL nebo PostgreSQL) a webovým serverem Apache je často využíván k tvorbě webových aplikací. Pro tuto kombinaci se vžila zkratka LAMP – tedy spojení Linux, Apache, MySQL a PHP nebo Perl. V PHP se tvoří i ty největší internetové projekty.

Výhody PHP:

- PHP je specializované na webové stránky.
- Rozsáhlý soubor funkcí v základní knihovně PHP (přes pět a půl tisíce).
- Nativní podpora mnoha databázových systémů.
- Multiplatformost (zejména Linux a Microsoft Windows).
- Šance použití nativních funkcí operačního systému (poté ale hrozí nekompatibilita s jiným OS).
- Velká podpora na hostingových službách – PHP je v podstatě standardem, který najdeme všude.
- Velké množství projektů a kódů, které lze zdarma využít (WordPress, phpBB a další).
- Velice slušná dokumentace.

2.2.5 Vývojové prostředí pro HTML, JavaScript a PHP – PSPad

PSPad je celosvětově rozšířený freewarový textový editor a editor zdrojových kódů pro platformu Microsoft Windows vyvíjený v prostředí Delphi. Program vyvíjí český programátor Jan Fiala, první verze vyšla v roce 2001.

Program neobsahuje nekonečnou řadu zbytečných funkcí, které většina z nás stejně nikdy nevyužije a soustředí spíše na jednoduchost, přehlednost a rychlost, to je také jeden z důvodů, proč jsem si ho tolik oblíbil při tvorbě webových aplikací. Rád bych vyzdvihl některé výhody toho programu:

- neomezená délka editovaného textu
- FTP klient - umožňuje editovat soubory přímo z webu

- umožňuje otevřít víc dokumentů současně
- zvýraznění syntaxe s automatickým nastavením dle typu souboru
- zvýraznění skriptů v rámci HTML
- porovnání obsahu textových souborů,
- export včetně zvýraznění do RTF a HTML
- definice externích programů, ve kterých je možné soubor otevřít, kompilace…,
- sloupcové a řádkové bloky, záložky v textu,
- jednoduchá integrace do kontextové nabídky systému Windows,
- přeformátování, komprese a kontrola HTML kódu,
- interní HTML prohlížeč,
- změny velikosti písmen, odstranění a přidání diakritiky, formátování kódu

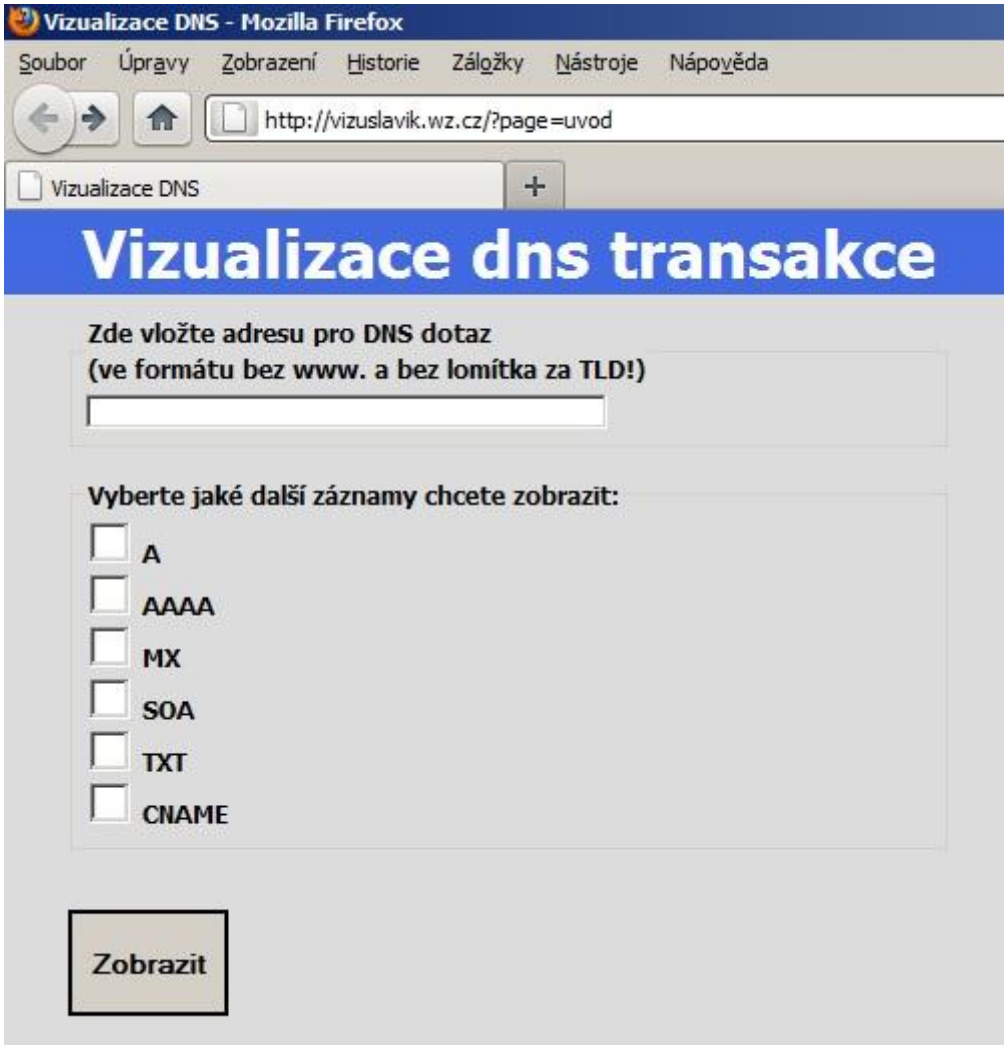
2.2.6 Vývojové prostředí pro CSS – TopStyle Lite

TopStyle je program vytvořený v první řadě pro tvorbu CSS. Hlavní výhodou je, že program pomáhá k tvorbě takového kódu, který se správně zobrazuje ve všech předem definovaných prohlížečích. Výhody použití tohoto softwaru:

- Barevné rozlišení. Rozlišení uživatelem měnit nelze, ale můžeme pozměnit velikost a font písma.
- Automatické mezery při začátku dokumentu. Jednak se může nastavit tzv. strouha, což jsou tři mezery od okraje dokumentu (implicitně definované, což je, že je nemůžeme pozměnit), ale automatické tři mezery po zmáčknutí enteru v CSS Definici.
- Seznam všech atributů a hodnot atributů.
- Průvodce vytváření selektorů.
- Zjištění kompatibility s mnoha verzí prohlížečů.
- Možnost integrace do jiných editorů.
- Zabudovaný W3C validátor.
- Opravář chyb, který je velmi užitečný. Najde chybky, které W3C Validátor neoznačí za chybu.
- Paleta CSS barev. Tj. Barvy, které se vyskytují ve vašem CSS dokumentu. Napíše počet a příslušný kód barvy.

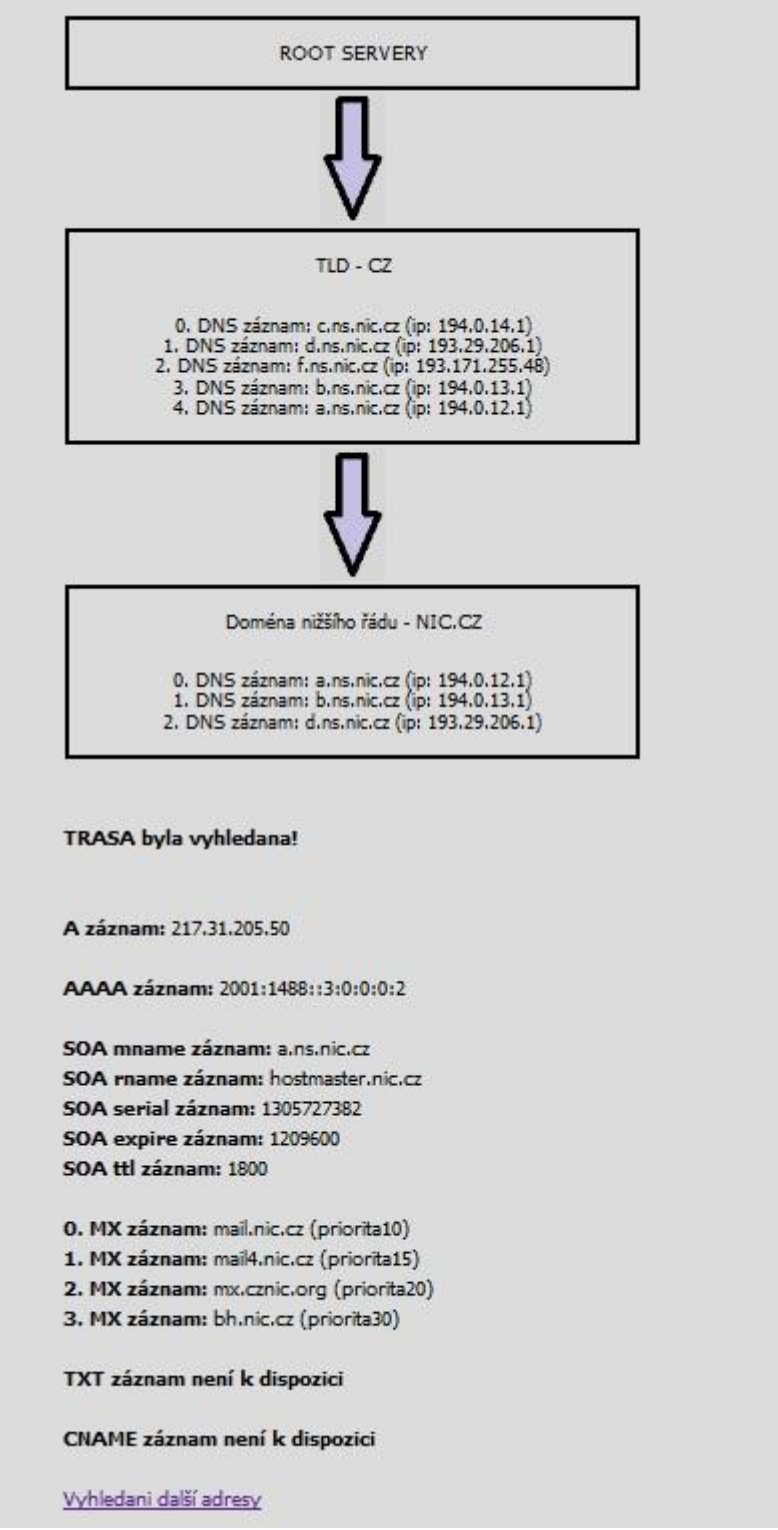
2.3 Grafické rozhraní aplikace

Při vytváření grafického rozhraní aplikace jsem se zaměřil hlavně na tři věci: přehlednost, jednoduchost a funkčnost. Protože už nějaký čas webové aplikace tvořím a zabýval jsem se tím i před touto bakalářskou prací, občas i ve svém volném čase, tak jsem měl již předem představu, jak budu postupovat při tvorbě tohoto programu. Nejprve jsem si navrhl funkci programu, co budu dělat, jaké funkce k tomu využiji a jaké budu mít vstupy a jaké výstupy z aplikace. Poté jsem si podle požadavků programu navrhl úvodní stranu a s pomocí implementace CSS stylů jsem jí vizuálně přizpůsobil do podoby, kterou vidíte na Obr. 2. O implementaci CSS a jejich tvorbě se rozepíši v následujících kapitolách, kdy budu rozebírat jednotlivé součásti programu. Při tvorbě tohoto grafického rozhraní a laděni CSS jsem na mnoho problémů nenarazil, až na několik drobných nepříjemností, vše fungovalo jak má.



Obr. 2 – Rozhraní, úvodní strana

Po doladění úvodní strany jsem se pustil do tvorby grafické podoby výstupu z mého programu. Zde to již bylo o trochu složitější, vzhledem k provázanosti zobrazení s vlastním programem a cykly či podmínkami, které v něm byly vytvořeny. I tuto část programu a použité CSS popíši podrobněji v následujících kapitolách. Samotné grafické rozhraní zobrazování výsledku je vidět na Obr. 3.



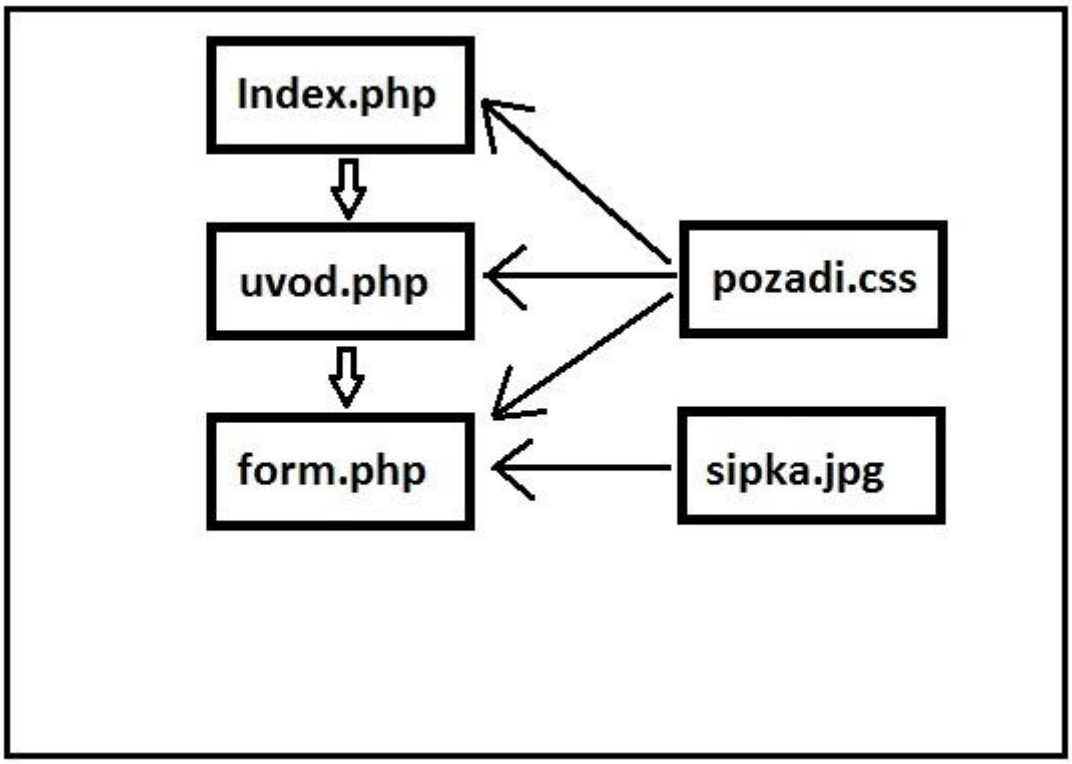
Obr. 3 – Rozhraní, zobrazení výsledku dotazu

2.4 Programová část

A nyní se začnu zabývat samotným jádrem této práce, webovou aplikací. V první kapitole popíši systém souborů a v dalších kapitolách blíže rozeberu kód programu a jeho funkci

2.4.1 Systém souborů v programu

Tato aplikace je rozdělena do tří typů souborů. V souborech *.php je programová část, v souboru pozadí.css je uloženo nastavení stylů pro grafické zpracování webu a konečně v souboru *.jpg je uloženo pozadí (šipka), která je použita u grafického zpracování výstupu. Blíže je toto přiblíženo na Obr. 4, kde je schéma systému souborů. Stručně popíši role jednotlivých souborů. Index.php je základním souborem, který se načte při zadání webu. Ihned poté se načítá uvod.php kde je vstupní formulář pro údaje. Po odeslání dat se načítá soubor form.php, kde je jádro celé aplikace a provádějí se příslušné algoritmy. Ke všem těmto souborům se načítá soubor pozadi.css, kde jsou popsány styly pro vzhled jednotlivých komponent stránek. Form.php ještě navíc pracuje se souborem sipka.jpg, který používá k dotvoření grafického výstupu.



Obr. 4 – Systém souborů

2.4.2 Kód souboru - index.php

První soubor webové aplikace je klasicky index.php. Tento soubor, pokud se nachází v adresáři, tak se automaticky volá. V první části souboru jsou uvedeny základní deklarace, meta tagy. V tagu <link> se volá a načítá soubor pozadí.css. V dalším pokračování kódu si těle (tag *body*) stránku rozděluji pomocí tagů * * na jednotlivé sekce, která má každá své vlastní *id*, podle kterého je také v budoucnu rozeznám a mohu s nimi pracovat, ovlivňovat jejich vzhled pomocí CSS. V poslední části kódu se poté již pracuje s php kódem, kde se načítá do sekce „obsah“ aktuální strana, případně úvodní okno (uvod.php).

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
 <head>
  <meta http-equiv="content-type" content="text/html; charset=windows-1250">
  <meta name="generator" content="PSPad editor, www.pspad.com">
  <meta name="Description" content="Vizualizace DNS" />
  <meta name="Keywords" content="Vizualizace, DNS ,jmenný server" />
  <title>Vizualizace DNS
  </title>
  <link rel="stylesheet" type="text/css" href="pozadi.css">
 </head>
 <body>
  <div id="hlavni">
   <div id="logo"><h1>Vizualizace dns transakce</h1>
   </div>
   <div id="obsah">
<?php
  $page=$_GET["page"];
  $uvodni="uvod.php";
 if ($page) {
  if (file_exists($page.".php"))
     include($page.".php");
     else { include "404.php"; }
  }
  else {
  include  ($uvodni);
 }
          ?>
   </div>
  </div>
 </body></html>
```

2.4.3 Kód souboru – uvod.php

Následující soubor je úvodní strana, která se vám zobrazí po zadání adresy webu (vizuslavik.wz.cz) viz Obr. 2. Je zde vložen nadpis stránky, a poté již pokračuje Javascriptová kontrolní funkce. Tato kontrola se provádí, jak již bylo zmíněno dříve, na straně klienta, tudíž lze obejít. Dále se tu nachází kód formuláře, který ve svém tagu říká, že data získaná z formuláře se po stisknutí tlačítka submit, odešlou do souboru form.php (rozebereme v další kapitole) metodou post a ještě před odesláním se zkontrolují JS funkcí *kontrola*…pokud proběhne korektně (navrátí hodnotu true), tak se odešlou data z textového pole formuláře s názvem „adresa“ a další data z doplňujícího výběru provedeného „checkboxy“. Funkce *kontrola* ověřuje, zda jsou data v poli „adresa“ správně zadána. Kontroluje se, zda je pole vůbec vyplněno, dále se pomocí funkcí na práci s řetězci kontroluje, zda není na začátku „www“, poté zda nejsou v adrese vyplněny dvě tečky vedle sebe a nakonec jestli je řádně zadána TLD. Pokud nastane nějaký z těchto chybových stavů, uživateli se objeví chybové hlášení a formulář se neodešle.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
 <head>
  <meta http-equiv="content-type" content="text/html; charset=windows-1250">
  <meta name="generator" content="PSPad editor, www.pspad.com">
  <title>DNS transakce
  </title>
  <link rel="stylesheet" type="text/css" href="pozadi.css">
 </head>
<script type="text/javascript" language="JavaScript"><!--
function kontrola()
 {
  var adresa = self.document.forms.f.adresa.value;
  var je_ok = adresa != "";
  if (je_ok == false) alert('Adresa musí být zadána!');
  var tecka = adresa.indexOf(".");
  var cast_pred_teckou = adresa.substring(0,tecka);
  if (cast_pred_teckou == "www")
     {
     alert('V adrese nesmí být WWW!');
     return false;
     }
  var pozice_dvou_tecek_vedle_sebe = adresa.indexOf("..");
  if (pozice_dvou_tecek_vedle_sebe >= 0)
     {
     alert('V adrese nesmí být dvì teèky vedle sebe!');
     return false;
     }
```

```
    var pozice_posledni_tecky = adresa.lastIndexOf(".");
    var pocet_znaku_za_posledni_teckou = adresa.length - pozice_posledni_tecky - 1;
    if (pocet_znaku_za_posledni_teckou < 2  ||  pocet_znaku_za_posledni_teckou > 6)
       {
       alert('V adrese musí být øádnì zadaná TLD!');
       return false;
       }
    return je_ok;
  }
// -->
</script>
<form name='f' action="?page=form" onSubmit='return kontrola();' method="post">
  <fieldset>
<legend align="top" class="lg">Zde vložte adresu pro DNS dotaz <br /> (ve formátu bez www. a bez lomítka za TLD!)</legend>
<label class="lg"><input type="text" name="adresa" size="60"></label><br />
  </fieldset>
    </br>
  <fieldset>
<legend align="top" class="lg">Vyberte jaké další záznamy chcete zobrazit:</legend>
<label class="lg"><input type="checkbox" name="A" value="A" class="l"> A</label><br />
<label class="lg"><input type="checkbox" name="AAAA" value="AAAA" class="l"> AAAA</label><br />
<label class="lg"><input type="checkbox" name="MX" value="MX" class="l"> MX</label><br />
<label class="lg"><input type="checkbox" name="SOA" value="SOA" class="l"> SOA</label><br />
<label class="lg"><input type="checkbox" name="TXT" value="TXT" class="l"> TXT</label><br />
<label class="lg"><input type="checkbox" name="CNAME" value="CNAME" class="l"> CNAME</label><br />
</fieldset>
</br> </br>
  <input type="submit" value="Zobrazit" class="btn">
</form>
</html>
```

2.4.4 Kód souboru - Form.php

A nyní projdeme hlavní programovou část aplikace. Toto je nejsložitější část webové aplikace, řeší se zde jádro problému.

V tomto souboru je obsáhnut kód php, který se na rozdíl od předchozího JS vykonává na straně serveru. Poté co se z formuláře z předchozího souboru odeslala hodnota pole adresa, tak toto pole načteme do proměnné *$adresa.* Začne se tvořit tabulka, kde jednotlivé určené buňky obalíme třídou „se“. U Root serveru jmenné servery nevypisuji, protože ty jsou neměnné. Poté začne cyklus, kdy proměnou *$*adresa rozdělíme na části a postupně přidáváme jednotlivé části domény od nejvyšší (TLD) po nejnižší. Nejdříve zjišťujeme u domény jmenné servery funkcí *dns_get_record.* Pomocí DNS dotazu získáme NS záznamy jmenných serverů pro danou doménu. U každého NS ještě funkcí *gethostbyname* pošlu DNS dotaz na A záznam (ipv4 adresa), a poté vypisuji d závorek IP adresu NS, aby bylo vidět, že se jedná o různé NS. V dalším kroku vkládám obrázek šípky mezi jednotlivé buňky tabulky, a pak se vše v cyklu opakuje, dokud se nevypíše i poslední část domény a její NS. Poté co se zobrazí celá trasa Name Serveru až k požadované doméně, začne vypisování dalších RR záznamů domény dle volby ve formuláři. Všechny tyto algoritmy jsou ošetřeny chybovými stavy a podmínkami kdy se mají provést.

```
<?
$adresa = $_POST['adresa'];
  ?>
  <table>
    <tr >
      <td class="se" >
     ROOT SERVERY
      </td>
    </tr>
  </table><a>
    <img src="sipka.jpg" width="50" height="100" hspace="192"></a>
  <table>
    <tr>
    <td class="se">
<?php
$pole = Explode (".", $adresa);
$poc=count($pole);
$dom = $pole[$poc-1];
echo ("<p>TLD  -  ".StrToUpper($dom)."</p>");
$result = dns_get_record("$dom", DNS_NS);
 $a = count ($result);
    if ($a!="" ){
        for ($i = 0; $i < $a; $i++):
        $a0= $result[$i][target];
```

```
      $vysledek = gethostbyname ($a0);
      echo "</br>".$i.". DNS záznam: ";
      print_r($a0);
      echo " (ip: ";
      print_r($vysledek);
      echo ")";
      endfor;
     }
   echo "</td></tr>";
$result = dns_get_record("$adresa", DNS_A);
$test = $result[0][ip];
if ($test!="" )
{

  $poc--;
  while ($poc > 0):
  $dom= $pole[$poc-1] . "." . $dom;
  $result = dns_get_record("$dom", DNS_NS);
  $a = count ($result);
   if ($a!="" ){
      echo ("</table><a><img src='sipka.jpg' width='50' height='100'
hspace='192'></a><table><tr><td class='se'><p>Doména nižšího řádu -
".StrToUpper($dom)."</p>");
      for ($i = 0; $i < $a; $i++):
      $a0= $result[$i][target];
      $vysledek = gethostbyname ($a0);
      echo "</br>".$i.". DNS záznam: ";
      print_r($a0);
      echo " (ip: ";
      print_r($vysledek);
      echo ")";
      endfor;
     }
   echo "</td></tr>";
  $poc--;
  endwhile;
  echo "</table></br></br><b>TRASA byla vyhledana!</b>";
}
if ($test=="" )
{
echo "</table></br></br><b>TRASA nebyla vyhledána, neplatná adresa!</b>";
}
?>
  </table>
<?php
  if($_POST["A"])
  {
     $result = dns_get_record("$adresa", DNS_A);
     $a0= $result[0][host];
      if ($a0!="" ) {
      echo "</br></br></br><b>A záznam:</b> ";
      print_r($result[0][ip]);
```

```
    }
    if ($a0=="" )
    {
    echo "</br></br></br><b>A záznam není k dispozici</b>";
    }
}
echo "</br>";
if($_POST["AAAA"]) {
   $result = dns_get_record("$adresa", DNS_AAAA);
   $a0 = $result[0][host];
    if ($a0!="" ) {
    echo "</br><b>AAAA záznam:</b> ";
    print_r($result[0][ipv6]);
    }
    if ($a0=="" )
    {
    echo "</br><b>AAAA záznam není k dispozici</b>";
    }
}
 echo "</br>";
if($_POST["SOA"])
{
   $result = dns_get_record("$adresa", DNS_SOA);
   $a = count ($result);
   if ($a!="" ){
    for ($i = 0; $i < $a; $i++):
    $a0= $result[$i][mname];
    $a1= $result[$i][rname];
    $a2= $result[$i][serial];
    $a3= $result[$i][expire];
    $a4= $result[$i][ttl];
    echo "</br><b>SOA mname záznam:</b> ";
    print_r($a0);
    echo "</br><b>SOA rname záznam:</b> ";
    print_r($a1);
    echo "</br><b>SOA serial záznam:</b> ";
    print_r($a2);
    echo "</br><b>SOA expire záznam:</b> ";
    print_r($a3);
    echo "</br><b>SOA ttl záznam:</b> ";
    print_r($a4);
    endfor;
   }
   if ($a=="" )
    {
    echo "</br><b>SOA záznam není k dispozici</b>";
    }
}
echo "</br>";
if($_POST["MX"])
{
   $result = dns_get_record("$adresa", DNS_MX);
```

```
   $a = count ($result);
   if ($a!="" ){
    for ($i = 0; $i < $a; $i++):
    $a0= $result[$i][target];
    $a1= $result[$i][pri];
    echo "</br><b>".$i.". MX záznam:</b> ";
    print_r($a0);
    echo " (priorita";
    print_r($a1);
    echo ")";
    endfor;
   }
   if ($a=="" )
    {
    echo "</br><b>MX záznam není k dispozici</b>";
    }
}
echo "</br>";
if($_POST["TXT"])
{
   $result = dns_get_record("$adresa", DNS_TXT);
   $a = count ($result);
   if ($a!="" ){
    for ($i = 0; $i < $a; $i++):
    $a0= $result[$i][txt];
    echo "</br><b>".$i.". TXT záznam:</b> ";
    print_r($a0);
    endfor;
    }
    if ($a=="" )
    {
    echo "</br><b>TXT záznam není k dispozici</b>";
    }
}
  echo "</br>";
if($_POST["CNAME"])
{
   $result = dns_get_record("$adresa", DNS_CNAME);
   $a = count ($result);
   if ($a!="" ){
    for ($i = 0; $i < $a; $i++):
    $a0= $result[$i][target];
    echo "</br><b>CNAME záznam:</b> ";
    print_r($a0);
    endfor;
    }
    if ($a=="" )
    {
    echo "</br><b>CNAME záznam není k dispozici</b>";
    }
}
?>
```

2.4.5 Ukázky kódu souboru – pozadi.css

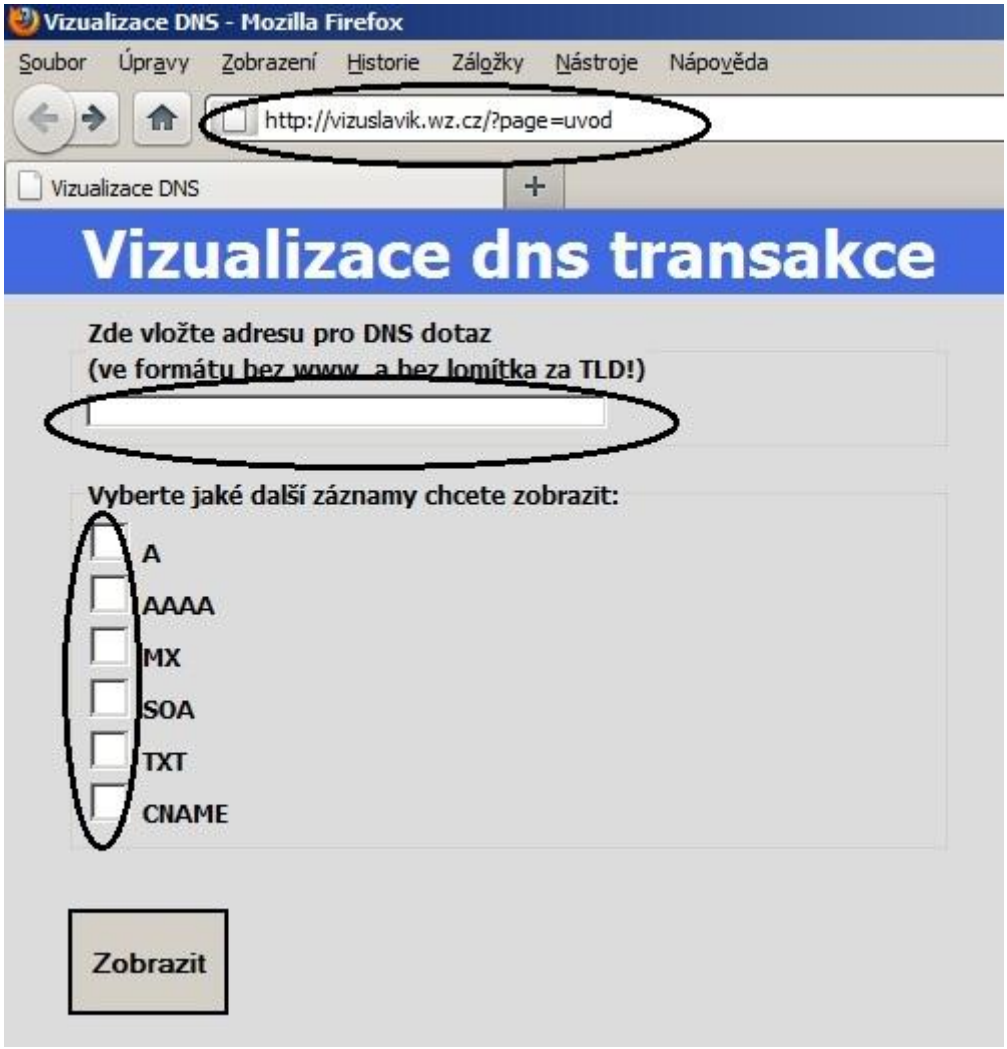
Jako poslední tu mám ukázku souboru kaskádových stylů, které používám a pomocí nichž je tvořeno grafické rozhraní. Jsou tu klasická nastavení od fontů přes zarovnání šířka a výška řádků atd. Vybral jsem pouze ty důležitější třídy, které se starají o korektní zobrazení úvodního formuláře a také výstupu.

```
.se  {
        text-align: center;
        line-height: 1.2;
        font-size: 13px;
        padding: 17px;
        width: 390;
        border: 3px solid Black;
        border-color: Black;
        border-width: 4px;

}
.btn {
        width: 120px;
        height: 80px;
        border: 3px solid Black;
        border-color: Black;
        font-size: 20px;
        text-align: center;
        border-width: 4px;
        font-weight: bold;
        vertical-align: middle;
}
.lg {
        font-weight: bold;
        font-size: large;
}
.l
{
        width: 30px;
        height: 30px;
        border: 3px solid Black;
        border-color: Black;
        border-width: 4px;
}
```

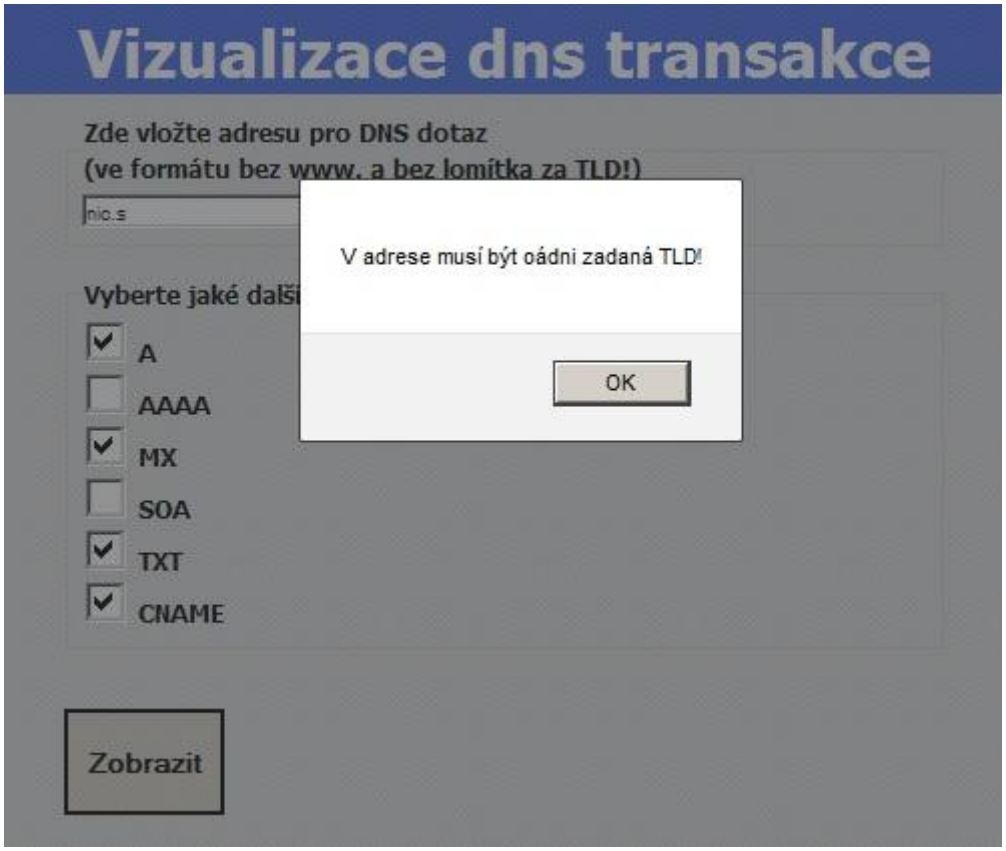
2.5 Ověření funkčnosti aplikace

Na závěr je třeba ještě ověřit funkčnost celé aplikace. Nejdříve zadáme adresu naší webové aplikace (vizuslavik.wz.cz) a poté je potřeba do formuláře vyplnit žádané údaje, tzn. doménu, kterou chceme vyhledat a zaškrtnout další dodatečné záznamy, které mají být zobrazeny (Obr. 5). Formulář odešleme tlačítkem zobrazit.

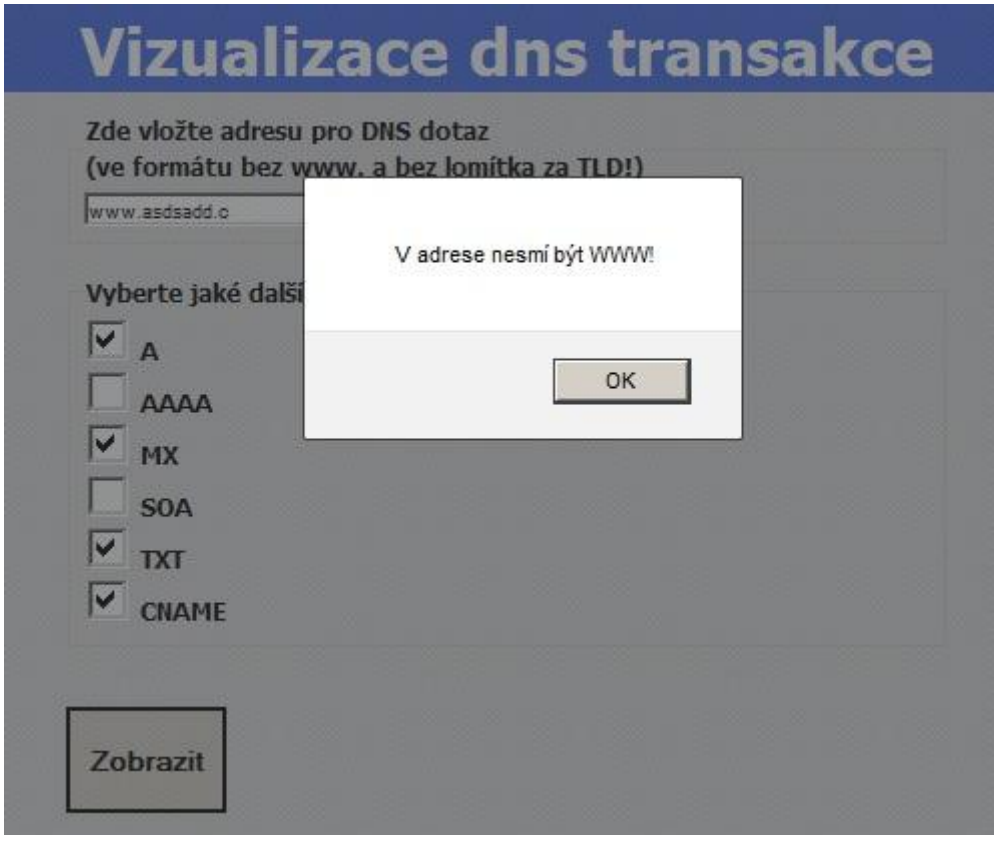


Obr. 5 – Vstupní formulář

V případě, že je zadaný špatný formát adresy, tak na nás „vyskočí“ jedno z následujících chybových ohlášení, Dvě ohlášení jsem pro nastínění vybral a jsou zobrazena na Obr. 6 a Obr. 7. JavaScript ošetřuje chyby na straně klienta, na straně serveru ošetřujeme chyby pomocí php.

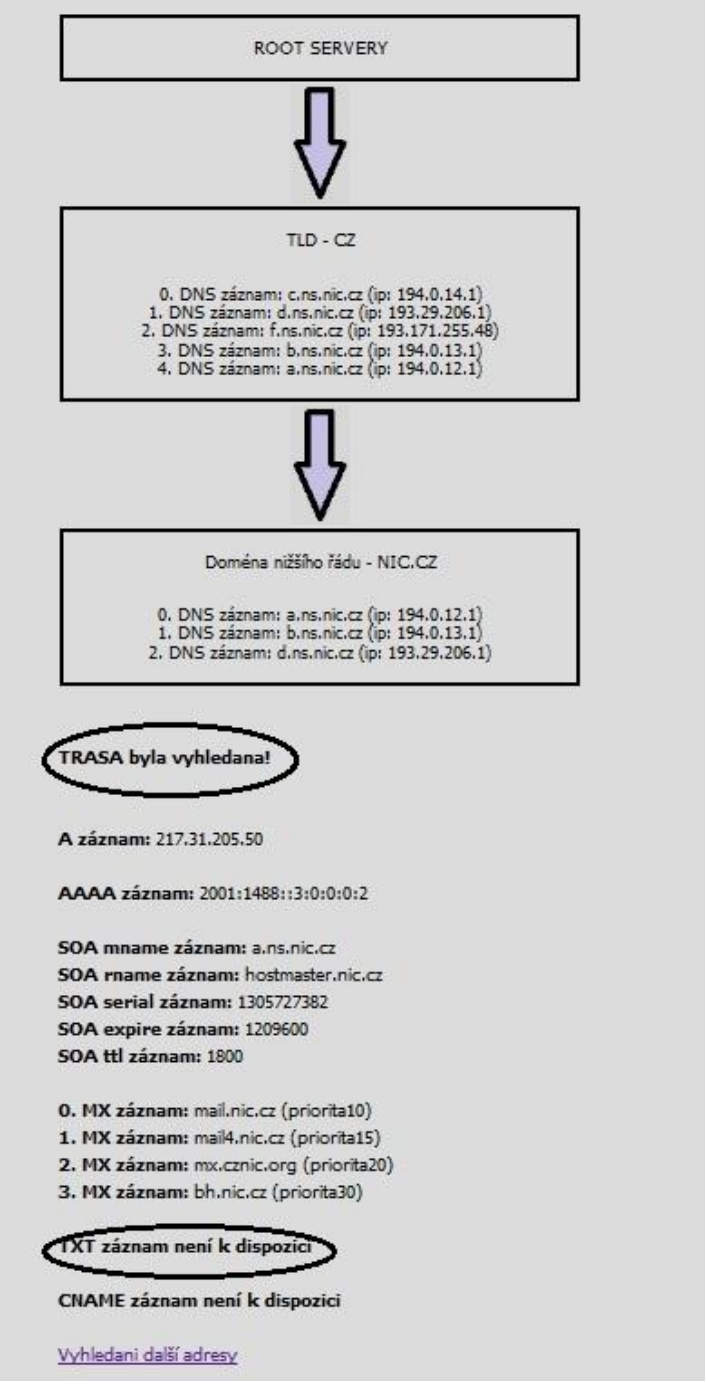


Obr. 6 – Funkčnost JavaScriptu1



Obr. 7 – Funkčnost JavaScriptu2

Poté co se odešle formulář a data projdou kontrolou, jsou všechny hodnoty odeslány jádru aplikace (form.php). Původní záměr bylo tvořit přímo DNS pakety, ale nenalezl jsem vhodnou funkci na ovládání resolveru. Proto program ukazuje iterační komunikaci, než se dostane k žádanému výsledku. Cyklus postupně skládá domény od root až po nejnižší stupeň a vypisuje jejich NS a IP adresy (v závorkách), aby byla vidět struktura DNS. Po zobrazení komunikace se vypisují další dodatečné informace ze zdrojových vět dle požadavků uživatele, které zadal ve formuláři. Výsledný výstup je vidět na Obr.8.



Obr. 8 – Popis výstupu z programu

3 Závěr

3.1 Možnosti využití a rozšíření nástroje

Tuto aplikaci je možné využít pro vyhledání autoritativní serverů a jejich IP adres, abychom mohli zjistit, s kterými servery probíhá komunikace našeho resolveru, při zjišťování odpovědi na náš dotaz (překlad) a jak se nakonec dostane k IP adrese patřící k námi žádané doméně. Další rozšíření tohoto nástroje jsou možná. Do odpovědí je možné zahrnout i další typy záznamů, které jsem prozatím nepoužil. Je tu i příležitost, přidat více možností interaktivní volby, například, kterého ze serverů se na danou doménu zeptat, zobrazení dalších časových informací, atd.

3.2 Výsledky práce

Při tvorbě této práce jsem získal mnoho nových znalostí jak systému DNS, tak i dalších aspektů webového prostředí. Dále jsem díky tomuto tématu prohloubil své znalosti tvorby webových aplikací a zlepšil své schopnosti programování, zejména v jazyce PHP a JavaScript, a doufám, že tyto znalosti využiji a budu i nadále prohlubovat nejen při mém studiu, ale také v mém profesním životě.

4 Bibliografie

1. **Surý, Ondřej.** www.root.cz. [Online] 22. 7 2010. http://www.root.cz/clanky/hotovo-dns-je-podepsano/.

2. php.net. [Online] 2010. http://php.net/manual/en/function.dns-get-record.php.

3. **Zajíc, Petr.** www.linuxsoft.cz. [Online] 8. 11 2004. http://www.linuxsoft.cz/article.php?id_article=502.

4. **Vrána, Jakub.** www.interval.cz. [Online] 10. 8 2007. http://interval.cz/clanky/vicestrankovy-formular-v-php-a-javascriptu/.

5. **Grimmich, Šimon.** www.tvorba-webu.cz. [Online] 2008. http://www.tvorba-webu.cz/php/pole.php.

6. **Stevens, W. R.** *TCP/IP Illustrated Vol. 1 - The Protocols,*. místo neznámé : Addison-Wesley, 1994.

7. **Lahvička, Jiří.** *www.interval.cz.* [Online] 12. 6 2000. http://interval.cz/clanky/php-zakladni-informace/.

8. **Dostálek, Libor a Kabelová, Alena.** *Velký průvodce protokoly TCP/IP a systémem DNS.* Brno : CCP Books, a.s., 2005. ISBN 80-7226-675-6.

9. **McClure, Stuart, Shah, Saumil a Shah, Shreeraj.** *Web Hacking: Útoky a Obrana.* Praha : SoftPress s.r.o., 2003. ISBN 80-86497-53-4.

10. **Satrapa, Pavel.** *Web design.* Havlíčkův Brod : Neokortex spol. s.r.o., 1997. ISBN 80-902230-1-X.

11. **Lavin, Peter.** *PHP objektově orientované.* Praha : Grada Publishing, a.s., 2009. ISBN 978-80-247-2137-8.

12. **Kosek, Jiří.** *PHP a XML*. Praha : Grada Publishing, a.s., 2009. ISBN 978-80-247-1116-4.

13. **Sosinsky, Barrie.** *Mistrovství - počítačové sítě.* Brno : Computer press, a.s., 2010. ISBN 978-80-251-3363-7.