FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Komunikace skrze Captive portal
Student:	Bc. Martin Černáč
Vedoucí:	Ing. Aleš Padrta, Ph. D.
Studijní program:	Informatika
Studijní obor:	Počítačové systémy a sítě
Katedra:	Katedra počítačových systémů
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

1. Seznamte se s problematikou Captive portals a způsoby jejich obcházení.
2. Navrhněte protokol umožňující obejití Captive portals s důrazem na co nejvyšší propustnost.
3. Navržený protokol implementujte.
4. Výsledky vyhodnoťte a porovnejte s dostupnými řešeními.

Seznam odborné literatury

Dodá vedoucí práce.

prof. Ing. Róbert Lórencz, CSc.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 10. listopadu 2017

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Komunikace skrze Captive portal

Bc. Martin Černáč

Katedra počítačových systémů
Vedoucí práce: Ing. Aleš Padrta, Ph. D.

30. dubna 2018

Poděkování

Rád bych poděkoval svému vedoucímu za cenné rady, věcné připomínky a vstřícnost při konzultacích.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 30. dubna 2018 .....

Odkaz na tuto práci

Černáč, Martin. *Komunikace skrze Captive portal.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018. Dostupný také z WWW: ⟨`https://github.com/octaroot/CTU-FIT-MasterThesis`⟩.

Abstrakt

TODO V několika větách shrňte obsah a přínos této práce v češtině. Po přečtení abstraktu by měl mít čtenář dost informací pro rozhodnutí, zda chce Vaši práci číst.

Klíčová slova Závěrečná práce, LaTeX.

Abstract

TODO Sem doplňte ekvivalent abstraktu Vaší práce v angličtině.

Keywords Thesis, LaTeX.

Obsah

Seznam obrázků

Úvod

Bezdrátové sítě se staly zcela běžným prostředkem mezilidské komunikace. Uživatelé bezdrátové sítě mají možnost si navzájem vyměňovat informace a nebýt přitom omezeni kabelovým spojením. Velkým přínosem bezdrátové sítě je tedy zvýšená mobilita uživatelů. Ta vedla k vlně popularity bezdrátových sítí počínaje mobilními telefony, využívajících bezdrátovou síť `GSM`, až po dnešní chytré spotřebiče a jejich zapojení do *Internet of Things*.

S rostoucími nároky uživatelů prošly rozsáhlým vývojem i bezdrátové sítě (vyšší prostupnost, nižší latence a další aspekty). Mezi dlouhodobě populární a velmi rozšířené typy bezdrátových sítí se řadí technologie `Wi-Fi`. Jedná se o technologii podporovanou širokým spektrem spotřební elektroniky (například televizory, tiskárny, mobilní telefony nebo počítače). Technologie `Wi-Fi` využívá bezlicenčním pásmo `ISM` a díky tomu je provozování vlastní `Wi-Fi` sítě legislativně nenáročné. Na trhu je navíc dostupná celá řada produktů, zajišťující provoz `Wi-Fi` sítě.

Z těchto důvodů došlo k velkému rozmachu takzvaných *hotspotů*, tedy veřejně přístupných míst s pokrytím `Wi-Fi` sítě. Taková `Wi-Fi` síť je zpravidla veřejně přístupná a uživatelům nabízí přístup do sítě Internet. Ačkoliv je velice snadné začít s provozem *hotspotu*, je nutné dbát na další aspekty provozu takové služby – zejména právní aspekty.

Uživatelé *hotspotu* by měli být srozuměni s pravidly používání konkrétní sítě, limitovanou odpovědností provozovatele a před začátkem užívání sítě doložit svůj souhlas s pravidly. Provozovatel navíc může mít zájem o některé identifikující informace o uživatelích *hotspotu*.

Technologie `Wi-Fi` však sama o sobě neumožňuje nic z výše uvedeného. Takovou situaci lze vyřešit například zapojením recepce v prostředí hotelu (uživatel písemně vyjádří souhlas s pravidly používání sítě, recepční vydá přístupové údaje do sítě). Častěji se však setkáváme s automatizovaným přístupem, realizovaným pomocí *captive portálu* (z angličtiny *Captive portal*) – a to jak na návštěvnických sítích drátových, tak i bezdrátových.

Řešení s pomocí *captive portálu* spočívá v detekci nově připojených uživa-

telů, které je nutné informovat o pravidlech provozu sítě. Po udělení souhlasu s pravidly je uživateli poskytnut přístup do Internetu a všechny následné interakce uživatele se sítí *captive portál* ignoruje (nezasahuje do nich).

Z principu věci tedy *captive portál* musí být schopen **nejprve zasahovat do veškerého síťového provozu** (uživatel doposud nedal souhlas s pravidly, neměl by mít možnost síť využívat) a **následně do provozu konkrétního uživatele nezasahovat vůbec**. Existuje celá řada technologických postupů pro docílení popsaného efektu. Mnohé z nich jsou však neefektivní a nepočítají s „neposlušným“ uživatelem, který se bude snažit omezující techniky překonat.

Právě proto jsem se rozhodl vypracovat diplomovou práci na téma obcházení *captive portálu*, zdůrazňující jejich technologickou nedokonalost a poukázat na lepší řešení řízení síťového přístupu (*Network Access Control*).

V této práci se proto budu zabývat popisem problematiky *captive portálů* a obecnými způsoby jejich obcházení. Jako demonstraci technologické nedokonalosti užití *captive portálu* pro zajištění řízení síťového přístupu rovněž navrhnu a implementuji protokol s důrazem na maximální prostupnost. Implementovaný protokol otestuji a provedu srovnání s dostupnými nástroji pro obcházení *captive portálů*.

Kapitola 1

Analýza současné situace

Tato kapitola se věnuje problematice *captive portálů*, motivací jejich nasazení v síti a častými problémy s používáním *captive portálu* jako nástroje pro zajištění řízení síťového přístupu.

1.1 Captive portál

Captive portál[1] představuje webovou aplikaci, často nasazovanou na veřejně přístupných sítích. Aplikace má za úkol informovat nově připojené klienty o podmínkách užití sítě a požadovat uživatelův souhlas s jejich dodržováním. Až do momentu souhlasu s podmínkami užití sítě je uživateli odepřen přístup do zbytku sítě. Z toho plyne první část názvu ***Captive*** *portál* – uživatel je „zajatý“, „uvězněný“ (v angličtině *captive*).

Pojem *captive portál* nemá v češtině ustálený překlad, v jiných akademických publikacích [2] [3] autoři používají počeštěný termín *captive portál* a proto je tento termín použit i v této práci.

1.1.1 Motivace nasazení

Captive portál je do provozu sítě často nasazován jako nástroj pro zajištění řízení síťového přístupu. Přístup do sítě je umožněn pouze klientům, kteří splní podmínky přístupu do sítě. Takovou podmínkou může být pouhé vyjádření souhlasu s používáním konkrétní sítě, ale může se jednat i o podmínku složitější, například:

- shlédnutí reklamního spotu dle výběru provozovatele
- uhrazení poplatku pro přístup do sítě
- poskytnutí některých osobních údajů a souhlasu s jejich zpracováním
- doložení oprávnění pro přístup do sítě (kód z účtenky, číslo hotelového pokoje, ...)

- zviditelnění provozovatele pomocí sociálních médií (například Facebook *check-in*)

Jak plyne z výše uvedeného výčtu, vyjma právních aspektů může být *captive portál* použit i pro shromažďování údajů o uživatelích sítě. Jedním z nástrojů pro takovou činnost je nabízení „přihlášení se“ do *captive portálu* pomocí účtu na některé ze sociálních sítí. Pokud uživatel takovou možnost využije, *captive portál* si od sociální sítě vyžádá informace o uživateli, jako například jméno, fotografii, pohlaví nebo datum narození. Po shromažďování takových informací je uživateli poskytnut přístup do zbytku sítě. Provozovatel tedy může uživatele například identifikovat nebo detekovat opakované návštěvy *hotspotu*. Na oplátku je uživateli „zdarma“ poskytnut přístup do sítě Internet.

Pro usnadnění nasazení takového řešení nabízí společnost Facebook službu *Facebook Wi-Fi*[4], cílenou na majitele obchodů. Jedná se o řešení na bázi *captive portálu*, které vyžaduje aby nově připojený uživatel měl konto na sociální síti Facebook. Po připojení na *hotspot* je uživatel vyzván ke sdílení informace o jeho návštěvě obchodu, jehož *hotspot* právě používá (jako protislužbu za poskytnutý přístup do Internetu).

Poněkud méně invazivní motivací pro zavedení *captive portálu* je monetizace *hotspotu*. Například prodejem reklamního místa – uživatel po připojení do sítě musí shlédnou reklamní spot, nebo vyplnit krátkou anketu. Provozovatel *hotspotu* získá z takové aktivity finanční odměnu a uživateli je odměněn přístupem do sítě Internet.

Některé *captive portály* alternativně umožňují uživateli doložit nárok na přístup do sítě. Například jednorázový kód z účtenky, čímž dokládá útratu v podniku, který *hotspot* provozuje. Nebo číslo hotelového pokoje, čímž dokládá svůj pobyt v hotelu, který zahrnuje (jinak zpoplatněný) přístup do sítě Internet.

1.1.2 Technologické pozadí

Úkolem *captive portálu* je detekovat nově připojené uživatele sítě, omezit jim přístup do sítě a nasměrovat je na webovou aplikaci captive portálu. Po splnění podmínek pro plnohodnotný přístup uživatele do zbytku sítě nesmí *captive portál* do komunikace dále zasahovat (tj. musí *detekovat*, že síťový provoz patří oprávněnému uživateli).

Ačkoliv se jedná o přímočarý cíl, je možné ho dosáhnout s pomocí celé řady technologií a postupů. Proto se v praxi setkáváme s velmi velkým počtem různorodých implementací *captive portálu*. Některé z nich jsou dostupné pod svobodnou licencí, jiné jsou součástí placeného produktu a v neposlední řadě existují řešení *na míru* – a to nejen *na míru* provozovateli, ale rovněž *na míru* konkrétnímu zařízení/hardware. V této práci jsou blíže zkoumány zejména

softwarová řešení, která jsou snadno dostupná (zveřejněná pod svobodnou licencí).

Přestože efektu *captive portálu* lze s velkou úspěšností docílit pouhým odkloněním HTTP provozu, existují mnohem sofistikovanější varianty, využívající například oddělené VLAN sítě. Obecně však platí, že *captive portál* při své práci může vycházet pouze z informací, které putují po síti. Detekce nově připojených uživatelů a identifikace oprávněných uživatelů je tedy zpravidla založena dvojici identifikátorů:

- globálně unikátní `MAC` adresa zařízení
- přidělená `IP` adresa zařízení

Captive portál lokálně ukládá informace o autorizovaných uživatelských zařízeních v síti (zaznamenává jejich `MAC` a `IP` adresy). Síťový provoz takových zařízení není narušován. Pokud však uživatel využívá zařízení, které *captive portál* na svém seznamu nenalezne, *captive portál* síťový provoz buď zahodí, nebo zmanipuluje takovým způsobem, aby se uživatel dostal na webovou aplikaci *captive portálu* a mohl se identifikovat. Záznamy na seznamu autorizovaných uživatelů sítě zpravidla podléhají periodickému mazání neaktivních uživatelů – uživatel je tedy nucen se po delší době nečinnosti opakovaně identifikovat *captive portálu*.

Alternativně k periodickému promazávání seznamu autorizovaných klientů může *captive portál* vyžadovat, aby uživatel po celou dobu používání sítě měl v prohlížeči otevřené speciální okno, jehož přítomnost instruuje *captive portál* k přidělení plnohodnotného síťového přístupu.

Ve chvíli, kdy je *captive portál* schopen rozeznat autorizované a neautorizované uživatele, musí rovněž mít možnost neautorizované uživatele nasměrovat na webovou aplikaci *captive portálu*. Takový cíl *captive portál* často naplňuje prováděním `MITM` útoku na nově připojené uživatele. Například při přístupu neautorizovaného uživatele na libovolnou webovou stránku protokolem `HTTP` je jeho provoz odkloněn a vrácena odpověď od *captive portálu*, která prohlížeč uživatele nasměruje na webovou aplikaci *captive portálu*. Kromě této techniky uvádím v následující části textu i několik dalších.

1.1.2.1 `ICMP` host redirect

Protokol `ICMP` specifikuje zprávy, které může směrovač poslat koncové stanici, pokud detekuje, že stanice v rámce své komunikace používá neoptimální síťovou cestu. Je zcela v režii cílové stanice, zda-li si nechá o svém směrování radit od ostatních zařízení v síti. Tato metoda spoléhá na situaci, kdy koncová stanice skutečně upraví svou směrovací tabulku a zanese do ní informace z `ICMP` *host redirect* zprávy. Právě s tímto úmyslem odesílá *captive portál* `ICMP` *host redirect* zprávu, když detekuje pokus o spojení uživatele se serverem v Internetu. `ICMP` zpráva se pokusí cílovou stanici uživatele přesvědčit, že ideální

cesta vede skrze server provozující *captive portál*. Koncová stanice upraví své směrování a začne komunikovat se svým protějškem skrze *captive portál*, který díky tomu může komunikaci manipulovat za účelem nasměrování uživatele na webovou aplikaci *captive portálu*.

1.1.2.2 HTTP 3xx redirect

Při pokusu o přístup na webovou stránku `www.example.com` je požadavek klienta odkloněn a odpověď na požadavek zaslána přímo z *captive portálu*. V odpovědi je zpravidla využita `HTTP` hlavička `302 Found`, která prohlížeč klienta nasměruje na webovou aplikace *captive portálu*, viz Ukázka 1.1.

```
> GET / HTTP/1.1
> Host: www.example.com
>
< HTTP/1.1 302 Found
< Location: http://192.168.1.1/captive/
```

Ukázka 1.1: Ukázka přesměrování HTTP požadavku (zkráceno)

1.1.2.3 Podvržení DNS odpovědi

Captive portál monitoruje `DNS` dotazy klientů. Pokud `DNS` požadavek patří neautorizovanému klientovi, *captive portál* mu nazpět zašle odpověď s `IP` adresou webové aplikace *captive portálu* bez ohledu na dotazované doménové jméno. Jedná se o značně nebezpečnou techniku, protože může snadno dojít k otrávení `DNS` cache klienta. Pro minimalizaci takového vedlejšího efektu bývá v podvržené DNS odpovědi nastavena nulová životnost (hodnota `TTL = 0`). Takové nastavení by mělo zajistit, že podvržená odpověď nebude zanesena do lokální `DNS` cache. Ukázka 1.2 zachycuje evidentní podvržení `IP` adresy serveru `google.com`.

```
$ nslookup google.com
Server:         192.168.1.1
Address:        192.168.1.1#53

Non-authoritative answer:
Name:   google.com
Address: 192.168.1.1
```

Ukázka 1.2: Ukázka podvržení DNS odpovědi

1.1.3 Technické problémy

Největším problémem *captive portálů* je závislost na technologii `WWW`. Cílení na tuto technologii pramení ze značně rozmanitého pojetí Internetu napříč jeho uživateli. Pro mnohé uživatele je totiž tvrzení „Nefunguje Internet“ synonymem pro „V prohlížeči se nepodařilo načíst mou domovskou stránku“. Díky tomu lze mnohé uživatele přesvědčit k provedení úkonů, které *captive portál* vyžaduje. Uživatel úkony provede, protože mu „nefunguje Internet“ a *captive portál* slibuje nápravu situace.

Z předcházejících tvrzení však plyne fakt, že *captive portál* je **závislý** na `WWW` a tím pádem **závislý na webovém prohlížeči**. V historii se ukázalo, že to představuje velký problém pro zařízení s podporou `Wi-Fi`, ale bez webového prohlížeče (nebo s velmi omezeným webovým prohlížečem). Demonstrovat takovou situaci lze na populárním[1] mobilním herním zařízení *Nintendo DS*. Tento problém v současnosti řeší protokol *WISPr*[6], který usnadňuje (v některých případech zcela eliminuje) nutnou interakci uživatele s webovou aplikací *captive portálu*.

S rostoucím rozmachem `HTTPS` na úkor nešifrovaného `HTTP` mají *captive portály* obtížnější práci s nasměrováním uživatele na webovou aplikaci *captive portálu*. *Captive portály* využívající podvržené certifikáty se budou muset od dubna 2018 vyrovnat s ještě větším stupněm nedůvěryhodnosti, díky zavedení nutnosti *Certificate Transparency* v prohlížeči Google Chrome[7]. *Captive portál* by se neměl snažit manipulovat s šifrovaným spojením, namísto snahy o modifikaci a *rozbití šifrování* by takový provoz měl být zahazován. Takový postup však nesdílejí všechny implementace *captive portálu*, jak je dále popsáno v podkapitole Netechnické problémy 1.1.4.

Návrhovým problémem mnoha *captive portálů* je snaha manipulovat s obsahem komunikace uživatelů sítě. V mnohých případech je manipulace dosaženo pomocí `MITM` útoku. Síť, která zcela úmyslně provádí útoky na své uživatele (ať už s jakýmkoliv účelem) pochopitelně nemůže získat jakoukoliv důvěru uživatelů. **Síť s nulovou důvěrou by uživatelé neměli vůbec využívat**.

Mnohé softwarové produkty dokáží detekovat omezený síťový provoz – například operační systém Microsoft Windows, nebo webové prohlížeče Firefox a Chrome. Nadměrná manipulace se síťovým provozem neautorizovaných uživatelů však může tuto funkcionalitu potlačit, což je pro uživatele nežádoucí.

Jak bylo uvedeno v podkapitole Realizační technologie 1.1.2, *captive portál* při své práci vychází z dat, která putují po síti. Do veřejné sítě *hotspotu* je však jednoduché získat přístup. Útočník na zmíněné síti může naslouchat a například pomocí naklonování `MAC` a `IP` adres se následně vydávat za jiné účastníky sítě, čímž se neautorizovaný útočník jeví *captive portálu* jako autorizovaný uživatel.

[1] prodáno přes 150 milionů kusů[5]

1.1.4 Netechnické problémy

V některých případech se *captive portály* chovají velmi invazivně. Na začátku roku 2015 společnost Gogo (poskytovatel připojení na palubách letadel) ve své síti začala využívat falešné certifikáty pro produkty firmy Google. Na situaci upozornila na svém Twitteru[8] Adrienne Porter Felt, zaměstnankyně firmy Google. Certifikáty byly vystaveny pro doménová jména `*.google.com`, tedy všechny domény třetího řádu domény `google.com`.

Mnoho uživatelů Internetu má ve svých prohlížečích nastavenou domovskou stránku na `www.google.com`. Po připojení se na palubní `Wi-Fi` síť v letadle a zapnutí prohlížeče byl uživatel okamžitě varován před nedůvěryhodným certifikátem. Vzhledem k tomu, že uživatel sám žádnou stránku nenavštívil (prohlížeč pouze načetl domovskou stránku), je pro uživatele snadné propadnout dojmu, že chyba není způsobena jeho počínáním a proto bude varování ignorovat.

Takové počínání samozřejmě není správné a poučená osoba by se ho měla vyvarovat. Zdaleka ne všechny uživatele Internetu však lze označit jako *poučené* uživatele. Takoví uživatelé nedisponují dostatečnými znalostmi pro porozumění problému, před kterým je prohlížeč varuje a varování budou ignorovat. **Vytvářet u uživatelů návyky „všechno potvrď a pak se dostaneš na Internet“ je neetické** a nemělo by k tomu docházet.

V případě *captive portálu*, který vyžaduje poskytnutí osobních informací by jejich počet měl být minimální a nakládání s nimi obezřetné. Uživatelé *hotspotu* zpravidla nemají zájem o *newsletter* provozovatele, ani si nepřejí být provozovatelem statisticky zkoumáni. Provozovatel si na takové akce samozřejmě vyhradí nárok v pravidlech používání sítě, které však (zpravidla na mobilních zařízeních) přečte jen malý zlomek uživatelů.

1.1.5 Alternativy captive portálů

Motivací *captive portálu* je řízení síťového přístupu. Takovou funkci však mnohem lépe[2] plní dedikované protokoly a softwarová řešení. Pro řízení přístupu na `Wi-Fi` hotspot lze například použít populární bezpečnostní protokol `WPA2`. Nikoliv však v módu *WPA-Personal*[2], nýbrž v režimu *WPA-Enterprise*. Tento režim vyžaduje, aby se uživatel identifikoval ještě **před** faktickým připojením do sítě – typicky pomocí uživatelského jména a hesla[3]. K ověření údajů tedy není zapotřebí webový prohlížeč, ale klientské zařízení musí podporovat *WPA-Enterprise* režim – nutná podpora pro *IEEE 802.1X* protokol. Příkladem takové sítě je celosvětová síťová infrastruktura *eduroam*, která pro autentizaci využívá protokol *IEEE 802.1X* a hierarchickou strukturu RADIUS serverů. Nasazení *WPA-Enterprise* je však z důvodu nutnosti provozu RADIS serveru náročnější, než *WPA-Personal*. I přesto se však jedná o technicky vhodnější

[2] Často označován jako *WPA-PSK*

[3] Protokol *IEEE 802.1X* podporuje i ověření pomocí certifikátu nebo tokenu

alternativu *captive portálu*, pokud je možné provozovat *hotspot* v režimu *WPA-Enterprise*.

1.2 Metody pro obcházení captive portálů

Captive portál s uživateli komunikuje pomocí `WWW`. Aby bylo možné uživatele nasměrovat na webovou aplikaci *captive portálu*, musí být uživatel úspěšně připojen do sítě. Díky takovému „odložení“ autentizace bylo popsáno několik způsobů pro obcházení *captive portálů*. Všechny dále popisované způsoby jsou založeny na neúplné nebo dokonce záměrně „špatné“ konfiguraci *captive portálu*.

Konfigurace firewallu, která úmyslně nefiltruje některý síťový provoz nemusí být dílem nezkušeného administrátora (proto tento stav označuji jako **„špatnou“** konfiguraci). Může se zkrátka jednat o jediný způsob, jak splnit požadavky pro provoz sítě – například kvůli proprietárnímu software, který vyžaduje nerušenou komunikaci na některých portech. Z hlediska síťové architektury by bylo lepší provozovat veřejnou síť s *captive portálem* bez takových klientů, tj. **pouze** jako síť pro hosty, nicméně hardware podporující pokročilé techniky jako provoz více oddělených `Wi-Fi` sítí nebo podporu `VLAN` je zpravidla dražší a pro nezkušené správce obtížnější na správu.

1.2.1 DNS tunelování

Protokol `DNS` je jedním z nejstarších protokolů dnešního Internetu. Slouží primárně k překladu mezi doménovými jmény (například `fit.cvut.cz`) a IP adresami uzlů v síti (například `147.32.232.248`). Častou nedokonalostí *captive portálů* je směrování `DNS` požadavků do Internetu. Pokud k takovému chování dochází i u neautentizovaných uživatelů, lze protokol `DNS` využít ke komunikaci se serverem v Internetu a tím pádem k obejití *captive portálu*.

1.2.2 ICMP tunelování

Protokol `ICMP` je rovněž velmi důležitým síťovým protokolem. Je využíván zpravidla k přenosu služebních informací jako například nedostupnost služby nebo nedosažitelnost uzlu v síti. I přesto, že není v praxi využíván aplikacemi pro přenos informací, lze ho k tomuto účelu využít. Vhodným využitím zpráv `Echo Request` a `Echo Reply` lze mezi dvěma síťovými uzly přenášet libovolná data. Protokol *ICMP* spadá do stejné *rodiny* protokolů jako `TCP` a `UDP`, ale nevyužívá ani jeden z nich. Právě proto bývá v konfiguraci firewallu často opomíjen. Pokud taková situace nastane, lze protokol `ICMP` využít ke komunikaci se serverem v Internetu a tím pádem k obejití *captive portálu*.

Tunelování pomocí `ICMP` je technicky možné díky RFC 792[9], kde je u typů zpráv `0` a `8` (*echo reply*, resp. *echo message*) specifikována proměnlivá délka zpráv.

1.2.3 Využití nefiltrovaných portů

Jak bylo uvedeno na začátku podkapitoly 1.2 *Metody pro obcházení captive portálů*, v konfiguraci firewallu se mohou z různých důvodů vyskytovat výjimky, které lze zneužít k tunelování provozu bez nutnosti maskovat komunikaci jako `DNS` nebo `ICMP` provoz. Zpravidla[10] se jedná o porty

- TCP/22 – pro vzdálenou správu zařízení
- TCP/3128 – HTTP proxy servery (například za účelem cache obsahu)
- UDP/53 – `DNS`, diskutováno v podkapitole 1.2.1 DNS tunelování

Důvodem k udělení výjimky pro port TCP/22 bývá nutnost vzdálené správy některých zařízení pomocí protokolu `SSH`. Samotný protokol `SSH` lze využít pro tunelování, *port forwarding* nebo přímo jako `SOCKS` proxy. Klient `OpenSSH` tyto operace umožňuje provést velmi snadno, například lokální `SOCKS` proxy na portu 8080 lze spustit příkazem `ssh -D 8080 uzivatel@domaci-server`.

TCP port 3128 bývá na firemních sítích využíván jako cache proxy pro často navštěvované webové stránky, aby se šetřilo síťovým provozem. Neautentizovaný klient se může pokusit takového proxy serveru využít pro obejití omezení *captive portálu* a úspěšně komunikovat se serverem v Internetu.

Tyto praktiky jsou však méně časté než dříve zmíněné `ICMP` a zejména `DNS` tunelování, zkrátka proto že `SSH` ani kešující proxy server nejsou na rozdíl od služby `DNS` pro provoz Internetu klíčové.

1.3 Existující software pro obcházení captive portálů

Idea tunelování síťového provozu pomocí protokolu `DNS` není nová. Už na přelomu tisíciletí[4] se objevil nástroj *NSTX* s podtitulkem *tunneling network-packets over DNS*. Od té doby byla zveřejněná řada nástrojů založených na stejných principech a se stejným cílem. Mezi populární[11] nástroje se řadí například *iodine*, *OzymanDNS* a *DNSCat*. Různé nástroje nabízejí různé funkce, podporují rozdílné platformy a liší se v konkrétních detailech `DNS` komunikace (autentizace, šifrování, užité typy `DNS` zpráv, ...). Mnohé aplikace jsou v současnosti funkční, ale dále nevyvíjené ve prospěch jiných nástrojů (například domovská stránka *NSTX* odkazuje zájemce na stránky *iodine*). Podobná je i situace s nástroji pro tunelování pomocí `ICMP`.

Tunelování síťového provozu pomocí `DNS` je populární[11] i mezi tvůrci škodlivého software (*malware*), kteří se tak snaží vyhnout detekčním nástrojům. Paradoxně tunelování pomocí `DNS` lze zpravidla úspěšně detekovat[12].

[4] soudě dle data první veřejné verzovacího systému nástroje *NSTX*

Kapitola 2

Návrh a implementace

Stěžejním cílem této diplomové práce je vytvoření protokolu pro obejití Captive portálů s důrazem na co největší prostupnost. Tato kapitola shrnuje návrh takového protokolu a následnou implementaci.

2.1 Dosažení maximální prostupnosti

Navržený protokol by při obcházení omezení *captive portálu* měl upřednostňovat síťovou propustnost. Za tímto účelem bude vyvinutý software mít k dispozici více možných způsobů obejití *captive portálu* a na základě naměřených dat se bude schopen rozhodnout, který způsob tunelování je nejefektivnější, případně jaká kombinace více tunelů poskytuje nejlepší výsledky.

2.2 Možné technické prostředky

Integraci síťového tunelu do operačního systému lze řešit řadou způsobů v závislosti na operačním systému, jeho verzi a v závislosti na požadavcích pro přenositelnost. Snahu o nalezení ideálního řešení projevila celá řada softwarových projektů, jako například:

- *Tor*, známý *open-source* software pro stejnojmenou síť, která umožňuje anonymizaci uživatelů při pohybu na Internetu,
- *OpenVPN*, populární *open-source* software zejména pro vytváření šifrovaných tunelů (VPN).

Každé z výše uvedených softwarových řešení přistupuje k problému jinou cestou. Software anonymizační sítě *Tor* na klientské stanici vytváří `SOCKS` proxy a umožňuje jiným programům komunikovat skrze tuto proxy službu. Jedná se o velmi dobře přenositelné řešení, protože nespoléhá na specifickou podporu operačního systému. Nevýhodou tohoto řešení je přenesení problému

kompatibility z operačního systému na jednotlivé aplikace. Pokud aplikace nepodporuje, nebo uživateli nedovolí nastavit komunikace skrze `SOCKS` proxy, nebude schopna anonymizační sítě *Tor* využít.

Oproti tomu technologie *OpenVPN* je silně vázána na podporu ze strany operačního systému. Na klientské stanici vytváří virtuální síťové rozhraní, které se uživateli jeví jako jakékoliv jiné síťové rozhraní. Lze upravit systémovou směrovací tabulku, aby preferovala virtuální síťové rozhraní a komunikující aplikace tudíž nemusí podporovat komunikaci skrze `SOCKS` proxy. Nevýhodou je závislost na podpoře virtuálních síťových rozhraní v operačním systému.

2.3 Vybrané technické prostředky

Pro tuto práci byla zvolena implementace pomocí virtuálního síťového rozhraní, neboť toto řešení lze považovat za obecnější, neboť není nutná přímá podpora koncových aplikací. Podpora virtuálních rozhraní je zahrnuta v Linuxovém jádře od verze 2.2 (vydáno v roce 1999), *FreeBSD* 3.0 a *Solaris* 2.6. Z ostatních rodin operačních systémů je částečná podpora zahrnuta rovněž v *macOS*, *iOS* a *Andorid*. Pro platformu *Microsoft Windows* existují doplňky třetích stran, které přináší podporu virtuálních rozhraní.

Pro implementaci tunelu byl zvolen typ rozhraní *TUN*, které simuluje síťové rozhraní na třetí vrstvě ISO/OSI modelu a síťová data předává danému uživatelskému programu. Vzhledem k velmi silné vazbě linuxového jádra na jazyk *C* je software pro diplomovou práci vytvořen rovněž v jazyce *C*. Velká část softwarového řešení provádí nízkoúrovňové operace, pro které mají jiné jazyky omezenou podporu. Jedná se zejména o pokročilou práci s *file descriptory* virtuálních síťových rozhraní a pokročilou práci se síťovými sokety.

Virtuální síťové rozhraní navíc poskytuje uživateli volnost užití libovolného `IP` síťového rozsahu. Pokud by uživatel vyžadoval existenci *SOCKS* proxy, lze ji vytvořit po navázání tunelu například pomocí nástroje `SSH` jak bylo popsáno v podkapitole 1.2.3 *Využití nefiltrovaných portů*.

2.4 Struktura softwarového řešení

Vzhledem k důrazu na snadnou rozšiřitelnost je softwarové řešení navrženo do dvou částí:

- hlavní část programu, která vytváří a spravuje virtuální síťové rozhraní,
- samostatné pluginy, které řeší pouze omezenou funkcionalitu spojenou s tunelováním pomocí konkrétní technologie.

Hlavní část programu má rovněž na starost vytvoření prostředí pro jednotlivé pluginy. Každý plugin totiž pracuje ve svém vlastním vlákně a komunikuje s jednotným virtuálním rozhraním. Komunikace probíhá skrze funkce, které

mají standardem `POSIX` garantovánu bezpečnost při paralelním přístupu[5]. Hlavní část programu tedy při spuštění zpracuje uživatelské vstupy, vytvoří virtuální síťové rozhraní, spustí pluginy a na konci práce programu pluginy ukončí a rozhraní zruší. *File descriptor* síťového rozhraní je v programu sdílen (pouze pro čtení) všemi pluginy.

Jednotlivé pluginy jsou schopny ověřit možnost navázání spojení skrze *captive portál* a změřit datovou prostupnost tunelu. Každému dostupnému pluginu je věnována jedna z následujících podkapitol, vysvětlující vnitřní činnost pluginu.

2.5 Ověření nového klienta

Serverová část softwarového řešení před navázáním tunelového spojení vyžaduje autentizaci klienta. I přesto, že tato diplomová práce explicitně neřeší otázky soukromí a bezpečnosti Internetových služeb, považuji alespoň základní úroveň kontroly přístupu za adekvátní. Právě proto je jako součást praktické části diplomové práce implementováno ověřování klientů pomocí protokolu *výzva-odpověď* (známý také jako *Challenge-response protocol*).

Protokol *výzva-odpověď* přímo nespecifikuje konkrétní prostředky pro autorizaci – může se například jednat o triviální dvojici údajů jako přihlašovací jméno a heslo, nebo komplexnější metody využívající asymetrické kryptografie. Pro účely této práce je jako technický prostředek použito sdílené tajemství (tzv. `keyfile`), který může obsahovat jako značné množství (řádově tisíce) bajtů dat, tak i relativně krátké (řádově desítky) znaků. Je tudíž možné jako sdílené tajemství použít buď soubor s binárními daty, distribuován mezi server a klient alternativním kanálem, nebo pouze krátký textový soubor se zapamatovatelným heslem.

Na základě sdíleného tajemství vytvoří klient otisk (*hash*), který na *výzvu* serveru zašle jako *odpověď*. Server stejnou cestou vytvoří svůj otisk dat a výsledek porovná. Pokud jsou otisky stejné, patrně klient a server znají sdílené tajemství a klientovi je povoleno navázat tunelové spojení.

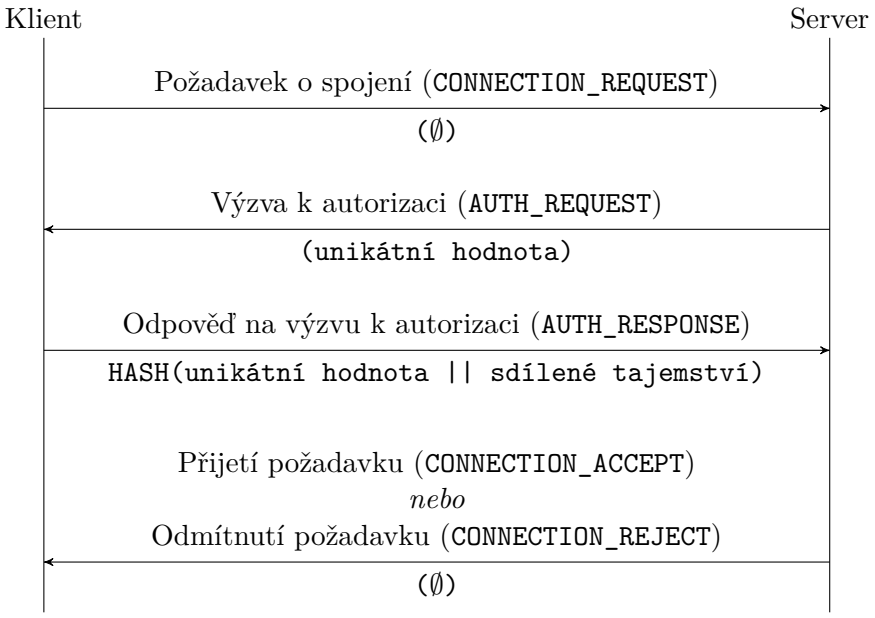
Otisk, nebo-li *hash*, je z dat tajemství generován předvolenou bezpečnou *hashovací funkcí* (výchozí volbou je *SHA256*). Klient zasílá serveru pouze otisk, aby bylo zabráněno odposlechnutí tajemství. Vyzrazení sdíleného tajemství je zabráněno díky jedné ze stěžejních vlastností *hashovací funkce – jednosměrnost*. Díky této vlastnosti je výpočetně extrémně náročné z výstupu (*otisk*) *hashovací funkce* získat vstupní hodnotu (*sdílené tajemství*). Vyzrazení tajemství není však jediným bezpečnostním rizikem při implementaci protokolu *výzvy-odpověď*.

[5] Anglicky vlastnost *thread-safety*

2.5.1 Bezpečnostní aspekty

Útočník, který je schopen odposlouchávat síťový provoz během úspěšné autentizace klienta, může odposlechnutou komunikaci využít k následné vlastní autentizaci. Pro zabránění *útoků přehráním* (známého také jako *Replay attack*) je každý proces autentizace doplněn o unikátní hodnotu (často označovanou jako *sůl*, anglicky *salt*) vázanou na konkrétní požadavek o spojení. Server tuto unikátní hodnotu vygeneruje, zapamatuje si ji a zašle klientovi. Klient následně při vytváření otisku tuto unikátní hodnotu spojí s daty sdíleného tajemství a výsledný otisk tedy závisí nejen na sdíleném tajemství, ale rovněž na unikátní hodnotě požadavku. Díky tomu má každý požadavek jiný *správný* otisk.

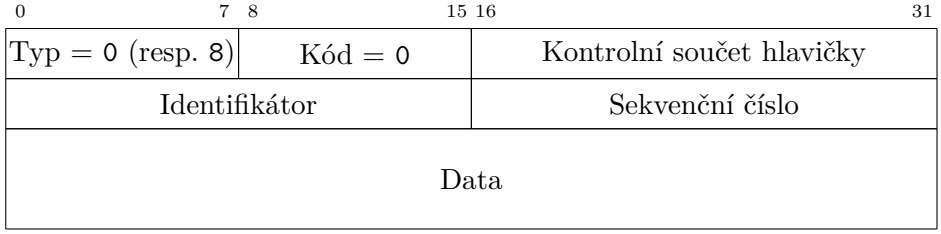
Při použití soli je klíčové vygenerování dostatečně náhodné hodnoty. Pokud není zajištěna dostatečná náhodnost dat, může s dostatečnou znalostí protokolu útočník hodnotu soli uhodnout nebo případně dopředu předvídat. Pro generování pseudonáhodných hodnot je v práci použita knihovna *OpenSSL*, konkrétně funkce `RAND_bytes()`, která dle specifikace[13] generuje kryptograficky silná pseudonáhodná čísla.



Obrázek 2.1: Sekvenční diagram autentizace klienta pomocí protokolu *výzva-odpověď*

2.6 Plugin pro ICMP tunelování

Tunelování dat pomocí ICMP je technicky možné díky zprávám typu 0 a 8 (echo reply, resp. echo request), pro které je specifikována proměnlivá délka zpráv. Zprávy těchto typů jsou používány zejména nástrojem ping – například pro zjištění dostupnosti cíle, nebo pro informace o latenci spojení. Zdrojová stanice vyšle na cílovou stanici zprávu typu echo request a cílová stanice odpoví zprávou echo reply. Toto chování je zajištěno v [14, RFC1122]. Strukturu zmíněných echo ICMP zpráv znázorňuje následující diagram 2.2.



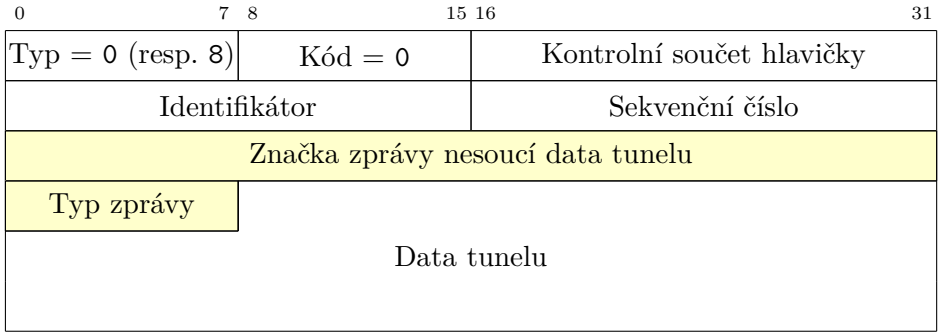
Obrázek 2.2: Diagram ICMP zprávy typu echo request (resp. echo reply)

Protože ICMP pakety echo reply, (resp. echo request) na rozdíl od UDP a TCP paketů nenesou informaci o zdrojovém/cílovém portu, jsou pro sdružování souvisejících zpráv použity hodnoty *identifikátor* a *sekvenční číslo*. Dle [9, RFC 792] a [15, RFC 3022] mohou síťové prvky, jako například NAT, pro rozpoznání souvisejících zpráv tyto hodnoty používat. Specifikace se však nezmiňuje o délce platnosti *identifikátoru*. Díky tomu je možno udržovat obousměrnou komunikaci, i pokud je jeden z účastníků omezen NAT, protože je možné klientovi „za NATem“ zaslat více paketů se shodnými hodnotami *sekvenční číslo* a *identifikátor*. Na základě hodnoty *identifikátor* je identifikována i žádost klienta o připojení a následná odpověď na výzvu serveru.

Systém, který dodržuje [14, RFC1122] se však bude snažit na příchozí ICMP *echo request* pakety reagovat. Toto chování je pro správnou funkci tunelu nežádoucí. V linuxových distribucích je možné povolit ignorování příchozích zpráv ICMP echo změnou souboru /proc/sys/net/ipv4/icmp_echo_ignore_all.

Aby obě strany tunelu byly schopny identifikovat tok dat tunelu a rozeznat takové zprávy od jiných ICMP zpráv, je nutné „tunelové“ ICMP zprávy označit. Právě proto začíná každá taková ICMP zpráva stejnou sekvencí čtyř bajtů, která označuje ICMP zprávy, nesoucí data síťového tunelu. Komunikující protějšky kontrolují zdrojovou IP adresu označených ICMP zpráv, aby nebylo triviálně možné injektovat komunikaci do tunelu. Těchto dodatečných pět bajtů je potřebných pro správnou funkci tunelu a předávání servisních zpráv. Umístění těchto informací v ICMP paketu je vyznačeno na diagramu 2.3.

Značku zpráv tunelu má uživatel možnost změnit při kompilaci programu. Výchozí značkou je sekvence bajtů 0x63 0x76 0x75 0x74, tedy CVUT.



Obrázek 2.3: Diagram ICMP zprávy typu `echo request` (resp. `echo reply`) s vyznačenou hlavičkou dat tunelu

2.6.1 Popis jednotlivých typů ICMP zpráv

Typ zprávy označuje jeden z typů, zachycených v ukázce kódu 2.6.1:

```
1  typedef enum ICMP_PACKET_TYPE
2  {
3          ICMP_CONNECTION_REQUEST,
4          ICMP_AUTH_CHALLENGE,
5          ICMP_AUTH_RESPONSE,
6          ICMP_CONNECTION_ACCEPT,
7          ICMP_CONNECTION_REJECT,
8          ICMP_NATPACKET,
9          ICMP_KEEPALIVE,
10         ICMP_DATA
11 } ICMP_PACKET_TYPE;
```

Ukázka kódu 2.1: Výňatek souboru `plugins/icmp/packet.h` definující typy ICMP zpráv

CONNECTION_REQUEST Tyto zprávy odesílá klient a přijímá server. Pokud by klientovi dorazila zpráva tohoto typu, bude ignorována. Server zprávu přijme a pokud momentálně není k serveru připojen žádný jiný klient, server odpoví zprávou typu AUTH_CHALLENGE, čímž klienta vyzve k autentizaci. V případě, že při přijetí zprávy CONNECTION_REQUEST server již navázal spojení s jiným klientem, bude žadateli odeslána zpráva typu CONNECTION_REJECT a požadavek na navázání tunelu tím pádem zamítnut.

AUTH_CHALLENGE Tento typ zpráv posílá server klientovi. Server příchozí zprávy tohoto typu ignoruje. Klient po přijetí zpracuje výzvu k autorizaci (protokol *Challenge-Response*) a výsledek předá serveru (viz následující odstavec **AUTH_RESPONSE**. V závislosti na tom, jestli server uživatele autorizuje

bude klientovi odeslána zpráva typu CONNECTION_ACCEPT a tunelové spojení zahájeno, nebo bude jeho požadavek zamítnut zprávou CONNECTION_REJECT.

AUTH_RESPONSE Zprávy tohoto typu odesílá klient serveru. Klient příchozí zprávy tohoto typu ignoruje. Obsahem zprávy je odpověď *Challenge-Response* protokolu na základě v minulosti přijaté zprávy AUTH_CHALLENGE. V závislosti na tom, jestli server uživatele autorizuje bude klientovi odeslána zpráva typu CONNECTION_ACCEPT a tunelové spojení zahájeno, nebo bude jeho požadavek zamítnut zprávou CONNECTION_REJECT.

CONNECTION_ACCEPT Touto zprávou server stvrzuje úspěšné navázání spojení s adresovaným klientem. Server příchozí zprávy tohoto typu ignoruje. Po celou dobu spojení odmítá server jakékoliv další pokusy o připojení jiných klientů.

CONNECTION_REJECT Touto zprávou server oznamuje zamítnutí klientova požadavku na vytvoření tunelového spojení se serverem. Server příchozí zprávy tohoto typu ignoruje. Zprávu klient obdrží pokud se pokusí připojit k již obsazenému serveru, nebo se mu nepodaří autorizovat.

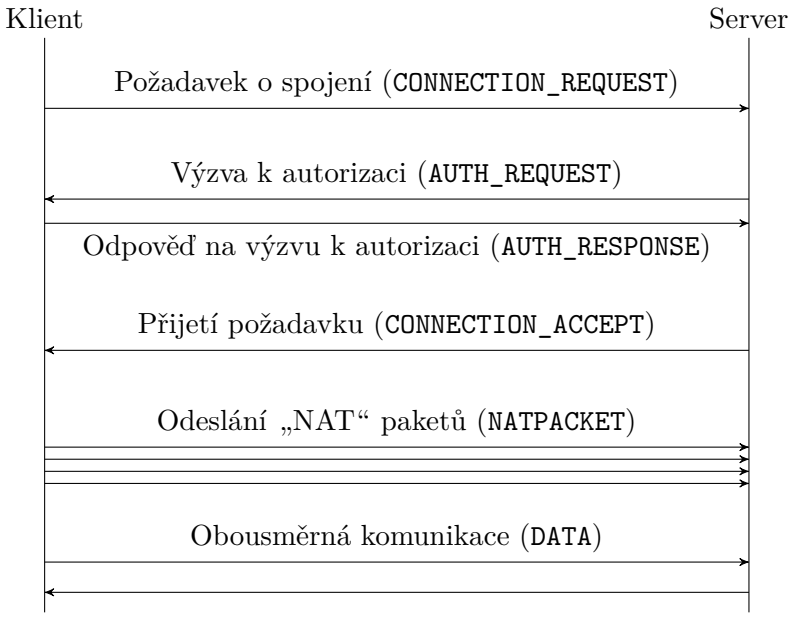
NATPACKET Tyto zprávy odesílá klient a přijímá server. Pokud by klientovi dorazila zpráva tohoto typu, bude ignorována. Server zprávu přijme a poznamená si *identifikátor* a *sekvenční číslo* zprávy. Tyto údaje následně server použije při odesílání zpráv klientovi. Jedná se o techniku překonání překážek, které způsobuje NAT.

KEEPALIVE Zprávy KEEPALIVE slouží k detekci problémů se spojením klienta a serveru. Zprávy odesílá klient (ohlašuje svou aktivitu serveru) a server na ně stejnou zprávou odpovídá (ujišťuje klienta, že spojení je v pořádku). Do jisté míry pakety KEEPALIVE napodobují nástroj ping. Pokud opakovaně nedojde k přijetí zprávy KEEPALIVE, klient a server považují spojení za přerušené nebo ukončené. Server navíc začne opět přijímat požadavky pro připojení CONNECTION_REQUEST.

DATA Zprávy tohoto typu nesou data tunelu. Odesílá a přijímá je jak klient, tak server.

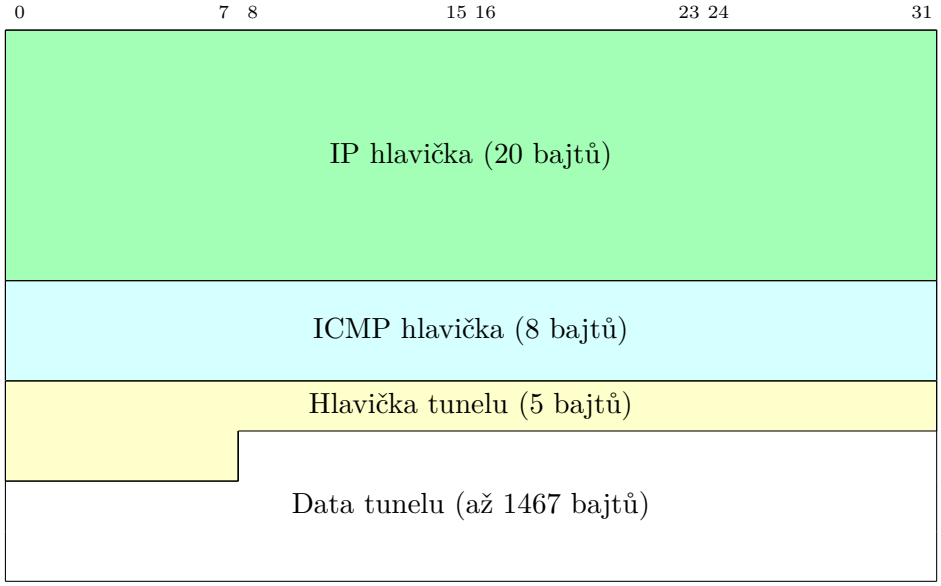
Situace, kdy se klient připojuje k volnému server je zachycena na diagramu 2.4.

Celková velikost paketu nesoucího zprávu je standardně nastavena na 1500 bajtů – jedná se o MTU technologie *Ethernet*. To zahrnuje IP hlavičku (20 bajtů), ICMP hlavičku (8 bajtů) a hlavičku zpráv tunelu (5 bajtů). Pro přenášená data je tedy k dispozici až 1467 zbylých bajtů v jedné zprávě. Režie



Obrázek 2.4: Sekvenční diagram navázání `ICMP` tunelu

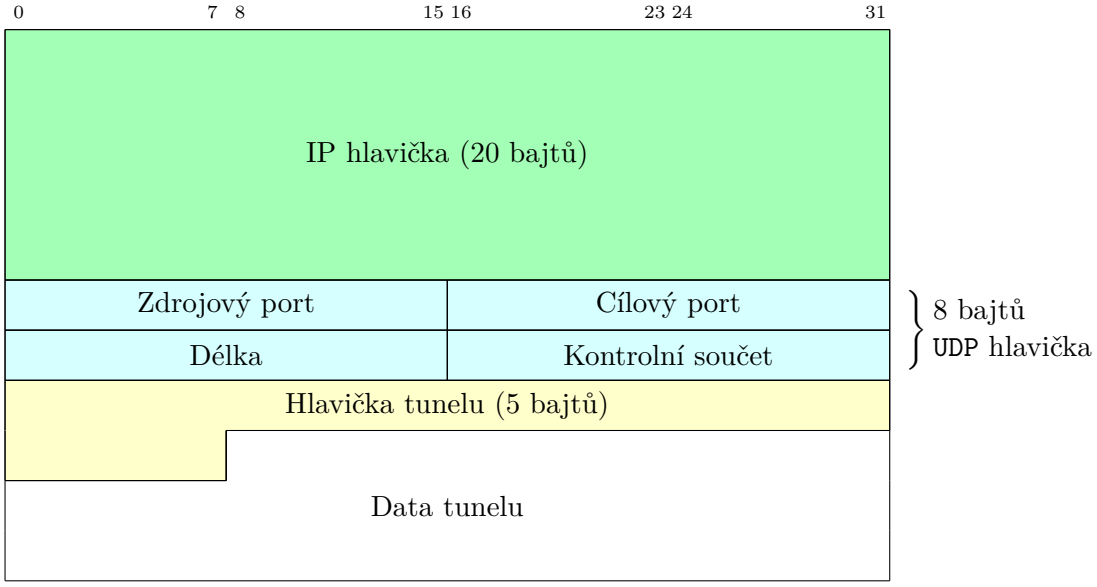
přenosu dat tunelem tvoří 21 bajtů. Vytvářené pakety se drží standardní hodnoty `MTU` aby bylo (pokud možno) zamezeno fragmentaci. Diagram paketu se všemi hlavičkami je znázorněn na obrázku 2.5.



Obrázek 2.5: Diagram paketu včetně rozložení `ICMP` zprávy tunelu a hlaviček

2.7 Plugin pro UDP tunelování

Tunelování dat skrze UDP tunel je primárním principem technologie *OpenVPN*[6]. Jedná se o jednoduchý způsob tunelování dat sítí, kdy jsou data tunelu zapouzdřena do UDP datagramů. Protokol UDP nezajišťuje doručení zpráv v původním pořadí a neobsahuje ani mechanismy pro detekci ztracených nebo duplicitních zpráv. Tato zodpovědnost je ponechána na protokolech vyšších vrstev. Strukturu UDP datagramu znázorňuje diagram 2.6.



Obrázek 2.6: Diagram paketu obsahující UDP datagram s daty tunelu

Rozdílem oproti tunelování pomocí ICMP je volba portu pro komunikaci. Volbu portu závisí na předpokládané výjimce na firewallu *captive portálu*. Mezi běžné aplikace protokolu UDP patří zejména:

- *Domain Name System*, protokol pro překlad doménových jmen, port 53,
- *Network Time Protocol*, protokol pro synchronizaci času, port 123,
- *OpenVPN*, SW pro síťové tunely, port 1194,
- *Session Initiation Protocol*, protokol pro přenos signalizace, port 5060

Výše zmíněné služby jsou široce rozšířené a je tudíž možné, že pro ně na firewallu *captive portálu* bude existovat výjimka. Například IP telefony typicky nejsou schopny se ověřovat *captive portálu* a právě proto by pro takové zařízení mohla existovat výjimka ve firewallu. Má proto smysl provozovat UDP tunel

[6] *OpenVPN* samozřejmě podporuje i TCP tunelování

na takovém portu – tedy za účelem zvýšení šance nalezení bezpečnostních nedostatků firewallu *captive portálu*. Z výčtu jsem záměrně vynechal některé známé aplikace protokolu UDP, jako například DHCP – protože má silně lokální charakter a je velmi nepravděpodobné, že by neautentizovaným klientům bylo povoleno komunikovat s DHCP serverem napříč Internetem. Protokol DNS rovněž není vhodným kandidátem, protože pro DNS je v rámci této práce implementována sofistikovanější metoda tunelování dat, blíže popsaná v podkapitole 2.9 *Plugin pro DNS tunelování*.

Vhodnými kandidáty jsou tedy porty 123 (NTP) a 5060 (SIP). Službu *OpenVPN*, která do jisté míry plní podobný účel, lze pohodlně provozovat bok po boku softwarového řešení této diplomové práce. Není proto důvodu zbytečně okupovat výchozí port jiného software.

2.7.1 Popis jednotlivých typů UDP zpráv

Typ zprávy označuje jeden z typů, zachycených v ukázce kódu 2.7.1:

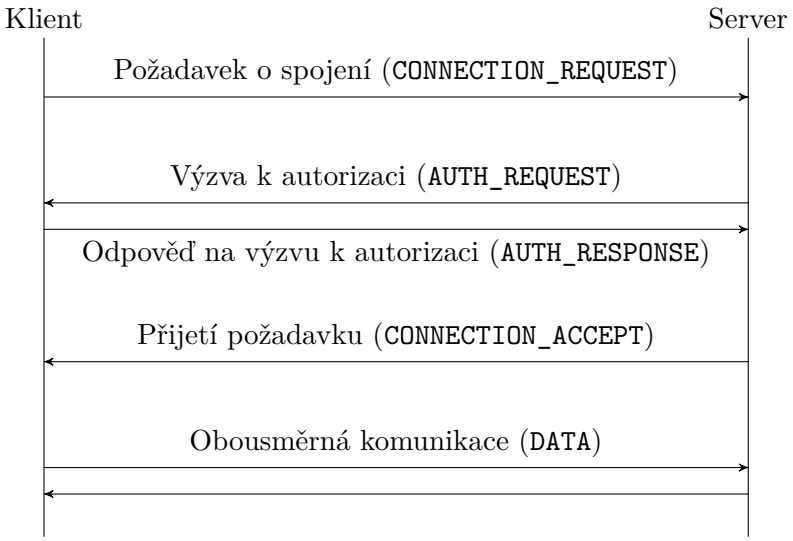
```
1  typedef enum UDP_PACKET_TYPE
2  {
3          UDP_CONNECTION_REQUEST,
4          UDP_AUTH_CHALLENGE,
5          UDP_AUTH_RESPONSE,
6          UDP_CONNECTION_ACCEPT,
7          UDP_CONNECTION_REJECT,
8          UDP_KEEPALIVE,
9          UDP_DATA
10 } UDP_PACKET_TYPE;
```

Ukázka kódu 2.2: Výňatek souboru `plugins/udp/packet.h` definující typy UDP zpráv

Z výňatku kódu je patrná analogie s výčtem zpráv 2.6.1 z podkapitoly *Plugin pro ICMP tunelování*. Jediným rozdílem oproti dříve popsaným typům zpráv je absence zpráv typu NATPACKET, protože protistrana (server) je schopen s klientem za NAT komunikovat díky překladu portů přímo. Klíčové jsou tím pádem KEEPALIVE zprávy, které slouží zejména k přesvědčení NAT zařízení o tom, že komunikace ještě neustala. Díky tomu může server komunikovat s klientem i skrze NAT.

Protože se však význam ostatních zpráv nijak nemění, je jejich opětovný popis vynechán s odkazem na popis v podkapitole 2.6.1 *Popis jednotlivých typů ICMP zpráv*.

Velmi podobný je také sekvenční diagram 2.7 komunikace klienta a serveru při pokusu o navázání spojení. V diagramu se nadále nevyskytují zprávy typu NATPACKET.



Obrázek 2.7: Sekvenční diagram navázání UDP/TCP tunelu

2.8 Plugin pro TCP tunelování

2.9 Plugin pro DNS tunelování

2.10 Plugin pro SCTP tunelování

SCTP je protokolem transportní vrstvy ISO/OSI modelu, stejně jako protokoly TCP nebo UDP. Přebírá některé vlastnosti těchto protokolů (TCP garanci doručení paketů ve správném pořadí, nebo UDP způsob komunikace (oproti TCP proudové komunikaci). Navíc přidává funkce jako možnost více proudové komunikace v rámci jednoho spojení. Jedná se o *relativně* mladý protokol, definovaný v [16, RFC 4960] v roce 2007.

Stejně jako u ICMP je možné, že implementace *captive portálu* nebude pamatovat na jiné protokoly než TCP a UDP. Právě proto je v rámce této práce implementován rovněž tunel pomocí SCTP protokolu.

Kapitola 3

Testování

Doplňte vhodný text.

Závěr

Doplňte závěr.

Literatura

[1] Kumari, W.; Gudmundsson, O.; Ebersman, P.; aj.: Captive-Portal Identification Using DHCP or Router Advertisements (RAs). RFC 7710 (Proposed Standard), Prosinec 2015. Dostupné z: https://tools.ietf.org/html/rfc7710

[2] Lauer, O.: *Porovnání systémů pro pokročilou správu připojení k síti*. Bakalářská práce, České vysoké učení technické v Praze, 2017.

[3] Smitka, J.: *Systém pro řízení přístupu do kolejní sítě ZČU*. Bakalářská práce, Západočeská univerzita v Plzni, 2016.

[4] Facebook: Get Facebook Wi-Fi for Your Business [online]. 2013, [cit. 2018-02-20]. Dostupné z: https://www.facebook.com/business/facebook-wifi

[5] Nintendo Co., Ltd.: Consolidated Sales Transition by Region [online]. 4 2016, [cit. 2018-02-23]. Dostupné z: https://www.nintendo.co.jp/ir/library/historical_data/pdf/consolidated_sales_e1603.pdf

[6] Wireless Broadband Alliance: WISPr 2.0 [online]. 4 2010, [cit. 2018-02-23]. Dostupné z: https://bitbucket.org/tamias/pywispr/downloads/WBA-WISPr2.0v01.00.pdf

[7] Google Chromium: Certificate Transparency in Chrome - Change to Enforcement Date [online]. 4 2017, [cit. 2018-02-23]. Dostupné z: https://groups.google.com/a/chromium.org/forum/#!msg/ct-policy/sz_3W_xKBNY/6jq2ghJXBAAJ

[8] Felt, A. P.: Twitter [online]. 1 2015, [cit. 2018-02-23]. Dostupné z: https://twitter.com/__apf__/status/551083956326920192

[9] Postel, J.: Internet Control Message Protocol. RFC 792 (Internet Standard), Září 1981. Dostupné z: https://tools.ietf.org/html/rfc792

[10] Laliberte, M.: Lessons from DEFCON 2016 – Bypassing Captive Portals [online]. 8 2016, [cit. 2018-03-15]. Dostupné z: https://www.secplicity.org/2016/08/26/lessons-defcon-2016-bypassing-captive-portals/

[11] Farnham, G.: Detecting DNS Tunneling [online]. 2 2013, [cit. 2018-03-18]. Dostupné z: https://www.sans.org/reading-room/whitepapers/dns/detecting-dns-tunneling-34152

[12] Rosa, Z.: *Detekce síťových tunelů v počítačových sítích*. Bakalářská práce, České vysoké učení technické v Praze, 2014.

[13] OpenSSL Cryptography and SSL/TLS Toolkit [online]: Manual page of RAND_bytes(). 3 2018, [cit. 2018-04-23]. Dostupné z: https://www.openssl.org/docs/man1.1.0/crypto/RAND_bytes.html

[14] Braden (Ed.), R.: Requirements for Internet Hosts - Communication Layers. RFC 1122 (Internet Standard), Říjen 1989. Dostupné z: https://tools.ietf.org/html/rfc1122

[15] Srisuresh, P.; Egevang, K.: Traditional IP Network Address Translator (Traditional NAT). RFC 3022 (Informational), Leden 2001. Dostupné z: https://tools.ietf.org/html/rfc3022

[16] Stewart (Ed.), R.: Stream Control Transmission Protocol. RFC 4960 (Proposed Standard), Září 2007. Dostupné z: https://www.rfc-editor.org/rfc/rfc4960.txt

Příloha A

Seznam použitých zkratek

DNS Domain Name System

ICMP Internet Control Message Protocol

XML Extensible markup language

ISM Industrial, Scientific and Medical radio bands

NAC Network Access Control – řízení síťového přístupu

GSM Global System for Mobile Communications

MAC Media Access Control

IP Internet Protocol

SIP Session Initiation Protocol

MITM Man-in-the-middle

HTTP Hypertext Transfer Protocol

MTU Maximum transmission unit

HTTPS HTTP Secure

TTL Time to live

SOCKS Socket Secure

NAT Network Address Translation

SSH Secure Shell

TCP Transmission Control Protocol

UDP User Datagram Protocol

RADIUS Remote Authentication Dial-In User Service

VLAN Virtual local area network

WWW World wide web

WPA Wi-Fi Protected Access

WISPr Wireless Internet Service Provider roaming

VoIP Voice over IP

VPN Virtual private network

Příloha B

Obsah přiloženého CD