VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

IMPLEMENTACE SERVEROVÝCH SLUŽEB DO ZÁTĚŽOVÉHO TESTERU

IMPLEMENTATION OF SERVER SERVICES IN STRESS TESTER

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — František Berg

AUTHOR

VEDOUCÍ PRÁCE — Ing. Vlastimil Člupek, Ph.D.

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: František Berg ***ID:*** 211132

Ročník: 3 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Implementace serverových služeb do zátěžového testeru

POKYNY PRO VYPRACOVÁNÍ:

V bakalářské práci se seznamte se softwarovým prostředím zátěžového testeru vyvíjeného na Ústavu telekomunikací. Proveďte analýzu serverových služeb (DNS, NTP, DHCP, …) a několik jich implementujte do zátěžového testeru do modulu Server Emulator. Grafické rozhraní pro ovládání serverových služeb bude umožňovat spustit/vypnout službu a definovat IP adresu (verze 4 i 6), port a síťové rozhraní, na kterém daná služba poběží. Otestujte funkčnost implementovaných služeb. Přehledně prezentujte dosažené výsledky.

DOPORUČENÁ LITERATURA:

[1] JMETER, Apache. Apache software foundation, https://jmeter.apache.org/.

[2] LENCSE, Gábor. Benchmarking authoritative DNS servers. IEEE Access, 2020, 8: 130224-130238.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 26.5.2023

Vedoucí práce: Ing. Vlastimil Člupek, Ph.D.

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Tato bakalářská práce se zabývá analýzou a implementací serverových služeb DNS, DHCP a NTP do zátěžového testeru vyvíjeného na Ústavu telekomunikací. Všechny služby jsou realizovány v Linuxové distribuci Ubuntu, přes programové balíčky bind9, isc-dhcp-server a ntp. Všem službám bylo vytvořeno grafické rozhraní, pomocí kterého je možné služby spustit a vypnout. Dále je možné definovat IP adresu verze 4 i 6, port a síťové rozhraní, na kterém daná služba naslouchá. Všechny služby pro svůj provoz využívají nově vytvořené konfigurační soubory pomocí modulu Server Emulator, přičemž původní konfigurace zůstává nedotčena. V práci je popsán teoretický základ vybraných serverových služeb a následně je popsána samotná implementace do prostředí zátěžového testeru do modulu Server Emulator.

KLÍČOVÁ SLOVA

JMeter, Java, Server Emulator, Domain Name System, Dynamic Host Configration Protocol, Network Time Protocol, Implementace, Serverové služby

ABSTRACT

This bachelor thesis deals with the analysis and implementation of DNS, DHCP and NTP server services into a stress tester developed at the Institute of Telecommunications. All services are implemented in the Ubuntu Linux distribution through the software packages bind9, isc-dhcp-server and ntp. A graphical interface has been created for all services that can be used to start and stop the services. It is also possible to define the version 4 and 6 IP address, port and network interface on which the service listens. All services use the newly created configuration files for their operation using the Server Emulator module, while the original configuration is retained. This article describes the theoretical basis of the selected server services and then the actual implementation in a load tester environment in the Server Emulator module.

KEYWORDS

JMeter, Java, Server Emulator, Domain Name System, Dynamic Host Configration Protocol, Network Time Protocol, Implementation, Server services

BERG, František. *Implementace serverových služeb do zátěžového testeru*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2022, 57 s. Bakalářská práce. Vedoucí práce: Ing. Vlastimil Člupek, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	František Berg
VUT ID autora:	211132
Typ práce:	Bakalářská práce
Akademický rok:	2022/23
Téma závěrečné práce:	Implementace serverových služeb do zátěžového testeru

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Vlastimilu Člupkovi Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci a panu Ing. Mgr. Pavlu Šedovi Ph.D. za otevřenost, podnětné debaty a odborné znalosti, které mi byly neocenitelným zdrojem inspirace a povzbuzení.

Obsah

Seznam obrázků

Úvod

Internet nás v dnešní době obklopuje na každém kroku a je dostupný ze všech koutů světa. Chod internetu zajišťují síťové prvky, které propojují komunikaci mezi jednotlivými zařízeními. Síťová zařízení jsou fyzické přístroje, která jsou nutná pro komunikaci a interakci mezi technickým vybavením počítačů a zajišťují tak spolehlivou výměnu dat mezi zařízeními. Mezi taková zařízení patří například opakovač, rozbočovač, most nebo směrovač. Fyzická zařízení jsou pouze kusem techniky, pokud neobsahují potřebné programového vybavení, které dává přístrojům daný účel a smysl. Pro zajištění spolehlivé komunikace mezi síťovými prvky slouží jednotlivé komunikační protokoly, jako je například TCP, DHCP, HTTP, FTP, DNS a mnoho dalších. Nedílnou součástí komunikace je také síťová architektura, která definuje strukturu řízení komunikace v systémech a tok dat. Prvky v internetu se dělí mezi koncová a mezilehlá zařízení. Koncovým zařízením je například osobní počítač nebo server. Mezilehlé zařízení je prvek, který propojuje síť. Server je označení pro počítač, který poskytuje určitou službu, nebo počítačový program, který danou službu realizuje. V systémech Unix se lze setkat s označením démon (anglicky deamon), což je program, který běží dlouhodobě bez zásahu uživatele. V systémech Microsoft Windows se používá označení služba (anglicky service).

Tato bakalářaká práce se věnuje implementaci serverových služeb do zátěžového testeru vyvíjeného na Ústavu telekomunikací VUT. Cílem práce je analyzovat serverové služby a implementovat je do zátěžového testeru do modulu Server Emulator. Dalším krokem je navrhnout a implementovat grafické rozhraní pro ovládání vybraných služeb a naprogramovat základní funkce pro jejich běh.

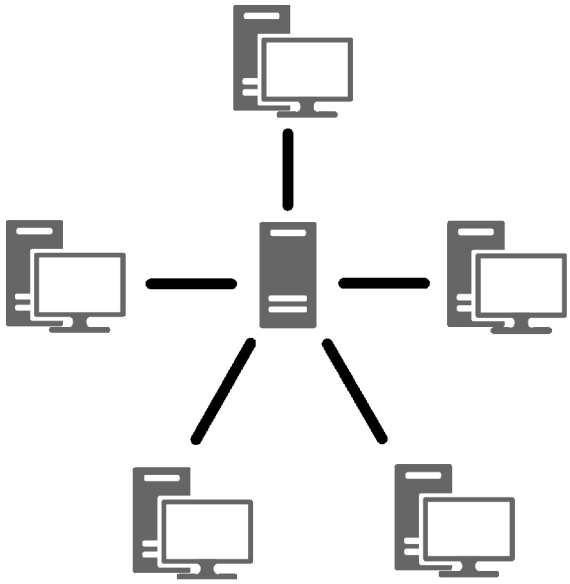
Praktická část bakalářké práce je zaměřena na implementaci a programové náležitosti pro realizaci daných serverových služeb v zátěžovém testeru v Linuxové distribuci Ubuntu. Cílem práce je analyzovat serverové služby a provést jejích implementaci do modulu Server Emulator.

1 Server

Existuji dva významy pojmu server v oblasti informačních technologií, a to server jako technické vybavení a server jako programové vybavení.

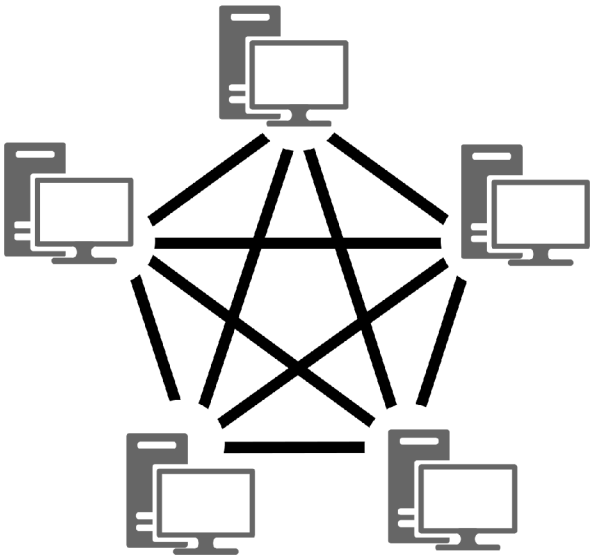
Pod pojmem server jako technické vybavení se rozumí koncové zařízení v síti. Hlavní úlohou serveru je poskytnutí určité služby dalším koncovým zařízením, tedy počítačům v sítí, neboli klientům, pomocí mezilehlých zařízení, které síť propojují. Jedná se tedy o centralizované zařízení, ke kterému může být připojeno více klientů přes internet, nebo lokální síť za účelem získání určité serverové služby, jako je například přístup k určitým datům, emailům nebo webovým stránkám [1].

Model, kde se vyskytuje centralizovaný počítač (server), ke kterému se připojují počítače, neboli klienti, se označuje jako klient-server a je znázorněn na obrázku 1.1. Dalšími modely mohou být například model peer-to-peer (P2P) (znázorněn na obrázku 1.2), nebo model friend-to-friend (F2F). [2].



Obr. 1.1: Síťový model klient-server

Pod pojmem server jako programové vybavení se rozumí soubor serverových služeb, které jsou k dispozici danému klientovi v síti. Tento server většinou poskytuje služby na osobním počítači, který je zároveň klientem. Klient se tedy odkazuje na server, který je zároveň i realizátorem na speciální vyhrazené IP adrese 127.0.0.1 v protokolu IPv4, nebo ::1 v protokolu IPv6. Tato vyhrazená IP adresa se nazývá localhost a slouží tedy pro komunikaci sama se sebou. Tento způsob realizace serveru se využívá například pro testování zátěže webových stránek, nebo pro ověření stavu TCP/IP stacku vlastního počítače. Jestliže se provede příkaz ping na adresu localhost a ten odpovídá, znamená to, že TCP/IP stack je funkční [3].



Obr. 1.2: Síťový model peer-to-peer.

1.1 Rozdíly mezi osobním počítačem a serverem

Mezi serverem a osobním počítačem existují určité podobnosti, ale jsou to dvě různá zařízení. Osobní počítače dnes dokáží sloužit jako jednoduché servery, ale takový server dokáže obsluhovat pouze velice omezený počet klientů. Hlavními rozdíly mezi serverem a osobním počítačem je ve fyzickém a programovém vybavení. Zatímco u osobního počítače se klade důraz na pohodlnost užívání a jednoduchost, u serverů se klade důraz na spolehlivost, rychlost, stálý běh a jednoduchou vyměnitelnost komponentů v případě poruchy [1].

1.2 Serverová služba

V oblasti serveru jako technického vybavení se termínem serverové služby označuje programová funkcionalita nebo soubor programových funkcionalit, které jsou poskytnuty klientům v modelu klient-server pro splnění určitého úkolu, jako například získání specifických informací nebo provedení určité operace.

V oblasti serveru jako programového vybavení se myslí typ programu, který umožňuje opakované využití funkcionalit pomocí rozhraní, které používá společný komunikační jazyk sítě [4].

2 Komunikace v síti

Přenos informace se v počítačové síti skládá z mnoha kroků. Žádné zařízení nedokáže provést všechny úkony najednou, aby provedlo celkový přenos dat. Z toho důvodu je komunikace v počítačové síti rozdělena na jednotlivé úrovně, neboli vrstvy, kde každá vrstva plní svůj specifický účel. Při společné práci všech vrstev dohromady poté vznikne výsledná komunikace mezi jednotlivými zařízeními. Pro tento účel vznikl obecný referenční model sítě, který je podkladem pro vytváření norem, který slouží k propojení jednotlivých zařízení v počítačové síti.

2.1 Referenční modely sítě

Prvním referenčním modelem je ISO/OSI, který vytvořila organizace ISO (International Organization for Standardization) pro účel standardizace komunikace v počítačových sítích OSI (Open System Interconnection). Tento referenční model rozděluje komunikaci v počítačových sítích do 7 úrovní, neboli vrstev, které realizují všechny nezbytné aspekty komunikace. Jedná se tedy o výchozí model architektury komunikace v počítačových sítích pro řízenou výměnu dat. Každá vrstva tak plní svůj specifický účel a poskytuje služby vrstvě vyšší.

Druhým existujícím referenčním modelem v počítačových sítích je model TCP/IP. Původně byl tento model rivalem obecného sedmivrstvého modelu ISO/OSI, ale dnes se stal velmi vhodným kandidátem pro praktickou implementaci budování komunikace v počítačové síti. Model TCP/IP odkazuje na dva přenosové protokoly a to TCP (Transmission Control Protocol) a IP (Internet Protocol). V této zkratce je zahrnuta celá řada internetových protokolů a ucelená představa jak se mají počítačové sítě budovat, a jak by měly fungovat [5].

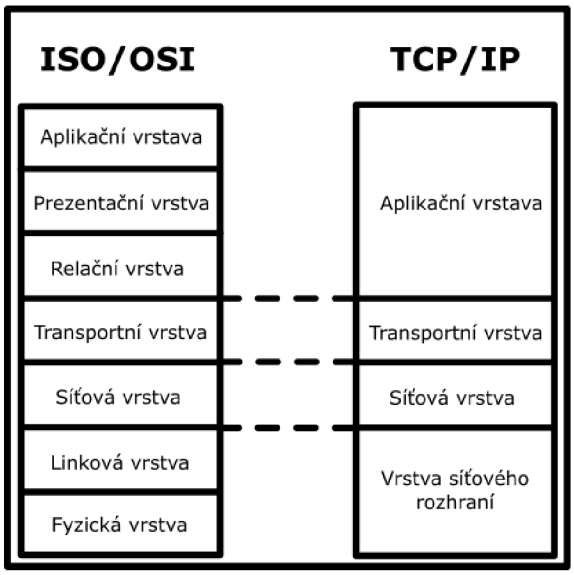
Srovnání modelu ISO/OSI a modelu TCP/IP je vidět na obrázku 2.1.

Pro označení datové jednotky v libovolné vrstvě se používá zkratka PDU, neboli Protocol Data Unit.

V souvislosti s přenosem dat v počítačových sítích se využívají dva termíny pro určení přenášených dat, a to paket a rámec. V literatuře existuje několik odlišných vysvětlení a také využívání těchto výrazů je různé. Důležitým faktorem při používání těchto pojmů je také technologie, u které se využívají.

Paket (packet) v počítačových sítích znamená v překladu balíček a jedná se o formátovaný blok dat. O paketu se mluví v souvislosti se síťovou vrstvou. Struktura paketu obsahuje IP hlavičku, TPC hlavičku a data.

Rámec (frame) je pak to, co doopravdy putuje skrze počítačovou síť. Rámec vzniká až na fyzické vrstvě modelu ISO/OSI a obsahuje celý paket, společně s přidanou ethernetovou hlavičkou a patičkou [6].

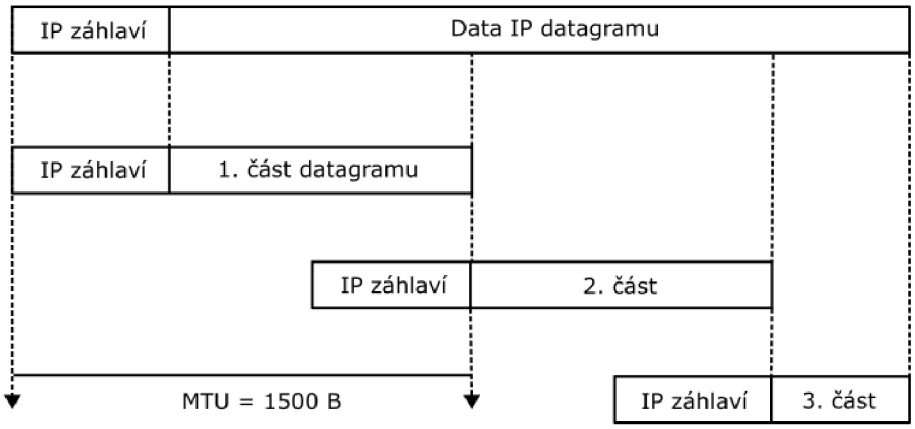


Obr. 2.1: Srovnání referenčních modelů sítí.

Maximální velikost dat přenášeného rámce, tedy paketu, je dána pomocí MTU (Maximum transmision unit), která je v jednotlivých sítích odlišná. Na začátku komunikace se stanoví velikost MTU, pomocí protokolu ICMP. Velikost MTU bývá většinou 1500 bytů. Přehled velikostí MTU v daných sítích je znázorněn v tabulce 2.1. Pokud je vyžadována menší MTU, je provedena fragmentace, neboli rozdělení rámce do více menších, jak znázorňuje obrázek 2.2.

Tab. 2.1: Srovnání velikostí MTU v jednotlivých technologiích.

Linkový protokol	MTU [bajty]
Ethernet II	1500
Ethernet 802.3 SNAP	1492
Frame Relay	1600
FDDI	4352
PPP	296
ATM	48



Obr. 2.2: Fragmentace IP datagram.

2.2 Popis vrstev modelu ISO/OSI

Vrstvy, které se zabývají zpracováním dat:

- Aplikační vrstva
- Prezentační vrstva
- Relační vrstva

Vrstvy, které se zabývají fyzickým přenosem dat:

- Fyzická vrstva
- Linková vrstva
- Síťová vrstva

Transportní vrstva slouží pro propojení mezi fyzickým přenosem dat a jejich zpracováním [5].

Popis jednotlivých vrstev ISO/OSI:

1. Fyzická vrstva
    - Fyzická vrstva zajišťuje fyzický přenos dat v podobě bitů mezi odesílatelem a příjemcem. Řeší typy konektorů, druhy kabelů, kódování, modulaci a synchronizaci přenosu.
2. Linková vrstva
    - Linková vrstva zajišťuje řízení toku dat, aby bylo možné data zpracovávat ze strany příjemce a nebyl jimi zahlcen. Data jsou reprezentována v podobě rámců. Linková vrstva musí určit, kde rámec končí, a kde začíná. Dále pak kontroluje jejich správnost a řídí spolehlivost přenosu dat.
3. Síťová vrstva
    - Síťová vrstva řeší směrování, neboli to, kam data budou přeposlána na základě síťové topologie. Na síťové vrstvě se tvoří tvz. pakety, které obsahují IP adresu odesílatele a příjemce.

4. Transportní vrstva
    - Transportní vrstva bývá implementována až v koncových zařízeních. Úkolem transportní vrstvy je vyrovnat rozdíly mezi vrstvami, které se zabývají fyzickým přenosem dat, od těch, které se zabývají zpracováním dat. Umí rozpoznat a někdy i opravit chyby vzniklé při přenosu a dokáže odhalit špatné pořadí paketů, které jsou doručeny, poskládat je do správného pořadí a na základě informace, kterému programu data patří, je předat konkrétnímu příjemci v rámci uzlu.
5. Relační vrstva
    - Relační vrstva se stará o relaci, neboli dobu, po kterou spolu uzly komunikují. Stará se o navázání spojení, jeho udržení, zrušení a rozhodování, o jaké spojení půjde – duplexní, poloduplexní, Dále se také stará o to, jestli půjde o šifrované spojení.
6. Prezentační vrstva
    - Prezentační vrstva se stará o kompresi a kódováním dat. Zabývá se pouze strukturou dat, protože na stejnou posloupnost bitů může každá aplikace nahlížet jinak, z důvodu jiného kódování. Dalším úkolem prezentační vrstvy je zajišťování šifrování a dešifrování dat.
7. Aplikační vrstva
    - Účel aplikační vrstvy je poskytovat služby aplikacím. Jsou zde zahrnuty pouze takové součásti aplikací, které má smysl standardizovat jako je například řízení databází [7].

2.3 Popis vrstev modelu TCP/IP

1. Vrstva síťového rozhraní
    - Vrstva síťového rozhraní se stará o přístup k fyzickému médiu a řízení spojení.
2. Síťová vrstva
    - Síťová vrstva zajišťuje předávání datagramů, síťovou adresaci a směrování.
3. Transportní vrstva
    - Na transportní vrstvě slouží protokol TCP, který slouží pro spolehlivou komunikaci mezi zařízeními, nebo protokol UDP, který slouží pro nespolehlivou, nekontrolovanou komunikaci.
4. Aplikační vrstva
    - Obsahem aplikační vrstvy jsou aplikační protokoly jako je například DNS, FTP, HTTP nebo DHCP [5].

3 Typy serverových služeb

Tato kapitola se zabývá teorií vybraných serverových služeb, které jsou v praktické části této práce implementovány do ICT testeru do modulu Server Emulator.

Jsou zde popsány základní principy a funkce služby DNS, DNSSEC, DHCP a NTP.

3.1 Domain Name System (DNS)

Domain Name System slouží pro překlad IPv4, nebo IPv6 adresy síťové vrstvy s jmenným názvem, který je pro člověka lehce zapamatovatelný. Pokud by neexistoval systém DNS, každý uživatel by musel při požadavku na komunikaci zadat konkrétní IP adresu daného serveru. Je prakticky nemožné si pamatovat IP adresy všech serverů, které by daný uživatel používal, na rozdíl od těch jmenných. Dalším důležitým bodem z hlediska existence DNS je změna adresy daného serveru, například v důsledku přečíslování stanic, nebo přesunu celé sítě k jinému poskytovateli internetového připojení. Dalším důsledkem může být také fyzické přestěhování daného serveru. Tato informace o změně adresy serveru se musí nějakým způsobem dostat k uživatelům, kteří chtějí službu i nadále využívat.

DNS obsahuje systém odkazů, které odkazují na skutečné adresy serverů a stanic. Systém odkazů obsahuje informace o dekadické reprezentaci IP adresy s příslušnou jmennou reprezentací. Pokud dojde k přejmenování adresy stanice, stačí změnit záznam v příslušné tabulce DNS serveru. Tím se uživatel automaticky dozví novou adresu serveru a může službu i nadále využívat [8].

3.1.1 Protokol DNS

DNS patří do balíčku aplikačních protokolů, které pracují na aplikační vrstvě modelu TCP/IP, využívající porty UDP/53 a TCP/53. Na úrovni síťové vrstvy se pracuje s logickými adresami, neboli IP adresami, které slouží pro univerzální identifikaci daného uzlu, aby síťová vrstva mohla poskytovat přenos mezi koncovými zařízeními. Tyto IP adresy slouží pro globální identifikaci daného uzlu a přiřazují se z daného rozsahu.

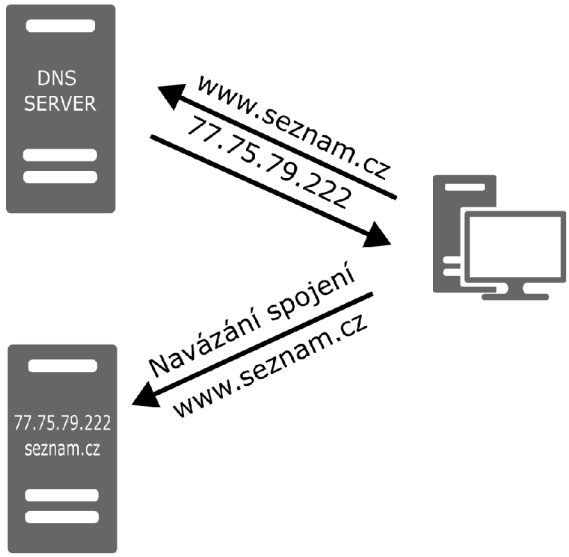
Z hlediska většího počtu IP adres, který by byl pro člověka těžko zapamatovatelný. Na síťově vrstvě představují IP adresy určitou míru abstrakce. DNS protokol přináší ještě větší míru abstrakce na aplikační úrovni.

Základem DNS je model klient-server. Díky tomuto modelu se tedy jedná o distribuovanou službu. Tato služba má hierarchickou organizaci, podobně jako je tomu

u názvů domén. Vazby mezi IP adresami jsou uloženy v DNS databázi, která je celosvětově distribuovaná [5].

3.1.2 Princip komunikace DNS

Základním principem DSN komunikace je v prvotní fázi zaslání požadavku na DNS server, který obsahuje jmenný název příslušného serveru nebo stanice. DNS server vyhledá ve svých záznamech příslušnou odpověď, kterou zašle v podobě odpovídající IP adresy, kterou dále může stanice přímo kontaktovat. Obecný princip DNS komunikace je znázorněn na obrázku 3.1 [5].



Obr. 3.1: Obecný princip DNS komunikace.

Pokud nastane situace, kdy v DNS databázi není potřebný záznam v paměti, kontaktuje se nadřazený DNS server, neboli kořenový (root) DNS server. Komunikace mezi DNS servery je uživateli skryta [5].

3.1.3 Detailní popis DNS komunikace

V prvotní fázi, kdy se hledá IP adresa podle jmenného názvu stanice, hledá počítač příslušnou odpověď v podobě IP adresy ve vlastní lokální dočasné paměti, neboli paměti cash. Pokud odpověď není nalezena v dočasné paměti, zašle dotaz na další úroveň, kterou je resolver server.

Resolver server je poskytovatel internetu, neboli ISP (Internet Service Provider). Pokud resolver server obdrží dotaz, ve kterém je jmenný název serveru, podívá se do

své vlastní dočasné paměti cash. Jestliže resolver server nenajde příslušný záznam na jmennou adresu, komunikace pokračuje na další úroveň, kterou je ROOT server. Těchto serverů je na světe celkem 13 a jsou strategicky umístěny po celém světě, přičemž se vždy vybere ten, který je nejblíže.

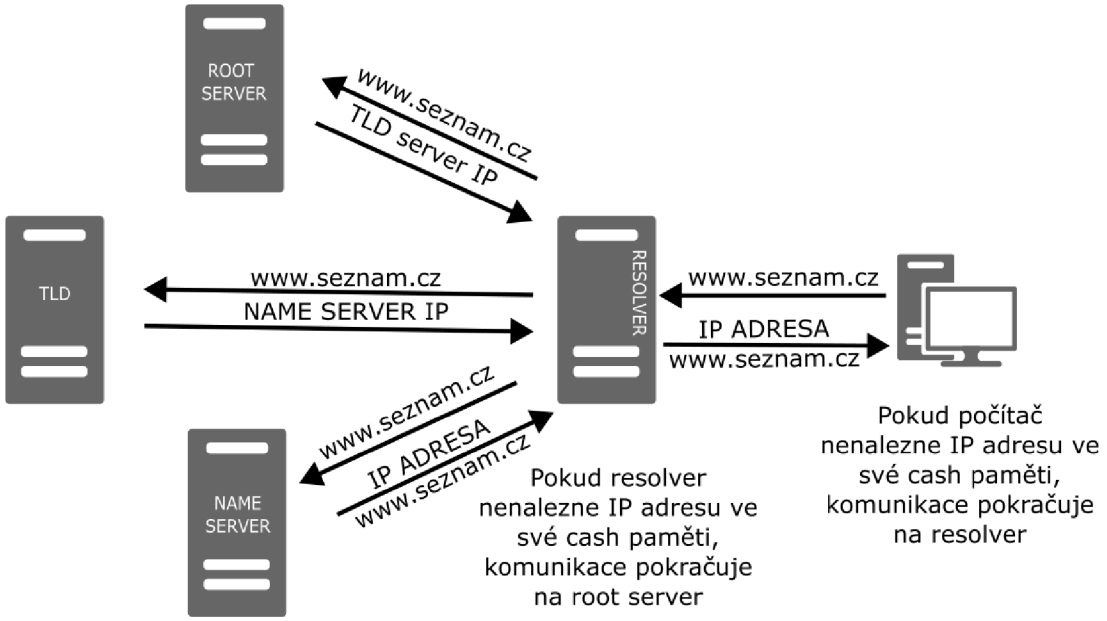
Root server nezná adresu, ale ví, kam poslat dotaz, aby byla zjištěna. Komunikace tak pokračuje na server s domény nejvyšší úrovně, neboli TLD (Top Level Domain), ve kterém jsou všechny domény, které spadají pod určitou doménu jako je .com, .cz, .net a další.

TLD nezná odpověď na daný dotaz. Komunikace tak dále pokračuje na jmenný server, neboli Name Server, ve kterém jsou uloženy veškeré informace o doménách, včetně IP adres. Jedná se tak o finální autoritu, která při obdržení dotazu od resolveru na jmenný název stránky zašle za svých záznamů požadovanou IP adresu.

Resolver server tak zašle počítači IP adresu pro hledanou stránku, aby bylo možné se stránkou komunikovat.

Jakmile resolver server obdrží IP adresu, uloží ji do své vlastní cash paměti pro případ, že by byla opět potřeba IP adresy příslušné stanice nebo serveru [5].

Celá DNS komunikace je znázorněna na obrázku 3.2.



Obr. 3.2: Detailní průběh DNS komunikace.

3.2 Domain Name System Security (DNSSEC)

DNSSEC zvyšuje bezpečnost systému doménových jmen, neboli DNS serveru a poskytuje tak uživateli jistotu, že informace, které byly získány z DNS poskytl správný zdroj, jsou úplná a při přenosu nebyla narušena jejich integrita.

Obecné fungování DNS je popsáno v kapitole 3.1. Počítač vyšle dotaz na jméno domény a pomocí DNS serveru je mu vrácena příslušná IP adresa. V situaci, kdy by někdo dokázal napodobit komunikaci z DNS serveru, pro zaslání nesprávné IP adresy, klient bude komunikovat s nesprávnou službou, aniž by cokoliv tušil [9].

3.2.1 Riziko DNS

Základním principem DNS je, že umožňuje ve vyhledávání používat místo IP označení jména, která jsou srozumitelná a jednoduše zapamatovatelná.

V situaci, kdy by někdo dokázal podvrhnout IP adresu, kterou zasílá DNS server, za adresu jiného místa, která by vypadala stejně, hrozí riziko úniku osobních dat jako například číslo platební karty.

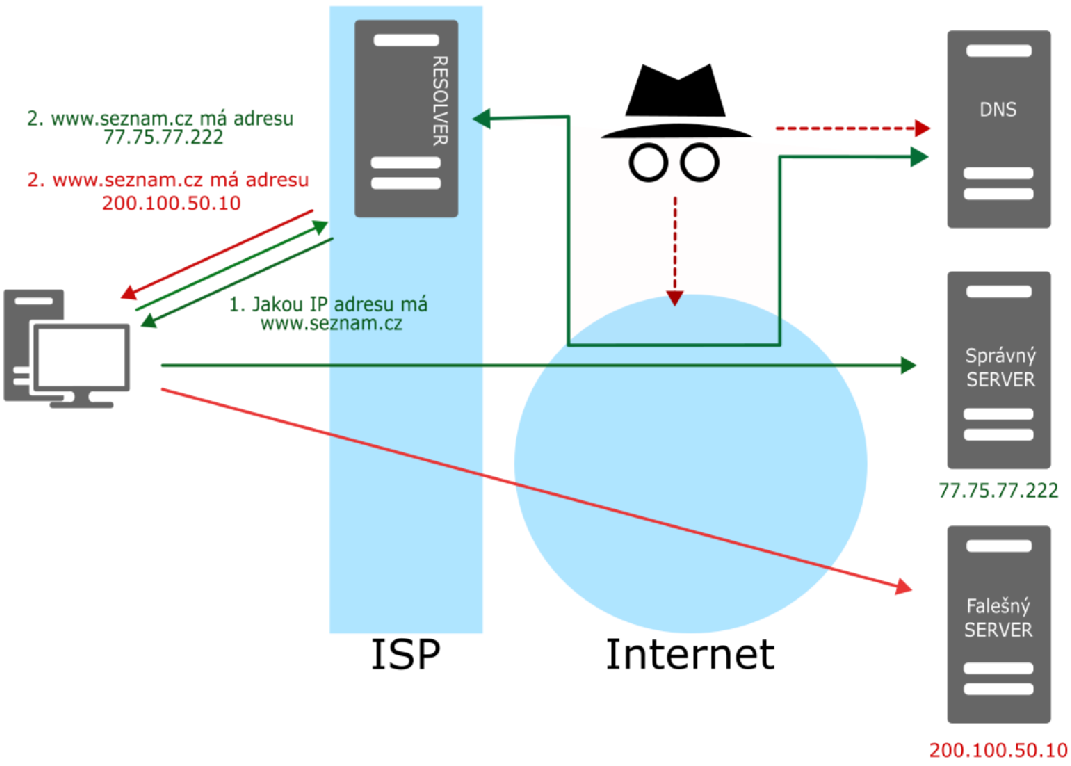
Komunikace v případě podvrhnutí DNS odpovědi je vidět na obrázku 3.3

Za normálních okolností, kdy uživatel zadá jmennou adresu do prohlížeče, komunikace putuje po zelené trase k DNS serveru poskytovatele internetového připojení (ISP). Od ISP dotaz dále putuje ke globálnímu DNS serveru, který odpoví číselnou adresou, se kterou se uživatel spojí a komunikuje s chtěnou službou.

V případě, kde je IP adrese serveru, který poskytuje službu, podvržena, komunikace putuje po červené trase. Klientovi je vrácena falešná IP adresa na službu, která se tváří stejně, ale všechny citlivé informace putují nesprávným směrem. To může být problém například v situaci, kdy uživatel zadává na internetovém e-shopu číslo své platební karty nebo odesílá email s citlivými informacemi [9].

3.2.2 Fungování DNSSEC

DNSSEC je rozšíření DNS pomocí asymetrické kryptografie, tedy vygenerování dvojice klíčů k šifrování a k dešifrování dat. Pomocí soukromého klíče si držitel domény podepíše údaje, které o své doméně vkládá do DNS. Pravost tohoto podpisu je pak možné ověřit pomocí veřejného klíče. Aby byl veřejný klíč dostupný všem, musí ho držitel publikovat u nadřazené autority, kterou je pro všechny .cz domény registr domén .cz. Data jsou šifrována i na úrovni registru domén .cz a veřejný klíč drží nadřazená autorita. Díky tomu se vytváří řetězec, který zajišťuje důvěryhodnost údajů, ale to pouze v případě, že není porušen žádný z článků řetězce [10].



Obr. 3.3: Riziko DNS komunikace.

3.3 Dynamic Host Configuration Protocol (DHCP)

Je názvem protokolu, který působí na aplikační vrstvě modelu TCP/IP na principu klient-server. Dynamic Host Configuration Protocol slouží pro automatickou konfiguraci síťových parametrů u zařízení, které jsou připojené do počítačové sítě. Díky tomu výrazně ulehčuje práci správcům sítě, kteří nemusí nastavovat parametry každému síťovému zařízení zvlášť. Protokol informuje všechny klienty o sadě parametrů, které jsou nutné pro komunikaci přes IP protokol. Díky DHCP, klienti nemusí znát informace o síti, do které se připojují, aby v síti mohli bezproblémově komunikovat s ostatními zařízeními. Jak už název napovídá, protokol zajišťuje dynamickou konfiguraci nutných parametrů, které jsou potřeba k tomu, aby klienti mohli v síti komunikovat. Konfigurace přidělených parametrů se tedy v průběhu času mění, jelikož dochází k jejímu vypršení. DHCP server tak klientům propůjčuje parametry, aby v síti mohli klienti komunikovat. Po vypršení intervalu přidělených parametrů si musí klienti opět zažádat u serveru obnovení konfiguračních parametrů, které mohou, ale nemusí, být jiné [11].

DHCP nejčastěji nastavuje tyto parametry:

- IP adresa
- maska sítě

- výchozí brána
- list dostupných DNS serverů
- další údaje jako například NTP, WINS...

3.3.1 Výhody DHCP

Protokol výrazně ulehčuje a centralizuje správu počítačové sítě například, pokud se do sítě budou přidávat nové stanice nebo při hromadné změně parametrů stanic.

Protokol dále poskytuje jednodušší správu a šetření adresního prostoru, kdy nemůže dojít k tomu, že se v síti objeví dvě stejné IP adresy, což nelze zaručit, při ručím zadávání.

Jelikož protokol pracuje sám bez vnějšího zasahování, uživatelé tak nemusí nastavovat na svých stanicích nic v souvislosti s připojením do sítě. Jedinou podmínkou je mít zapnutou službu DHCP, která je dnes součástí všech operačních systémů a ve výchozím stavu je vždy spuštěna. Klient tak automaticky po připojení do sítě komunikuje s DHCP serverem, který mu zašle potřebné parametry, aby měl klient přístup do sítě [5].

3.3.2 Princip komunikace DHCP

V prvním kroku komunikace klient neví, jakou adresu má kontaktovat, aby navázal spojení s DHCP serverem, který by mu poskytl žádané parametry. Z tohoto důvodu vyšle klient zprávu DHCP_DISCOVER, která značí, že klient potřebuje přidělit IP adresu. Zpráva je ve formě broadcast, což znamená všesměrová. Zpráva tak zaplaví celou síť, ale odpoví na ni pouze příslušný DHCP server. U ostatních zařízeních je zpráva DHCP_DISCOVER ignorována.

V dalším kroku zašle DHCP server zprávu DHCP_OFFER, která říká, že pro má pro klienta, který zaslal zprávu DHCP_DISCOVER, IP adresu. Počítač zprávu přijme a odpoví na ni zprávou DHCP_REQUEST, která znamená požadavek na přidělení nutných parametrů. Na tuto zprávu server odpoví zprávou DHCP_ACK, která znamená potvrzení přiřazení parametrů a v systému si zaeviduje přiřazenou IP adresu, aby nemohla být přiřazena jinému zařízení.

Důležitým prvkem DHCP komunikace je i tvz. předávací agent, neboli DHCP relay agent, který slouží pro propojení sítí, které jsou oddělené směrovačem. Obvykle se broadcast, neboli všesměrové zprávy nedostanou mimo síť, kde není DHCP server, ale počítač z jiné sítě chce přidělit adresu.

Z tohoto důvodu je možnost na směrovači zapnout předávacího agenta, který zajistí, že všechny všesměrové zprávy ze sítě, kde není DHCP server doputují přes směrovač i do sítě, kde se DHCP server vyskytuje a bude klientovi doručena zpráva s IP adresou, kterou může použít. Agent také k přijatému dotazu přidá informaci

o tom, z jaké sítě dotaz pochází, aby DHCP věděl, z jakého adresního rozsahu má klientovi poskytnou IP adresu. [5].

3.4 Network Time Protocol (NTP)

Network Time Protocol slouží pro synchronizaci vnitřních hodin zařízení, tak aby měla všechna zařízení synchronizovaný čas. Čas v internetu může být jednoduše přehlédnut, ale pokud nejsou zařízení synchronizovány na stejný čas, může rozdíl jen pár sekund způsobit potíže. Veškerá zařízení v sobě udržují vlastní vnitřní hodiny, které uchovávají datum a čas. V případě desynchronizace času u jednotlivých zařízeních vzniknou problémy. Příkladem důležitosti synchronizace hodin je například nákup na e-shopu. Vznik objednávky musí byt zaznamenán s přesným časem vytvoření objednávky v návaznosti na další transakce, které jsou v reálném světě, jako například zaplacení faktury ve splatnosti, dodání ve lhůtě nebo také vyřízení reklamace.

Dále je synchronizace času nutná i pro běh některých aplikací, jako jsou například různé kalendáře, budíky nebo start procesů na serverech. Některé operace mají přesně daný čas spuštění na různých zařízeních v rámci jedné sítě. Je proto klíčové, aby spouštění aplikací proběhlo na všech zařízeních ve stejný čas, nebo v předem dané časové posloupnosti. Z těchto důvodu je synchronizace času v zařízeních nutná.

3.4.1 Princip protokolu NTP

NTP protokol využívá nespolehlivou, nespojitou službu User Datagram Protocol, neboli UDP na portu 123. Základní myšlenka protokolu NTP je taková, že existuje spoustu různých zařízení, které mohou poskytnout přesný čas. Jsou to zařízení jako rádia, GPS nebo v extrémním případě cesiové atomové hodiny. Tato zařízení se ve struktuře NTP protokolu nazývají vrstvou stratum 0. Tato vrstva poskytuje přesný čas serverům, které jsou na vrstvě nazývající se stratum 1. Na další, v pořadí již třetí vrstvě nazývající se stratum 2, jsou již zařízení, kterým jsou předávány informace o přesném čase. Tyto zařízení však nedostávají pouze jednu odpověď z jednoho serveru, ale v rámci určení přesného času komunikují s více servery a nebo také mezi sebou [12].

Existuje tedy několik různých vztahů mezi zařízeními synchronizace času:

- Server / klient
  - K synchronizace času jednoho klienta poskytuje i několik serverů na požádání přesný čas.

- Peer
  - Jedná se o dvojici strojů, které si spolu předávají informace o přesném čase. Ten, který má nejpřesnější čas pak vystupuje jako server a ostatní jako klienti.
- Broadcast / multicast server a klient
  - V pravidelných intervalech jsou ze strany serveru odesílány informace o časech na broadcastovou nebo multicastovou adresu.

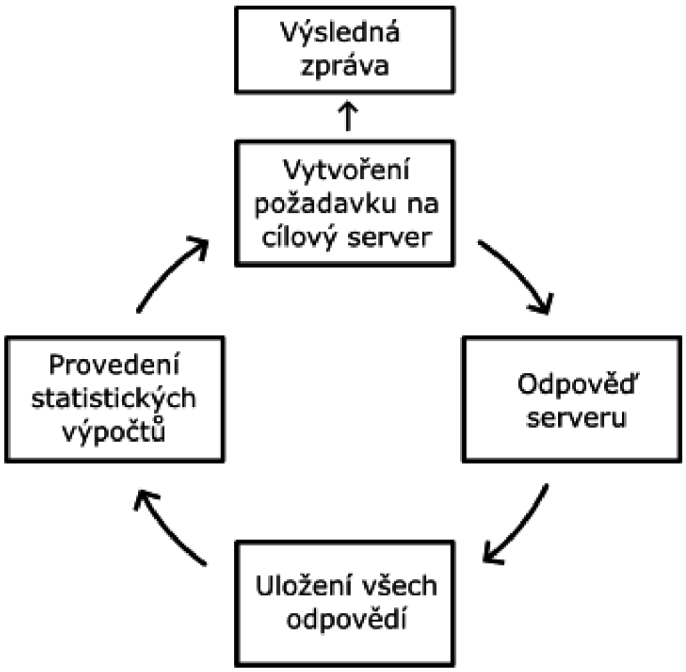
4 Zátěžové testování

Zátěžové testování je dnes v oblasti komunikačních technologií široce využívaný pojem, se kterým se dnes často setkáváme u moderních technologií a jejich služeb. V obecném použití tohoto termínu se jím myslí řízení zatížení pro určité zařízení, systém, nebo službu a sledování reakcí na danou zátěž. Zátěžové testování slouží k výkonovému testování síťových prvků a serverů, které bývají často předmětem DDoS útoků, které jsou detailně popsány v kapitole 5. Důvody k provedení výkonnostních testů nejsou jen útoky z okolí, ale také ke zkoušce zátěže například určité webové stránky, na kterou uživatelé v jednu chvíli zašlou tisíce požadavků. Výsledkem jsou poté cenné informace o průběhu zatížení, omezeních a funkčnosti daného systému při zátěži, které je vystaven.

Primárním záměrem zátěžového testu je získat informace o chování objektu při určité zátěži, které je vystaven a získat informace, které určí, kde je kritická hranice zátěže u daného zařízení a jaké chování lze od zařízení v takovémto extrémním případě očekávat. Další informace, které by zátěžové testování mělo poskytnout je odhalení slabin daného objektu a tím i předcházení vzniku takových situacím.

Obecně se zátěžové testování provádí pomocí zaslání určitého požadavku na daný objekt, kterým může být zařízení či určitá služba. Dalším krokem je získání statistických informací z cílového objektu nebo služby, jako je délka odezvy, nebo jestli zařízení stále komunikuje a zpracovává požadavky. Posledním krokem je vygenerování výsledné testovací zprávy, která obsahuje souhrn o průběhu testu a výsledcích testu. Všechny kroky jsou znázorněny na obrázku 4.1.

Dnes již existuje celá řada zátěžových testerů, které jsou schopny určit výkon daného objektu, nebo služby. Tato bakalářská práce se zaměřuje na zátěžový tester Apache JMeter, který slouží pro testování chování, analýzu a měření výkonu různých aplikací a služeb [12].



Obr. 4.1: Průběh testovaní pomocí nástroje Apache JMeter.

5 Kybernetické útoky typu odepření služby (DoS)

Kybernetické útoky typu odepření služby (anglicky Denial of Service (DoS)) je typ útoku, při kterém se útočník snaží docílit výpadku určité služby a uvést ji do nedostupného stavu pro ostatní uživatele v síti.

Obecným principem útoku typu odepření služby je zahlcení cílové stanice nebo služby velkým množstvím požadavků, dokud nedojde k vyčerpání systémových prostředků a tím pádem k přetížení systému a následnému blokování dostupnosti dané služby [14].

5.1 Dělení DoS útoků

DoS útoky se dělí podle svých charakteristických vlastností do několika kategorií, aby bylo možné jednodušeji určit specifické vlastnosti útoků, jako je například provádění nebo zaměření útoku.

Tato kategorizace se využívá k systematickému přístupu k analýze a ochraně proti těmto útokům.

5.1.1 Podle druhu útoku

Organizace CERT Cordination Center definuje tři základní druhy DOS útoků [15]

- Zaměřené na spotřebu vzácných, limitovaných nebo neobnovitelných zdrojů,
- zaměřené na poškození či záměna konfiguračních informací,
- zaměřené na fyzické poškození či záměnu síťových zařízení.

5.1.2 Podle zdrojů cíle

Útoky, které se zaměřují na maximální využití omezených zdrojů cíle se dále dělí na útoky, které jsou zaměřeny na vyčerpání:

- Systémových zdrojů,
- síťových zdrojů,
- aplikačních zdrojů.

Vyčerpání systémových zdrojů

Tento typ útoku má za cíl maximálně využít zdroje cíle, jako je paměť RAM nebo CPU, což má za následek odepření služby pro ostatní uživatele v síti [16].

Vyčerpání síťových zdrojů

Útok zaměřený na vyčerpání síťových zdrojů se zaměřuje na využití veškeré kapacity síťového připojení (bandwidth) oběti tím, že generuje velké množství nelegitimního provozu. V malém množství se útok může jevit jako normální provoz, avšak při velkému objemu může síť cíle zahltit [16].

Vyčerpání aplikačních zdrojů

Tento typ útoku se zaměřuje na slabiny aplikačních protokolů, jako je Hypertext Transfer Protocol (HTTP), či jeho zabezpečená varianta HTTPS, ale také protokoly jako je Domain Name System (DNS), File Transfer Protocol (FTP) nebo Voice over Internet Protocol (VoIP) [16].

5.1.3 Podle rychlosti

Dále lze DoS útoky dělit podle jejich intenzity na:

- Floods, neboli záplavové,
- low and slow, neboli pomalé.

Cílem floods útoků je zahltit cíl obrovským množstvím dotazů, díky čemuž je dosaženo zamezení dostupnosti provozu dané služby. Pro tyto útoky je typickou jednotkou množství paketů za sekundu (pps), nebo objem datového toku (Mbps).

Low and slow útoky je těžké rozeznat, protože vytvářejí provoz, který je těžko rozeznatelný od toho normálního, díky menšímu generování požadavků na cílový objekt. Tyto typy útoky většinou využívají slabiny aplikačních protokolů [16].

5.1.4 Jedno-zdrojový a distribuovaný útok

Škodlivý provoz může být odeslán z jedné stanice nebo z více stanic, které má útočník pod kontrolou. Pokud škodlivý provoz pochází z jedné stanice, je nazýván single-source denial of service (SDoS) attack. V případě, kdy škodlivý provoz pochází z více stanic, které jsou v různých sítích, distributed denial of service (DDoS) attack.

U DDoS útoků figurují dvě komponenty, a to agent a handler. Agent je často označován jako zombie nebo bot, který je spuštěn na infikované stanici kde generuje škodlivý provoz v síti. Souhrn více agentů, neboli botů, či zombies, které ovládá jeden útočník je označován jako botnet. Handlerem je označován program, který má pod kontrolou všechny agenty, které ovládá a sděluje jim, kdy a na koho mají v danou chvíli zaútočit.

DDoS útoky bývají efektivnější než SDoS útoky z hlediska součtu výkonu stovek až tisíců nakažených stanic, kterému jednotlivé komponenty většinou nedokáží

dlouho odolávat. Taktéž i odhalování a prevence proti takovým útokům je mnohem náročnější než proti SDoS útokům [16].

5.2 Typy DoS útoků

Existuje mnoho typů DoS útoků, které se liší svým provedením, zaměřením a dopady na cílový systém. Každý typ útoku má své vlastní charakteristické vlastnosti, podle kterých lze určit, o jaký druh útoku se jedná.

V této kapitole je vyjmenováno a popsáno několik základních typů DoS útoků.

5.2.1 SYN flood

Tento typ útoku využívá slabiny protokolu TCP. Konkrétně navázání spojení pomocí tvz. three-way handshake mechanismu.

TCP protokol má na rozdíl od UDP protokolu spojitý charakter. Snaží se tedy o spolehlivý přenos dat. K tomu využívá potvrzovacích mechanismů, podle kterých se ověřuje, zda došlo k úspěšnému přenosu. Před samotným přenosem dat musí TCP protokol navázat spojení. Pro navázání spojení využívá TCP three-way handshake, který obsahuje příznaky SYN, SYN-ACK, ACK. Příznakové bity (flags) slouží k určení, zda se jedná o navázání, či ukončení spojení, ale také, zda se jedná o naléhavou zprávu urgent (URG), push function (PSH) nebo reset the connection (RST).

Při navazování spojení pomocí three-way handshake, vytváří každá žádost SYN polo-otevřené spojení, kdy se očekává odpověď SYN-ACK a nakonec je žadatelem potvrzeno přijetí odpovědí pomocí příznaku ACK.

Při útoku SYN flood je na cílovou stanici zasláno velké množství SYN příznaků, které se zpočátku jeví jako běžný provoz. Tyto žádosti ale obsahují neexistující a podvržené IP adresy. Server, na který je útok zaměřen, vytváří pro každý SYN příznak nové vlákno a tím alokuje prostředky ve své vyrovnávací paměti (anglicky buffer) pro přípravu na vlastní spojení. Server se dále pokouší odeslat paket s příznakem SYN-ACK zpět zdroji zprávy, avšak protože jsou zadány neexistující adresy, server se paketu s příznakem SYN nikdy nedočká. Server tak musí stále udržovat aktivní vlákno s alokovanými prostředky ve vyrovnávací paměti pro každý požadavek a pokouší se o zaslání požadavku SYN-ACK do doby, než vyprší časový limit pro zpracování konkrétní žádosti. SYN flood tímto způsobem vyčerpává omezené prostředky díky snaze zpracovávat velké množství žádostí, dokud nedojde k zahlcení serveru a tím odepření služby pro ostatní uživatele [16].

5.2.2 UDP flood

Pro komunikaci v sítí se využívá kromě TCP protokolu, který slouží pro spolehlivou komunikaci také User Datagram Protocol (UDP), který je charakteristický nespojitým, nespolehlivým provozem. UDP tedy nevyžaduje sestavení spojení pro komunikaci mezi stanicemi. Cílová stanice tím pádem ani nepotvrzuje, že obdržela zprávu od odesílatele.

UDP flood využívá slabin serveru, než protokolu jako takového. V běžné komunikaci zdrojová stanice vysílá UDP datagramy na cílovou stanici. Komunikace je tak díky nízké režii provozu rychlá a s minimálním zpožděním. Pokud však UDP datagram obsahuje neplatnou neexistující IP adresu, zašle se ICMP odpověď destination unreachable, což v překladu znamená cíl nedosažitelný.

V případě velkého objemu provozu jsou porty cílové stanice zahlceny UDP datagramy s neplatnou IP adresou, na které zasílá ICMP pakety o nedosažitelnosti stanic. Cílová stanice tak využije veškeré své síťové prostředky na odesílání odpovědí, aby oznámila, že na daných portech nenaslouchá žádná aplikace [16].

5.2.3 HTTP flood

HTTP flood využívá legitimních dotazů HTTP GET nebo POST, které jsou mířené na webovou stránku serveru. Díky legitimním dotazům se stává detekce a obrana DDoS útoku HTTP flood velice obtížnou.

Metoda HTTP GET se využívá pro stažení klasického statického obsahu, jako jsou například obrázky. Data se v metodě předávají v URL adrese jako parametry. Žádost POST slouží pro získání přístupu k dynamicky generovaným aplikačním zdrojům serveru. Data v metodě POST putují skrytě společně s požadavkem na stránku. HTTP většinou využívá sítě botnet, nebo sít agentů, botů nebo zombies, kde stanice hromadně zasílají HTTP žádosti a tím zahlcují cílový webový server, který využívá všechny své aplikační prostředky na zpracování těchto požadavků, díky čemuž dojde k odepření služby všem ostatním uživatelům.

Metody POST obsahují parametry, které na straně serveru mohou spouštět komplexní operace, které jsou náročné na procesní výkon. Metody GET jsou naopak snadnější na proveditelnost a s růstem sítě botnet roste také jejich efektivita.

Výhodou DDoS útoku HTTP flood jsou nižší požadavky na rychlost připojení (bandwidth), než u ostatních DDoS útoků, aby se docílilo zamezení služby pro ostatní uživatele v síti [16].

5.2.4 ICMP flood

ICMP flood taktéž známý jako ping flood je útok, který využívá Internet Control Message Protocol (ICMP) požadavky typu Echo Request, který je také známý jako příkaz ping.

ICMP flood pracuje na čtvrté vrstvě síťového modelu ISO/OSI. Jeho princip spočívá v zahlcení cílové stanice velkým objemem příkazů Echo Request (ping). U cílové stanice, ve snaze odpovědět na všechny ICMP zprávy, dochází k vyčerpání kapacity připojení pod záplavou ICMP požadavků, což má za následek odepření služby ostatním uživatelů v síti. Tento typ útoku vyžaduje, aby útočník měl k dispozici vyšší rychlost připojení než má oběť, aby oběť nestíhala zpracovávat ping požadavky [17].

5.2.5 DNS Flood

DNS flood cílí na DNS servery, které zahltí velkým množstvím dat, díky čemuž vznikne odepření služby DNS. Vznikne tak odepření překladu jména zadaného klientem na IP adresu dané služby.

Realizace útoku DNS Flood

V současné době se nejvíce využívá k realizace daného útoku malware, který napadá zařízení s operačním systémem Linux. Nemusí se ale nutně jednat pouze o počítače a servery, ale mohou se zde zařadit veškerá zařízení Internet of Things, neboli internetů věci, což mohou dnes být veškerá domácí elektronická zařízení s přístupem na internet.

Stroje, které jsou napadené nepřátelským malwarem hromadně posílají dotazy na DNS server. Ten je takovým provozen zahlcen a ztrácí schopnost odpovídat na legitimní dotazy klientů [19].

DNS Amplification

Útok typu DNS Amplification zneužívá službu DNS, tedy překlad doménových jmen. Konkrétně vlastnost, kdy je dotaz na DNS server mnohonásobně menší, než jeho odpověď.

Princip útoku DNS Amplification je v zamaskování IP adresy oběti (neboli IP spoofing) v paketu, který se dotazuje na DNS službu. Při využití protokolu UDP je jednoduché zamaskovat, nebo podvrhnout zdrojové IP adresy, jelikož protokol, na rozdíl od TCP, neověřuje IP adresy. IP adresa útočníka je tedy velice těžko dohledatelná.

DNS Amplification znamená v překladu zesílení DNS útoku. Využívá se v něm proměnlivá délku DNS paketu. DNS paket neobsahuje v případě dotazu část pro odpověď, pro autoritativní servery a pro dodatečné informace. DNS služba je schopna vrátit odpověď o velikosti až 4096 bajtů na relativně malý dotaz.

Tímto způsobem se DNS server chová jako útočník, který zasílá pakety o velké velikosti na nic netušící oběť, která je nutná pakety přijímat, protože je v paketech změněná IP adresa na adresu oběti [20].

5.2.6 NTP Flood

NTP Flood využívá funkčnost NTP protkolu. Konkrétně jeho příkaz s názvem monlist, který klientům zasílá zpět seznam IP adres posledních 600 klientů, kteří byli připojeni k NTP serveru. Na server je tedy zaslán dotaz MON_GETLIST_1, který obsahuje cílovou adresu oběti, na kterou je směřován útok z NTP serveru.

Tento příkaz primárně slouží pro monitoring provozu na serveru, ale je velmi obtížné rozeznat, jestli se jedná o legitimní dotaz či útok na server.

Jedná se tedy amplifikační útok, vek kterém chce útočník dosáhnout, aby odpověď zaslaná serverem byla několikanásobně větší, než dotaz ze strany útočníka. Server je tedy zahlcen odesíláním odpovědí na dotazy monlist. Tím způsobem je zahlcen a nezbývá mu dostatek šířka pásma a přestává odpovídat na legitimní dotazy klientů. Dochází tedy k odepření služby a úspěšnému DDoS útoku NTP Flood [18].

6 ICT Tester

V této kapitole je popsán ICT tester, který je vyvíjen na Ústavu telekomunikací na Vysokém učení technickém v Brně. Je zde popsána souvislost ICT testeru s programem Apache JMeter a modulem Server Emualtor, který je v této práci rozšířen o vybrané serverové služby.

6.1 Zátěžový tester ICT

Zátěžový tester ICT je souhrn modulů pro testování zátěže pomocí DDoS flood útoků, jako je například DNS flood, ICMP flood, NTP flood či SYN flood. Moduly umí mimo jiné i definovat například různé filtry pro síťová rozhraní, jejich přenosovou rychlost nebo ztrátovost. Hlavním modulem, kterým se tato bakalářská práce zabývá je modul Server Emulator, který slouží pro řízení HTTP serveru Apache a FTP serveru Vsftp. Modul umí tyto serverové služby spustit, zastavit, zobrazit jejich stav a v případě HTTP serveru, má modul také konfigurační možnosti prostřednictvím GUI. Server Emulator také disponuje možností vytvoření vzorové HTML stránky o zadané velikosti, kterou si zvolí uživatel. Stránka je následně zaslána zpět HTTP serverem jako odpověď na dotaz na základní URL serveru. Modul umí také odesílat dotazy jedním síťovým rozhraní na serveru a zároveň naslouchat na druhém síťovém rozhraním stejného stroje.

Všechny tyto výše zmíněné moduly využívají jako základ Apache JMeter [20].

6.1.1 Apache JMeter

Apache JMeter je open-source program, který je vytvořen programovacím jazykem Java. Jeho hlavním účelem je výkonnostní testování různých služeb. Původní účel programu JMeter spočíval v testování webových aplikací, ale s růstem možností testování, které do něj byly postupně přidávány, se z něj stal robustní nástroj pro testování chování webových aplikací či různých služeb a jejich zátěžové testování [21].

6.2 Impelemtace serverových služeb

V této kapitole jsou popsané základní funkce vybraných serverových služeb a jejich možná implementace a možnosti přes programové balíčky v Linuxové distribuci Ubuntu.

6.2.1 DNS

Implementace vlastního DNS serveru je možné skrze mnoho programů, které slouží pro vytvoření vlastního DNS serveru. Velice populárním DNS serverem je server bind9 (Berkeley Internet Name Domain), který poskytuje mnoho funkcí jako je například vyvažování zátěže, dynamické aktualizace, rozdělení DNS nebo možnost IPv6.

Pro vytvoření serveru je možné použít jak operační systém Windows, tak i více preferovaný systém Linux. Z hlediska bezpečnosti, ceny a podpory komunity je vhodnější použít Linux server. Windows server má výhodu ve více uživatelsky přívětivějším přístupu k uživateli ve formě grafického rozhraní, namísto linuxové příkazové řádky [23].

6.2.2 DHCP

DHCP protokol umožňuje dynamickou konfiguraci nutných parametrů klienta pro připojení se do sítě, aby bylo možné v síti komunikovat. Detailní popis komunikace je popsán v kapitole 3.3.

V prostředí Ubuntu lze DHCP službu provozovat pomocí balíčku ins-dhcp-server, který slouží pro nastavení základních parametrů klienta jako je:

- IP adresa,
- maska,
- IP adresa výchozí brány,
- IP adresa DNS serveru.

Dalšími parametry, které balíček umožňuje předat je:

- Název hostitele,
- název domény,
- časový server,
- tiskový server [24].

6.2.3 NTP

Implementace služby NTP byla v bakalářské práci zprostředkována skrz balíček NTP. Linuxová distribuce Ubuntu má v svém základu pro synchronizaci času službu systemd-timesyncd. Tato služba však slouží pouze jako SNTP (Simple Network Time Protocol). SNTP je zjednodušená verze NTP, která pracuje pouze v modu klient a je tedy neschopna poskytovat funkce NTP serveru. SNTP funguje na jednoduchém dotazování časových serverů, které obratem zasílají informace o čase, podle kterých jsou aktualizovány hodiny v systému.

7 Výsledky studentské práce

V praktické části se tato bakalářská práce zabývá implementací serverových služeb do zátěžového testeru ICT. Konkrétně do modulu Server Emulator, který slouží pro řízení HTTP serveru Apache a FTP serveru Vsftpd v programovém prostředí linuxové distribuce Ubuntu.

Do modulu Server Emulator byly přidány tři služby. Konkrétně služba DNS, která je implementována pomocí balíčku *bind9*, služba DHCP zprostředkovaná pomocí balíčku *ics-dhcp-server* a služba NTP, která je zprostředkována balíčkem *ntp*.

Nejprve bylo v bakalářské práci provedeno vytvoření uživatelského rozhraní pro ovládání daných služeb. Toho bylo docíleno vytvořením nových záložek, které se přidaly do modulu Server Emulator. Všechny služby zachovávají původní konfiguraci, která je nastavena v konfiguračních souborech daných služeb. Spuštění služeb tedy vytvoří nový konfigurační soubor a původní zachová. Vypnutí služby naopak smaže nově vytvořenou konfiguraci a vrátí původní. S ukončením programu JMeter jsou automaticky zastaveny všechny služby a jsou vráceny všechny původní konfigurační soubory nově implementovaných služeb v modulu Server Emulator.

V rámci modulu Server Emulator je k dispozici okno, do kterého se vypisují odpovědi z ovládání daných služeb. Toto okno poskytuje okamžitou zpětnou vazbu, pokud se daná služba spustí, nespustí, nebo restartuje. Ve výpisech je také vidět, zdali se povedlo vytvořit nové konfigurační soubory při spuštění, nebo vrátit původní konfigurační soubory při zastavení dané služby.

7.1 Implementace služby DNS

Všechny níže specifikované základní konfigurace DNS serveru, kromě nainstalování balíčku *bind9*, zprostředkovává modul Server Emulator v záložce Domain Name System. Původní konfigurace modul pouze přejmenuje a vytvoří nové, se kterými dále pracuje DNS server. Při zastavení služby pomocí tlačítka stop se nově vytvořené konfigurace vymažou a původní vrátí zpět. Díky této funkcionalitě se uživatel nemusí bát, že by Server Emulator smazal původní konfiguraci.

Při spuštění serveru DNS pomocí tlačítka start se v hlavní konfigurační složce `/etc/bind/` vytvoří základní zónový soubor `db.ubuntu.local`, ve kterém jsou specifikovány základní nastavení pro definovanou zónu.

Dále Server Emulator vytvoří nový soubor `named.conf.local`
a soubor `named.conf.options`. První z těchto dvou souborů slouží k deklaraci vytvořené zóny pomocí souboru `db.ubuntu.local`. Ve druhém souboru je specifikováno naslouchání služby DNS na specifikované adrese a portu pomocí grafického rozhraní služby DNS v modlu Server Emulator.

Implementace DNS služby byla zprostředkována pomocí balíčku *bind9*. Bylo tedy nutné tento balíček nainstalovat pomocí příkazu `sudo apt install bind9` a provést základní konfiguraci, která je vidět na obrázku 7.1.

```
options {
        directory "/var/cache/bind";

        // If there is a firewall between you and nameservers you want
        // to talk to, you may need to fix the firewall to allow multiple
        // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

        // If your ISP provided one or more IP addresses for stable
        // nameservers, you probably want to use them as forwarders.
        // Uncomment the following block, and insert the addresses replacing
        // the all-0's placeholder.

        forwarders {
                1.1.1.1;
                8.8.8.8;
        };

        //========================================================================
        // If BIND logs error messages about the root key being expired,
        // you will need to update your keys.  See https://www.isc.org/bind-keys
        //========================================================================
        dnssec-validation auto;

        listen-on-v6 { any; };
};
```

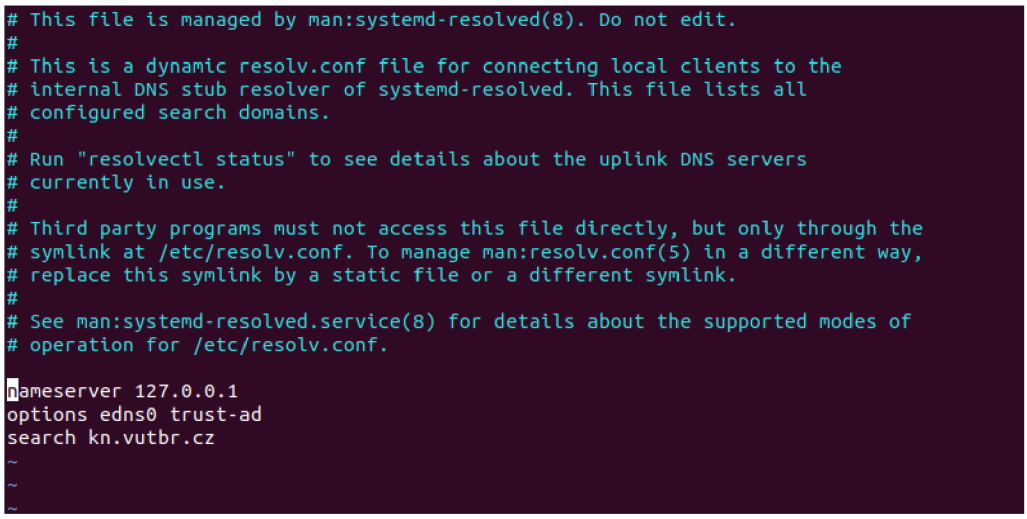
Obr. 7.1: Základní nastavení souboru named.conf.options.

Nastavením forwarders na 1.1.1.1 a 8.8.8.8 budou dotazy přeposílány na tyto externí DNS servery pro obdržení odpovědi s IP označení serveru. Ubuntu má vlastní DNS rozhraní, které je potřeba vypnout, aby byla využívána výhradně DNS služba *bind9*.

Toho bylo docíleno příkazem `sudo systemctl stop systemd-resolved`. Pro zajištění, aby služba nebyla opět aktivována byl použit příkaz
`sudo systemctl disable systemd-resolved`.

Dále v souboru `stub-resolv.conf` byla nastavena položka nameserver na 127.0.0.1, která reprezentuje naši IP adresu localhost, tedy náš počítač. Tímto bylo docíleno, že výchozí služba `systemd-resolved` nebude využívána a bude využívána služba *BIND* pro vytvoření DNS serveru. Nastavení je vidět na obrázku 7.2.

V neposlední řadě je nutné DNS serveru specifikovat oblast jmenného prostoru, neboli zónu. Zóny slouží k rozdělení jmenného prostoru na logické části a tím usnadňují administrátorům správu DNS záznamů. Nejdříve je potřeba pro zónu vytvořit zónový soubor v konfiguraci *bind9* ve složce `/etc/bind/`. Název souboru je volitelný. Do tohoto záznamu je potřeba zadat parametry pro nastavení zóny. Příklad konfigurace je vidět na obrázku 7.3. IP adresy v této konfiguraci jsou adresy Ubuntu DNS serveru. V tomto souboru se definují tyto parametry:



Obr. 7.2: Konfigurace souboru stub-resolv.conf.

- $TTL: Definuje dobu, po kterou bude záznam uložen v dočasné paměti DNS serveru. Hodnota je zadávána v sekundách (604800 = 7 dní).
- SOA: neboli Start Of Authority, slouží k řízení chování DNS zóny. IN je označení třídy pro internet.
  - Serial: Sériové číslo, které se z každou změnou záznamu zvětší.
  - Refresh: Čas, po kterém se obnoví verze zóny, udávaná v sekundách.
  - Retry: Čas, po kterém sekundární server obnovit data z primárního serveru.
  - Expire: Čas, po kterém budou sekundární servery považovat zóny za neplatnou.
  - Negative Cache TTL: Čas, po který budou ostatní DNS servery uchovávat záznamy z této zóny.
- ns: Neboli Name Server record. Jedná se o definici autoritativního serveru DNS.
- A: Slouží pro spojení IPv4 adresy s doménovým jménem.
- AAAA: Slouží pro spojení IPv6 adresy s doménovým jménem.

Dále je nutné do souboru `named.conf.local` zónu deklarovat pomocí přidání odkazu na zónový soubor. Příklad konfigurace je na obrázku 7.4. V tomto souboru je specifikováno:

- zone: Určení jména zóny.
- type master: Hlavní jmenný server.
- file: Zónový soubor, na který se odkazuje.

Posledním krokem je restartovat službu *BIND*, aby se aktualizovala nastavená konfigurace příkazem `sudo systemct status bind9` a zobrazení, zda-li je služba *BIND* aktivní pomocí příkazu `sudo systemctl status bind9.service`. Dále je

```
$TTL    604800
@       IN      SOA     ubuntu.local. root.localhost. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                          86400 )       ; Negative Cache TTL
;
        IN      A       192.168.56.103
        IN      AAAA    2001:0db8:85a3:0000:0000:8a2e:0370:7334
@       IN      NS      localhost.
```

Obr. 7.3: Příklad konfigurace DNS zóny.

```
zone "ubuntu.local" {
        type master;
        file "/etc/bind/db.ubuntu.local";
};
```

Obr. 7.4: Deklarace DNS zóny.

nutné nastavit, aby firewall neblokoval jakoukoli komunikaci na portu 53. Nastavení bylo provedeno příkazem `sudo ufw allow 53/udp`. Status zabezpečení portů pomocí firewall lze zobrazit pomocí `sudo ufw status`.

7.1.1 Ovládání DNS serveru

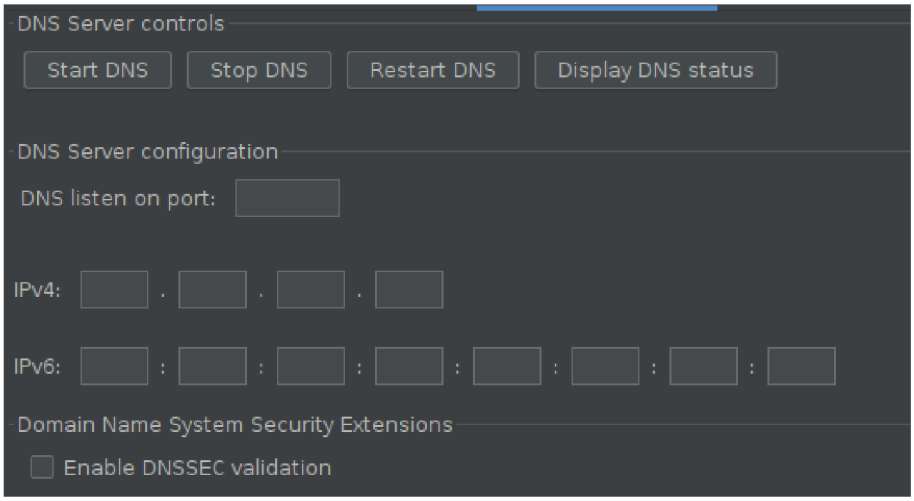
První záložka v modulu Server Emulator nese jméno Domain Name System a slouží pro ovládání služby DNS. Možnosti grafického rozhraní jsou znázorněny na obrázku 7.5.

Přes grafické rozhraní v modulu Server Emulator je možné službu DNS spustit, zastavit, restartovat a zobrazit aktuální stav.

Dále je možné definovat port a IP adresu verze 4 i 6, na kterých bude server pro DNS dotazy naslouchat. Zadané parametry se zapíší do konfiguračního souboru DNS služby, jakmile je služby spuštěna. Uživatel musí zadat port a adresu IPv4 nebo IPv6. Výchozím portem pro službu DNS je port 67. Port, který se zadá do nastavení DNS serveru bude svázán s adresou, která bude zadaná. Služba DNS ve svém konfiguračním souboru neumožňuje definovat přímo rozhraní, na kterém bude DNS služba naslouchat. Volba rozhraní, respektive IP adresy, která musí být vždy přidělena na dané rozhraní, je řešeno v rámci služby DHCP.

Poslední možností je možnost aktivovat služby DNSSEC, která slouží pro zajištění integrity DNS záznamů. Podmínkou funkčnosti DNSSEC je, aby stránka, na kterou se chce uživatel připojit také podporovala DNSSEC. Pokud stránka tuto

službu podporuje, zašle zpět odpověď s příznakem ad (authenticated data), jak je vidět na obrázku 7.6.



Obr. 7.5: Grafické rozhraní pro službu DNS.

```
C:\Users\František Berg>dig isc.org

; <<>> DiG 9.16.34 <<>> isc.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45670
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: d7907c83f9762aa8010000006463a939e100f2e6e5c87dc6 (good)
;; QUESTION SECTION:
;isc.org.                       IN      A

;; ANSWER SECTION:
isc.org.                300     IN      A       149.20.1.66

;; Query time: 164 msec
;; SERVER: 192.168.56.103#53(192.168.56.103)
;; WHEN: Tue May 16 18:03:06 St;; MSG SIZE  rcvd: 80
```

Obr. 7.6: Odpověď DIG isc.org s příznakem AD.

7.1.2 Otestování DNS serveru

Test serveru byl proveden pomocí příkazu `dig`, který byl vyslán na doménu google.com a isc.org.

Příkaz `dig`, neboli Domain Information Groper slouží pro vyhledávání informací DNS serverů a pro odhalování problémů v rámci DNS.

Nejprve byla na hostitelském počítači nastavena IP adresa DNS serveru 192.168.56.103, která reprezentuje Ubuntu DNS server, jak je vidět na obrázku 7.7.

Obr. 7.7: Nastavení serveru DNS na IP serveru Ubuntu.

Nastavení DNS serveru je také možné zobrazit pomocí příkazu `ipconfig /all`, kde je vidět aktuálně využívána adresa DNS serveru. Výpis je možné vidět na obrázku 7.8.

```
Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : kn.vutbr.cz
   Description . . . . . . . . . . . : Realtek USB GbE Family Controller
   Physical Address. . . . . . . . . : 00-E0-4C-3B-66-0D
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   IPv4 Address. . . . . . . . . . . : 147.229.217.183(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.252.0
   Lease Obtained. . . . . . . . . . : čtvrtek 18. května 2023 9:17:48
   Lease Expires . . . . . . . . . . : čtvrtek 18. května 2023 14:39:29
   Default Gateway . . . . . . . . . : 147.229.216.1
   DHCP Server . . . . . . . . . . . : 147.229.191.143
   DNS Servers . . . . . . . . . . . : 192.168.56.103
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

Obr. 7.8: Zobrazení DNS serveru v příkazovém řádku.

Příkazem `dig google.com` bylo dosaženo odpovědi z Ubuntu DNS serveru se statusem NOERROR, jak je vidět na obrázku 7.9. Na obrázku lze také vidět, jak dlouho trvala odpověď a jaká adresa na jakém portu byla kontaktována. V tomto případě Ubuntu server na výchozím portu TCP/UDP 53.

Celá komunikace byla sledována i v Ubuntu DNS serveru na portu 53 na rozhraní ethernet enp0s8 pomocí příkazu `tcpdump -vv -n -i enp0s8 port 53`. Po odeslání příkazu `dig google.com` se paket se zprávou zobrazil také na Ubuntu serveru, jak je vidět na obrázku 7.10.

Pomocí nástroje Wireshark je také možné zachytit DNS dotazy, které jsou vidět ve výpisu na obrázku 7.11.

Ubuntu DNS server, který je spuštěn ve virtuálním prostředí VirtulBox, tedy funguje jako článek, který je pod hostitelským počítačem, přes který je možné se komunikovat s internetem, protože zprostředkovává DNS překlad doménových jmen. Pokud se zadá jiná adresa pro DNS server na hostitelském počítači, není možné se připojit na internet, což je očekávané chování a kontrola, že překlad DNS pracuje správně.

```
C:\Users\František Berg>dig google.com

; <<>> DiG 9.16.34 <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5646
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 69851da12a32941201000000063924684f5f20f04850e9f8b (good)
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             300     IN      A       142.251.37.110

;; Query time: 222 msec
;; SERVER: 192.168.56.103#53(192.168.56.103)
;; WHEN: Thu Dec 08 21:18:12 St;; MSG SIZE  rcvd: 83
```

Obr. 7.9: Odpověď dig google.com.

```
ubuntujeheslo@ubuntujeheslo-VirtualBox:~/Desktop$ sudo tcpdump -vv -n -i enp0s8
port 53
[sudo] password for ubuntujeheslo:
tcpdump: listening on enp0s8, link-type EN10MB (Ethernet), capture size 262144 b
ytes
21:18:12.205199 IP (tos 0x0, ttl 128, id 34196, offset 0, flags [none], proto UD
P (17), length 79)
    192.168.56.1.54552 > 192.168.56.103.53: [udp sum ok] 5646+ [1au] A? google.c
om. ar: . OPT UDPsize=4096 (51)
21:18:12.421232 IP (tos 0x0, ttl 64, id 22621, offset 0, flags [DF], proto UDP (
17), length 111)
    192.168.56.103.53 > 192.168.56.1.54552: [bad udp cksum 0xf225 -> 0x9e14!] 56
46 q: A? google.com. 1/0/1 google.com. A 142.251.37.110 ar: . OPT UDPsize=4096 (
83)
```

Obr. 7.10: Zachycení komunikace přes příkaz DIG google.com.

1	0.000000000	192.168.56.1	192.168.56.103	DNS	93	Standard query 0x71ba A
2	0.000360675	192.168.56.103	192.168.56.1	DNS	125	Standard query response
3	6.618826440	192.168.56.1	192.168.56.103	DNS	89	Standard query 0xae4e A
4	6.634059538	192.168.56.103	192.168.56.1	DNS	121	Standard query response
5	6.747762289	192.168.56.1	192.168.56.103	DNS	77	Standard query 0x91df A
6	6.747762790	192.168.56.1	192.168.56.103	DNS	77	Standard query 0xcedc Un
7	6.756291577	192.168.56.103	192.168.56.1	DNS	185	Standard query response
8	6.756846171	192.168.56.103	192.168.56.1	DNS	221	Standard query response
9	14.860657314	192.168.56.1	192.168.56.103	DNS	89	Standard query 0xda30 A
10	14.899575854	192.168.56.103	192.168.56.1	DNS	121	Standard query response

Obr. 7.11: Zachycení komunikace DNS přes Wireshark.

7.1.3 Zabezpečení DNS serveru

Zabezpečení DNS serverů slouží pro ochranu proti DoS a DDoS útokům, které zapříčiňují odepření a nedostupnost překladu doménových jmen. Existuje několik způsobů, jak je možné zabezpečit DNS server proti útokům:

1. Omezení rychlosti pomocí Response Rate Limit (RRL)
2. Whitelist a blacklist
3. Distribuované DNS služby

Zabezpečení balíčku *bind9*, který zprostředkovává DNS server poskytuje RRL modul (Response Rate Limit), který omezuje počet odpovědí na dotazy na DNS server a chrání tak DNS server proti DoS a DDoS útoky. V kapitole 5 bylo podrobně vysvětleno fungování DNS flood, který slouží pro odepření služby DNS. Základním principem tohoto útoku je posílání velkého množství DNS dotazů a tím zahlcení DNS serveru, který vlivem odpovídání na nelegitimní dotazy nemá prostředky na odpovídání legitimních dotazů. Dochází tak k zahlcení serveru a odepření služby DNS.

Z jednoho paketu s falešnou adresou DNS nemůže poznat nelegitimní provoz. Pokud však dotazů, které mají stejnou IP adresu, přijdou stovky, které mají stejnou, nebo podobnou adresu, je zde vysoká pravděpodobnost, že se jedná o útok s cílem vyřadit DNS službu.

RRL modul omezuje odezvu DNS serveru na základě rychlosti, jakou jsou dotazy zasílány. Pokud DNS server zaznamená vysokou rychlost, kterou jsou zasílány dotazy z jednoho zdroje, omezí počet odpovědí, které jsou zaslány zpět na tuto adresu.

RRL službu lze nakonfigurovat pomocí přidání doložky rate-limit do souboru `named.conf`, kterou lze vidět na obrázku 7.12.

```
options {
        …
         rate-limit {
             responses-per-second 10;
         };
    };
```

Obr. 7.12: Přidání doložky do named.conf.

Po přidání limitu dotazů je nutné server restartovat pomocí příkazu `systemctl reload bind9`. Pomocí této metody se snižuje zátěž a tím zvyšuje odolnost vůči DoS nebo DDoS útokům.

7.2 Implementace služby DHCP

Všechny níže zmíněné kroky pro přípravu základních konfiguračních souborů DHCP serveru, kromě nainstalování balíčku `iss-dhcp-server`, vykonává modul Server Emulator, který při startu DHCP serveru vytvoří základní soubor `/etc/dhcp/dhcpd.conf`, který je znázorněn na obrázku 7.14. Základní konfigurace souboru pokrývá síť 192.168.56.0 s maskou 255.255.255.0. Rozmezí přidělovaných adres je od 192.168.56.20–25. Adresa přidělované výchozí brány je 192.168.56.1 a doba přidělení těchto parametrů je nastavena na 600 sekund, neboli 10 minut.

Do souboru `/etc/default/isc-dhcp-server` se zapíše rozhraní, podle nastavení DHCP serveru v modulu Server Emulatoru, na kterém bude DHCP server naslouchat.

Všechny původní konfigurace modul pouze přejmenuje a při zastavení služby se nová konfigurace vymaže a původní vrátí.

Implementace služby DHCP na Ubuntu serveru byla provedena pomocí balíčku *isc-dhcp-server*. Bylo tedy nutné balíček nainstalovat pomocí příkazu `sudo apt install isc-dhcp-server`.

V první fázi je nutné nastavit v souboru `isc-dhcp-server`, který se nachází v `/etc/default/`, rozhraní pro DHCP žádosti. Nastavení souboru je vidět na obrázku 7.13.

```
# Defaults for isc-dhcp-server (sourced by /etc/init.d/isc-dhcp-server)

# Path to dhcpd's config file (default: /etc/dhcp/dhcpd.conf).
#DHCPDv4_CONF=/etc/dhcp/dhcpd.conf
#DHCPDv6_CONF=/etc/dhcp/dhcpd6.conf

# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
#DHCPDv4_PID=/var/run/dhcpd.pid
#DHCPDv6_PID=/var/run/dhcpd6.pid

# Additional options to start dhcpd with.
#       Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
#       Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="enp0s8"
INTERFACESv6=""
```

Obr. 7.13: Nastavení rozhraní pro DHCP.

Hlavním konfiguračním souborem pro DHCP server pomocí balíčku *isc-dhcp-server* je soubor `dhcpd.conf`, který se nachází v `/etc/dhcp/`. V tomto souboru jsou definovány základní nastavení, jako adresní rozsah, který bude server klientům přidělovat, nebo doba, po kterou bude IP adresa zaznamenána v Ubuntu serveru a klient ji může využívat. Nastavení je vidět na obrázku 7.14.

Dalším krokem v nastavení DHCP je restartování služby příkazem `sudo systemctl restart ics-dhcp-server` a zobrazení statusu, pro potvrzení, že

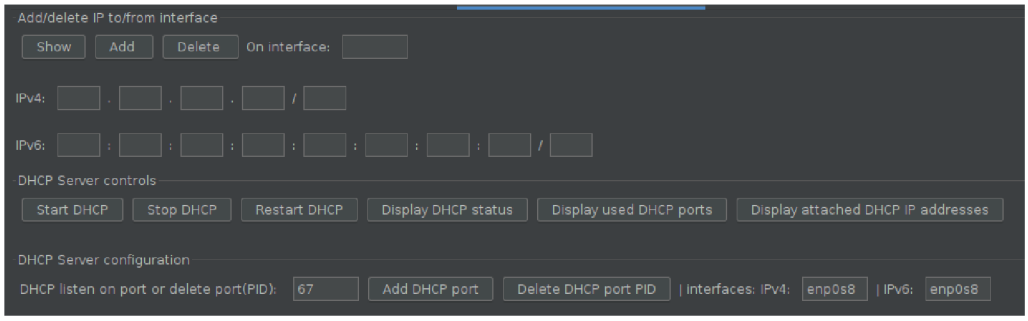
```
subnet 192.168.56.0 netmask 255.255.255.0 {
  range 192.168.56.20 192.168.56.25;
  option domain-name-servers server.example.org;
  option domain-name "example.org";
  option subnet-mask 255.255.255.0;
  option routers 192.168.56.1;
  option broadcast-address 192.168.56.255;
  default-lease-time 600;
  max-lease-time 7200;
}
```

Obr. 7.14: DHCP konfigurace v souboru dhcpd.conf.

je DHCP služba správně nakonfigurovaná a aktivní. To je provedeno příkazem `systemct status isc-dhcp-server`. Je také nutné nastavit firewall brány, aby neblokovala udp komunikaci při DHCP zprávách, které jsou posílány na port 67, příkazem sudo ufw allow 67/udp.

7.2.1 Ovládání DHCP serveru

Grafické rozhraní pro ovládání DHCP serveru přes modul Server Emulator je vidět na obrázku 7.15. Konfigurační soubor DHCP služby umožňuje definovat pouze rozhraní, na kterém bude DHCP naslouchat. V rámci grafického rozhraní proto byla do DHCP panelu přidána možnost přidání nebo odebrání IP adresy verze 4 i 6. Uživatel musí definovat nejdříve rozhraní a poté adresu, která má být přidělena. Bez definice rozhraní Server Emulator upozorní, že rozhraní není definováno a nedovolí přidat adresu. Poslední možností je zobrazení si všech rozhraní a jejich adres, pomocí kliknutí na tlačítko Show. Pokud není zvoleno rozhraní, zobrazí se IP adresy všech rozhraní. Pokud je rozhraní definováno, zobrazí se pouze adresy, které jsou přiděleny právě tomuto rozhraní.



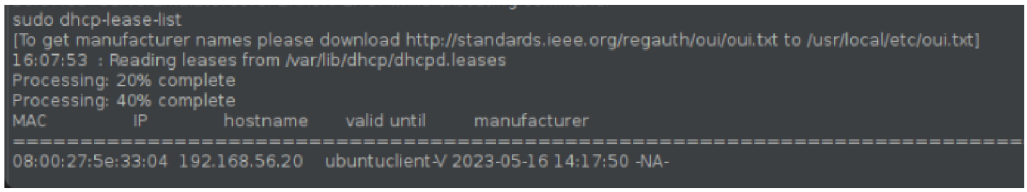
Obr. 7.15: Grafické rozhraní pro službu DHCP.

Dále je pomocí grafického rozhraní možné službu spustit, zastavit, restartovat, zobrazit status, zobrazit si aktivní porty, na kterých DHCP naslouchá a také zobrazit si list zařízení a jejich adres.

Poslední možností je definovat porty a rozhraní, na kterých bude služba naslouchat. Porty je možné následně smazat pomocí PID identifikátoru procesu, který je vidět po zobrazení portů, na kterých DHCP naslouchá pomocí tlačítka Display used DHCP ports.

7.2.2 Otestování DHCP serveru

Služba byla otestována vytvořením jednoho virtuálního Ubuntu klienta se jménem ubuntuclient, který se připojil do stejné sítě a byla mu přiřazena adresa ze zadaného adresního rozsahu ze souboru `dhcpd.conf`. Přidělená adresa klienta je vidět na obrázku 7.16.



Obr. 7.16: Přiřazení adresy klientovi.

Na klientovi je také možné zobrazit, který DHCP server přidělil adresu, nahlédnutím do složky `/var/lib/dhcp/dhclient.leases`. Zde pod řádkem dhcp-server-identifier je IP adresa DHCP serveru, od kterého byla daná adresa přidělena. Názorná ukázka souboru je vidět na obrázku 7.17.

Jak je vidět, klientovi byla přiřazena první adresa ze zadaného adresního rozsahu DHCP serveru Ubuntu. Na obrázku jsou vidět i další parametry, které byly nastaveny DHCP serveru, jako například délka přidělení adresy (dhcp-lease-time 600), adresa výchozí brány (option router), nebo od jakého serveru byla adresa přidělena (dhcp-server-identifier).

Pomocí nástroje CyberShadow/dhcptest je možné zaslat DHCP žádost a tím také otestovat funkci DHCP serveru. Pomocí příkazu `d` CyberShadow vyšle broadcast paket DHCP discover. Přijatá žádost a odeslaná odpověď je zachycena v Ubuntu DHCP serveru na obrázku 7.18.

Na obrázku 7.19 je vidět odpověď DHCP serveru na odeslanou žádost v programu CyberShadow. Zde je vidět jakou adresu DHCP server nabízí (192.168.56.20), z jakého serveru nabídka přichází (192.168.56.103) a další parametry, které byly dříve specifikovány v souboru `/etc/dhcp/dhcpd.conf`, jako doba přidělená adresy, jméno domény, nebo maska sítě.

```
lease {
  interface "enp0s8";
  fixed-address 192.168.56.20;
  option subnet-mask 255.255.255.0;
  option dhcp-lease-time 600;
  option routers 192.168.56.1;
  option dhcp-message-type 5;
  option dhcp-server-identifier 192.168.56.103;
  option broadcast-address 192.168.56.255;
  option domain-name "example.org";
  renew 0 2023/05/21 12:45:35;
  rebind 0 2023/05/21 12:45:35;
  expire 0 2023/05/21 12:45:35;
```

Obr. 7.17: Identifikace DHCP serveru na klientovi.

```
    1 0.000000000   192.168.56.1        255.255.255.255      DHCP      288 DHCP Discover
    3 0.001385182   192.168.56.103      255.255.255.255      DHCP      344 DHCP Offer
```

Obr. 7.18: Zachycení paketu v programu Wireshark na DHCP serveru.

```
Received packet from 192.168.56.103:67:
  op=BOOTREPLY chaddr=7E:5E:41:02:F1:EC hops=0 xid=9291FFB1 secs=0 flags=8000
  ciaddr=0.0.0.0 yiaddr=192.168.56.20 siaddr=192.168.56.103 giaddr=0.0.0.0 sname= file=
  7 options:
     53 (DHCP Message Type): offer
     54 (Server Identifier): 192.168.56.103
     51 (IP Address Lease Time): 600 (10 minutes)
      1 (Subnet Mask): 255.255.255.0
      3 (Router Option): 192.168.56.1
     28 (Broadcast Address Option): 192.168.56.255
     15 (Domain Name): example.org
```

Obr. 7.19: Odpověď v programu CyberShadow.

Celá komunikace je zachycena v programu Wireshark na obrázku 7.20, kde je nejdříve od klienta vyslána broadcast zpráva DHCP Discover pomocí příkazu `dhclient`, která slouží pro nalezení DHCP Serveru. Server odpovídá klientovi pomocí DHCP Offer, kde nabízí možnou adresu s dalšími parametry. Dále je na obrázku vidět zpráva DHCP Request. Tato zpráva říká, že si klient vybral adresu a žádá o ni. Poslední zprávou, kterou je DHCP ACK, si DHCP server potvrzuje vybranou adresa a od této chvíle může klient adresu používat ke komunikaci.

```
   10 89.449721433  0.0.0.0            255.255.255.255      DHCP      344 DHCP Discover
   19 90.475437066  192.168.56.103      192.168.56.20        DHCP      344 DHCP Offer
   20 90.477247950  0.0.0.0            255.255.255.255      DHCP      347 DHCP Request
   21 90.485596498  192.168.56.103      192.168.56.20        DHCP      344 DHCP ACK
   39 111.564041575 192.168.56.20       192.168.56.103       DHCP      344 DHCP Release
```

Obr. 7.20: Celá DHCP komunikace přidělení IP adresy.

7.3 Implementace služby NTP

Všechny níže zmíněné kroky pro konfiguraci základních souborů, kromě nainstalování balíčku `ntp`, zprostředkovává služba Network Time Protokol v modulu Server Emulator. Se spuštěním služby pomocí tlačítka start je vytvořen nový konfigurační soubor `/etc/ntp.conf` a původní zachován. Se zastavením služby pomocí tlačítka stop je služba zastavena a původní konfigurace vrácena. Uživatel se tak nemusí obávat přepsání původního konfiguračního souboru `/etc/ntp.conf`.

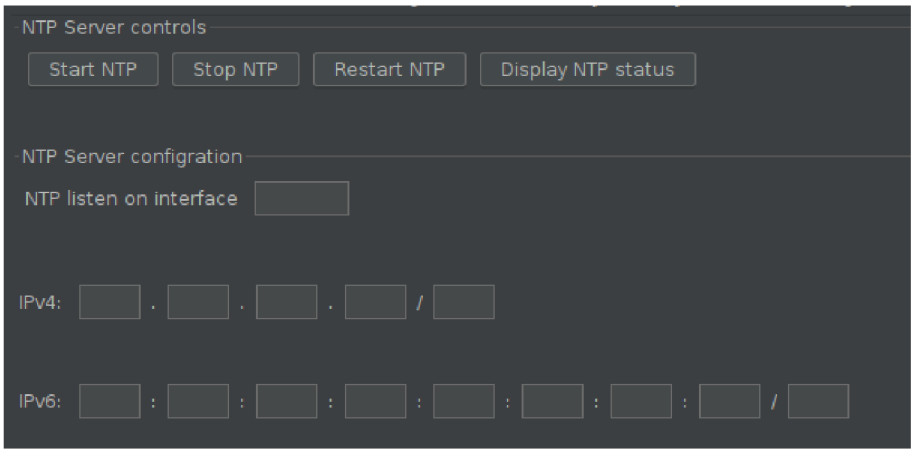
Implementace NTP do Ubuntu serveru byla zprostředkována skrz balíček *ntp*. Pro chod služby je zapotřebí balíček nainstalovat pomocí příkazu
`sudo apt install ntp`. Tímto příkazem se nainstalují všechny potřebné soubory pro chod služby. Aby komunikaci nebránil firewall serveru, je zapotřebí vypnout ochranu portu 123, který je využíván službou, pomocí příkazu
`sudo ufw allow 123/udp`. Po instalaci balíčku *ntp* je vytvořen hlavní soubor
`ntp.conf`, který služba využívá ke konfiguraci nastavení. V tomto souboru je možné definovat hlavní NTP servery pro aktualizaci času, ip adresy nebo rozhraní, na kterých bude služba naslouchat. V souboru je také možné definovat, kam se budou ukládat logy ze služby NTP pomocí přidání `logfile /var/log/ntp.log` a
`logconfig =all` na konec konfiguračního souboru. Tím se vytvoří nový soubor, kam se budou ukládat veškeré logovací záznamy. Tímto je služba připravena ke spuštění pomocí příkazu `sudo service ntp restart` a
`sude service ntp-systemd-netif restart`. Pro ověření správné konfigurace nastavení lze využít příkaz `sudo service ntp status`, v jehož výpisu lze vyčíst, jestli je služba aktivní, neaktivní, nebo v chybovém stavu.

7.3.1 Ovládání NTP serveru

Grafické rozhraní pro ovládání NTP serveru je vidět na obrázku 7.21. Přes Server Emulator je možné službu spustit, zastavit, restartovat a zobrazit její aktuální status. Dále je možné definovat rozhraní, nebo adresu verze 4 nebo 6, na kterých bude služba naslouchat.

7.3.2 Otestování NTP serveru

Pro test funkčnosti NTP služby je zapotřebí zkontrolovat výpis z logu souboru `ntp.log`, ve kterém je informace o tom, na jakém rozhraní a IP adrese služba naslouchá. Příklad výpisu je uveden na obrázku 7.22, kde je vidět, že NTP naslouchá mimo jiné i na adrese 192.168.56.103. Tato adresa musí být použita pro komunikaci s NTP serverem.

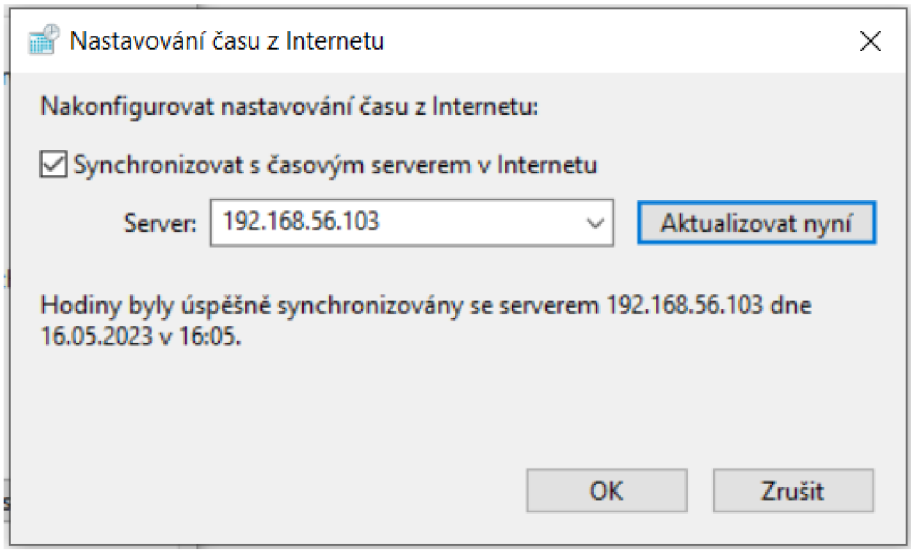


Obr. 7.21: Grafické rozhraní pro službu NTP.

```
16 May 16:02:46 ntpd[784]: Listen normally on 4 enp0s3 10.0.2.15:123
16 May 16:02:46 ntpd[784]: Listen normally on 5 enp0s8 192.168.56.103:123
16 May 16:02:46 ntpd[784]: Listen normally on 6 enp0s3 [fe80::b08e:dce8:2378:bcf
```

Obr. 7.22: IP adresa a rozhraní pro naslouchání NTP dotazů.

Test funkčnosti NTP služby se zajistí pomocí hostitelského počítače, na kterém lze vyzkoušet zaslat žádost o aktualizaci času. To je provedeno pomocí nastavení času v ovládacích panelech, kam se zadá adresa Ubuntu NTP serveru, na které NTP naslouchá. Po určité době by se měl čas aktualizovat z Ubuntu serveru. Úspěšná aktualizace času je vidět na obrázku 7.23.



Obr. 7.23: Úspěšná synchronizace času z NTP Ubuntu serveru.

Pro ověření funkčnosti je také možné použít příkaz `w32tm /query /peers`, `w32tm /query /source`, `w32tm /resync /rediscover`, nebo `w32tm /stripchart /computer:192.168.56.103`. Příklady výpisů těchto příkazů jsou vidět na obrázku 7.24.

```
C:\Windows\system32>w32tm /query /peers
#Peers: 1

Peer: 192.168.56.103,0x9
State: Active
Time Remaining: 32127.0855799s
Mode: 3 (Client)
Stratum: 3 (secondary reference - syncd by (S)NTP)
PeerPoll Interval: 10 (1024s)
HostPoll Interval: 10 (1024s)

C:\Windows\system32>w32tm /query /source
192.168.56.103,0x9

C:\Windows\system32>w32tm /resync /rediscover
Sending resync command to local computer
The command completed successfully.
```

Obr. 7.24: Výpisy příkazů pro ověření funkčnosti NTP.

Výpis příkazu `w32tm /query /peers` obsahuje:

- Peers: 1
  - Informace o tom, s kolika servery je čas synchronizován. V tomto případě s jedním serverem.
- Peer: 192.168.56.103,0x9
  - Označení IP adresy časového serveru, 0x9 slouží pro označení protokolu NTP.
- Active
  - Stav Active slouží pro indikaci stavu probíhající synchronizace.
- Time Remaining
  - Hodnota, který udává zbývající čas v sekundách, po kterém budou hodiny opět synchronizovány s časovým serverem.
- Mode: 3 (Client)
  - Informace i tom, že počítač je v režimu klienta.
- Stratum: 3 -
  - Stratum označuje hierarchickou úroveň synchronizace s časovými servery. Stratum 3 označuje 3. úroveň, která přijímá časové informace od úrovně stratum 2.

- PeerPoll Interval: 10 (1024s)
  - Čas udávaný v sekundách, po kterém se počítač znovu dotáže časového serveru na synchronizaci času.
- HostPoll Interval: 10 (1024s)
  - Čas udávaný v sekundách, po kterém klient bude kontrolovat dostupnost časového serveru

Výpis příkazu `w32tm /resync /rediscover` slouží pro manuální vyvolání synchronizace času. Výpis příkazu říká, že čas byl úspěšně synchronizován.

Na obrázku 7.25 je vidět časový rozdíl mezi místním počítačem a časovým serverem. Výpis poskytuje jak číselné statistiky rozdílu času lokálního počítače, tak i vizualizaci ve formě hvězdičky v baru, která se posouvá podle velikosti rozdílu času mezi serverem a klientem. Na tomto obrázku je hvězda uprostřed, což značí minimální rozdíl mezi časem klienta a časem serveru. Konkrétní rozdíly jsou pak na obrázku vidět vlevo.

```
C:\Users\František Berg>w32tm /stripchart /computer:192.168.56.103
Tracking 192.168.56.103 [192.168.56.103:123].
The current time is 17.05.2023 12:37:26.
12:37:26, d:+00.0016386s o:-00.0608333s  [                       *                       ]
12:37:28, d:+00.0013720s o:-00.0614852s  [                       *                       ]
12:37:30, d:+00.0020945s o:-00.0616067s  [                       *                       ]
12:37:32, d:+00.0013728s o:-00.0624549s  [                       *                       ]
12:37:34, d:+00.0012051s o:-00.0630593s  [                       *                       ]
12:37:36, d:+00.0012199s o:-00.0636037s  [                       *                       ]
12:37:38, d:+00.0011344s o:-00.0641049s  [                       *                       ]
```

Obr. 7.25: Časové rozdíly mezi lokálním počítačem a časovým serverem.

Ve Wiresharku byla NTP komunikace z hostitelského počítače zachycena na Ubuntu NTP serveru, jak je vidět na obrázku 7.26.

7.3.3 Zabezpečení NTP serveru

Útok NPT flood, který se využívá k odstavení služby NTP využívá zneužití příkazu monlist, který zasílá seznam IP adres posledních 600 klientů, kteří jsou připojeni k serveru NTP. Dotaz monlist se tak využívá díky vlastnosti amplifikace, tedy odeslání malého množství dat k vyžádání značně většího množství dat ze strany serveru. Server ale nedokáže rozpoznat, jestli byl dotaz vyslán za účelem poškození, nebo se jedná o běžný provoz.

5	1.539192395	192.168.56.1	192.168.56.103	NTP	90	NTP Version 3, client
6	1.539396273	192.168.56.103	192.168.56.1	NTP	90	NTP Version 3, server
11	10.749756283	192.168.56.1	192.168.56.103	NTP	90	NTP Version 3, client
12	10.749950000	192.168.56.103	192.168.56.1	NTP	90	NTP Version 3, server

Obr. 7.26: Zachycení NTP komunikace přes Wireshark.

Útočník tak zašle dotaz monlist s falešnou zdrojovou adresou a server zašle odpověď, která obsahuje několikanásobně vyšší objem dat, na oběť, které nezbývá dostatek šířky pásma pro legitimní komunikaci.

Pokud je server zranitelný lze zjistit pomocí zaslání dotazu monlist. Pokud server odpoví, je zranitelný na NTP flood. Otestování, zdali je server náchylný k amplifikaci lze zjistit pomocí příkazu `ntpdc -n -c monlist <IP> | head`. Pokud server odpoví, je žádané přidat do konfiguračního souboru pro *ntp* `/etc/ntp.conf` direktivu `noquery` na řádek začínající `restric default`.

Nejjednodušší ochranou proti zneužití příkazu monlist je aktualizace NTP verze v Ubuntu na verzi 4.2.7p26. V této verzi je dotaz `MON_GETLIST_1` odstraněn.

Služby NTP využívána na Ubuntu NTP serveru má verzi 4.2.8p12, jak je vidět na obrázku 7.27, takže je služba zabezpečená proti útoku NTP flood.

```
||/ Name           Version
+++-==============-==================================
ii  ntp            1:4.2.8p12+dfsg-3ubuntu4.20.04.1
```

Obr. 7.27: Verze používaného balíčku NTP.

Závěr

V této bakalářské práci bylo provedeno seznámení se základními internetovými službami jako DNS, DNSSEC, DHCP a NTP. Bylo zde popsáno jejich fungování a princip, jak dané služby pracují a jaký je jejich daný účel. Dále se bakalářská práce zabývá modulem Server Emulator ICT testeru, který je vyvíjen na Ústavu telekomunikací na Vysokém učení technickém v Brně. Po provedení analýzy serverových služeb byla do modulu Server Emulator vybrána služba DNS, DHCP a NTP. Služby, které byly vybrány pro implementaci běží na Linuxové distribuci Ubuntu.

V praktické části bakalářské práce bylo pro implementaci základních funkcí a možnosti nastavení rozhraní, IP adresy 4 a 6, a portu, navrhnuto a naprogramováno grafické rozhraní, pomocí kterého je možné služby ovládat. Chod služeb zajišťují balíčky *bind9* pro DNS server, *isc-dhcp-server* pro DHCP server a *ntp* pro NTP server.

Bylo také provedeno otestování fungování daných služeb v modulu Server Emulator přes nově vytvořené záložky daných služeb, ve kterých jsou prvky pro jejich ovládání. V praktické části bylo pro otestování služeb provedeno reálné odzkoušení, zdali jsou služby funkční, díky čemuž se potvrdilo správné konfigurační nastavení a chování služeb. Komunikace a odezvy daných služeb byly zachyceny přes program Wireshark a další programy, které slouží na zachytávání síťové komunikace nebo pro otestování impelemtovaných služeb, ve kterých byla zachycena komunikace služeb mezi virtuálním serverem Ubuntu a hostitelským počítačem Window.

Literatura

[1] *What is a Server?* [online]. Brien Posey, c1999-2022 [cit. 2022-12-08]. Dostupné z: https://www.techtarget.com/whatis/definition/server

[2] *Comparison of "peer-to-peer" vs "client-server" Network Models. NETWORKS TRAINING* [online]. Harris Andrea, c2022 [cit. 2022-12-08]. Dostupné z: https://www.networkstraining.com/peer-to-peer-vs-client-server-network

[3] *What Is Localhost? HOSTINGER TUTORIALS* [online]. Kaunas, Litva: Domantas G., 2022, 15. 8. 2022 [cit. 2022-12-08]. Dostupné z: https://www.hostinger.com/tutorials/what-is-localhost

[4] *What is service-oriented architecture (SOA)?* Rad Hat [online]. Raleigh, State of North Carolina, USA: Rad Hat, 2020 [cit. 2022-12-08]. Dostupné z: https://www.redhat.com/en/topics/cloud-native-apps/what-is-service-oriented-architecture

[5] *Komunikační technologie* Jeřábek, J. , verze 2022. Brno: Vysoké učení technické v Brně, 2014. s. 1-172, ISBN 978-80-214-4713-4. (CS)

[6] *TCP/IP - model, encapsulace, paket vs. rámec.* SAMURAJ-cz [online]. Hluboká nad Vltavou, Česko: Petr Bouška, 2007 [cit. 2022-12-08]. Dostupné z: https://www.samuraj-cz.com/clanek/tcpip-model-encapsulace-paketu-vs-ramec/

[7] *Internet a jeho služby: Multimediální podpora předmětu Internet a jeho služby.* IJS: Internet a jeho služby [online]. [cit. 2022-12-08]. Dostupné z: http://ijs.8u.cz/index.php/standardizace-v-pocitacovych-sitich/referencni-model-iso-osi

[8] *JAK NA INTERNET: Doména, IP adresa, DNS.* Nic.cz: Doména, IP adresa, DNS [online]. Prague, Czech Republic: nic.cz, 2012, 2012 [cit. 2022-12-08]. Dostupné z: https://www.jaknainternet.cz/page/1261/domena,-ip-adresa,-dns/

[9] *O DNSSEC: PROČ POTŘEBUJETE DNSSEC?.* Nic.cz: PROČ POTŘEBUJETE DNSSEC? [online]. Prague, Czech Republic: nic.cz, 2022, 2022 [cit. 2022-12-08]. Dostupné z: https://www.nic.cz/page/513/about-dnssec/

[10] *JAK FUNGUJE DNSSEC: JAK DNSSEC FUNGUJE?.* Nic.cz: JAK DNSSEC FUNGUJE? [online]. Prague, Czech Republic: nic.cz, 2022, 2022 [cit. 2022-12-08]. Dostupné z: https://www.nic.cz/page/444/jak-funguje-dnssec/

[11] DROMS, R., LEMON, T. DHCP – *Příručka administrátora.* 1. vydání. Brno: Computer Press, a. s., 2004. 490 s. ISBN 80-251-0130-4.

[12] *NTP: Filozofie synchronizace času po Internetu.* LUPA.CZ [online]. Praha: Karel Chvalovský, 2003, 2003 [cit. 2022-12-08]. Dostupné z: https://www.lupa.cz/clanky/ntp-filozofie-synchronizace-casu-po-internetu/

[13] *What is JMeter? Introduction Uses: How does JMeter work?.* GURU99.com: How does JMeter work? [online]. Thomas Hamilton, 2022, 2003 [cit. 2022-12-08]. Dostupné z: https://www.lupa.cz/clanky/ntp-filozofie-synchronizace-casu-po-internetu/

[14] S. Farraposo, L. Gallon, P. Owezarski *Network Security and DoS Attacks.* [online]. April 2005 [cit. 08.12.2022]. Dostupné z: <https://homepages.laas.fr/owe/METROSEC/Security_and_DoS.pdf>

[15] CERT Division, *Denial of Service Attacks.* [online]. 1997 [cit.08.12.2022]. Dostupné z <https://resources.sei.cmu.edu/library/assetview.cfm?assetID=496599>

[16] KENIG, R., et al. DDoS Survival Handbook. *Radware*[online]. 2013, [cit. 08.12.2022]. Dostupné z URL: <https://security.radware.com/ddos-knowledge-center/ddospedia/dns-amplification-attack/>.

[17] Ping Flood | ICMP Flood, *Incapsula* [online]. 2017, [cit. 08.12.2022]. Dostupné z URL: <https://www.incapsula.com/ddos/attack-glossary/ping-icmp-flood.html>.

[18] NTP Amplification, *Incapsula* [online]. 2017, [cit. 08.12.2022]. Dostupné z URL <https://www.incapsula.com/ddos/attack-glossary/ntp-amplification.html>.

[19] DNS flood DDoS attack. *(Cloudflare)* [online]. [cit. 08.12.2022]. Dostupné z <https://www.cloudflare.com/learning/ddos/dns-flood-ddos-attack//>.

[20] MÍŠANÝ, Daniel. *DETEKCE ÚTOKU DNS AMPLIFICATION Z PASIVNÍ ANALÝZY DNS PROVOZU:* DNS AMPLIFICATION ATTACK DETECTION USING PASSIVE DNS ANALYSIS. Brno, 2014. Bakalářská práce. VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ, FAKULTA INFORMAČNÍCH TECHNOLOGIÍ, ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ. Vedoucí práce Ing. Michal Kováčik.

[21] ICT tester - Uživatelský návod k modulu Server Emulator. VUT v Brně, Ústav telekomunikací, 2019.

[22] *Apache JMeter™* [online]. THE APACHE SOFTWARE FOUNSDATION, c1999 – 2022 [cit. 2022-12-08]. Dostupné z: https://jmeter.apache.org/

[23] *BIND DNS: Pros, Cons and Alternatives:* What is BIND?. Ns1.com [online]. ns1.cz, c2021 [cit. 2022-12-08]. Dostupné z: https://ns1.com/resources/bind-dns-pros-cons-and-alternatives

[24] *Ubuntu: Dynamic Host Configuration Protocol (DHCP).* Https://ubuntu.com/server/docs/network-dhcp: Dynamic Host Configuration Protocol (DHCP) [online]. Great Britain: ubuntu.com, c2022 [cit. 2022-12-08].

Seznam symbolů a zkratek

TCP/IP	Transmission Control Protocol/Internet Protocol
UDP	User Datagram Protocol
ISO/OSI	International Organization for Standardization/Open System Interconnection
P2P	Peer-to-peer
F2F	Friend-to-friend
PDU	Protocol data unit
MTU	Maximum transmission unit
DNS	Domain Name System
DNSSEC	Domain Name System Security Extensions
DHCP	Dynamic Host Configuration Protocol
NTP	Network Time Protocol
DoS	Denial-of-Service
DDoS	Distributed Denial-of-Service
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
ICT	Information and Communication Technologies
BIND	Berkeley Internet Name Domain
TTL	Time to live
DIG	Domain Information Groper
RRL	Response Rate Limiting

A Obsah elektronické přílohy

Obsahem přiloženého souboru ServerEmulator je soubor ServerEmulator.jar, společně se zdrojovým kódem modulu Server Emulator, který je rozšířený o dané serverové služby.

Modul ServerEmulator.jar je určen k vložení do programu Apache JMeter a rozšíření jej o možnosti vytvoření daných serverových služeb, které jsou poskytnuty modulem Server Emulator.

Soubor Server Emulator obsahuje mimo jiné také soubory, ve kterých je kód grafického rozhraní a funkcí modulu pro ovládání serverových služeb v progamovacím jazyku Java.

```
/ ..... kořenový adresář přiloženého archivu
└── src
    └── main ..... adresář obsahující Java kód aplikace
        └── java
            └── eu
                └── gity
                    └── jmeter
                        └── serveremulator
                            ├── properties
                            │   └── strings
                            │       └── ServerEmulatorPropertiesStrings.java
                            ├── ServerEmulator.java
                            ├── ServerEmulatorCore.java
                            ├── ServerEmulatorGui.java
                            ├── ServerEmulatorInputVerification.java
                            └── ServerEmulatorTreeGuard.java
```