



**FAKULTA
INFORMAČNÍCH
TECHNOLGIÍ
ČVUT V PRAZE**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název: Ochrana anonymity uživatele v Internetu
Student: Jakub Dvořák
Vedoucí: Ing. Josef Kokeš
Studijní program: Informatika
Studijní obor: Bezpečnost a informační technologie
Katedra: Katedra počítačových systémů
Platnost zadání: Do konce letního semestru 2018/19

Pokyny pro vypracování

- 1) Nastudujte problematiku anonymity v síti Internet.
- 2) Analyzujte známé útoky na anonymitu ve vztahu k webovým prohlížečům.
- 3) Zaměřte se na odhalení pravé IP adresy uživatele používajícího anonymizační nástroje. Demonstrujte provedení útoku ve virtuálním stroji.
- 4) Formulujte možné způsoby obrany.
- 5) Implementujte obranný mechanismus jako rozšíření prohlížeče.
- 6) Diskutujte své výsledky.

Seznam odborné literatury

Dodá vedoucí práce.

prof. Ing. Róbert Lórencz, CSc.
vedoucí katedry

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

V Praze dne 27. ledna 2018



**FAKULTA
INFORMAČNÍCH
TECHNOLGIÍ
ČVUT V PRAZE**

Bakalářská práce

Ochrana anonymity uživatele v Internetu

Jakub Dvořák

Katedra počítačových systémů
Vedoucí práce: Ing. Josef Kokeš

19. května 2018

Poděkování

Na tomto místě bych rád poděkoval Ing. Josefovi Kokešovi za věcné rady při zpracování bakalářské práce a implementačních částí. Dále bych rád poděkoval své rodině za podporu při studiu a pomoc při hledání gramatických chyb, stejně tak jako mé přítelkyni a její rodině.

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. Dále prohlašuji, že jsem s Českým vysokým učením technickým v Praze uzavřel dohodu, na základě níž se ČVUT vzdalo práva na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona. Tato skutečnost nemá vliv na ust. § 47b zákona č. 111/1998 Sb., o vysokých školách, ve znění pozdějších předpisů.

V Praze dne 19. května 2018

.....

České vysoké učení technické v Praze
Fakulta informačních technologií

© 2018 Jakub Dvořák. Všechna práva vyhrazena.

Tato práce vznikla jako školní dílo na Českém vysokém učení technickém v Praze, Fakultě informačních technologií. Práce je chráněna právními předpisy a mezinárodními úmluvami o právu autorském a právech souvisejících s právem autorským. K jejímu užití, s výjimkou bezúplatných zákonných licencí a nad rámec oprávnění uvedených v Prohlášení na předchozí straně, je nezbytný souhlas autora.

Odkaz na tuto práci

Dvořák, Jakub. *Ochrana anonymity uživatele v Internetu*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

V práci se zabývám ochranou anonymity uživatele v Internetu. V teoretické části vysvětlím důležité pojmy pro pochopení principů útoků a obranných mechanismů. Odhalím jádra problémů vybraných útoků a jejich náležitou obranu proti těmto metodám. V praktické části navazuji demonstrací vybraného útoku na anonymitu a implementací rozšíření do prohlížeče Google Chrome, jako obranného mechanismu. Útok je implementován pomocí programovacího jazyka JavaScript, který používá rozhraní WebRTC. Rozšíření pro prohlížeč Google Chrome je implementován v HTML a JavaScript.

Klíčová slova Chrome rozšíření, JavaScript, napadení internetové anonymity, ochrana anonymity na Internetu, únik IP adresy, WebRTC

Abstract

In this thesis I deal with protection of user anonymity on the Internet. In theoretical section I explain important concepts for understanding principles of attacks and defense mechanisms. I reveal the core aspects of chosen problems and proper defense against these methods. In the practical section I demonstrate a chosen attack on anonymity and implementation of extension for Google Chrome web browser as a defense mechanism. Attack is implemented in JavaScript programming language and uses the framework WebRTC. Extension for web browser Google Chrome is implemented in HTML and JavaScript.

Keywords Chrome extension, JavaScript, Internet anonymity attack, IP leakage, protection of anonymity on the Internet, WebRTC

Obsah

Úvod	1
1 Anonymita v síti Internet	3
1.1 Význam anonymity	3
1.2 Historie Internetu	4
1.3 Komunikační protokoly	5
2 Analýza známých útoků na anonymitu	11
2.1 Anonymizéry	11
2.2 Útoky na anonymitu	13
3 Analýza obrany proti odhalení	19
3.1 Obrana proti nedůvěryhodným prostředníkům	19
3.2 Obrana proti MITM	19
3.3 Obrana proti modifikaci dat	20
3.4 Obrana proti DNS leak	20
3.5 Obrana proti (D)DoS	21
3.6 Obrana vývoje a implementace	21
3.7 Obrana proti Action Script	21
3.8 Obrana proti cross-site útoku	22
3.9 Obrana Tor sítě	22
3.10 Obecné doporučení zachování anonymity	23
4 Demonstrace útoku	25
4.1 WebRTC P2P protokol	25
4.2 Popis útoku	27
5 Realizace obrany	33
5.1 Rozšíření	33
5.2 Použití rozšíření	36

5.3 Implementace	38
5.4 Diskuse	41
Závěr	43
Bibliografie	45
A Seznam použitých zkratk	49
B Obsah přiloženého CD	51

Seznam obrázků

1.1	OSI model [3]	5
1.2	TCP/IP model [3]	6
1.3	Porovnání OSI a TCP/IP modelů [3]	7
1.4	Spojení mezi serverem a klientem pomocí UDP [7]	8
1.5	Spojení mezi serverem a klientem pomocí TCP [8]	8
1.6	Spojení mezi serverem a klientem pomocí RTP [9]	9
1.7	Spojení mezi serverem a klientem pomocí FTP [11]	9
2.1	Komunikace pomocí proxy serveru [14]	11
2.2	Komunikace v síti Tor [17]	12
2.3	Komunikace v I2P síti [18]	13
2.4	MITM útok na šifrovanou komunikaci [21]	14
2.5	Útok na skryté služby [18]	16
3.1	Nastavení DNS	20
3.2	Vypnutí WebRTC pro Mozilla Firefox	22
4.1	Komunikace se STUN serverem [37]	26
4.2	Webová stránka s útokem	27
5.1	Popup okno rozšíření	34
5.2	Architektura rozšíření [41]	35
5.3	Okno s nastavením	35
5.4	Kontextové menu	36
5.5	Okno s rozšířeními	36

Úvod

Dnes žijeme v době, kdy je Internet nepostradatelnou součástí našich životů. Využíváme ho denně, ať aktivně, či pasivně. Současně s touto skutečností se nám otevírají otázky bezpečnosti, ochrany citlivých údajů a snaha o maximální zabezpečení našeho soukromí. Ty jsou dnes řešeny nejen bezpečnostními složkami a aktivními jednotlivci, ale i státy, politickými stranami, či světovými společenstvími.

Při používání Internetu zanecháváme po sobě vědomě či nevědomě mnoho informací, které narušují naše soukromí, a tím přestáváme být anonymními uživateli. Uchovávání informací o uživateli má své klady i zápory. Za klady je možné považovat snadnější odhalení nelegálního obchodu s drogami, zbraněmi a jiných trestných činů, za zápory naopak ztrátu práva svobodného slova nebo anonymního průchodu webovými stránkami.

Anonymita a soukromí bylo vyžadováno již v minulosti. Stejného stavu chceme docílit při používání Internetu. Do jisté míry toto můžeme ovlivnit sami, ale při neopatrném jednání se můžeme odhalit. Práce je určena pro každého uživatele Internetu, který chce ovlivnit své soukromí ve virtuálním světě.

Téma jsem si vybral z důvodu velkého osobního zájmu o danou problematiku, která dodnes není zcela vyřešena. Dalším faktorem motivace je podpora řešení daného problému a zajistit co největší anonymitu pro každého, kdo ji požaduje a vyvarovat se častým problémům, které mohou nastat při neopatrnosti.

V práci uvedu čtenáře do problematiky anonymity v Internetu a doporučím, co udělat pro to, aby byla zachována anonymita uživatele. Ze začátku se v práci budu věnovat teoretickému pojednání o problematice. Vysvětlím důležité pojmy, co je to anonymita a Internet. V další části shrnu historii Internetu, a poté se začnu zabývat analýzou nejznámějších útoků a dosavadní obranou. Dále navází částí praktickou s podpůrnou implementační částí, která se v jedné kapitole zabývá demonstrací útoku, jenž slouží k odhalení IP adresy a popisu

jeho funkčnosti a ve druhé realizací obrany proti danému útoku.

Cílem této práce je získat přehled o dnešních problémech anonymity v síti Internet, dosavadních řešení vybraných problémů, implementace vybraného útoku a implementace obrany.

Anonymita v síti Internet

Dříve, než se začneme zabývat problematikou ochrany soukromí uživatele v Internetu a jeho anonymity, si vysvětlíme důležité pojmy, abychom znali vše důležité, co je nutné k porozumění vybraných metod a technologií popsaných v této práci a jaká je historie technologií. V útocích a obranných mechanismech v kyberprostoru je důležitým faktorem perfektní znalost jádra technologií a povědomí o tom, jak přesně věci fungují, aby útok a náležitá obrana byly co nejefektivnější a bezchybné.

1.1 Význam anonymity

Slovo anonymita pochází z řeckého slova *anonymia*, ve významu bezejmenný. Anonymita zde byla již před příchodem internetu. Už dříve lidé nechtěli, aby někdo znal jejich pravé jméno, a vymýšleli si vlastní pseudonymy. [1] Příkladem může být Studentovo rozdělení, které je využíváno ve statistice. Autor je již známý, ale publikoval pod pseudonymem Student. S anonymitou se dnes můžeme setkat například v matematických oborech, případně v literatuře, kdy autoři používají pseudonymy nebo zůstávají bezejmennými.

Uživatelé se mohou v některých prohlížečích setkat se speciální funkcí nazvanou „Anonymní okno“. Tato funkce do jisté míry anonymitu zajišťuje, avšak pouze na používaném počítači, ale nijak člověka neskrývá v internetové síti, protože mezi běžnými metodami této anonymity je pouhé neukládání historie, hesel, či cookies. Na kteroukoliv stránku v tomto anonymním okně se uživatel dostane, stále je jeho IP adresa neskryta a dá se dohledat, kdo sedí za počítačem.

Anonymita má své klady, ale také zápory. Stejně tak, jako se jí dá využít pro účely vyhledávání informací, které nejsou všude dostupné, tak ji lze i zneužít pro trestné činy. Mezi klady můžeme považovat:

- Anonymní prohlížení webových stránek.
- Veřejný názor, bez omezení nebo případného stíhání.

- Komunikace s ostatními lidmi beze strachu.
- Řešení osobních záležitostí bez studu.

Mezi zápory považujeme:

- Obtížná dohledatelnost útočníka.
- Obchodování se zbraněmi, drogami, či lidmi.
- Pomluva osob, bez možnosti dohledání pachatele.
- Podpora a propagace ideologií potlačující práva a svobody lidí, či skupin.
- Možnost kyberšikany.

Naskytá se otázka, jestli nám naše anonymita stojí za ta rizika, která jsou s ní spojena. Mnoho záporných věcí je samozřejmě hlídanych a anonymita není nikdy stoprocentní, takže se pachatel dohledá.

1.2 Historie Internetu

Kořeny internetu sahají do 60. let 20. století. Hlavním důvodem pro vývoj internetu byla studená válka, kde hrozilo zničení telekomunikačních prostředků jadernými zbraněmi. Ministerstvo obrany USA se tedy rozhodlo zadat studii odolné komunikační sítě. [2]

První sociální interakce, které byly popsány v srpnu 1962 v poznámkách J. C. R. Licklidera z MIT, byly nazvány „galaktická síť“. Tento koncept představoval množinu počítačů, vzájemně propojených, skrze které budou moci lidé rychle komunikovat a přistupovat k datům. V říjnu 1962 se J. C. R. Licklider stal vedoucím projektu odolné komunikační sítě výzkumné skupiny The Advanced Research Projects Agency, zkráceně ARPA, která koncem roku 1971 změnila svůj název na The Defense Advanced Research Projects Agency, zkráceně DARPA. [2]

V roce 1969 se začal vyvíjet první software pro spuštění Internet Message Processor, zkráceně IMP, který sloužil jako nástroj pro výměnu bloků dat s informacemi mezi síťovými uzly. Zpočátku tato síť měla pouze 4 uzly. [2] Postupem času se však tato síť začala rozšiřovat. Zpočátku se rozšiřovala průměrně jeden uzel měsíčně, toto číslo ale poté začalo růst exponenciálně. Současně se internetová síť začínala i technologicky zdokonalovat a vymýšlely se nové protokoly, jedním z nejznámějších byl komunikační protokol TCP/IP. V roce 1991 bylo registrováno přes 100 připojených zemí a 600 000 uživatelů. O rok později, 13. února 1992, se otevřela první internetová síť v Praze, v tehdejší České a Slovenské Federativní Republice.

1.3 Komunikační protokoly

Dříve, než popíši druhy komunikačních protokolů, je nutné zmínit modely, na nichž jsou postavené a které pokrývají. Dnešními nejběžněji používanými modely jsou The Open Systems Interconnection (OSI) a Transmission Control Protocol/Internet Protocol (TCP/IP).

1.3.1 OSI model



Obrázek 1.1: OSI model [3]

Na obrázku 1.1 je ilustrace modelu OSI, ze kterého je zřejmé, že se dělí na 7 částí, takzvaných vrstev:

- Aplikační
- Prezentační
- Relační
- Transportní
- Síťová
- Linková
- Fyzická

Každá vrstva má svůj speciální význam. Popis těchto vrstev, který následuje, začíná od nejspodnější vrstvy a postupuje se až k nejvyšší. Fyzická vrstva se stará o elektrické signály v hardwarové části. Jedná se v podstatě o signály vedoucí kabely, repeatery, či rozdělovače. Činnost linkové vrstvy spočívá v kódování

a dekodování jednotlivých signálů na bity a řešení chyb vycházejících z 1. vrstvy. Tuto část zpracovává přepínač, který kontroluje počítače v síti, pro které jsou signály, resp. bity určeny. Funkcionalita síťové vrstvy spočívá ve vytváření cest mezi dvěma uzly a směrování paketů k cíli. Transportní vrstva je zodpovědná za přenos dat mezi koncovými systémy a zotavení při chybě při přenosu. Relační vrstva tohoto modelu zajišťuje zprostředkování spojení mezi aplikacemi. SQL nebo NFS pracují na této vrstvě. Prezentační vrstva, jak již napovídá název, slouží k prezentaci dat koncovému uživateli. Mezi její funkcionality také patří šifrování a dešifrování dat. Poslední vrstva, která je na úplném vrcholu, je zodpovědná za služby, které například slouží k posílání e-mailů nebo souborů. [4, 5]

1.3.2 TCP/IP model



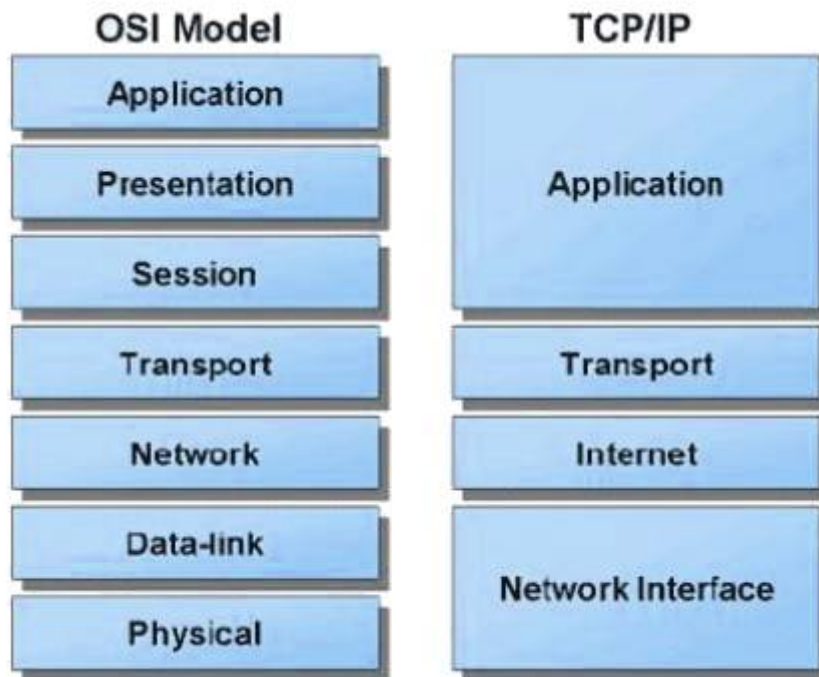
Obrázek 1.2: TCP/IP model [3]

Tento model, viz obrázek 1.2, využívá méně vrstev než OSI, i když z něho vychází. V tomto modelu se vrstvy vycházející z OSI sdružují do jedné. Vrstvy v tomto modelu jsou:

- Aplikační
- Transportní
- Síťová
- Vrstva síťového rozhraní

Ve vrstvě síťového rozhraní jsou spojeny vrstvy linková a fyzická. Její hlavní činností je přenos rámců mezi dvěma přímo připojenými zařízeními. Definiuje se zde metodika přístupu na medium. [4, 5] Vrstvy síťová a transportní mají stejnou úlohu, jako tomu bylo v předešlém modelu. Aplikační vrstva

je sdružením aplikační, prezentační a relační vrstvy do jedné. Její funkcí je správa srozumitelnosti a přenosu zpráv mezi klienty. Vazba mezi jednotlivými vrstvami obou modelů je zobrazeno na obrázku 1.3.



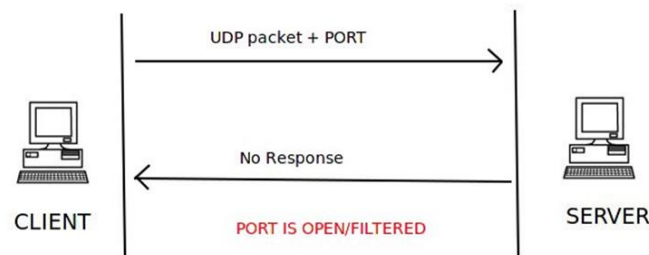
Obrázek 1.3: Porovnání OSI a TCP/IP modelů [3]

1.3.3 Druhy komunikačních protokolů

Komunikace na Internetu je souhrn pravidel, pomocí kterých si počítače v síti vzájemně vyměňují informace. Tyto souhrny definují komunikační protokoly. Těchto protokolů je několik a v této sekci si popíšeme ty nejznámější a nej-používanější.

1.3.3.1 UDP

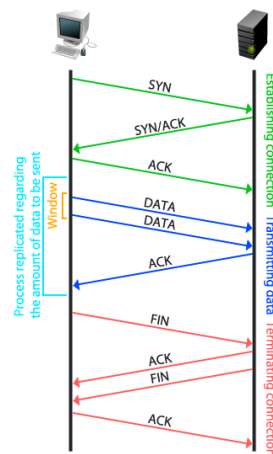
Jedním z nejjednodušších je User Datagram Protocol (UDP). Pro rozpoznání chyby při přenosu se používají kontrolní součty, avšak stále není zabráněno jejich ztrátě. [6] UDP se nejběžněji používá při streamování videí, hovorech nebo videohovorech. Jeho výhodou je rychlejší přenos, neboť se nečeká na potvrzení příjmu, za nevýhodu by bylo možno považovat možné výpadky paketů při přenosu. Na obrázku 1.4 je znázorněno spojení mezi klientem a serverem pomocí UDP.



Obrázek 1.4: Spojení mezi serverem a klientem pomocí UDP [7]

1.3.3.2 TCP

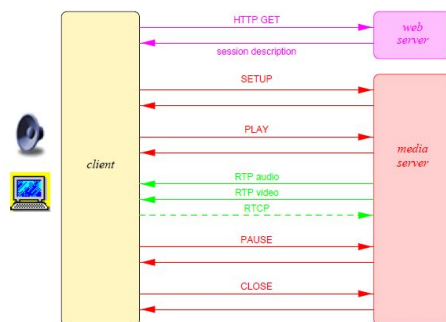
Zásluhou úspěšného systému TCP má i své jméno v protokolech zajišťujících komunikaci. Tento je poněkud složitější než předešlé UDP. Pro bezchybný přenos je využito systémů, které správně číslovají datagramy, potvrzení správnosti dat a jejich případné opakování, kdyby došlo k chybě v přenosu. [6] Nejvíce se využívá pro přenos souborů, e-mailů, vzdáleného připojení. Z obrázku 1.5 lze vyčíst, jak probíhá komunikace mezi serverem a klientem. Je zde znázorněno, v první části navázání spojení, v druhé přenos dat a v třetí ukončení spojení.



Obrázek 1.5: Spojení mezi serverem a klientem pomocí TCP [8]

1.3.4 RTP

Real Time Protocol (zkráceně RTP) slouží k lepšímu přenosu mediálního charakteru, pro který je důležité, kdy budou doručeny a aby byl zaručen rychlý přenos. [6] Příkladem může být živé vysílání, kdy data musí být pravidelně

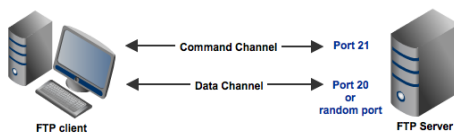


Obrázek 1.6: Spojení mezi serverem a klientem pomocí RTP [9]

doručována, aby nedocházelo k trhanému obrazu nebo výpadkům zvuku. Demonstrace přenosu a jeho ukončení pomocí tohoto protokolu je na obrázku 1.6.

1.3.5 FTP

File Transfer Protocol (FTP) je určen k přenosu souborů mezi zařízeními v síti. Není závislý na konkrétním operačním systému. Lze k němu tedy přistupovat z kteréhokoli systému. Jeho funkčnost je založená na principu klient-server. [10] To znamená, že server je článek poskytující data a klient je ten, kdo chce data získat. Při některých konfiguracích serveru může host na úložišti například měnit názvy souborů, přidávat je nebo mazat. Popis fungování spojení je na obrázku 1.7.



Obrázek 1.7: Spojení mezi serverem a klientem pomocí FTP [11]

1.3.6 HTTP

Hypertext Transfer Protocol (HTTP) se používá pro přenos dat na WWW. Návrh tohoto protokolu byl určen pro komunikaci mezi webovým prohlížečem a webovým serverem, avšak může být použit i pro jiné účely. [12] Komunikace tohoto protokolu probíhá stejně jako při TCP spojení. Nevýhodou tohoto protokolu je, že komunikace mezi webovým prohlížečem a serverem je nešifrovaná, a díky tomu jí může každý odposlouchávat. Proto byl vytvořen protokol HTTPS, kde písmeno S značí slovo Secure. Ten využívá protokolu HTTP

1. ANONYMITA V SÍTI INTERNET

společně s protokoly SSL nebo TLS, které slouží pro bezpečný zašifrovaný přenos.

Analýza známých útoků na anonymitu

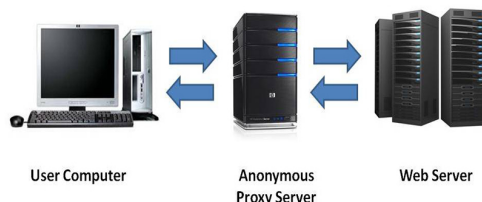
V této kapitole budou nejdříve stručně popsány nástroje, které nám pomáhají k anonymitě na Internetu, a popíšeme si, jak fungují. Poté budou popsány útoky, které se dají použít na všechny anonymizéry, a dále budou ukázány i některé útoky na konkrétní nástroj, který je následně použit i v demonstraci útoku popsaný v kapitole 4.

Jak jsem již dříve zmiňoval, je nutné, abychom znali jádro věcí, na které útočíme, případně bráníme. To stejné platí i pro fungování anonymizačních nástrojů, aby naše následná obrana byla důkladná, bezchybná a účinná.

2.1 Anonymizéry

Anonymizér je nástroj, který slouží k co největší anonymitě během používání internetu. [13] Obecně tyto nástroje fungují tak, že veškeré dotazy, které jsou zasílané na servery, prochází skrze jiné počítače. Následná odpověď od dotazovaného serveru je vrácena zpět skrze anonymizační nástroj.

2.1.1 Proxy servery



Obrázek 2.1: Komunikace pomocí proxy serveru [14]

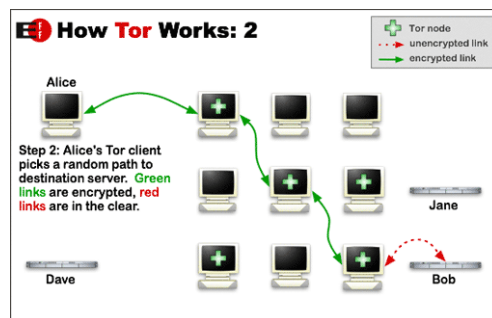
2. ANALÝZA ZNÁMÝCH ÚTOKŮ NA ANONYMITU

Anonymní proxy servery fungují na principu prostředníka mezi klientem a cílovým serverem. Všechny požadavky, které směřují na cílový server, jsou zprostředkovány pomocí proxy serveru, který pošle požadavek na cílový server. [15] Odpověď od cílového serveru je opět veden přes prostředníka. Popis zobrazující komunikaci lze vidět na obrázku 2.1.

2.1.2 Anonymní VPN

Metoda anonymizace pomocí VPN (Virtual Private Network) funguje podobně, jako tomu je v případě proxy serverů. V této metodě je jediný rozdíl, kterým je připojení vlastního zařízení do jiné sítě, kde nám je přiřazena vlastní privátní IP adresa v rámci sítě. [16] V této síti poté komunikujeme skrze server.

2.1.3 Tor

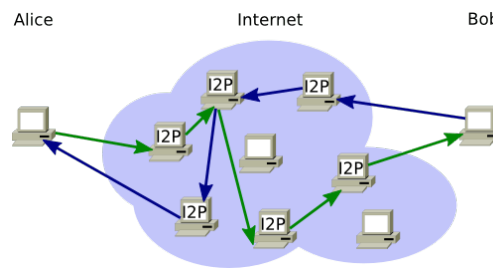


Obrázek 2.2: Komunikace v síti Tor [17]

TOR (The Onion Router) síť je množina serverů, které poskytují lidé dobrovolně. Do této sítě tedy může přispět kdokoli. [17] Tento nástroj funguje tak, že klient nejdříve získá seznam serverů od takzvaných „Directory serverů“, jenž slouží k držení databáze všech spojujících zařízení v síti. Po jejich získání vygeneruje náhodnou trasu skrze tato zařízení, viz obrázkový popis 2.2.

2.1.4 I2P

I2P (Invisible Internet Project) je projekt pro budování a správu anonymní komunikace. Síť podporuje dynamické konfigurace v reakci na různé útoky a její návrh je veden k rychlému využívání dalších zdrojů, jakmile jsou k dispozici. [19] Anonymizační síť tohoto projektu je tvořena množinou uzlů s řadou jednosměrných příchozích a odchozích tunelů. Komunikace mezi dvěma klienty nevede stejnou cestou, jelikož každý z nich využívá vlastních cest, tunelů. Viz obrázek 2.3).



Obrázek 2.3: Komunikace v I2P síti [18]

2.2 Útoky na anonymitu

Útoky na anonymitu jsou prováděny v souvislosti s použitými anonymizéry nebo na jejich obecné problémy. V této části kapitoly budu popisovat útoky využívající těchto souvislostí. Nejdříve popíši obecné problémy anonymizérů, kde nezáleží na tom, jaký typ je zrovna používán, a následně navážu útoky na anonymizér Tor, který jsem vybral pro demonstraci útoku a obrany.

2.2.1 Obecné problémy anonymizérů

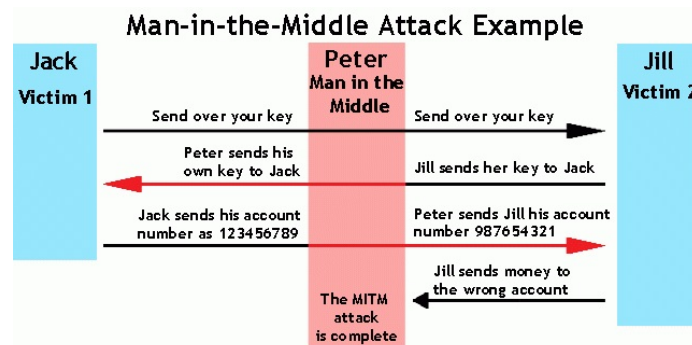
Ačkoliv uvedené anonymizéry fungují odlišně, přesto mají některé věci společné. Tím jsou útoky, u kterých nezáleží, jaký anonymizér je právě používán. Mohou využívat různých funkcionalit, které jsou spouštěny na straně klienta, pomocí kterých je například navázáno spojení, aniž by o tom uživatel věděl. Tento typ útoků je popsán obecně v této kapitole níže a následně i v kapitole 4, kde je popis podrobnější i s demonstrací. Druhý útoků, kde nezáleží na použitém anonymizéru, je více. V této sekci se nachází výběr takových útoků.

2.2.1.1 Nedůvěryhodná komunikace

V každém typu anonymizéru, který jsme si uvedli, je možnost dobrovolného zapojení se do systému anonymizace. [20] Stejně tak to ale mohou dělat i ti, kteří chtějí narušit naši anonymitu. Je zde riziko, že naše používání internetu nemusí být až tak anonymní, jak se domníváme, protože při komunikaci se servery může docházet k odposlechům. Záleží tedy na vlastní důvěře používané služby, respektive důvěra v servery, přes které je naše komunikace vedena.

2.2.1.2 Man in the middle

Známým útokem je útok zvaný Man in the middle (MITM). Tento útok spočívá v postavení třetí osoby mezi komunikací dvou klientů, kde odposlouchává komunikační kanál. Při běžné nezabezpečené komunikaci stačí, když se prostředník dostane pouze ke komunikačnímu kanálu. Při zabezpečených,



Obrázek 2.4: MITM útok na šifrovanou komunikaci [21]

resp. šifrovaných, komunikačních kanálech nestačí pouhé odposlouchávání komunikace. [21] Pokud dochází k výměně klíčů pro šifrování mezi klienty, je nutné, aby prostředník tuto výměnu zachytil a každému z účastníků komunikace zaslal svůj klíč. Příklad takového útoku na šifrovanou komunikaci lze vidět na obrázku 2.4. Účastníci Jack a Jill jsou oběti a Peter je narušitel, který se snaží získat důležitá data, která jsou zasílána skrze komunikační kanál. V ukázce jde o změnu bankovního účtu. Oběť 1 nejprve zašle svůj klíč oběti 2. Tento klíč narušitel zachytí a použije ho i jako svůj a přepošle oběti 2. Při zaslání klíče 2. oběti ho narušitel opět zachytí, ale už nepoužije a vygeneruje místo něj svůj. V této fázi má klíč pro šifrování narušitel a může změnit data ve zprávě.

2.2.1.3 Modifikace protékajících dat

Tato metoda využívá nezašifrovaného spojení. Útočník, který odposlouchává komunikaci, může zachytit některou ze zpráv a pozměnit její data. [22] V tomto útoku se většinou jedná o vložení skriptu nebo aplikace, která má za úkol odhalit uživatele. Může zde být například i periodické načítání stránky, aby bylo možné zjistit adresu, pokud se uživatel rozhodne vypnout anonymizér. Ve své podstatě se jedná o podobný princip útoku, který je popsán výše.

2.2.1.4 DNS leak

Při používání anonymizérů je nutné, aby celá naše komunikace procházela skrz anonymizační síť. Jakýkoliv únik z ní může sloužit k monitorování naší aktivity na internetu. Po připojení do anonymní sítě může za určitých podmínek operační systém nadále používat výchozí DNS namísto anonymních DNS serverů, jak je popsáno v [23]. DNS servery jsou provozovány poskytovateli internetu. Tito poskytovatelé tak mohou sledovat naši aktivitu na Internetu.

2.2.1.5 (D)DoS

(Distributed) denial of service útok může být znám jako způsob, jak vyřadit z provozu některý ze serverů. Tohoto útoku lze využít i k odhalení anonymity, ale je nutné mít vytipovaného potencionálního útočníka. [24] Server, skrze který útočník komunikuje, se zahlcuje nesmyslnými požadavky. Pak už jen stačí sledovat latenci podezřelého uživatele, pokud se mění při zahlcení serveru. Pro tento útok je nutné mít velký počet zdrojů, zařízení, které dokážou zahltit servery. Otázkou je, zda se tento útok opravdu vyplatí, jestli cena nákladů a energie vložená do tohoto útoku se nám vrátí nalezením daného uživatele.

2.2.1.6 Útok na implementaci a vývoj

Tyto dva útoky nevyužívají různých statistických metod nebo různých modifikací, ale spíše chyb, které mohli napáchat vývojáři. Při implementačním útoku se spíše zaměřujeme na odhalení chyb, které mohou být v konfiguracích nebo v kódu softwaru, kterých by se dalo zneužít k odhalení pravé IP adresy. V útoku na vývoj se útočník snaží podílet na vývoji některého ze softwaru, konfigurace sítě, apod. a nechávat si takzvaná zadní vrátka pro své pozdější útoky. Takovéto útoky mohou být někdy těžko prokazatelné, zda-li se jedná o chybu, která byla přidána za nějakým účelem nebo jen nepozornost programátora, a tak tedy tento člověk může dále zůstat mezi vývojáři a ohrožovat anonymitu. Jedním z příkladů útoku na implementaci je útok, respektive nalezení chyby v prohlížeči Mozilla Firefox, který využíval špatného manipulování s URL začínající `file://`, jak je uvedeno v [25]. Pokud uživatel přistoupí na speciálně vytvořenou adresu URL, operační systém se může připojit ke vzdálenému serveru a obejde tím nastavení prohlížeče. Chyba se týkala operačních systémů macOS a GNU/Linux.

2.2.1.7 Action script

Řada webových stránek využívá doplňků jako jsou například Flash Player nebo QuickTime. Tyto doplňky je možné využít k odhalení pravé IP adresy tím, že je možné díky nim vytvořit přímé spojení. [26] Flash obsahuje standardní funkce pro přímé spojení dvou systémů. Jeho spouštěč ignoruje jakékoliv nastavení proxy prohlížeče a vytvoří separátní TCP připojení z počítače oběti ke specifickému serveru. Podobně i QuickTime má možnost nastavení parametru, který slouží k vytvoření spojení pro sledování videa. Tato metoda útoku již přestává být aktuální hrozbou, protože se tyto doplňky nahrazují jinými a zmiňovaný Flash bude společností Adobe zrušen v roce 2020 [27]. Podobný princip, který je založen na spojení mezi dvěma systémy, bude detailněji popsán v kapitole 4, kde bude provedena i demonstrace daného útoku.

2.2.1.8 Cross-site útok

Při prohlížení webových stránek je někdy třeba, aby stránka reagovala na akci uživatele a k tomu nemusí čistě HTML jazyk stačit. Proto se využívá například programovacího jazyka JavaScript. Tento jazyk využívá prostředků na straně klienta, takže při některém z procesů, tvořených tímto jazykem, se používá procesor a operační paměť na straně klienta. Pomocí tohoto jazyka lze uskutečnit spojení mezi dvěma systémy. Princip útoku na anonymitu je tedy v podstatě stejný, jako v předešlém případě, s tím rozdílem, že se nevyužívá doplňků, ale funkcionality některých z jazyků pro implementaci webových aplikací, stránek. Pro tento jazyk jsou vyvíjeny mnohé podpůrné aplikace, které nás dokáží spojit s jiným klientem a například uskutečnit hovor. K tomuto slouží WebRTC. Detailnější popis útoku za pomoci WebRTC rozhraní bude popsán v kapitole 4.

2.2.2 Útoky na Tor síť

Tor síť se dnes potýká s mnoha typy útoků, které využívají například různých statistických metod k odhalení uživatelů nebo chyb v implementaci. Nyní si popíšeme některé z nich, které jsou dnes populární.

2.2.2.1 Korelační analýza

Tento útok se snaží zjistit závislost vstupního a výstupního datového toku na úrovni jednotlivých uzlů sítě nebo na jejich okrajích. Je zde nutná znalost sítě. Je možné také uvažovat pouze vybranou podmnožinu sítě a na jeho vstupech a výstupech data korelovat. [28] V principu se jedná o analýzu síťového provozu. Čím více dat je nasbíraných, tím je větší úspěšnost tohoto útoku. Dalším faktorem pro úspěšnosti je také zřetelná charakteristika přijímaných dat. Příklad dat, které se dají korelovat v tomto útoku, může být čas, který může dávat nezkreslené výsledky, pokud síť má nízké zpoždění nebo objem posílaných dat, kde jeho podstata spočívá v korelování velikosti posílaných a přijímaných dat na obou koncích komunikace.

2.2.2.2 Útok na skryté služby

```
# torsocks ssh root@msydgstlz2kzerdg.onion
RSA key fingerprint a7:93:84:a6:97:fa:25:65:77:c9:58:bb:fe:8e:e2:2f
# ssh root@ahmia.fi
RSA key fingerprint a7:93:84:a6:97:fa:25:65:77:c9:58:bb:fe:8e:e2:2f
```

Obrázek 2.5: Útok na skryté služby [18]

Cílem tohoto útoku je využití chyby, díky které můžeme získat důležité informace o Tor webových stránkách nebo skrytých službách. K tomuto útoku

lze využít službu SSH, která slouží ke vzdálenému připojení. [29] Pokud se stejná služba použije na anonymní adresu provozovanou pomocí Tor služby i na veřejnou IP adresu, pak lze díky otisku, který si tato služba vytváří, odhalit pravou adresu. Názorná ukázka je na obrázku 2.5.

Analýza obrany proti odhalení

Tato kapitola navazuje na předešlou a popisuje zde obranné mechanismy proti útokům, které byly zanalyzovány v minulé kapitole. Opět popíšeme obranné mechanismy proti obecným problémům anonymizérů a následně i obranu nástroje Tor. V závěru kapitoly doplním doporučení pro zachování si anonymního užívání prohlížení internetu.

3.1 Obrana proti nedůvěryhodným prostředníkům

Při obraně proti tomuto typu útoku v nástrojích typu proxy server nebo VPN by bylo vhodné, kdybychom si tyto sítě spravovali sami. To by nám ovšem mohlo zařídit anonymitu jen v případě, že by nebyl znám vlastník těchto sítí, což nemusí být úplně jednoduché. Jednodušší variantou by mohl být pronájem některé ze sítí, přičemž úhrada ceny služby by byla prováděna prostřednictvím anonymní platby, například bitcoiny, které k tomuto účelu slouží. Příkladem takové stránky může být <https://www.privateinternetaccess.com/pages/gift-card-payment>.

3.2 Obrana proti MITM

Jako obranu proti tomuto útoku lze využít certifikátů. Připojení mezi dvěma systémy se ověří pomocí těchto certifikátů. [30] Útočník, který stojí mezi spojením dvou systémů, by mohl opět zachytit a zaslat vlastní certifikát, a pokud by klienti nekontrolovali podvrhy certifikátů, opět by došlo k útoku. Proti tomuto slouží takzvané důvěryhodné certifikační autority (CA), které kontrolují certifikáty, digitální podpisy a klíče klientů [30]. Při jejich neplatnosti nebo nedůvěře jsou odmítnuty a ke komunikaci nedojde. Další možnou obranou je využití infrastruktury veřejných klíčů, kde klient má na svém systému uložený svůj privátní klíč určený k dešifrování a uživatelé, kteří s ním chtějí komunikovat, šifrují zprávy pomocí jeho veřejného klíče.

3.3 Obrana proti modifikaci dat

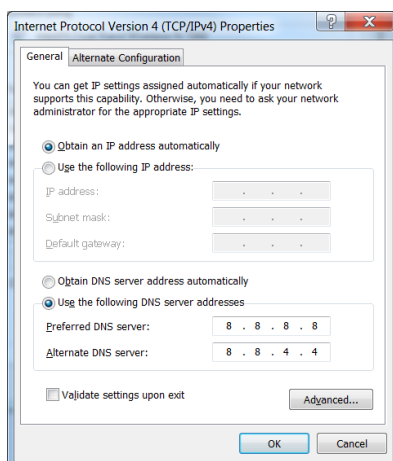
Jedním ze způsobů, jako obranných mechanismů, se používají kontrolní aplikace, které zajišťují integritu a spolehlivost dat, která jsou přijímána. [31] Při jejich příjmu se nejdříve zkontroluje, jestli neobsahují nějaké pochybné části kódu. Pokud se kód jeví jako škodlivý, pak je paket s těmito daty zahozen. Toto vyžaduje velkou pečlivost, protože někdy se nám může kód jevit jako pochybný a přitom nemusí být vůbec škodlivý.

3.4 Obrana proti DNS leak

Jak již víme, operační systém může používat výchozí DNS servery. Proto je nutné zajistit, aby byly využity přesně ty, které potřebujeme. Toto lze zajistit nastavením statické IP adresy některého z DNS serverů. [32] Například v operačním systému Microsoft Windows 7 toho lze docílit nastavením v příkazové řádce pomocí programu `netsh.exe`, ukázka na výpisu 3.1, kde je nastavení pro připojení pomocí ethernetu. Možné nastavení je i pomocí vizuálního nastavení ve vlastnostech síťové jednotky, kde vybereme vlastnosti nastavení protokolu IPv4 nebo IPv6 a následně v políčku pro nastavení DNS serveru zvolíme námi preferované adresy, jak je zobrazeno na obrázku 3.1.

```
netsh interface ip add dns name="Local Area Connection" addr  
=8.8.4.4 index=1  
netsh interface ip add dns name="Local Area Connection" addr  
=8.8.8.8 index=2
```

Listing 3.1: Nastavení DNS z příkazové řádky



Obrázek 3.1: Nastavení DNS

Pro ověření, jestli jsme chráněni proti tomuto typu útoku, je možné využít

některých webových služeb, které jsou určeny právě k tomu, aby zjistily, zda-li je pro nás tento útok potencionální hrozbou, či nikoliv.

3.5 Obrana proti (D)DoS

Jako obrana proti tomuto druhu útoku je využíváno více způsobů. Jednoduchý, ale naivní způsob obrany, je mít hlídací uzel, který kontroluje délku fronty dotazů na daný server. Pokud počet dotazů přesáhne námi stanovené číslo, tak celý okruh spojení ukončí. [33] Tato obrana je možná při použití anonymizérů typu Tor nebo I2P, protože pokud by bylo ukončeno spojení mezi námi a proxy serverem nebo VPN, tak by opět bylo jasné, že skrze ně komunikujeme. Pokud by provozovatelé těchto anonymizérů měli své záložní servery, přes které by byla vedena komunikace, dalo by se útoku zabránit. Dalším způsobem obrany, který byl navržen pro vývojáře projektu Tor, je implementace vlastního správce komunikačních okruhů přímo do projektu Tor, který při nedostatku paměti, tedy zahlcení, vybere nejméně používaný uzel v okruhu a vede se komunikace skrze něj.

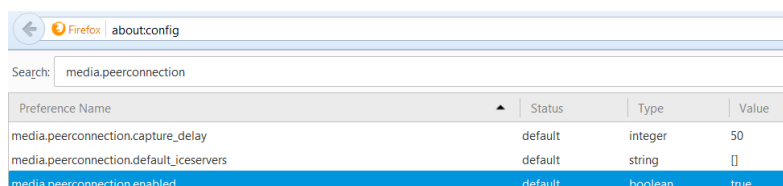
3.6 Obrana vývoje a implementace

Anonymizéry Tor a I2P popsané v předešlé kapitole jsou open-source projekty, to znamená, že je možné se podílet na vývoji těchto nástrojů. Jak jsme si řekli, tak tento typ útoků spočívá ve vytváření například zadních vrátek pro pozdější útoky nebo zanechávání chyb odhalující naši pravou identitu. Jako obrana pro tyto nástroje se dělá bug-tracking, což je kontrola, která odhalí případné chyby v kódu. Chyby jsou nahlášený, například prostřednictvím stránky pro hlášení chyb v nástroji Tor <https://trac.torproject.org/projects/tor/>. Pro zbylé popsané anonymizéry se dělají kontroly nastavení sítí. Existují skupiny, které se snaží napadat tyto nástroje a hlásit jejich chyby provozovatelům s případným řešením, jak danou chybu opravit.

3.7 Obrana proti Action Script

Obrana je u tohoto typu útoku závislá na klientovi. Jelikož útok spočívá ve spouštění kódu na klientské straně, je nutné, aby byl uživatel v tomto případě obezřetný a využíval například rozšíření do prohlížeče, která spouštějí kód jen na důvěryhodných webových stránkách. Ty jsou definovány buď aplikací samostatně nebo individuálně uživatelem. [34] Příkladem takového doplňku může být NoScript, který uživatele chrání před různými typy útoků, kde jedním z nich jsou právě doplňky jako Flash nebo QuickTime. Detailnější popis rozšíření do prohlížečů a jejich obrana bude popsána v kapitole 5.

3.8 Obrana proti cross-site útoku



The screenshot shows the Firefox 'about:config' page. A search bar at the top contains the text 'media.peerconnection'. Below the search bar, a table lists configuration preferences. The table has four columns: 'Preference Name', 'Status', 'Type', and 'Value'. Three rows are visible, with the last row, 'media.peerconnection.enabled', highlighted in blue.

Preference Name	Status	Type	Value
media.peerconnection.capture_delay	default	integer	50
media.peerconnection.default_iceservers	default	string	[]
media.peerconnection.enabled	default	boolean	true

Obrázek 3.2: Vypnutí WebRTC pro Mozilla Firefox

Jako obranný mechanismus lze využít stejného principu jako v předešlém případě. Opět je možné doinstalovat do prohlížeče doplňky, které nás budou chránit. Pokud bychom samozřejmě nechtěli doinstalovávat žádný z doplňků nebo by nám v tom bránila oprávnění, tak stejného cíle lze dosáhnout i pomocí vlastního nastavení v prohlížeči. V prohlížeči Mozilla Firefox toho lze docílit tak, že do URL řádky napíšeme `about:config`, kde se nám objeví bezpečnostní upozornění, které potvrdíme. Objeví se nám okno s různými typy konfigurací. V tomto okně se nachází i vyhledávací řádek, do kterého zadáme řádek s funkcí, kterou chceme změnit. Například pro zákaz WebRTC napíšeme do vyhledávacího řádku `media.peerconnection`. Po jeho vyhledání nastavíme řádek s hledanou položkou na `false`. Pro ukázkou viz obrázek 3.2. Avšak ne každý prohlížeč má tuto možnost. Například v prohlížeči Google Chrome pro osobní počítače nemůžeme vypnout WebRTC pomocí nastavení (platí pro verzi 65). Zde, jako jediná možnost, je doinstalování příslušného rozšíření. V kapitole 5 bude popsána realizace doplňku pro tento prohlížeč, která nám řídí funkcionalitu WebRTC.

3.9 Obrana Tor sítě

Útoky popsané pro tento nástroj byly záměrně vybrány, protože se mi jeví mezi ostatními jako jedny z nejzajímavějších a více moderních. Proto by bylo vhodné si ukázat, jak se tomu lze bránit.

3.9.1 Obrana proti korelační analýze

Mohlo by se zdát, že tento útok patří mezi složitější útoky, avšak jeho obrana není ve výsledku až tak složitá. Jako jeden typ obrany lze použít plánovač pro přeposílání určitého množství paketů za sekundu z výchozích bran. [35] Druhý typ obrany spočívá v tom, že jsou do sítě Tor přeposílány pakety v určitých intervalech, které neobsahují žádnou věcnou informaci, aby bylo obtížné zjistit, s kterými uzly v síti probíhá přesně komunikace. Pro tuto obranu se bere v úvahu fakt, že útočník není schopen zanalyzovat celou síť, ale že i on má některé své limity.

3.9.2 Obrana skrytých služeb

V útoku na skryté služby byla využita fingerprinting technika. Existují tři způsoby, jak této technice zabránit, jedním je podvrhnutí informací například o operačním systému nebo verzi prohlížeče. Jiným způsobem je náhodná hodnota, která je přidána do zpracování výsledné informace. Třetí možností je zablkování přístupu k dané informaci za pomoci rozšíření do prohlížeče blokující skripty, avšak tento způsob obrany nám může zabránit úplnému využití aplikace, či přístupu k webové stránce. Touto problematikou se detailněji zabývá práce [36].

3.10 Obecné doporučení zachování anonymity

Doporučením pro zachování anonymity je dodržování některých pravidel. Jedním z těchto pravidel je používáním vhodného webového prohlížeče pro anonymní prohlížení webových stránek. Příkladem může být aplikace Tor Browser, kde se jeho vývojáři snaží co nejvíce zabránit útokům. Například proti (D)DoS útokům využívají toho, že se nám přibližně každých 15 minut mění okruh, přes který je naše komunikace vedena. Samozřejmě tento typ obrany chrání i proti jiným typům útoků, jako je například korelační analýza. Správci tohoto projektu se snaží nabízet lidem peněžitou odměnu za hlášení chyb v tomto prohlížeči, odměny jsou rozděleny do skupin, podle úrovně hrozby. Dále prohlížeč obsahuje doplňky třetích stran, které slouží například k zabránění spouštění action skriptů, omezují funkce JavaScriptu, apod. Určitě je vhodné se ptát, jestli nemůže být mezi vývojáři někdo, kdo by se snažil zanechávat chyby v programu. Tím si ale nemůžeme být jistí u žádného z programů. Tento prohlížeč jsem vybral z důvodu, že je testován širokou komunitou, a spíše se nalezne rychleji chyba v tomto programu, než v některém z méně používaných.

Dalším pravidlem je snaha o komunikaci skrz HTTPS, aby naše komunikace byla vždy šifrovaná. V neposlední řadě je důležité nesdělovat žádné osobní údaje, snažit se nedát najevo nic o svých osobních údajích. Nepřihlašovat se nikam, co by mohlo být spojováno s naší osobou, neboť služby, do kterých se přihlašujeme, můžou zaznamenávat adresy, přes které jsme se připojili.

Je nutné abychom si pečlivě rozmysleli, zda-li chceme stahovat některé dokumenty. Opět bychom mohli stáhnout něco, co může být spojováno s naší osobou. Stejně tak můžou aplikace, přes které dokumenty čteme, vyžadovat aktualizace a může zde docházet k zaznamenávání adresy nebo dokonce může přenos probíhat přes jiný port, který odhalí naši reálnou IP adresu. To stejné platí i pro pouštění videí. Ani zde si nemůžeme být jistí, co přesně přehrávač videa dělá.

Vedle používání programů k zachování anonymity je také vhodné vést veškerou svoji komunikaci skrze vlastní zařízení, které funguje jako směrovač.

3. ANALÝZA OBRANY PROTI ODHALENÍ

Je možné použít některé z IoT zařízení jako tzv. Tor směrovač. Ten nám zajišťuje, že komunikace přes veškeré porty půjde právě přes toto zařízení.

Dalším obranným faktorem je používání virtuálních systémů. Pokud by se někomu povedlo nahrát do našeho systému program, který by se o nás snažil zjistit co nejvíce, bylo by vhodné, kdyby tento program opět neměl přímý přístup k našemu počítači, ale pouze do nějakého omezeného prostředí. Dokonce existují volně dostupné operační systémy¹, které jsou právě vytvořené za účelem co největší bezpečnosti a udržení naší anonymity.

¹<https://tails.boum.org/>

Demonstrace útoku

V této kapitole nejdříve popíší demonstraci útoku na anonymitu, respektive odhalení pravé IP adresy, pomocí WebRTC. Aby nedocházelo k předčasnému zmatení, rozhraní WebRTC primárně neslouží k odhalení, jen se využívá jeho funkcí k provedení útoku. Nejprve popíší, co je rozhraní WebRTC a následně jeho funkce, které jsou využívány k útoku na anonymního uživatele. Dále bude následovat popis útoku a jeho implementace, psaný v jazyce JavaScript, který bude pro lepší představu doprovázen obrázky.

4.1 WebRTC P2P protokol

WebRTC je open-source projekt, využívající jednoduchá API pro P2P komunikaci, nahrávání videí a zvuků, k uskutečnění hovorů nebo videohovorů mezi uživateli. Využívá se pro vývoj aplikací pro webové prohlížeče, mobilní platformy a IoT zařízení k uskutečnění komunikace za využití běžných protokolů. K použití tohoto rozhraní není třeba, aby se doinstalovávaly speciální balíčky, záleží na prohlížeči a platformě, jsou-li podporované. Mezi podporovanými prohlížeči a platformami jsou:

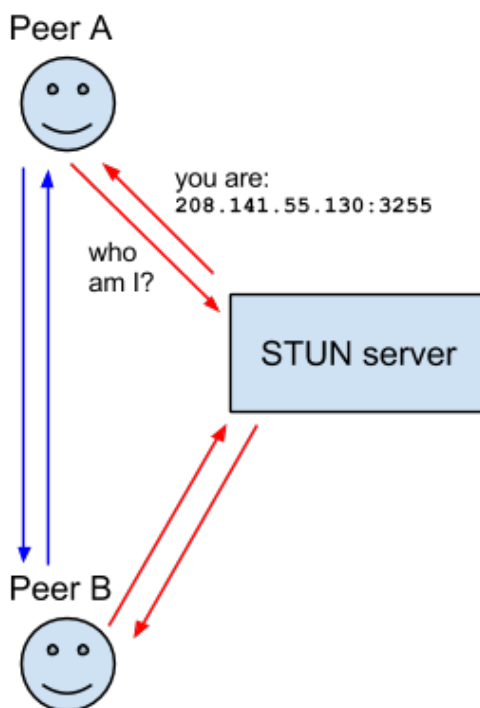
- Google Chrome
- Mozilla Firefox
- Opera
- Android
- iOS

Na ostatních prohlížečích a platformách není zaručena funkčnost. Více o WebRTC lze nalézt v [37].

4.1.1 ICE

Aby mohlo dojít ke spojení mezi dvěma klienty, je k tomu potřeba zvláštní rozhraní. Toto rozhraní se využívá z důvodu, že existují sítě, které mají zakázané přímé připojení ke vzdálenému systému. WebRTC proto obsahuje Interactive Connectivity Establishment rozhraní, které umožňuje webovým prohlížečům připojit se k ostatním uživatelům. [38] Snaží se najít alternativní řešení pro komunikaci, pokud nastavení firewallu nebo přepínače tuto akci nepovolují. Je schopno také přiřadit unikátní IP adresu, pokud ji zařízení nemá, skrze kterou poté komunikuje s okolním světem. ICE používá některé z technik, které jsou popsány níže.

4.1.2 STUN



Obrázek 4.1: Komunikace se STUN serverem [37]

Jedna z technik, které ICE využívá, je protokol Session Traversal Utilities for NAT (STUN), který zjišťuje veřejnou IP adresu a omezení ve směrovači, která by zabránila přímému spojení mezi vrstevníky.

Klient nejdříve zašle dotaz na STUN server a čeká na odpověď. V té přijde veřejná IP adresa klienta, nezávisle na tom, jakou má v síti adresu za NAT směrovače [38], viz obrázek 4.1.

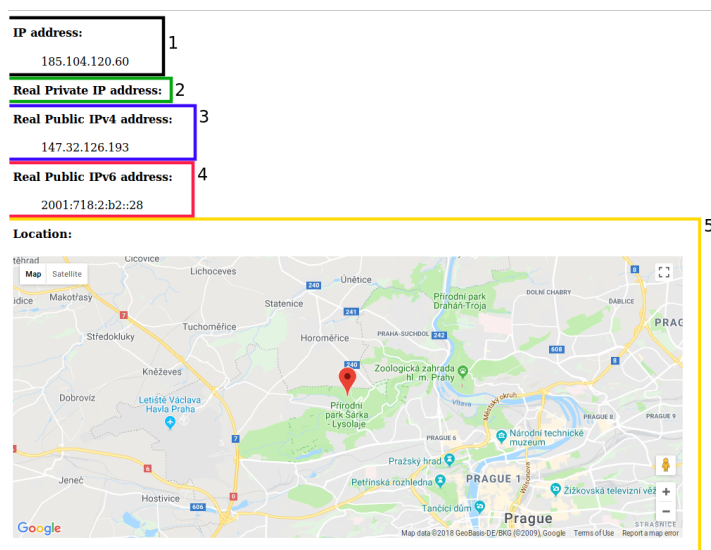
4.1.3 NAT

Pokud nemáme veřejnou IP adresu, je nutné komunikovat skrze směrovač. Ten musí ale vědět, jakému zařízení se má odpověď vrátit. Proto jsou všechna zařízení připojena v privátní síti směrovače. Aby směrovač věděl, kam poslat odpověď, používá se systém pro překlad adres v lokální síti do jedné unikátní veřejné adresy, zvaný Network Address Translation (NAT). [38]

4.1.4 SDP

Jakmile se spojíme se zařízením, je nutné, abychom si dohodli nějaká pravidla, jak spolu budeme komunikovat. K tomu se používá standard Session Description Protocol (SDP), který domluví a definuje popis multimediálního obsahu spojení, jako rozlišení, formáty nebo šifrování, aby oba komunikující znali své parametry pro přenos a dekodování posílaných dat. [38] Jedná se v podstatě o metadata popisující obsah a nikoliv samotný obsah médií.

4.2 Popis útoku



Obrázek 4.2: Webová stránka s útokem

Aby útok mohl být proveden, je nutné, aby oběť měla ve svém prohlížeči povolené spouštění JavaScriptového kódu a P2P komunikaci. Vše je demonstrováno ve virtuálním stroji. Použitá distribuce je Arch Linux, prohlížeč byl

4. DEMONSTRACE ÚTOKU

použít Chromium a pro simulaci anonymní IP adresy byl použit anonymizér Tor. Detailnější popis je zobrazen ve výpise 4.1.

```
$uname -a
Linux arch 4.15.12-1-ARCH #1 SMP PREEMPT Wed Mar 21 15:14:56 UTC
    2018 x86_64 GNU/Linux
$chromium --version
Chromium 65.0.3325.181
```

Listing 4.1: Výpis verzí programů

```
var is = {iceServers: [{urls: "stun:stun4.l.google.com:19302"}]};
var conn = new window.RTCPeerConnection(is) ||
new window.webRTCPeerConnection(is) ||
new window.mozRTCPeerConnection(is);
if (!conn) {
    console.log("cannot create connection");
    return;
}
conn.onicecandidate = function(ice) {
    if(ice.candidate)
        parseAndWriteIP(ice.candidate.candidate);
};
conn.createDataChannel("");
conn.createOffer(function(result) {
    conn.setLocalDescription(result, dummyFoo, dummyFoo);
}, dummyFoo);
```

Listing 4.2: Tělo funkce payload

Pro demonstraci útoku jsem vytvořil webovou stránku, která pouze ukazuje IP adresu uživatele, pod kterou se hlásí, jeho reálnou adresu a přibližnou pozici, za použití Google Maps. Jak lze vidět na obrázku 4.2, v kolonce číslo 1, označené černou barvou, se nachází adresa, pod kterou se uživatel skrývá. Například podle <https://www.whois.net/> nebo UNIXového příkazu `whois` je možné dohledat, že se server, přes který je spojení vedeno, nachází v hlavním městě Velké Británie, v Londýně. V dalším rámečku zelené barvy, označeném číslem 2, se může nacházet adresa lokální sítě, ve které uživatel může být, pokud nemá přidělenou vlastní veřejnou adresu. V demonstraci se v této sekci nenachází žádná IP adresa, protože bylo využito veřejné adresy školní sítě ČVUT Silicon Hill na Strahově v Praze. Rámec číslo 3 modré barvy obsahuje reálnou veřejnou IP adresu, přesněji IPv4. Tato adresa může mít více významů. Buď se může jednat o veřejnou adresu, která je přímo přidělena uživateli, nebo to může být adresa směrovače, která patří poskytovateli internetu, pod kterou uživatel vystupuje. V dalším rámečku červené barvy, číslo 4, je adresa, která se nemusí vždy objevit. Jedná se o adresu IPv6, pod kterou nemusí vystupovat uživatel či poskytovatel internetu. Na konci stránky, ve žlutém obdélníku číslo 5, je přibližná poloha, která se určí podle první získané adresy. Díky značce na mapě je možné si všimnout toho, že poloha je opravdu nepřesná. Terčík se nachází u ulice Truhlářka, v místě Praha-Nebošice, avšak reálná po-

loha při demonstraci byla na kolejkách Strahov. Tato dvě místa jsou od sebe vzdálená přibližně 7 kilometrů.

Webová stránka s útokem je napsána v jazyce HTML a JavaScript. Ke zjištění adresy se využívá STUN server, který je provozován společností Google a je volně dostupný veřejnosti. Webová aplikace se k tomuto serveru připojí a v odpovědi, kterou od něho získá, si přečte pravou identitu uživatele. Tuto komunikaci zajišťuje rozhraní WebRTC.

```
var regex_ipv4 = /\b((25[0-5]|2[0-4]\d|1\d{2}|[1-9]\d|\d)
  ((\. (25[0-5]|2[0-4]\d|1\d{2}|[1-9]\d|\d)){3}))\b/;
var tmp_exec = regex_ipv4.exec(ip);
var address = null;
if (tmp_exec) {
  var address_tmp = tmp_exec[1];
  var converted = convertAddressToInt(address_tmp);
  if (converted === null)
    return;
  if ((converted >= 167772160 && converted <= 184549375) || (
    converted >= 2886729728 && converted <= 2887778303) || (
    converted >= 3232235520 && converted <= 3232301055))
    writeIP(address_tmp, true, false);
  else {
    address = address_tmp;
    writeIP(address_tmp, false, false);
  }
}
```

Listing 4.3: Tělo funkce zpracovávající odpověď 1. část

Ve funkci `payload` (viz výpis 4.2) je příklad toho, jak lze této komunikace dosáhnout. Celé nastavení je zpracováno třídou `RTCPeerConnection`. Od této třídy jsou poděděné třídy `webRTCPeerConnection` určená pro prohlížeč Google Chrome a `mozRTCPeerConnection` pro Mozilla Firefox. Tyto dvě třídy jsou specifické pro prohlížeče z důvodu vlastního zakomponování rozhraní. Konstruktor těchto tříd přijímá 1 povinný parametr. Jelikož chceme pouze využít připojení ke STUN serveru, stačí nám, pokud se zde nachází pouze jeho „url“. Dříve bylo nutné, pro prohlížeče Chrome a Opera, přidat jako druhý parametr kanál, přes které spojení bude vedeno, dnes už to ale není třeba. Pokud se vytvoření třídy povedlo, je nutné vytvořit callback pro kandidáta ICE. Vytvořený objekt má připravený atribut `onicecandidate`, který slouží jako funkce, která se zavolá, pokud se objeví nějaká odpověď ve spojení. Pro uvedení příkladu se v demonstraci volá funkce k získání IP adresy z odpovědi a jejího následného vypsání. Aby tato funkce byla zavolána musí se nejdříve vytvořit takzvaná nabídka pro spojení, která potřebuje datový kanál, jenž se vytvoří pomocí metody `createDataChannel`. Ta jako parametr přijímá řetězec specifikující zasílaná data. V ukázce je jako parametr předán prázdný řetězec, protože se žádná data neposílají a čeká se pouze na odpověď. Následně se pomocí metody `createOffer` vytvoří nabídka. Jejím parametrem je odkaz na funkci, přijímající parametr specifikující domluvené

nastavení, potřebné k lokální konfiguraci SDP. Dalšími parametry jsou funkce pro úspěch a neúspěch, v demonstrovaném případě to jsou pouze takzvané „dummy funkce“, které nedělají vůbec nic.

V první části funkce zpracovávající přijímanou odpověď (viz výpis 4.3), se snaží dostat z odpovědi adresu IPv4. K nalezení této adresy se používá regulární výraz. Pokud se v dané odpovědi nachází, nastavá rozřazení, jedná-li se o veřejnou adresu či privátní. Podle standardů jsou rozsahy privátní adresy následující [39]:

- 10.0.0.0 – 10.255.255.255
- 172.16.0.0 – 172.31.255.255
- 192.168.0.0 – 192.168.255.255

Zbylé se považují za veřejné. K efektivnějšímu porovnání, o jaký druh se jedná, je adresa převedena na celé bezznaménkové číslo, neboť je výpočetně rychlejší porovnání dvou čísel než porovnávání pomocí regulárního výrazu. Pro konverzi je implementována vlastní funkce, která si nejdříve rozdělí adresu na čtyři oktety a postupným přičítáním a bitovým posunem je získána výsledná číselná hodnota. Po zjištění o jaký druh se jedná, se zavolá funkce `writeIP`, která zapíše adresu do předem připravených kontejnerů, popsanými značkami jazyka HTML.

```
var regex_ipv6 = /((?<=\s|^)(([a-f\d]{1,4}:){7}[a-f\d]{1,4})
(?=$|\s)|((?<=\s|^):([a-f\d]{1,4}){1,6})(?=$|\s)|((?<=\s|^)([a-f\d]{1,4}:){1,6})(?=$|\s)|((?<=\s|^)([a-f\d]{1,4}:){1,2}([a-f\d]{1,4}){1,5})(?=$|\s)|((?<=\s|^)([a-f\d]{1,4}:){1,3}([a-f\d]{1,4}){1,4})(?=$|\s)|((?<=\s|^)([a-f\d]{1,4}:){1,4}([a-f\d]{1,4}){1,3})(?=$|\s)|((?<=\s|^)([a-f\d]{1,4}:){1,5}([a-f\d]{1,4}){1,2})(?=$|\s)|((?<=\s|^)([a-f\d]{1,4}:){1,6}([a-f\d]{1,4}){1})(?=$|\s)|((?<=\s|^)([a-f\d]{1,4}:){1,6})(?=$|\s))/i;
tmp_exec = regex_ipv6.exec(ip);
if (tmp_exec) {
    address = tmp_exec[1];
    writeIP(tmp_exec[1], false, true);
}
if (!ipSet && address) {
    ipSet = {'ip': address};
    getLocation();
}
```

Listing 4.4: Tělo funkce zpracovávající odpověď 2. část

Ve druhé části funkce, viz výpis 4.4, se hledá, zda se v odpovědi nachází IPv6 adresa. Toto je mnohem obtížnější, protože sice neplatí rozdělení na veřejnou a privátní část, ale možností, jak zapsat tuto adresu, je více, díky jejímu možnému zkracování. Operace probíhá stejně jako v minulé ukázce, opět se hledá pomocí regulárního výrazu, který je mnohem delší než pro starší

metodu. Při úspěšném ověření, že se adresa v odpovědi nachází, se opět vypíše do příslušného kontejneru. Na konci těla se asynchronně vykoná funkce pro zobrazení přibližné polohy uživatele za podmínky, že nalezená adresa je první, která využívá globální proměnnou `ipSet`, ve které je uložena daná adresa.

Pro použití Google Maps k zaměření polohy jsou nutné 3 kroky. První z nich je registrování klíče pro vlastní projekt využívající této mapy. Dále je nutné přidat do hlavičky webové stránky zdroj knihovny, která obsahuje třídy pro zobrazení mapy. Posledním důležitým faktorem je poloha místa, kam se má mapa zaměřit pomocí geografického souřadnicového systému. V ukázkové demonstraci ale máme pro zjištění polohy pouze IP adresu, která nepatří do souřadnicového systému. Abychom mohli polohu zjistit díky IP adrese, je nutné z ní souřadnice získat. K tomu slouží například webové stránky, které určí přibližnou polohu pro danou adresu. Tyto stránky zjistí přibližnou polohu díky světovým geolokačním službám, které využívají databáze, do které přispívají svými informacemi o svých adresách poskytovatelé Internetu. Pro demonstraci je využita stránka <https://ipapi.co/>.

```
$.getJSON('https://ipapi.co/' + ipSet.ip + '/json',
    setMap);
```

Listing 4.5: Příkaz k získání polohy

Příklad použití, za pomoci rozhraní pro JavaScript jQuery, které stejně jako Google Maps potřebuje do hlavičky webové stránky přidat zdroj knihovny, je možné vidět ve výpisu 4.5. Po získání polohy je zavolána funkce `setMap`, ukázka výpis 4.6, která přijímá parametr s informacemi o adrese, ve formátu pro výměnu dat JSON.

```
var pos = {lat: json.latitude, lng: json.longitude};
var map = new google.maps.Map(document.getElementById('
    map'), {
    zoom: 12,
    center: pos
});
var marker = new google.maps.Marker({
    position: pos,
    map: map
});
```

Listing 4.6: Tělo funkce nastavující mapu a zaměření

Mapa přijímá v konstruktoru parametr s nepovinnými údaji, avšak aby bylo možné mapu vycentrovat, předává se jí objekt s atributem `center`, popisující zeměpisnou šířku a délku. Pro vytvoření značky s polohou slouží třída `Marker`, která v konstruktoru přijímá opět objekt s atributem popisujícím polohu a objekt s mapou, na které se má nacházet.

Realizace obrany

Jak jsem již zmiňoval v kapitole 3, prohlížeč Google Chrome nemá ve svém nastavení možnost vypnutí WebRTC. Závěrečná kapitola tedy obsahuje popis rozšíření pro prohlížeč Google Chrome, který slouží jako defenzivní prvek proti útoku na anonymitu pomocí rozhraní WebRTC. Jako přidaná hodnota tohoto rozšíření je i nastavení spouštění JavaScriptu. V první části této kapitoly se nachází definice, co je to rozšíření pro prohlížeč, jeho důležitých aspektů pro implementaci rozšíření a funkcionalit, které jsou v realizaci použity, následně popíši implementaci.

5.1 Rozšíření

Aplikace umožňující jednoduché nastavení našeho prohlížení jsou vskutku užitečné, je tedy vhodné, stejně tak jako tomu bylo u útoku, vědět o nich vše nutné, abychom nepřehlédli nějaký detail, který by nás mohl poškodit.

5.1.1 Co je to rozšíření

Rozšíření je malá aplikace přizpůsobující prohlížení webových stránek. Umožňuje uživateli individuální nastavení pro prohlížení. Pro jednoduchost jsou postavena na webových technologiích jako HTML, JavaScript a CSS. [40] Jedná se v podstatě o webovou aplikaci, která zaštiťuje tato nastavení.

Každé rozšíření má pouze jeden účel, který je jasně definovaný a jednoduchý k pochopení. Může obsahovat více komponent a velkou škálu funkcionality, které se stále drží jediného účelu.

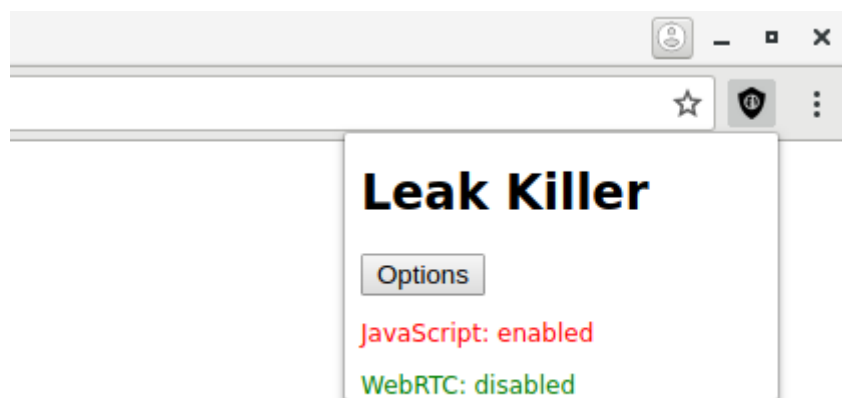
5.1.2 Konfigurační soubor `manifest.json`

Soubor `manifest.json` je nutný pro každé rozšíření. Nachází se zde základní informace o programu, jako je například název, popis, verze nebo akce, které

nastanou po vyvolání okénka aplikace. [41] V jednoduchosti se jedná o konfigurační soubor, který nám nastaví vše potřebné pro fungování. Kupříkladu se v tomto souboru nastavuje ikona rozšíření, cesta ke kódu s popup oknem nebo různá bezpečnostní nastavení, která jsou nutná pro přístup k některým z podpůrných tříd. Ve své implementaci používám lokální úložiště, pro jeho zpřístupnění je vyžadováno udělení oprávnění od aplikace. Na stránkách <https://developer.chrome.com/apps/manifest> je podrobnější popis různých nastavení.

5.1.3 Popup okno

Pro interakci mezi uživatelem a aplikací se používá takzvané popup okno. [41] Je to HTML soubor, který se zobrazuje ve speciálním okně po kliknutí na ikonu rozšíření, viz obrázek 5.1. Funguje stejně jako každá jiná webová stránka s tím rozdílem, že v tomto souboru nemůžou být z bezpečnostních důvodů použity přímo JavaScriptové kódy.



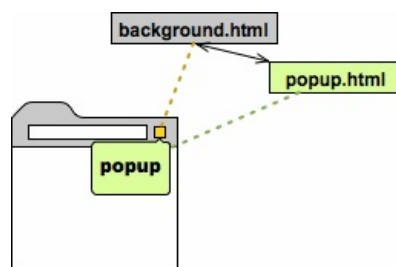
Obrázek 5.1: Popup okno rozšíření

5.1.4 Skript na pozadí

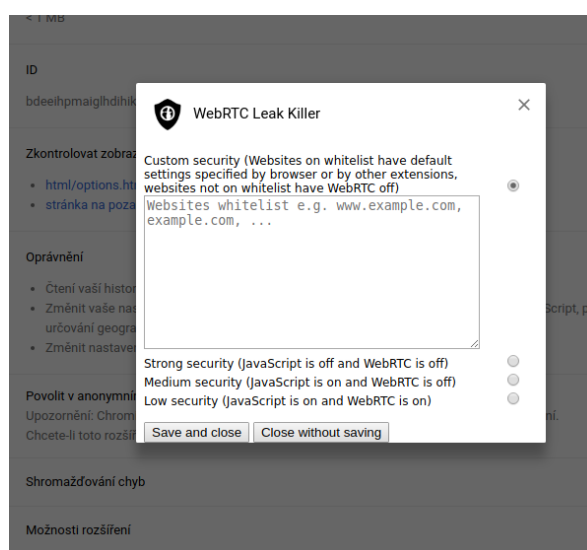
K nastavení různých stavů pro specifické stránky je potřeba využít některé funkce, které by HTML samostatně nemohlo vykonat. K tomu se využívá jazyk JavaScript, ale protože ho nelze použít v popup okně, tak je nutné vytvořit samostatný soubor, ve kterém jsou funkce v tomto jazyce implementovány. Poté je možné volat tyto funkce z vyvolávacího okna. Obrázkový popis viz 5.2.

5.1.5 Okno s nastavením

Rozhraní pro rozšíření nabízí možnost implementace vlastního okna s nastavením, nezávislé na ostatních skriptech. Jedná se opět o HTML soubor, který



Obrázek 5.2: Architektura rozšíření [41]

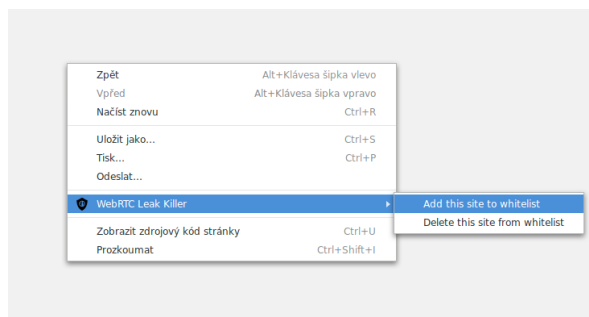


Obrázek 5.3: Okno s nastavením

má zajišťovat specifické nastavení definované uživatelem. Například v mém případě se jedná o nastavení různých bezpečnostních úrovní, které definují, na jaké stránce bude vypnuto WebRTC nebo zda bude vypnuto všude atp. Pro ukázkou takového okna viz obrázek 5.3.

5.1.6 Kontextové menu

Pokud chceme uživateli usnadnit některou z operací s rozšířením, je možné vyvolat kontextové menu pomocí pravého tlačítka myši na některé z webových stránek. Po zobrazení menu, kde se nachází typické operace spravované prohlížečem, jako například načíst znovu stránku nebo uložit stránku, se mezi těmito možnostmi nachází i možnosti definované v našem rozšíření. Pro ukázkou viz obrázek 5.4. Kontext může být například označení textu nebo pouhé kliknutí na stránce.

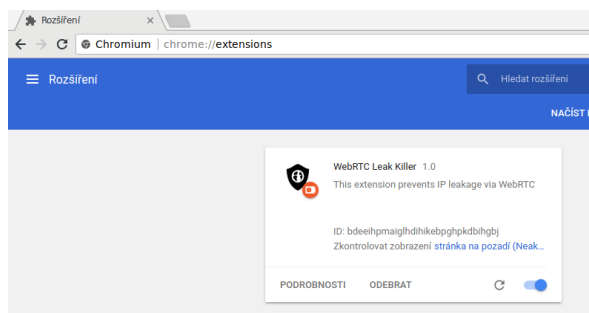


Obrázek 5.4: Kontextové menu

5.2 Použití rozšíření

Rozšíření je navrženo tak, aby práce s ním bylo pro uživatele co nejjednodušší. Aby nebylo porušeno nastavení, které v prohlížeči mohou být před nainstalováním rozšíření, je úroveň zabezpečení nastavena na volbu **custom**.

5.2.1 Instalace



Obrázek 5.5: Okno s rozšířeními

Pro použití rozšíření je nutné buď načíst rozbalený soubor do prohlížeče, nebo ho nainstalovat z internetového obchodu Chrome.

Instalace rozšíření do prohlížeče pomocí načtení lze jednoduše provést tak, že v prohlížeči do URL okna napíšeme **chrome://extensions**. V tomto okně se nám zobrazí námi instalované rozšíření, pokud nějaké máme. V pravém horním rohu se nachází možnost vypnutí či zapnutí režimu pro vývojáře. Po jeho aktivování se nám zobrazí 3 tlačítka. Jedno z těchto tlačítek je pojmenováno jako **Načíst nerozbalené**. Klikneme na něj a vybereme složku, kde jsou zdrojové kódy rozšíření společně s konfiguračním souborem **manifest.json**. Poté se nám zobrazí v okně námi přidáné rozšíření, viz obrázek 5.5.

Další možností instalace rozšíření je pomocí internetového obchodu Chrome. Adresa <https://chrome.google.com/webstore/category/extensions> nás přímo

navede na internetový obchod Chrome. K odkazu se také můžeme dostat v nastavovacím okně rozšíření prohlížeče na `chrome://extensions`. Na internetové stránce obchodu lze rozšíření nalézt pod názvem **WebRTC Leak Killer**. Instalace probíhá tak, že po jeho nalezení klikneme na tlačítko, vpravo od názvu rozšíření, **Přidat do Chromu** a potvrdíme oprávnění, které rozšíření požaduje.

5.2.2 Nastavení

Po úspěšné instalaci je nutné si toto rozšíření nastavit. K nastavení lze přejít po kliknutí na tlačítko **Options**, které se nachází v popup okně, vyvolaném po kliknutí na ikonu rozšíření, která se objevila na liště v pravém horním rohu, podobně jako je vyobrazeno na obrázku 5.1. Okno s nastavením je možné i vyvolat na stránce s nastavením rozšíření, kde klikneme na podrobnosti daného rozšíření, a poté klikneme na **Možnosti rozšíření**.

Nastavení rozšíření je rozděleno do 4 bezpečnostních úrovní. Těmito úrovněmi jsou:

- **Low security** – Tato úroveň je nejnižší možná. Na všech internetových stránkách je používání WebRTC povoleno a spouštění JavaScriptového kódu je také povoleno.
- **Medium security** – Střední úroveň zabezpečení umožňuje uživateli procházet webové stránky, aniž by se musel obávat úniků IP adresy pomocí rozhraní WebRTC, v této úrovni je totiž povoleno pouze spouštění JavaScriptového kódu.
- **Strong security** – Při této úrovni hrozí, že některé webové stránky nemusí být zobrazeny, či zpřístupněny, protože vyžadují spouštění JavaScriptového kódu. V tomto nastavení je vypnut JavaScript i WebRTC.
- **Custom security** – Tato úroveň umožňuje operaci jiných rozšíření, která mohou bránit únikům IP adresy. Pro toto nastavení je možné využít seznam stránek, jenž jsou spravovány nastavením prohlížeče nebo jinými rozšířeními. Pro ostatní stránky, které nejsou na tomto seznamu, platí nastavení střední úrovně.

Abychom nemuseli stále kontrolovat náš seznam stránek, nastavení prohlížeče nebo jiných rozšíření a mohli pohodlně zjistit, jaké je aktuální nastavení pro danou stránku, je do rozšíření přidána funkce, která je schopna toto zjistit. Pro zjištění aktuálního stavu stránky v záložce, která je nyní aktivní, stačí kliknout na ikonu rozšíření vyvolávající popup okno. Jak může tento výpis vypadat, lze vidět na obrázku 5.1.

Pro jednoduché přidání stránky na seznam, či odebrání stránky ze seznamu bez nutnosti otevírání nastavovacího okna je dostupné kontextové menu, pro ukázkou viz obrázek 5.4. Pro přidání stránky v aktuální záložce prohlížeče

stačí kliknout pravým tlačítkem myši, a v sekci WebRTC Leak Killer kliknout na požadovanou funkci buď `Add this site to whitelist`, která nám přidá stránku na seznam, nebo `Remove this site from whitelist` pro smazání stránky ze seznamu.

5.3 Implementace

Než jsem začal s implementací, byl nutný návrh a volba jazyka. Jak jsem již zmiňoval, jazyk je dán jednoznačně, protože se vývojáři prohlížeče snaží zachovat jednoduchost při implementaci. Jazyk, resp. jazyky pro implementaci byly využity HTML, CSS a JavaScript. Návrh rozšíření je postaven tak, aby mohl být doplněn o případné nové funkce, které zvýší naši bezpečnost při prohlížení. Struktura rozšíření je rozdělena do 3 souborů. Jeden soubor řídí vše, co nám běží na pozadí, druhý soubor řídí funkci popup okna a třetí soubor řídí nastavovací okno. Detailněji budou popsány níže. Abychom zredukovali duplikování funkcí, je k ostatním souborům přiložen soubor `global.js`, kde se nacházejí funkce, které jsou využívány více soubory. Jelikož při tvorbě rozšíření je v některých funkcích JavaScript omezen, například u funkce `import`, tak to svádí k duplikacím funkcí, avšak i přesto je možné toto obejít a využít funkcí z jiných souborů. Jednou z možností nám přímo nabízí konfigurační soubor `manifest.json`, avšak některé řídicí objekty, definované v konfiguračním souboru, nemají možnost nastavení více souborů se skripty. Je tedy nutné pro tento objekt vytvořit HTML soubor a v hlavičce uvést pomocí HTML značek `<script src="">` soubor s funkcemi jako zdroj.

5.3.1 Chrome API

Chrome nabízí API, které nám usnadňuje nastavení funkce JavaScript nebo WebRTC. V tomto rozhraní se nachází i funkce pro správu úložiště, se kterým může operovat pouze rozšíření.

Abychom byli schopni získat nastavení funkce JavaScript pro specifickou stránku, je nutné jí předat vzor této stránky. To se dělá pomocí regulárního výrazu. Zde může být matoucí, že pokud této funkci předáme například vzor `http://example.com/*`, tak nám nepřijme stránku `example.com`. Rozhraní ale podle vzoru tuto stránku přijme. Vzor si rozdělí do 3 částí:

- **schéma** – Schéma nám definuje protokol, skrze který komunikujeme, nebo zda se jedná o soubor uložený na počítači lokálně.
- **hostitel** – Hostitel je část URL, která se nachází za protokolem až po cestu, resp. port.
- **cesta** – Cesta je vše, co se nachází za lomítkem za hostitelem stránky, například `www.example.com/cesta`.

Zvláštním případem jsou soubory uložené lokálně, protože u nich musí být jasně specifikována cesta a nesmí být použit regulární výraz.

5.3.2 background.js

Tento soubor zastupuje běh skriptů na pozadí. Je volán, když zaregistruje nějakou aktivitu v prohlížeči, jako například přechod na některé z webových stránek nebo vytvoření nové záložky. Pokud uživatel nejeví žádnou aktivitu v prohlížeči, stává se soubor neaktivním.

V první fázi, po instalaci rozšíření, nám nastaví kontextové menu. Pro tvorbu kontextového menu je využito API přímo implementované vývojáři Google Chrome, společně s podpůrnými funkcemi, kterými jsou:

- **setContextMenu()** – Tato funkce slouží k vytvoření položek v menu a nastavení jednotného manipulátoru, který je zavolán po kliknutí na položku. V těle této funkce jsou objekty definující každou položku, které jsou poté přidány pomocí Chrome API. Pro definici je nutné zadat ID položky, titulek, který se objeví v menu, a kontext, ve kterém je spouštěn. Na konci funkce je nastaven manipulátor pro kontextové menu.
- **contextMenuEvent(data)** – Manipulátor potřebuje svoji funkci, která přijímá v parametru objekt s daty, uchovávající informace o položce kontextového menu a záložce, ve kterém bylo kontextové menu vyvoláno. V těle této funkce se rozhodne podle ID položky, jaká operace bude následovat, zda přidání na seznam nebo odebrání.
- **addToWhitelist(data)** – Funkce přijímá jako parametr objekt, který byl předán manipulátoru, aby mohla být zjištěna aktuální adresa záložky. Funkce nejdříve získá seznam stránek z lokálního úložiště rozšíření. V něm se pokusí danou stránku vyhledat. Pokud se již stránka na seznamu nachází, nepřidá se. V opačném případě je přidána na konec seznamu.
- **removeFromWhitelist(data)** – Mazání ze seznamu funguje v principu podobně jako předešlá funkce. Jako parametr je přijímán objekt s daty, který byl předán manipulátoru. Funkce vyhledá, zda-li se stránka nachází na seznamu, a při jeho úspěšném vyhledání je ze seznamu smazána.

Ve druhé fázi tento soubor řídí nastavení pro aktuální stránku záložky. V případě, že se nejedná o úroveň `custom`, nic se neděje. Pokud se jedná o úroveň `custom`, funkce nastaví WebRTC podle toho, zda se stránka nachází na seznamu nebo nikoliv. Pro aktualizaci nastavení je nutné, aby stránka byla znovu načtena.

5.3.3 popup.js

Soubor `popup.js` má jednoduchou funkci. Po načtení popup okna nastaví pro tlačítko `Options` funkci, která zavolá nastavovací okno v novém okně.

Druhou její funkcí je zobrazit aktuální nastavení stránky. Nejdříve funkce získá bezpečnostní úroveň. Pokud se nejedná o `custom` úroveň, nastaví příslušný výpis podle ní. V opačném případě získá informace o nastavení JavaScriptu pro příslušnou stránku, zjistí zda-li stránka se nachází na seznamu a přiřadí díky tomu příslušný výpis. K tomuto souboru náleží i `popup.html`, který má předem definované zobrazení popup okna.

5.3.4 options.js

Tento soubor řídí události vyvolané nastavením. K tomuto souboru patří soubor `options.html`, který definuje, jak nastavovací okno vypadá. Nejdříve funkce nastaví pro každý objekt znázorňující nastavení, její příslušnou funkci, která se zavolá po kliknutí na danou úroveň. Dále se nastaví odpovídající funkce pro tlačítka `Save and close` a `Close without saving`. Následně je nastaveno výchozí nastavení, pokud se jedná o první spuštění nebo nastavení definované uživatelem. Aby toto mohlo fungovat, jsou implementovány pomocné funkce:

- `getOption()` – Funkce slouží k získání aktuální bezpečnostní úrovně. Jejím výstupem je název konfigurace. Při chybném určení nastavení vrací prázdný objekt.
- `setOption()` – Tato funkce nastaví aktuální nastavení při opětovném vyvolání popup okna, definované uživatelem nebo výchozí nastavení po instalaci rozšíření.
- `saveOptions()` – V této funkci se uloží veškeré nastavení, které se v okně nachází. Do lokálního úložiště rozšíření se uloží název bezpečnostní úrovně a stránky uvedené na seznamu. Po uložení se zavře okno s nastavením.
- `loadOptions()` – Princip je načtení nastavení z lokálního úložiště. Pomocí rozhraní spravujícího úložiště se načte název bezpečnostní úrovně a seznam stránek. Pokud se jedná o první spuštění, nastavení je na úrovni `custom`.
- `setPrivacy(opt)` – Tato funkce zajišťuje správné nastavení soukromí definovaného uživatelem, prohlížečem a bezpečnostními úrovněmi rozšíření. Přijímá parametr `opt`, typu `string`, ve kterém se nachází název bezpečnostní úrovně.
- `showCustomConfig(enable)` – Tato funkce určuje, zda bude okno se seznamem aktivní podle parametru `enable`. Okno je aktivní pouze při zvolené úrovni `custom`.

5.3.5 global.js

Soubor `global.js` obsahuje funkce, které jsou využívány více soubory. Tyto funkce jsou:

- `setJavaScript(disabled, pattern)` – Tato funkce, jak už vyplývá z názvu, nastaví používání JavaScriptu pro danou stránku. Jsou přijímány dva parametry. Parametr `disabled`, typu `boolean`, určuje, zda-li má být blokován nebo zapnut. Parametr `pattern`, typu `string`, je vzor pro danou stránku.
- `setWebRTC(disabled)` – Parametr `disabled`, typu `boolean`, indikuje, zda-li má být WebRTC zapnuto, či vypnuto. Nastavení pro toto rozhraní neuvažuje vzor pro specifickou webovou stránku a vypne jej globálně.
- `parseHost(website)` – Výstupem této funkce, která má jako parametr internetovou stránku, je pouze její hostitel.
- `parseProtocol(website)` – Funkce funguje obdobně jako minulá, ze stránky, která je předána v parametru, získá protokol, skrze který je komunikace vedena, a je předán jako výstup této funkce.
- `onWhitelist(url, whitelist)` – Tato funkce určí, zda-li se nachází stránka na seznamu. Parametr `url`, typu `string`, je hledaná stránka. Parametr `whitelist`, typu `string`, je seznam všech stránek, ve kterém se má hledat. Výstupem této funkce je `boolean`, který indikuje výskyt stránky v seznamu.

5.4 Diskuse

Výsledkem této bakalářské práce je demonstrace útoku, která proběhla úspěšně. Dalším úspěchem a výsledkem této práce je implementace rozšíření pro prohlížeč Google Chrome.

Útok je schopen zobrazit reálnou IP adresu uživatele, který používá jakýkoliv anonymizační nástroj. Zobrazí také jeho přibližnou polohu, avšak s patrnou odchylkou. Pro zobrazení přibližné polohy útok využívá jiné webové služby, která časem může být neaktivní. To může postupem času útoku uškodit. Bylo by vhodné provozovat vlastní malou službu, která zjistí přibližnou polohu podle IP adresy.

Implementace rozšíření úspěšně zabrání útokům pomocí rozhraní WebRTC, avšak je důležité vzít v potaz, že nás chrání pouze před jedním typem útoku. Další výhodou tohoto rozšíření je, že se snažím v jeho implementaci spolupracovat i s jinými rozšířeními, které mohou být v prohlížeči nainstalovány nebo uživatelským nastavením prohlížeče.

Nastavení rozhraní WebRTC probíhá pomocí funkcí volané z Chrome API. Původní záměr byl, aby nemuselo být toto rozhraní voláno a nastavily se

návratové hodnoty třídy WebRTC na nedefinovanou hodnotu. Toto však není v rozšíření možné, proto bylo stejného efektu dosaženo pomocí Chrome API rozhraní.

Rozšíření je umístěno na internetovém obchodu Chrome, kde je volně dostupné ke stažení. Aby byla zaručena větší bezpečnost, kód bude zveřejněn v repozitáři a bude možné přispívat návrhy, které zlepší funkcionalitu rozšíření.

Závěr

Cílem této práce bylo seznámit se a zanalyzovat problematiku anonymity v Internetu a obranné mechanismy proti útokům, demonstrovat jeden z vybraných útoků a implementovat rozšíření pro prohlížeč Google Chrome, které brání demonstrovanému útoku.

V první kapitole jsem popsal historii internetu, význam anonymity a pokračoval popsáním komunikačních protokolů a jakými modely se dnešní protokoly řídí. Společně s tím je popsáno, jak komunikace na internetu funguje.

Ve druhé kapitole se zpočátku věnuji stručně anonymizérům, abychom věděli, jak anonymity na internetu dosáhnout. Následně navazuji obecnými útoky proti anonymizérům. Také se věnuji nástroji Tor, protože byl využit v demonstraci útoku.

Třetí kapitola se zabývá obrannými mechanismy proti útokům, které byly popsány v předešlé kapitole. Přidanou hodnotou k této kapitole je souhrn obecných pravidel, jak si zachovat svojí anonymitu na internetu.

Čtvrtá kapitola popisuje zpracovaný útok na anonymitu pomocí rozhraní WebRTC. V této kapitole je úspěšně provedený útok, který odhalí naši identitu a zobrazí naši přibližnou polohu, společně s naší reálnou IP adresou.

Poslední, pátá kapitola, se zabývá implementací rozšíření do prohlížeče Google Chrome. Popisuji nejdříve, co to vlastně rozšíření je, důležité faktory pro fungování rozšíření a základní architekturu. Na tyto skutečnosti navazuji vysvětlením mé implementace rozšíření, jaké jsou její funkce a bezpečnostní úrovně.

Výstup praktické části této práce je úspěšný útok a úspěšný obranný mechanismus proti danému útoku. Toto rozšíření by bylo vhodné obohatit o další funkce, metody, které by zabránily útokům sloužícím k odhalení uživatele, popsané ve třetí kapitole.

Bibliografie

1. PALME, Jacob; BERGLUND, Mikael. *Anonymity on the Internet [online]* [<https://people.dsv.su.se/~jpalme/society/anonymity.html>]. [cit. 2018-03-21].
2. COMPUTER HISTORY MUSEUM. Internet History 1962 to 1992 [online]. Dostupné také z: <http://www.computerhistory.org/internethistory/>. [cit. 2018-03-21].
3. IT GEARED. *TCP/IP Network Model [online]* [<http://www.itgeared.com/articles/1341-tcpip-networking-model-overview/>]. [cit. 2018-04-04].
4. WILKINS, Sean. OSI and TCP/IP Model Layers [online]. Dostupné také z: <http://www.pearsonitcertification.com/articles/article.aspx?p=1804869>. [cit. 2018-04-02].
5. PEDAGOGICKÁ FAKULTA MASARYKOVY UNIVERZITY. TCP/IP [online]. Dostupné také z: http://www.ped.muni.cz/wtech/03_studium/teps/teps-03.pdf. [cit. 2018-04-02].
6. PETERKA, Jiří. Protokoly TCP/IP [online]. Dostupné také z: <http://www.earchiv.cz/a98/a817k180.php3>. [cit. 2018-04-02].
7. INFOSEC INSTITUTE. *Port Scanning using Scapy [online]* [<http://resources.infosecinstitute.com/>]. [cit. 2018-04-04].
8. TOSCH PRODUCTION. *Transmission Control Protocol [online]* [<https://toschprod.wordpress.com/2012/01/30/osi-model-layer-4-transport/>]. [cit. 2018-04-04].
9. VCENTER COMPANY. *Real Time Streaming Protocol (RTSP) [online]* [<https://en.vcenter.ir/application/real-time-streaming-protocol-rtsp/>]. [cit. 2018-04-04].

10. POSTEL, Jonathan; REYNOLDS, Joyce. *FILE TRANSFER PROTOCOL (FTP)* [Internet Requests for Comments]. RFC Editor. Dostupné také z: <https://tools.ietf.org/html/rfc959>. RFC. RFC Editor. [cit. 2018-04-02].
11. AL-NAIMI, Maayouf. *Active v.s. Passive FTP Simplified* [online] [<http://memo.balnaimi.com/doku.php?id=services:ftp:avsp>]. [cit. 2018-04-04].
12. MOZILLA CONTRIBUTORS. *HTTP* [online] [<https://developer.mozilla.org/en-US/docs/Web/HTTP>]. [cit. 2018-04-05].
13. *How Anonymizers Work* [online] [https://www.livinginternet.com/i/is_anon_work.htm]. [cit. 2018-04-09].
14. TECHCRESS. *How to make a proxy server with an Amazon EC2 instance* [online] [<https://www.techcress.com/how-to-make-a-proxy-server-with-an-amazon-ec2-instance/>]. [cit. 2018-04-04].
15. MITCHELL, Bradley. *Best Free Anonymous Proxy Servers* [online] [<https://www.lifewire.com/free-anonymous-web-proxy-servers-818058>]. [cit. 2018-04-09].
16. SHEA, Mark; PAUL, Ian. *How to set up a VPN in Windows* [online]. Dostupné také z: <https://www.pcworld.com/article/210562/networking/how-set-up-vpn-in-windows.html>. [cit. 2018-04-09].
17. THE TOR PROJECT. *About Tor* [online] [<https://www.torproject.org/about/overview.html.en>]. [cit. 2018-04-09].
18. ANONYMOUS. *Amrd: Going Dark the next level i2p TOR* [online] [<https://charlescarrollsociety.com/2014/03/19/amrd-going-dark-the-next-level-i2p-tor-https-everywhere/>]. [cit. 2018-04-09].
19. I2P PROJECT. *A Gentle Introduction To How I2P Works* [online] [<https://geti2p.net/en/docs/how/intro>]. [cit. 2018-04-09].
20. DINGLEDINE, Roger; SASSAMAN, Len. *Attacks on Anonymity Systems: The Theory* [online] [<https://www.blackhat.com/presentations/bh-usa-03/bh-us-03-sassaman-dingledine/bh-us-03-sassaman.pdf>]. [cit. 2018-04-16].
21. CA TECHNOLOGIES VERACODE. *Man in the Middle Attack* [online] [<https://www.veracode.com/security/man-middle-attack>]. [cit. 2018-04-16].
22. MICROSOFT. *Common Types of Network Attacks* [online] [[https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/cc959354\(v=technet.10\)](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/cc959354(v=technet.10))]. [cit. 2018-04-16].
23. CAMPBELL, Jeremy. *Common Types of Network Attacks* [online] [<https://www.dnsleaktest.com/what-is-a-dns-leak.html>]. [cit. 2018-04-16].

24. RAYMOND, Jean-Francois. *Traffic Analysis: Protocols, Attacks, Design Issues and Open Problems*. 2000. Technická zpráva. Zero-Knowledge Systems, Inc. [cit. 2018-04-16].
25. THREAT POST. *Tor Browser Users Urged to Patch Critical ‘TorMoil’ Vulnerability [online]* [<https://threatpost.com/tor-browser-users-urged-to-patch-critical-tormoil-vulnerability/128769/>]. [cit. 2018-04-16].
26. ERDIN, Esra; ZACHOR, Chris; GUNES, Mehmet Hadi. *Traffic Analysis: Protocols, Attacks, Design Issues and Open Problems* [<https://www.cse.unr.edu/~mgunes/papers/CST15-Anonymity.pdf>]. [cit. 2018-04-16].
27. WARREN, Tom. *Adobe will finally kill Flash in 2020 [online]* [<https://www.theverge.com/2017/7/25/16026236/adobe-flash-end-of-support-2020>]. [cit. 2018-04-16].
28. COUFAL, Zdeněk. *Korelace dat na vstupu a výstupu sítě Tor*. 2014. Diplomová práce. Vysoké učení technické v Brně. [cit. 2018-04-16].
29. SAMEEH, Tamer. *An Overview of Modern Tor Deanonymization Attacks [online]*. Dostupné také z: <https://www.deepdotweb.com/2017/09/12/overview-modern-tor-deanonymization-attacks/>. [cit. 2018-04-16].
30. SHUBH, Tulika; SHARMA, Shweta. *Defence Techniques of Man-In-The-Middle-Attack using SSL and HTTPS [online]*. 2016. Technická zpráva. International Journal of Computer Science a Mobile Computing. [cit. 2018-04-16].
31. CHASAKI, Danai; WOLF, Tilman. *Attacks and Defenses in the Data Plane of Networks [online]*. 2012. Technická zpráva. IEEE. [cit. 2018-04-16].
32. REVIEWER, VPN. *What is a DNS Leak and how to fix it? [online]*. Dostupné také z: <https://vpnreviewer.com/dns-leak-fix>. [cit. 2018-04-16].
33. JANSEN, Rob. *New Tor Denial of Service Attacks and Defenses [online]* [<https://blog.torproject.org/new-tor-denial-service-attacks-and-defenses>]. 2014. [cit. 2018-04-16].
34. ORJIH, Obi. *Type 2 Cross-site Scripting: An Attack Demonstration [online]*. 2014 [cit. 2018-04-16]. Dostupné z: <https://www.cse.wustl.edu/~jain/cse571-07/ftp/xsscript/index.html>.
35. SHMATIKOV, Vitaly; WANG, Ming-Hsiu. *Timing analysis in low-latency mix networks: attacks and defenses [online]*. Springer, Berlin, Heidelberg, 2006. Č. 978-3-540-44605-7. [cit. 2018-04-16].

36. SIXTA, Jiří. *Obrana proti fingerprintingu do Privoxy*. 2018. Diplomová práce. České vysoké učení technické v Praze. [cit. 2018-04-16].
37. THE WEBRTC PROJECT. *WebRTC [online]* [<https://webrtc.org/>]. [cit. 2018-04-04].
38. MOZILLA CONTRIBUTORS. *Introduction to WebRTC protocols [online]* [https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API/Protocols]. [cit. 2018-04-04].
39. JONES, Robb. *IP Addressing and Subnetting [online]* [<https://www.scribd.com/doc/49445184/IP-Addressing-and-Subnetting-Workbook-Instructors-Version-1-5>]. [cit. 2018-04-20].
40. GOOGLE. *What are extensions? [online]* [<https://developer.chrome.com/extensions>]. [cit. 2018-04-16].
41. GOOGLE. *Overview [online]* [<https://developer.chrome.com/extensions/overview>]. [cit. 2018-04-16].

Seznam použitých zkratek

API Application Programming Interface

CSS Cascading Style Sheets

DNS Domain Name System

HTML Hypertext Markup Language

IoT Internet of Things

IP Internet protocol

JSON JavaScript Object Notation

NFS Network File System

P2P Peer-to-peer

SSH Secure Shell

SSL Secure Sockets Layer

SQL Structured Query Language

TLS Transport Layer Security

URL Uniform Resource Locator

Obsah přiloženého CD

readme.txt	stručný popis obsahu CD
src	
impl	
extension.....	zdrojové kódy implementace rozšíření
attack	zdrojový kód implementace útoku
thesis	zdrojová forma práce ve formátu \LaTeX
text	text práce
thesis.pdf	text práce ve formátu PDF