VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

KORELACE DAT NA VSTUPU A VÝSTUPU SÍTĚ TOR

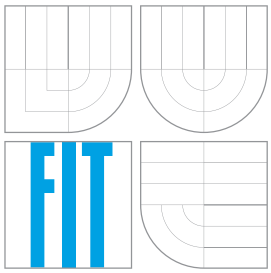
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ZDENĚK COUFAL

BRNO 2014

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

KORELACE DAT NA VSTUPU A VÝSTUPU SÍTĚ TOR

CORRELATION OF INBOUND AND OUTBOUND TRAFFIC OF TOR NETWORK

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. ZDENĚK COUFAL

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. LIBOR POLČÁK

BRNO 2014

Abstrakt

Komunikace ve veřejných sítích založených na protokolu IP není ve skutečnosti anonymní, protože je možné přesně určit zdrojovou a cílovou IP adresu každého paketu. Uživatelé, kteří chtějí komunikovat anonymně, využívají anonymizačních sítí jako je např. Tor. V případě, že je takový uživatel cílem zákonných odposlechů, je to problém, systém pro zákonné odsposlechy vidí pouze to, že uživatel komunikoval s anonymizační sítí a má podezření, že datový tok na výstupu anonymizační sítě Tor náleží tomu samému uživateli. Cílem této diplomové práce bylo navrhnout korelační metodu, která určí závislost datového toku na vstupu a výstupu sítě Tor. Navržená metoda využívá analýzy síťového provozu, kdy jsou porovnávány charakteristiky datových toků extrahované z metadat jako je čas výskytu a velikost paketu. Metoda se specializuje na korelaci datových toků protokolu HTTP, konkrétně odpovědi webového serveru. Byla testována na reálných datech ze sítě Tor a úspěšně rozlišila zavislost datových toků.

Abstract

Communication in public networks based on the IP protocol is not really anonymous because it is possible to determine the source and destination IP address of each packet. Users who want to be anonymous are forced to use anonymization networks, such as Tor. In case such a user is target of lawful interception, it presents a problem for those systems because they only see that the user communicated with anonymization network and have a suspicion that the data stream at the output of anonymization network belong to the same user. The aim of this master thesis was to design a correlation method to determine the dependence of the data stream at the input and the output of the Tor network. The proposed method analysis network traffic and compares characteristics of data streams extracted from metadata, such as time of occurence and the size of packets. This method specializes in correlating data flows of protocol HTTP, specifically web server responses. It was tested on real data from the Tor network and successfully recognized dependency of data flows.

Klíčová slova

Anonymita na Internetu, anonymizující síť, Tor, analýza síťového provozu, korelační metoda, závislost datových toků, zákonné odposlechy.

Keywords

Anonymity in the Internet, anonymization network, Tor, network traffic analysis, correlation method, dependency of data flows, lawful interception.

Citace

Zdeněk Coufal: Korelace dat na vstupu a výstupu sítě Tor, diplomová práce, Brno, FIT VUT v Brně, 2014

Korelace dat na vstupu a výstupu sítě Tor

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Libora Polčáka

.....
Zdeněk Coufal
27. května 2014

Poděkování

Tímto bych chtěl poděkovat vedoucímu práce Ing. Liboru Polčákovi za vedení této práce a ochotu při konzultacích.

Obsah

Kapitola 1

Úvod

Anonymita na Internetu vytváří dva odlišné pohledy a to v oblasti sociální týkající se lidských práv, a stejně tak i v oblasti kyberprostoru. Na jednu stranu [26] anonymizující technologie umožňují legální využití za účelem zajištění soukromí, svobody projevu, boje proti cenzuře, anonymních průzkumů a zpětné vazby od uživatelů. Na druhou stranu [26, 16] tyto technologie poskytují ochranu a útočiště pro kriminálníky, kteří se podílejí na pirátství, krádežích dat, informací a identit, spamování, sledování uživatelů tzv. *cyberstalking* a organizovaném zločinu.

Přestože uživatelé na Internetu využívají ke komunikaci na Internetu často pseudonymy a připadají si anonymní, opak je pravdou. Internet jako síť s přeposíláním paketů postavená na IP protokolech umožňuje u každého zaslaného paketu přesně určit odesílatele a příjemce paketů na základě jeho IP adresy [33].

Za účelem dosažení rozumné úrovně anonymity vzniklo mnoho anonymizujících technik. Principem těchto technik je zabránit dohledání pravé IP adresy odesílatele zprávy. Nejnižší úroveň anonymity poskytují obyčejné proxy servery [33], kdy jde skutečně o jediného prostředníka, přes kterého uživatel komunikuje s okolním světem. Vyšší úroveň anonymity poskytují distribuované anonymizující sítě [33], kde uživatel komunikuje přes řetěz proxy serverů, které vzájemně spolupracují, ale neznají nikoho jiného než svého předchůdce a následníka. Nejznámějším představitelem těchto sítí je projekt Tor, který si za více než deset let své existence vybudoval díky rozsáhlé komunitě přednostní postavení.

Tato diplomová práce souvisí s projektem *Moderní prostředky pro boj s kybernetickou kriminalitou na Internetu nové generace*, který se mimo jiné zabývá zákonnými odposlechy komunikace uživatelů na Internetu. S tím souvisí i cíl této práce, kterým je nalézt metodu, která by umožnila vyšetřujícím orgánům potvrdit, že daný uživatel komunikoval přes anonymizující síť s jiným uživatelem či službou na Internetu (např. webovým serverem). Odposlouchávající vidí pouze, že uživatel komunikoval s anonymizující sítí a potřebuje jeho datový tok asociovat s datovým tokem na výstupu sítě Tor.

Tento problém řeší korelační metody, ty se snaží nalézt závislost mezi dvěma datovými toky s využitím analýzy síťového provozu. Tor je síť s nízkým zpožděním doručení dat a díky tomu je možné sledovat charakteristiky datových toků, zejména časování a objem paketů. V minulosti byly na síti Tor úspěšně realizovány korelační útoky a autoři Toru veřejně uznávají [4], že je Tor vůči těmto útokům zranitelný. Nicméně efektivita korelačního útoku je stále otevřenou otázkou a autoři Toru vyzývají [4] komunitu, aby tento problém podrobněji analyzovala.

Písemná zpráva je členěna následovně. V kapitole 2 popisuji problematiku anonymizace na Internetu a jakým způsobem ji řešili předchůdci sítě Tor. V kapitole 3 se zabývám

analýzou sítě Tor a principy, které využívá. Pochopení těchto principů je nutné ke studiu korelačních metod a návrhu vlastní korelační metody. Dále popisuji v kapitole 4 vlastní korelační útok a podmínky pro jeho realizaci v síti Tor. Navazuje kapitola 5, kde jsou popsány existující korelační metody. V kapitole 6 podrobně popisuji návrh vlastní korelační metody, jakým způsobem navazuje na metody existující a co přináší nového. O implementaci nástroje realizujícího navrženou korelační metodu pojednávám v kapitole 7. Testování korelační metody a vyhodnocení její úspěšnosti na reálných datech ze sítě Tor je popsáno v kapitole 8. Závěr této diplomové práce a dosažené výsledky diskutuji v kapitole 9.

Kapitola 2

Historie anonymizačních technik

V této kapitole jsou popsány základní pojmy a definice, které tato práce využívá. Konkrétně jde o vytyčení problému anonymizace, tak jak je chápán v oblasti Internetu, viz sekci 2.1. Následuje krátký popis anonymizujících technik, které se problém anonymizace snaží řešit, viz sekci 2.2.

Poté uvádím krátký přehled předchozích anonymizačních systémů, ze kterých Tor vychází, viz sekci 2.3, a to z toho důvodu, že Tor přejímá významnou část jejich funkcionality.

2.1 Definice problému anonymizace

Uživatelé na Internetu realizují denně nespočet transakcí různých forem. Některé z těchto transakcí jsou komunikativního charakteru např. zaslání emailu, přečtení novinek, chat. Další mohou být komerčního charakteru např. nákupy, prodeje.

V každém případě účastníci mezi sebou vyměňují nějaký obsah, v případě komunikativních transakcí jsou to informace a v případě komerčních transakcí jsou to hodnoty. Jenže tyto transakce zahrnují mimo jiné výměnu (mezi uživateli) nebo odhalení (někomu zvenčí) meta-informací týkajících se účastníků nebo prováděných transakcí. Těmito meta-informacemi může být čas transakce, hodnoty vzájemně vyměněných položek, fyzická lokace nebo informace o identitách účastníků.

Goldberg definoval [26] základní pojmy v oblasti anonymizace.

Soukromí: Schopnost uživatele řídit šíření informací o něm samotném.

Anonymita: Forma soukromí identity. Systém, který poskytuje anonymitu, je takový systém, který umožní uživateli určit, kdo se dozví jeho pravou identitu (pravé jméno). Znamená to, že systém automaticky nevkládá informace o identitě uživatele do hlaviček protokolů. Pro útočníka není jednoduché, nebo může být nemožné, aby prolomil takový systém a odhalil pravou identitu uživatele.

Pseudonymita: Další forma soukromí identity. Uživatel udržuje jeden nebo více pseudonymů, což jsou zosobnění, která jsou spojena s jeho fyzickou identitou. Druhá strana komunikace si může být jista, že se pod pseudonymem skrývá jedna a ta samá osoba, ale skrývá fyzickou identitu. U anonymity žádný takový perzistentní identifikátor neexistuje a nelze říci, zda transakci provedla jedna a ta samá osoba.

Dopředné utajení (*Forward Secrecy*): Neschopnost odhalit bezpečnostně kritické informace jako je identita odesílatele zprávy potom, co byla zpráva odeslána. Systém poskytující anonymitu se snaží této vlastnosti dosáhnout např. tím, že si neudržuje žádné záznamy o provedených transakcích.

Spojitelnost (*Linkability*): Schopnost dát si do souvislosti transakce s identitou účastníků.

2.2 Anonymizující techniky

Li, et al. definovali [33] popis a dělení anonymizujících technik. Přestože účastníci využívají ke komunikaci na Internetu převážně pseudonymy, nezaručuje jim to zcela anonymitu, protože je lze stále dohledat prostřednictvím jejich IP adresy.

V sítích postavených na protokolu IP (Internet Protocol) obsahují pakety zdrojovou a cílovou IP adresu komunikujících uzlů, a proto na této úrovni nemohou být žádné datové pakety ve skutečnosti anonymní. Nicméně záměnou původní zdrojové adresy za falešnou lze anonymní komunikace dosáhnout, protože není jednoduché zpětně vysledovat původní zdroj komunikace.

Anonymizující techniky poskytují Internetovým uživatelům určitou úroveň soukromí tím, že zabraňují sběru identifikačních informací jako je jejich IP adresa a další specifické položky protokolů (většinou aplikačních), které by mohly odhalit uživatelovu identitu nebo lokaci. Tyto techniky si kladou za cíl utajit identitu zdroje komunikace, případně i cíle komunikace před vnějším pozorovatelem nebo nebo i mezi účastníky komunikace.

Anonymizující systémy mohou být kategorizovány podle jejich latence, úrovně důvěry, typu sítě, anonymizujících vlastností a možností případného útočníka. Z hlediska zpoždění doručení zpráv jsou anonymizující systémy klasifikovány do dvou kategorií: *systémy s vysokým zpožděním doručení dat*, které poskytují vysokou úroveň anonymity a *systémy s nízkým zpožděním doručení dat*, kde je vysoká míra interakce uživatele s aplikací a rychlá odezva je prioritou, většinou používané při anonymizovaném surfování na webu. Strategie přeposílání zpráv na uzlech je podmíněna nízkým zpožděním doručení dat, a proto tyto systémy volí kompromis mezi dosaženou úrovní anonymity a poskytovaným zpožděním doručení dat.

Z hlediska složitosti a úrovně poskytované anonymity se dělí anonymizující systémy na jednoduché anonymizující proxy servery, kde výše zmíněné zastává pouze jediný síťový uzel. Tam nastává problém, že uživatel musí plně důvěřovat pouze jedné entitě. A pak jsou to distribuované anonymizující systémy jako je Tor, tam na poskytnutí vysoké úrovně anonymity spolupracuje i více uzlů. Důvěra uživatele je rozložena na všech těchto spolupracujících uzlech.

2.3 Předchůdci sítě Tor

V této sekci jsou popsáni předchůdci sítě Tor, protože Tor přejímá významnou část jejich funkcionality.

Vznikly dva směry pro distribuované anonymizující systémy [39]. První skupina se snaží o maximalizaci anonymity za cenu vysokých latencí. Tyto anonymizující systémy odolávají i silným globálním útočníkům, ale jejich zpoždění je příliš veliké na to, aby byly používány u interaktivních úloh jako je prohlížení webu, chat nebo zabezpečené SSH spojení. Většinou fungují na principu, který představil systém *Mix-Net*.

Druhá skupina anonymizujících systémů se snaží o nízké zpoždění doručení zpráv, je vhodná pro interaktivní síťový provoz a umožňuje anonymizovat celou škálu aplikačních protokolů, nejen elektronickou poštu. Představitelem této kategorie je síť *Onion Routing*, ze které vychází Tor.

2.3.1 Mix-Net

Chaum představil [15] v roce 1981 koncept Mix-Netu jako sítě založené na přeposílání. Vznikl za účelem anonymizace komunikace elektronickou poštou.

Základní myšlenkou je využití asymetrické kryptografie ve spojení se sítí proxy serverů tzv. *mixů*, které slouží pro přeposílání zpráv mezi účastníky komunikace. Klient vybírá určitý počet mixů, které tvoří *cestu*, kterou jsou zprávy přeposílány. Každý mix vlastní pár soukromý a veřejný klíč.

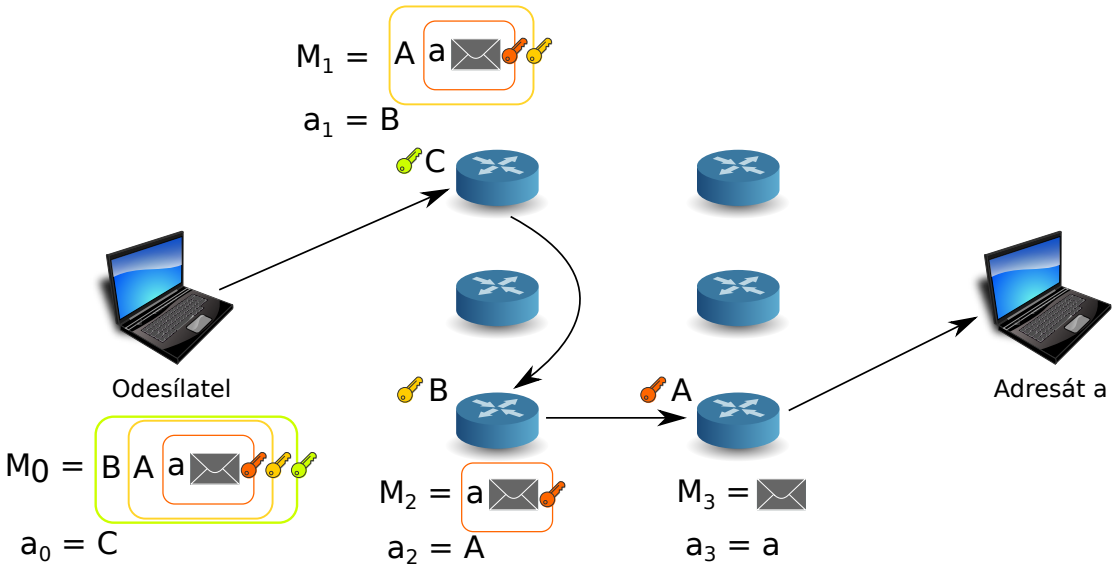
Uvedený princip přeposílání, viz obrázek 2.1, si lze jednoduše představit jako *loupání cibule*, každý mix na cestě odstraňuje jednu zašifrovanou vrstvu svým soukromým klíčem, aby získal směrovací informace a zprávu zasílá dalšímu mixu, kde se situace opakuje. To zabraňuje přeposílajícím uzlům zjistit cíl, původ a obsah zprávy. S výjimkou posledního mixu, ten může pozorovat obsah zprávy.

Jedná se o tzv. *store-and-forward* architekturu, zprávy jsou na mixech pozdrženy po předem neurčenou dobu a odtud plyne vysoké zpoždění doručení zpráv [44].

Mixy přijímají zprávy od více klientů, zamíchají je a v určitých časech takto vytvořenou dávku společně s vycpávkovými zprávami přeposílají dále.

Navržený způsob přeposílání zajišťuje, že každý uzel na cestě zná pouze svého předchůdce a následníka, ale nemá možnost zjistit totožnost komunikujících stran. Navíc dávkové přeposílání mixů znemožňuje korelaci zpráv při odposlechu komunikace jak v rámci anonymizující sítě, tak na jejich okrajích [39].

Asymetrická kryptografie a algoritmy pro dávkové přeposílání jsou klíčové pro dosažení vysoké úrovně anonymity, bohužel mají negativní celkový dopad na výkonnost sítě.



Obrázek 2.1: Princip doručení emailové zprávy v síti Mix-Netu. [39]

Předpokladem výše navrženého způsobu přeposílání je, že nesmí být možné určit vztah mezi množinou zašifrovaných zpráv a odpovídající množinou nezašifrovaných zpráv vstupujících, respektive vystupujících z mixu. Z toho důvodu mixy odstraňují redundantní zprávy. Stejně tak nesmí být možné vytvářet padělky zpráv bez znalosti náhodného řetězce a sou-

kromého klíče. Požadavek vychází z vlastností asymetrické kryptografie [1].

Mix-Net umožňuje na anonymně přijatou zprávu také anonymně odpovědět. K tomu využívá nevysledovatelné zpáteční adresy, která specifikuje zpáteční cestu k původnímu odesílateli.

2.3.2 Onion Routing

V roce 1996 Goldschlag et al. [27] přišli s návrhem anonymizujícího systému *Onion Routing*, který umožňoval anonymizaci spojení pro řadu aplikačních protokolů (HTTP, FTP, SMTP, rlogin, telnet a další). Uživatelská aplikace komunikuje s touto sítí pomocí aplikačně specifické proxy.

Primárním cílem Onion Routing bylo poskytnout bezpečnou anonymní komunikaci v reálném čase s využitím veřejně dostupné sítě. Směrovače si mezi sebou udržují TCP/IP spojení a tvoří tak logickou vrstvu *overlay* sítě Onion Routing. Informace o aktuálním stavu sítě (topologii) je propagována přes všechny aktivní směrovače a uživatelské proxy.

U Mix-Netu musel uživatel pro každou zasílanou zprávu vytvářet zvlášť cibuli a používat často operace asymetrické kryptografie. To bylo sice vhodné pro případ komunikace elektronickou poštou, ale u interaktivní aplikace by to znamenalo spoustu výpočetně náročných operací s veřejným klíčem.

V Onion Routing síti každý směrovač disponuje párem veřejný/soukromý klíč. Asymetrickou kryptografii (konkrétně algoritmus RSA) používá tato síť pouze pro zašifrování sdíleného symetrického klíče, kterým je zašifrován zbytek zprávy, tím je implementace daleko efektivnější.

Pro každou zprávu není potřeba vytvářet zvlášť cibuli, Onion Routing poskytuje obousměrnou komunikaci pomocí *virtuálních okruhů*. Princip rezervace okruhu vychází ze zasílání zpráv Mix-Netem. Při požadavku na nové anonymizované spojení vytváří Onion Routing náhodnou cestu skrze vlastní síť, na rozdíl od Mix-Netu, kde volil cestu uživatel. Aplikační proxy vytváří *rezervační cibuli*, která specifikuje ve vrstvách cestu k cíli a informace o použitých kryptografických klíčích a algoritmech. Průchod této cibule všemi směrovači zajišťuje rezervaci dopředné i zpětné cesty zvoleným okruhem. Následně mohou být zasílány vlastní aplikační datové toky.

Zasílá se také vycpávkový provoz, aby nemohl útočník pozorovat, že sítí proudí data. Protože nejsou ukládány zaslané zprávy jako u Mix-Netu, hrozí zde útok opakováním přenosu rezervační cibule[2].

Onion Routing nebyl v praxi příliš úspěšný, reálně byl nasazen pouze za účelem *proof-of-concept*. Mnoho problémů ať již v návrhu nebo zjištěných při nasazení nebylo vyřešeno a samotný návrh nebyl již léta aktualizován.

[1] Zná-li útočník kryptogram $K(X)$, což je pouze zpráva X zapečetěná veřejným klíčem K, pak si může dovolit odhadnout, zda $Y = X$ tím, že vyzkouší zda $K(Y) = K(X)$. Tuto hrozbu lze jednoduše eliminovat tak, že před vlastní zprávu X je přidán náhodný řetězec bitů R a kryptogram pak vypadá následovně $K(R, X)$.

[2] Útok *onion replay* mohl útočník využít k opětovnému vytvoření dříve existujícího virtuálního okruhu pomocí rezervační cibule, kterou dříve zachytil. Tím pádem získal přístup k použitým šifrovacím klíčům a algoritmům a mohl nechat dešifrovat dříve zachycenou komunikaci. [44]

Kapitola 3

Anonymizační síť Tor

Tato kapitola se zabývá popisem sítě Tor. Síť Tor vychází z původního konceptu sítě *Onion Routing* [27], avšak s řadou vylepšení a je označována jako druhá generace Onion Routing. Byla spuštěna v roce 2003 (skryté servery v roce 2004). Na rozdíl od sítě Onion Routing není Tor zatížen žádnými patenty, jedná se o *open-source* projekt s rozsáhlou komunitou [4].

Návrh sítě popsali Dingledine et al. [23, 22], z tohoto návrhu také vycházím v dalším textu. Od uvedení sítě do praxe uběhla již řada let a za tu dobu Tor prodělal řadu méně či více významných změn.

Při popisu Toru nerozebírám v dalších pasážích vyloženě nízkoúrovňové postupy, pokud to není pro práci relevantní. Podrobný popis je uveden v technické zprávě [17], která vznikla současně s touto prací. Detaily lze nalézt přímo ve specifikačních dokumentech Toru [6, 2, 1, 3].

3.1 Architektura a komponenty sítě Tor

Z pohledu návrhu je síť Tor koncipována tak, že využívá již existující Internetovou TCP/IP infrastrukturu, jedná se tedy o tzv. *overlay* síť[1], která je pouze další logickou vrstvou nad již existující vrstvou.

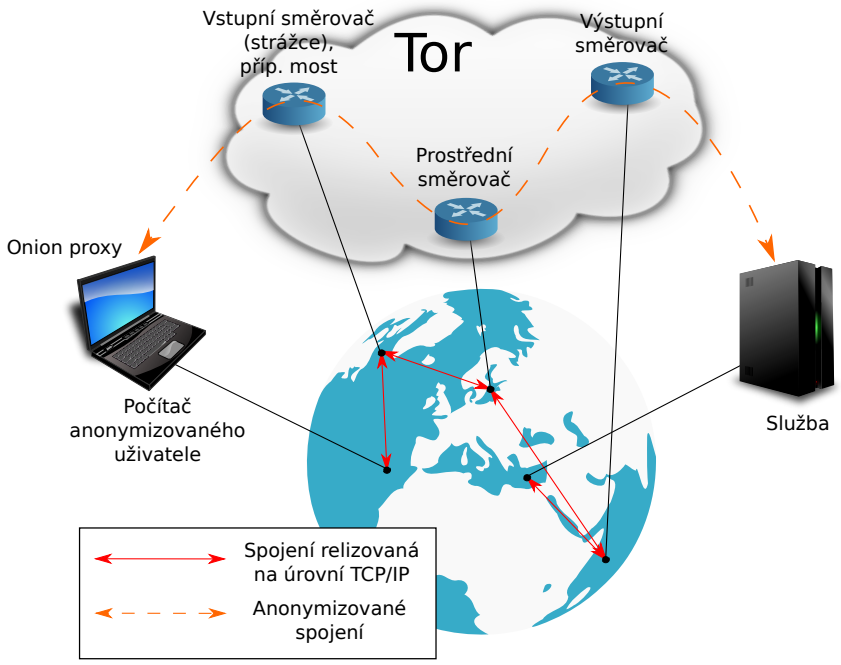
Síť Tor je tvořena následujícími základními komponentami, jejich použití bude dále rozvedeno v textu.

- **Směrovače** (*onion routers* - OR): Tyto směrovače jsou základním stavebním kamenem celé sítě Tor, vytváří její topologii a především se jedná o uzly, kterými prochází virtuální okruhy. Podle pozice ve tříčlenném okruhu se dělí na vstupní směrovače tzv. strážce (*guards*), prostřední směrovače (*middle*) a výstupní směrovače (*exits*). Ve výchozím stavu naslouchají na portu 9001 a čekají na příchozí požadavky na spojení.
- **Uživatelská proxy** (*onion proxy* OP): Získávají aktuální data o topologii a stavu sítě z adresářových serverů, vytváří virtuální okruhy a řídí přenos datových toků mezi uživatelskými aplikacemi a sítí Tor.
- **Adresářové servery** (*directory servers*): Udržují globální pohled na topologii a stav jednotlivých uzlů sítě Tor. Klientům je tento konsensus dostupný ve výchozím stavu na portu 9030.

[1] Na podobném principu pracují virtuální privátní sítě (VPN) a *peer-to-peer* sítě (P2P).

- **Skryté servery** (*hidden services*) (HS): Tor poskytuje tzv. skryté služby, což je integrovaný mechanismus, který umožňuje serverům anonymně poskytovat služby, bez toho aniž by byla prozrazena identita těchto serverů. K těmto serverů je přistupováno přes upravené doménové jméno v rámci sítě Tor.
- **Mosty** (*bridges*): Tyto uzly byly představeny až později [22] kvůli boji proti cenzuře. Jedná se o speciální vstupní směrovače, informace o nich není veřejně dostupná na adresářových serverech, tudíž je nelze hromadně blokovat. Uživatel si může vyžádat IP adresu těchto uzlů, ale přidělený počet je vázán na jeho vlastní IP adresu.
- **Uživatelé**: Využívají služeb sítě Tor, z hlediska přístupu k síti se dělí na přímé uživatele (přistupují přes veřejně dostupné OR) a cenzurované uživatele (přistupují přes mosty).
- **Maskování provozu** (*pluggable transports*): Transformují síťový provoz Toru mezi klientem a mosty. Touto cestou cenzoři, kteří monitorují síťový provoz mezi uživatelem a mostem, obecně mezi dvěma uzly sítě Tor, uvidí jiný druh síťového provozu, za který se protokol Toru maskuje. V každém případě je tento provoz šifrovaný, mění charakteristiku původního provozu např. délku paketů, rozložení délek paketů, rozestupy mezi pakety apod.

Obrázek 3.1 zachycuje typické použití sítě Tor. Uživatel sítě Tor nekontaktuje službu přímo. Namísto toho využije služeb sítě Tor pro vytvoření několika spojení napříč celým světem pro ukrytí skutečného cíle komunikace.



Obrázek 3.1: Základní funkcionalita Toru. Data jsou zasílána přes OR rozmístěné různě ve světě. [17]

3.2 Významné inovace

Tor přinesl oproti síti Onion Routing následující významné změny.

- Tor používá jako aplikační proxy standard **SOCKS**[2], což je proxy rozhraní, které podporuje většinu programů založených na TCP a to bez nutnosti dalších modifikací. Pro případy kdy není Tor schopen zaručit únik identifikujících informací např. DNS dotaz přes UDP, doporučují autoři používat Tor ve spojení s lokální proxy (např. *Privoxy*[3]).
- Tor využívá **důvěryhodné adresářové autoritativní servery**, které spolupracují při vytváření podepsaných adresářů obsahujících známé OR a jejich aktuální stav. OP uživatelů si adresáře periodicky stahují přímo z adresářových serverů pomocí protokolu HTTP tunelovaného přes virtuální okruh sítě Tor.
- Variabilní **politiky pro odchozí provoz** umožňují každému výstupnímu uzlu specifikovat, ke kterým hostům a portům je ochoten se připojit.
- Pro realizaci skrytých služeb Tor nepoužívá odpovědní cibule, ale využívá tzv. *rendezvous* uzlů, na kterých se klienti dohodnou jako na bodu setkání, aby se dostali k serverům poskytujícím skryté služby.
- Tor je **odolný vůči cenzuře** a pokusům zablokovat uživatelům přístup do jeho sítě. Zavedl nový uzel typu most *bridge*, který není nabízen adresářovými servery a jeho IP adresu zná jen malá množina individuálních osob. Navíc byl Tor navržen tak, aby se jevil na síti podobně jako protokol HTTPS (včetně využití typického portu 443, nicméně může využívat i jiný) a blokování Toru znamená zablokovat i HTTPS. Protože některé země blokují protokol HTTPS nebo mohou cenzoři provádět hlubokou inspekci paketů (DPI), zavedl Tor obfuskační rozšíření **pluggable transports**.
- Tor zavedl tzv. *Tor control* protokol [1], který poskytuje programátorské API dovolující spolupráci OP s dalšími uživatelskými či systémovými komponentami a umožňuje tak realizovat širokou škálu požadavků uživatelů na různé aplikace. Např. grafické uživatelské rozhraní *Vidalia* je samostatná aplikace, která komunikuje s OP přes výše uvedený řídicí protokol.

3.3 Virtuální okruhy

Anonymizace v Toru je postavena na využití tzv. virtuálních okruhů. Ty byly používány již v síti Onion Routing, ale Tor přišel s řadou změn a vylepšení.

- Okruhy buduje **inkrementálně**. Pro vytvoření okruhu se nepoužívá jediná rezervační cibule, ale klient se postupně dohodne na klíčích sezení s každým následujícím uzlem okruhu[4]. Tento přístup zajišťuje perfektní dopředné utajení (*forward secrecy*).

[2] SOCKS Protocol Version 5: http://tools.ietf.org/html/rfc1928

[3] Privoxy: http://www.privoxy.org

[4] K vyjednání sdíleného symetrického klíče se využívá algoritmu asymetrické kryptografie Diffie-Hellman. V novějších verzích Toru je založen na využití eliptických křivek.

- Tor využívá **multiplexování** a umožňuje tak využít jeden virtuální okruh pro více TCP spojení. Uživatel má kontrolu nad tím, které datové toky mohou virtuální okruh sdílet.
- Nevyužívá se přeuspořádání paketů, vycpávkový provoz ani rozprostření síťového provozu (*traffic shaping*).
- Tor verifikuje integritu dat na výstupních uzlech předtím než data opustí vlastní síť.
- Princip *leaky pipe* může OP uživatele využít při určení, který OR v okruhu bude pro datový provoz výstupním.
- OP zvolí 3 strážce a ty používá jako vstupní OR pro všechny vytvářené okruhy. Tyto 3 OR jsou obměňovány každých 30 až 60 dní. Zavedení OR typu strážce nesnižuje šanci, že by mohl být při vytváření prvního okruhu vybrán kompromitovaný OR, ale zajišťuje, že pokud se tak nestane, budou všechny další okruhy využívající tento OR bezpečné.
- Využívá **decentralizované řízení zahlcení**, které využívá potvrzování mezi koncovými uzly, což zajišťuje zachování anonymity a zároveň umožňuje uzlům na krajích sítě detekovat zahlcení a posílat méně dat, než zahlcení odezní.

3.4 Anonymizace aplikačního spojení

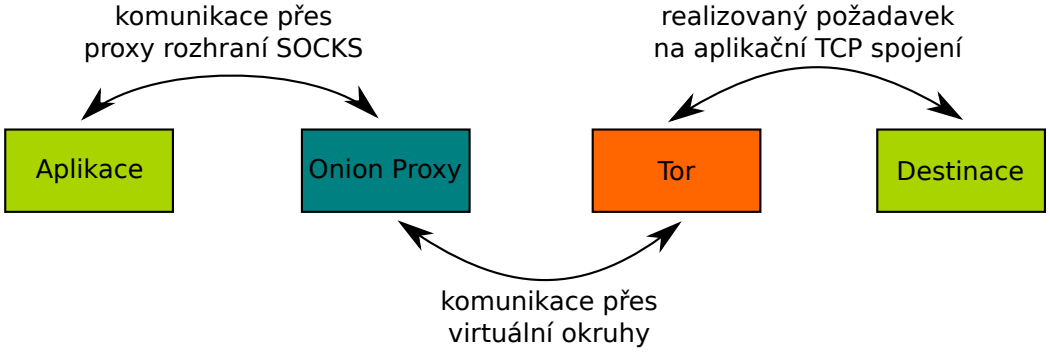
Postup realizace anonymizovaného spojení je znázorněn na obrázku 3.2. Vyžaduje-li uživatelská aplikace TCP spojení na danou adresu a port požádá OP skrze univerzální proxy rozhraní SOCKS (ve výchozím stavu naslouchá na portu 9050), aby spojení vytvořila. OP vybere nejnovější vhodný existující okruh nebo jej vytvoří. Využívá přitom preemptivní konstrukce okruhů (snaží se odhadnout, jaké okruhy by mohly být užitečné a dopředu je vytváří). Při výběru uzlů cesty, vybírá OP z konsensu (adresáře) OR na základě statusů a vah, které odráží aktuální přenosovou kapacitu, spolehlivost a politiky těchto uzlů.

Při postupném sestavování okruhu OP inkrementálně vyjednává symetrické klíče s každým OR v okruhu v pořadí od prvního (strážce) až po poslední (výstupní). V každém kroce rozšíření okruhu o další OR komunikuje OP s daným OR již přes vytvořenou část okruhu a tedy šifrovaně.

Před vlastním přenosem dat uživatelské aplikace zašle OP poslednímu OR požadavek na vytvoření spojení TCP s cílovou destinací. Ihned po vytvoření spojení na výstupu sítě Tor začíná OP přeposílat data uživatelské aplikace. Přenáší se užitečná data nad vrstvou TCP. Každý datový paket je přenášen v buňkách o pevné velikosti 512 B s tím, že dochází uvnitř buněk k zarovnávání na požadovanou velikost. Před odesláním OP buňku opakovaně zašifruje sdílenými symetrickými klíči a to v pořadí od posledního OR k prvnímu. Po průchodu buňky sítí Tor, kdy dochází k postupnému dešifrování vrstev buňky a směrování sítí Tor, zasílá výstupní OR[5] data uživatelské aplikace cílové destinaci.

Odpověď destinace zpět k uživatelské aplikaci probíhá analogicky, pouze s tím rozdílem, že při zpětném průchodu sítí Tor jednotlivé OR buňky postupně šifrují sdílenými symetrickým klíči ve směru od posledního OR k prvnímu OR okruhu. OP potom opakovaně buňku dešifruje a vlastní data odpovědi přeposílá přes rozhraní SOCKS uživatelské aplikaci.

[5] V tomto bodě mohou být data jak v otevřené podobě, tak šifrována. Záleží na tom, jaký datový tok uživatelská aplikace přeposílá. Každopádně Tor nezajišťuje *end-to-end* šifrování, pokud uživatel požaduje šifrovanou komunikaci mezi aplikací a cílovou destinací, musí explicitně používat např. protokol TLS.



Obrázek 3.2: Ilustrace realizace anonymizovaného spojení v síti Tor.

Celý postup anonymizace spojení je ve skutečnosti složitější, je třeba uvažovat algoritmus výběru cesty, šifrovací schémata, kontrolu integrity a mechanismus řízení zahlcení.

3.5 Zabezpečení komunikace v síti Tor

Tor používá hned několik různých kryptografických mechanismů k dosažení následujících cílů. Šifrování k zajištění důvěrnosti dat ve vlastní síti. Autentizace OR vůči klientům a podepisování k zajištění toho, že všichni klienti znají stejnou množinu OR.

Šifrování: Všechna spojení v síti Tor mezi jednotlivými OR používají protokol TLS. To znemožňuje pozorovatelům zjistit, ke kterému okruhu daná buňka patří. Dále klient zřizuje dočasný sdílený šifrovací klíč postupně s každým OR v okruhu a toto vrstvené zašifrování pomocí algoritmu AES zajišťuje, že pouze výstupní OR může vidět obsah aplikačního spojení. Po zániku okruhu klient i OR tyto klíče zahazují. Zaznamenávání provozu a následné kompromitování OR pak ztrácí smysl.

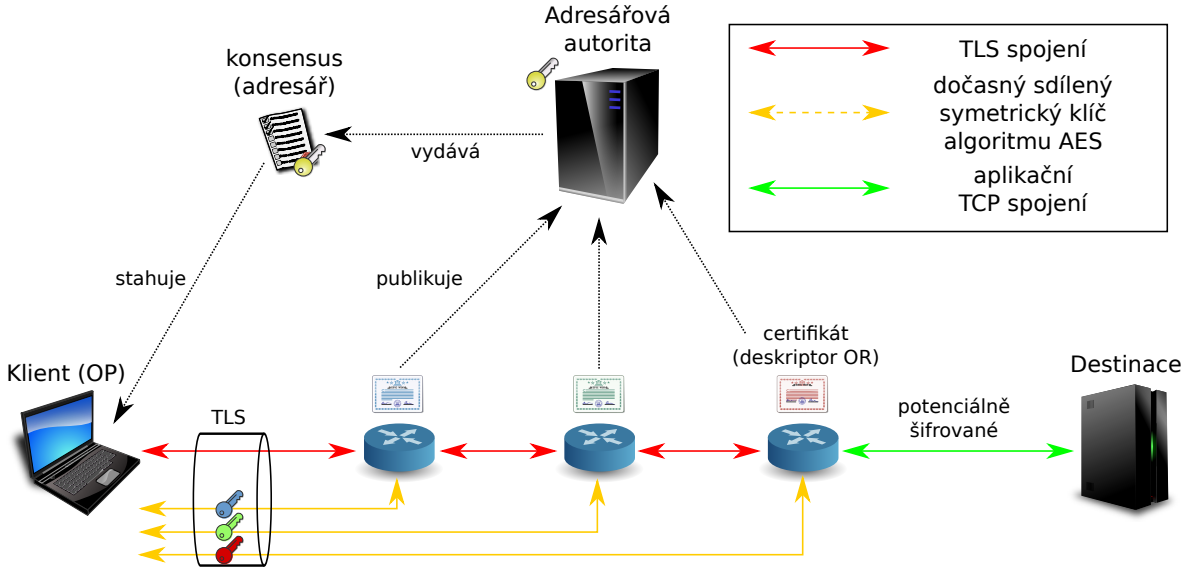
Autentizace: Ke každému OR je k dispozici jeho veřejný klíč zvaný *onion klíč*. OR rotují tyto klíče cca jednou za týden. Když klient zřizuje nový okruh, v každém kroku vyžaduje po OR znalost párového soukromého klíče a tím se OR klientovi autentizuje.

Koordinace: Aby měli klienti jistotu, že komunikují se správnými OR pomocí správných klíčů, vlastní každý OR dlouhodobý veřejný klíč zvaný *identifikační klíč*. Tímto klíčem podepisuje certifikát, který specifikuje jeho další klíče, lokaci, typ uzlu a další informace o daném typu uzlu. Adresářové autority disponují *adresářovým klíčem*. Tyto adresářové autority poskytují podepsaný seznam všech známých OR. V tomto seznamu se nachází množina certifikátů příslušejících jednotlivým OR, každý podepsaný příslušným identifikačním klíčem.

Obrázek 3.3 zachycuje použití výše definovaných mechanismů v síti Tor. V novějších verzích Toru se navíc kvůli zatížení adresářových autorit začalo používat kešování konsensu na tzv. adresářových serverech a nebo přímo na OR.

3.6 Režimy činnosti a výchozí konfigurace

Instance Toru může běžet ve čtyřech různých režimech, uživatel může dle potřeby používat jednu či více instancí v jednom z dále uvedených režimů činnosti.



Obrázek 3.3: Realizace zabezpečené komunikace v síti Tor.

1. **Klientský režim** – Uživatel využívá sítě Tor pouze jako klient.
2. **OR** – Uživatel přispívá svou vlastní přenosovou kapacitou k celkové přenosové kapacitě sítě Tor, je součástí vytvářených virtuálních okruhů ve prospěch jiných uživatelů. Informace o jeho uzlu je dostupná ve veřejném adresáři Toru.
3. **Most** – Podobné jako režim OR s tím rozdílem, že informace o uzlu uživatele není veřejně dostupná v adresáři Toru. Adresa jeho uzlu pak mohou ostatní uživatelé získat přes webovou stránku `https://bridges.torproject.org/` nebo elektronickou poštou. V tomto režimu lze navíc povolit rozšíření *pluggable transports*, které zajišťuje maskování charakteristik síťového provozu Toru.
4. **Skrytá služba** – Uživatel může anonymně provozovat libovolnou službu např. webový server a k té mohou ostatní přistupovat přes síť Tor skrze adresu, kterou je pro tyto účely speciálně modifikované doménové jméno.

Konfigurace je uložena v souboru `torrc` a ve výchozím stavu je Tor nakonfigurován následovně.

- Běží v klientském režimu.
- Virtuální okruhy mají délku 3 uzly.
- Není využíván mechanismus *leaky-pipe*.
- OP udržuje aktivní 4 okruhy, kterým přiděluje příchozí požadavky na TCP spojení od klienta.
- Životnost okruhu je omezena na 10 minut.
- Každých 30 sekund zvažuje OP konstrukci nového okruhu.

3.7 Statistiky sítě Tor

V této sekci uvádím statistiky sítě Tor, které jsou zajímavé pro realizaci korelačního útoku.

3.7.1 Počet uživatelů, velikost sítě a přenosové kapacity

V příloze v grafu A.1 je zachycen celkový počet přímo připojených uživatelů k síti Tor. V květnu 2014 se jednalo o cca 2 milióny uživatelů.

V příloze v grafu A.2 je znázorněn počet uzlů pro jednotlivé kategorie (strážce, výstupní uzel). Velikost sítě je již delší dobu konstantní s občasnými výkyvy. V dlouhodobějším časovém horizontu velikost výrazně narostla, z původního počtu desítek uzlů jsou to dnes tisíce. Z celkového počtu kolem 5000 uzlů je pouze 1000 uzlů výstupních, to svědčí o tom, že dobrovolníků, kteří na sebe převezmou riziko provozování výstupního uzlu, je méně než těch, kteří pouze přeposílají provoz uvnitř sítě Tor. Počet uzlů typu strážce je téměř 2000, stejně tak počet prostředních uzlů, které pouze přeposílají.

V příloze v grafu A.3 je znázorněno rozložení celkové přenosové kapacity mezi kategoriemi uzlů (strážce, prostřední, výstupní). OR typu strážce poskytují celkovou přenosovou kapacitu 2500 MiB/s. U výstupních OR je celková přenosová kapacita již pouze 500 MiB/s. Vzhledem k poměru počtu vstupních a výstupních uzlů (2:1), poměr rozložení přenosové kapacity (5:1) vypovídá o tom, že dobrovolnící poskytující výstupní uzly nejsou ochotni přeposílat velké objemy dat.

3.7.2 Přenášené protokoly

McCoy et al. [34] vytvořili v roce 2008 studii, která se zabývá využitím sítě Tor z hlediska přenášených protokolů a geografického rozložení klientů. Chaabane et al. [13] na tuto studii v roce 2010 volně navázali a zhodnotili výsledky po dvou letech.

V objemu přenesených dat pro klasifikovaný provoz je majoritní HTTP, ať už se jedná o prohlížení webu (převládá) nebo stahování souborů. Následuje BitTorrent, SSL, IM, E-mail, FTP a Telnet.

Nejvíce TCP spojení vytváří HTTP požadavky (uživatelé nejčastěji využívají Tor jako anonymizační HTTP proxy), na druhou stranu uživatelé, kteří provozují přes Tor P2P sdílení konzumují značné množství přenosové kapacity. Takové využití je z hlediska původního záměru využití sítě nežádoucí. Sice jsou v odchozích politikách blokovány TCP porty, které BitTorrent využívá, ale protože může využívat i nestandardní porty, není to koncovým řešením.

Vyskytují se i nezabezpečené protokoly, které zasílají přihlašovací informace v otevřené podobě. Jedná se nejčastěji o POP, IMAP, Telnet a FTP.

Vyskytuje se i neklasifikovaný provoz, ale Chaabane et. al s využitím metody založené na hloubkové inspekci paketů zjistili, že patří BitTorrentu. Tím pádem se BitTorrent stává hlavním přispěvatelem co do objemu přenášených dat (více než 50%). Přetěžuje tedy síť a navyšuje zpoždění.

3.7.3 Geografická lokalita klientů a OR

McCoy et al. a Chaabane et. al. se zabývali [34, 13] také geografickým rozložením klientů Toru. V uvedených studiích se rozložení mírně změnilo, ale shodují se na tom, že nejvíc klientů Toru a OR pochází z Německa a USA. Protože tyto studie jsou již několik let staré,

aktuální pohled na geografické rozložení klientů jsem čerpal přímo ze statistik Toru, které jsou veřejně dostupné.[6]

Z celkového počtu 2 miliónů přímo připojených uživatelů (nikoliv přes mosty) je pro deset nejčetnějších zemí uvedeno jejich procentuální zastoupení v příloze v grafu A.4. Procentuální zastoupení pro 10 nejčetnějších zemí co do počtu provozovaných OR je uvedeno v příloze v grafu A.5.

Necelou polovinu všech připojených uživatelů tvoří USA, Německo a Francie. Tyto samé země však provozují téměř dvě třetiny všech OR. Počet uživatelů připojených v ČR je v rozmezí 15 až 20 tisíc a počet provozovaných OR v ČR pak 77. Všechny zmiňované hodnoty jsou aktuální pro květen 2014.

3.7.4 Zpoždění v síti Tor

Tor je dynamická a heterogenní síť vzhledem k dostupné přenosové kapacitě a počtu jednotlivých uzlů. Následkem tohoto faktu je, že zpoždění a rozptyl v rámci sítě jsou proměnlivé veličiny. Ty pak závisí především na vzdálenosti a zatížení uzlů na okruhu.

Dhungel et al. [21] studují zpoždění okruhů i individuálních směrovačů v čase. Zpoždění okruhu měřili pomocí zaslání jediné buňky a přijetí odpovědi na ní (*tor ping*) a zpoždění na linkách mezi směrovači pomocí TCP pingu[7].

Z výsledků jejich studie vyplývá, viz graf A.6, že pro 23 % všech testovacích okruhů bylo zpoždění více než 1 sekunda[8], to je o dost více, než je běžné zpoždění bez použití sítě Tor. Z výsledků také vyšlo najevo, že hlavním důvodem vyššího zpoždění Toru je zpoždění linek mezi směrovači, protože při výběru cesty není uvažována fyzická vzdálenost OR. Významnou měrou přispívá ke zpoždění i směrování na OR na úrovni Toru.

Zjistili, že mezi přenosovou kapacitou okruhu a jeho zpožděním je nízká korelace a že u okruhů s vyšší přenosovou kapacitou kolísá zpoždění v čase méně. To přisuzují algoritmu výběru cesty. Zmiňují také, že na celkové zpoždění má vliv technika *token bucket*.

Závěrem studie bylo, že zpoždění na linkách mezi směrovači hraje významnou roli a stejně tak zpoždění při směrování na OR.

Panchenko et al. [38] se zabývají otázkou, zda je zpoždění Toru způsobeno přetíženými OR nebo přetíženými linkami mezi nimi. Na základě pozorované korelace mezi latencí a propustností se přiklánějí spíše k přetíženým OR, protože zpoždění Toru je často mnohem vyšší než u běžného spojení na Internetu (jednotky sekund vs. desítky milisekund). Výsledky experimentů jejich předpoklad potvrdily.

Dále porovnávají různé současné metriky a navrhují nové, které využívá algoritmus výběru cesty při výběru OR nového okruhu. Pro sadu testovaných okruhů s využitím těchto metrik měří dosaženou přenosovou kapacitu, RTT a rozptyl (*jitter*). Pro účely této práce jsou zajímavé hodnoty rozptylu pro současně implementovanou metriku IDLE-BW, viz graf A.7. Výsledky ukazují, že až pro 80 % všech testovaných okruhů byl rozptyl v jednotkách desítek milisekund.

Závěrem studie bylo, že současná metrika používaná algoritmem výběru cesty snižuje výkonnost sítě Tor až o 70 % ve srovnání s novou navrženou.

Otázkou zpoždění v sítí Tor se zabývaly i další studie. Bauer změřil [10] latenci v rámci sítě Tor. DeFabbia-Kane změřil [20] rozptyl mezipaketového zpoždění (buněk v síti Tor).

[6] https://metrics.torproject.org/

[7] Zasílá se paket s příznakem SYN a očekává se odpověď SYN/ACK.

[8] Při testování využívají okruhy o délce dvou uzlů, ve skutečnosti bude zpoždění okruhu ještě o něco vyšší.

Kapitola 4

Korelační útok v síti Tor

V této kapitole se zabývám možnostmi realizace korelačního útoku v anonymizačních sítích se zaměřením na síť Tor. Nejdříve definuji cíl této práce v kontextu systémů pro zákonné odposlechy, viz sekci 4.1. Následuje popis analýzy síťového provozu, kterou korelační útok využívá, viz sekci 4.2. Principy vlastního korelačního útoku jsou popsány v sekci 4.3. Podmínky pro realizaci útoku v síti Tor popisuji v sekci 4.4. A na závěr pojednávám o vlivu lokality provozu na úspěšnost korelace, viz sekci 4.5.

4.1 Formulace cíle

Anonymizační sítě představují pro zákonné odposlechy problém, protože systém vidí pouze to, že sledovaná osoba komunikovala s anonymizační sítí. K dispozici jsou tak pouze časové periody, kdy komunikace probíhala [39].

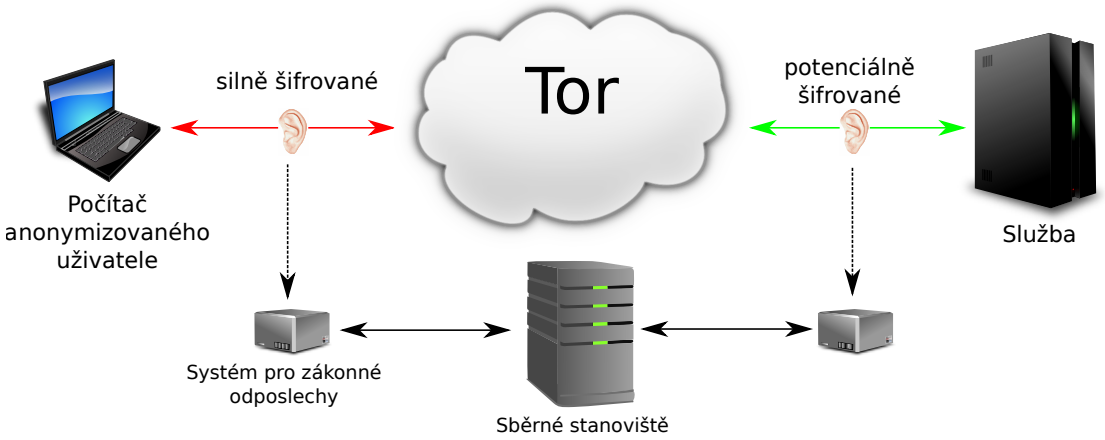
V této práci je předpokladem, že útočník (zákonné odposlechy) má k dispozici datový tok vstupující do sítě Tor a datový tok vystupující ze sítě Tor a podezření, že se jedná o jeden a ten samý. Cílem je s využitím korelačních metod určit, zda závislost mezi sledovanými datovými toky opravdu existuje. Pozice útočníka je znázorněna na obrázku 4.1.

Metoda bude pracovat se sítí Tor na vyšší úrovni abstrakce, bude ji tedy brát pouze jako černou skříňku. V této pozici lze předpokládat jako útočníka např. poskytovatele připojení k Internetu (ISP), internetovou ústřednu (IXP) nebo policejní systémy pro zákonné odposlechy podle norem ETSI [39].

Výčet výše uvedených útočníků spadá do kategorie pasivního útočníka, protože není žádoucí v případě policejních odposlechů měnit charakteristiku síťového provozu. Předpokladem pro tento typ útočníků je, že neprovozují kompromitované OR (nemají aktivně nasazené prostředky v síti Tor), ale mají přístup k síťovým linkám, kterými provoz Toru prochází. Lokalita provozu umožňuje těmto útočníkům pozorovat datové toky jak na vstupu, tak na výstupu sítě Tor.

4.2 Analýza síťového provozu

Když jsou přes Internet zasílané citlivé informace, typicky je použito šifrování za účelem zajištění důvěrnosti informace. Proces šifrování způsobí, že obsah paketů je pro útočníka nečitelný, ale i když jsou tato data úspěšně před útočníkem skryta, stále jsou k dispozici metadata v podobě informací z hlaviček paketů, časování paketů a velikosti paketů.



Obrázek 4.1: Ilustrace problému korelace datových toků na vstupu a výstupu sítě Tor v kontextu zákonných odposlechů.

V kombinaci s podobnými informacemi dalších paketů tyto informace jednoznačně identifikují datový tok. Tímto se zabývá analýza síťového provozu[1], jde o proces extrakce a odvození informací ze síťových metadat a je tak možné sledovat charakteristické vzory síťového provozu.

Způsobem, jak předcházet analýze síťového provozu a tím narušení bezpečnosti komunikace, je zasílání vycpávkového provozu nebo musí datový tok pro útočníka vypadat jako konstantní [40].

Song et al. využil [42] analýzy síťového provozu při útoku na protokol SSH. Wang et. al [48] ji použil při útoku na tzv. *stepping stones* systémy[2].

V minulosti byla analýza síťového provozu úspěšně nasazena i na datové toky anonymizačních sítí [8, 41, 18, 20, 46].

U síťového provozu anonymizačních sítí nelze jednoznačně určit datový tok na základě zdrojové a cílové IP adresy, protože při průchodu sítí prochází data několika nezávislými spojeními. Užitečnou informací je tedy časování, objem paketů a také zpoždění dat při průchodu sítí. Pokud má útočník k dispozici tuto informaci, může pomocí ní vytvořit model zpoždění sítě a odhadnout, jak by mohl vypadat sledovaný datový tok na výstupu sítě [19]. Hopper et al. ukazují [28], jak dokáže informace o zpoždění v síti ovlivnit úroveň poskytované anonymity.

Bauer vysvětluje [10] možné zranitelnosti sítí s nízkým zpožděním doručení. Čím je síť větší, tím je bezpečnější a zároveň je náročnější analýza síťového provozu.

Z hlediska toho, jak využívá útočník charakteristiky síťového provozu může být buď aktivní nebo pasivní.

Pasivní útočník využívá charakteristik síťového provozu, které jsou implicitně jeho součástí a nesnaží se do těchto charakteristik nijak zasahovat. Díky tomu může být složitější takového útočníka vystopovat, protože se příliš neprojevuje.

Aktivní útočník naopak explicitně přidává do síťového provozu vlastní charakteristiky, které mu umožní lépe pozorovat průchod dat sítí. Aktivní útočník musí mít k dispozici

[1] http://www.ietf.org/rfc/rfc3917.txt

[2] *Stepping stones* jsou v sérii umístěné bezpečností systémy, které slouží jako autentizační servery za účelem zřízení vyšší úrovně zabezpečení.

více prostředků k vedení útoku než útočník pasivní, např. musí mít pod kontrolou uzly anonymizující sítě.

4.3 Princip korelačního útoku

Cílem korelačního útoku je zjistit závislost vstupního a výstupního datového toku a to buď na úrovni jednotlivých uzlů anonymizující sítě a nebo na jejich okrajích. Další možností je uvažovat část anonymizující sítě (vybrané uzly na cestě) jako atomický celek a na jeho vstupu a výstupu data korelovat.

Předpokladem útoku je, že má útočník k dispozici charakteristiku vstupního datového toku. Ta je vstupem korelační metody společně s charakteristikou výstupního datového toku a výsledkem je míra závislosti obou datových toků.

Korelační útok patří do třídy útoků analýzy síťového provozu. Úspěšnost korelačního útoku závisí na množství nasbíraných dat a na tom, jak zřetelné charakteristiky tato data mají [51]. Platí, že čím více dat má útočník k dispozici, tím jsou výsledky korelační metody přesvědčivější. S tím souvisí i způsob vzorkování síťových dat a délka časového intervalu pozorování.

Sítě s vysokým zpožděním doručení dat jsou odolné i vůči globálním útočníkům a korelace dat v rámci těchto sítí nebo na jejich okrajích není účinná. Naopak sítě s nízkým zpožděním doručení dat jsou vůči korelačnímu útoku zranitelné, kvůli požadavku na nízké zpoždění doručení dat je možné sledovat s využitím analýzy síťového provozu charakteristiky přenášených datových toků.

Murdoch et al. [35] popisuje možnost korelace na různých úrovních abstrakce v anonymizující sítí. Na vyšší úrovni abstrakce pracuje útočník s anonymizující sítí jako s černou skříňkou a uvažuje pouze charakteristiky datových toků na vstupu a výstupu sítě. Na nižší úrovni abstrakce lze provádět inspekci síťového provozu přímo uvnitř anonymizující sítě, sledovat charakteristiky síťového provozu na vybraných linkách a tak sledovat průchod datového toku sítí, což může vést ke snížení poskytované úrovně anonymity a nebo k odhalení komunikujících stran.

Korelaci na nižší úrovni abstrakce (uvnitř sítě) lze klasifikovat do dvou kategorií v závislosti na tom, zda mají tyto útoky k dispozici informaci o jednotlivých datových tocích v rámci sítě či nikoliv [51]. První kategorie útoků získává informaci o časování jednotlivých datových toků buď z kompromitovaných uzlů [32], nebo z kompromitovaného serveru [35]. Druhá kategorie útoků se snaží nalézt známý vstupní datový tok v množině agregovaných výstupních toků [19]. V takovém agregátu pak bývají datové toky vzájemně nerozlišitelné a cílem útočníka je zjistit, zda se v něm nachází i jím sledovaný vstupní datový tok.

Korelace na vyšší úrovni abstrakce (na vstupu a výstupu anonymizační sítě) pracuje s faktem, že komunikace uživatele se sítí Tor je silně zašifrovaná a tedy považována za bezpečnou, ale na výstupu sítě Tor už může být komunikace v nešifrované otevřené podobě (pokud uživatel nekomunikuje zabezpečeně pomocí např. protokolu TLS nebo tunelování síťového provozu). Syverson popisuje [43] zranitelnost Toru vůči *end-to-end correlation útoku.*

V návrhovém dokumentu Toru [23] autoři uvádějí několik možných typů útoků, které spadají do kategorie pasivního korelačního útoků na vyšší úrovni abstrakce a nebo na něj mohou vést.

Časová korelace: Nízké zpoždění a absence dávkového přeposílání má pozitivní dopad na časové charakteristiky datových toků. Ty nejsou nijak výrazně zkresleny.

Objemová korelace: Počítání paketů je podobně efektivní jako časová korelace, ale mechanismus *leaky pipe* by mohl být problematický.

Nechráněné aplikační protokoly: Tor byl navržen jako univerzální anonymizující síť pro aplikační protokoly postavené nad TCP, ale nedokáže ochránit před tím, když je anonymita narušena na úrovni těchto aplikačních protokolů. Pokud tedy existuje zranitelnost na úrovni aplikačního protokolu, může útočník využít tyto citlivé informace k odhalení pravé identity uživatele.

Autoři Toru uznávají [5], že je Tor vůči korelačnímu útoku zranitelný, je to podloženo řadou korelačních metod, které jsou podrobněji popsány dále v textu. Nicméně zůstává otevřenou otázkou, jaké množství dat na síti musí útočník vidět, aby byly výsledky těchto korelačních metod přesvědčivé.

4.4 Podmínky pro korelační útok v síti Tor

Pro útočníka je důležité, aby měl informace o tom, jakým způsobem jsou zkresleny charakteristiky datových toků při průchodu sítí, protože na základě toho může vhodně specifikovat parametry korelační metody a tím dosáhnout lepších výsledků.

Z níže popsaných podmínek vyplývá, že realizace korelačního útoku v síti Tor je pro pasivního útočníka možná a podmínky jsou příznivé.

4.4.1 Zkreslení charakteristik přenášených datových toků

Tor jako síť s nízkým zpožděním doručení dat se nesnaží odolávat globálním útočníkům, kteří se mohou dostat do vhodné pozice pro realizaci korelačního útoku. Snaží se chránit pouze před útočníkem s omezeným přístupem k Internetu. Je to kompromis mezi zpožděním sítě a poskytovanou úrovní anonymity.

Tor zajišťuje dopředné utajení, útočník nemůže zpětně dešifrovat již jednou proběhlou komunikaci. Tím pádem musí data potřebná ke korelaci získávat především analýzou síťového provozu.

Architektura Toru je podobná klasickým sítím přepínání okruhů. Není přesně jasné [23], jaký je vztah mezi zvýšením úrovně anonymity za cenu vyšších latencí či zkreslení charakteristik síťového provozu. Z tohoto důvodu Tor nepoužívá na směrovačích (OR) strategii, která by přidávala zpoždění, měnila pořadí zasílaných zpráv, zarovnávala jejich velikost a nebo přidávala vycpávkový provoz. Zkreslení charakteristik datových toků Tor implicitně ponechává na integrovaném mechanismu řízení zahlcení.

Na OR je sice implementována technika *token bucket* pro usměrnění datového toku, ale stále se mohou v síťovém provozu objevit špičky (dány maximální přenosovou kapacitou každého OR) a může tak stále dojít k saturaci síťového spojení některých OR. Saturaci pak řeší integrovaný mechanismus pro řízení zahlcení. Strategie na OR by mohla ovlivnit charakteristiku datových toků tím, že při výběru další buňky k přeposlání uvažuje priority virtuálních okruhů (pro každý okruh má OR vlastní frontu). Dává přednost těm okruhům, které tolik nevytěžují přenosovou kapacitu daného OR. V případě souvislého přenosu bloku dat např. protolem BitTorrent se zcela jistě tato vlastnost projeví více než u běžného surfování na webu. Existuje zde přímá závislost mezi požadavky anonymizovaného protokolu na propustnost (*goodput*) a mírou zpoždění datového toku.

4.4.2 Aktivní vs. pasivní útočník

Všechny linky v síti Tor používají zabezpečenou komunikaci pomocí TLS, externí útočník nemá možnost modifikovat charakteristiku datových toků a tím získat lepší podmínky pro korelaci. Pro interního útočníka, u kterého se předpokládá, že má pod kontrolou kompromitované OR, je situace o něco lepší, ale výhodu při korelaci mu to stejně nepřináší. Tor kontroluje integritu zasílaných dat pouze na koncových uzlech virtuálního okruhu (typicky na vstupu a výstupu sítě), je tedy možné, aby interní útočník vhodně upravil charakteristiku datových toků při průchodu sítí. Autoři Toru tento fakt berou na vědomí [23] a tvrdí, že takto útočník nezíská žádnou další informaci proti pasivní korelaci. Navíc by nasazení protiopatření v podobě kontroly integrity i v rámci okruhu vyžadovalo zvýšení režie.

4.4.3 Detekce síťových toků Toru

Před vlastním korelačním útokem musí být útočník schopen rozeznat síťový provoz Toru. Síťový provoz je přenášen v buňkách o pevné velikosti 512 B pomocí protokolu TCP, pokud není naplno využita přenosová kapacita mezi OR, lze pozorovat zvýšené množství paketů s délkou užitečných dat o velikosti 512 B a nebo v násobcích této délky. Protože jsou data šifrovaná, mají vysokou entropii. Také může pomoci fakt, že seznam všech OR v síti Tor je veřejně dostupný [39].

Problémem je, že se Tor dokáže maskovat za jiný běžný síťový provoz. Při navazování spojení maskuje Tor procedury, které provádí v rámci TLS ustavení spojení, takovým způsobem, že se jeví jako HTTPS komunikace. Další možností je, že uživatel využije funkcionality *pluggable transports* rozšíření, pomocí kterého může maskovat síťový provoz Toru za libovolný jiný protokol.

4.5 Lokalita provozu

Pro realizaci korelačního útoku je nutné, aby byl útočník v pozici, kdy může sledovat vstupní a výstupní datový tok anonymizační sítě. Vhodná pozice útočníka je ovlivněna lokalitou provozu, velikostí anonymizační sítě a dostupnými prostředky útočníka.

4.5.1 Velikost anonymizační sítě

Syverson et al. definovali [44] složitost korelačního útoku v závislosti na velikosti anonymyzující sítě. Čím je tato síť větší, konkrétně čím více kombinací vstupních a výstupních uzlů sítě musí útočník uvažovat, tím je pro něj realizace korelačního útoku obtížnější.

Pokud je první a poslední OR okruhu vybírán náhodně z množiny všech uzlů, pravděpodobnost, že budou útočníka je $\frac{c^2}{n^2}$, kde c je počet uzlů, které má útočník pod kontrolou a n je celkový počet uzlů. Protože Tor používá *guard* OR, má útočník pravděpodobnost, že úspěšně kompromituje okruhy uživatele $\frac{c}{n}$, pokud bude jeho OR vybrán jako *guard*. Pokud se tak ale nestane, budou všechny další okruhy uživatele, které půjdou přes strážce, bezpečné.

4.5.2 Diverzita AS a IXP při výběru cesty

Feamster et al. zkoumají [25] možnosti útočníka na úrovní autonomního systému (AS). Tvrdí, že geograficky odlišné cesty mohou nepříznivě ovlivnit anonymitu, protože tyto cesty

ačkoliv prochází mnoha AS, tak je u nich více pravděpodobné než u kratších cest, že začínají i končí v tom stejném AS.

Edman et al. zkoumají [24] různorodost výběru cesty skrze AS a navrhují algoritmus výběru cesty, který chrání před korelačním útokem na úrovni AS.

Akhoondi et al. navrhují [7] metodu založenou na geografickém výběru Tor směrovačů nazvanou *LASTor*, která zajišťuje diverzitu AS při výběru cesty a spoléhá přitom na stručné Internetové atlasy. Wacek et al. poukazuje [47] na to, že stejný AS se může objevit na obou koncích okruhu v nejméně 18 % případů.

Murdoch et al. poukazuje [36] na to, že zajištění diverzity AS při výběru cesty virtuálních okruhů je nedostatečné protiopatření vůči korelačním útokům útočníků, kteří mohou sledovat více síťového provozu než jednotlivé AS. Síťový provoz mezi AS je směrován internetovými ústřednami (IXP) a ústředna tak může pozorovat síťový provoz procházející mezi více AS.

Juen navrhuje [31] rafinovaný algoritmus výběru cesty, který poskytuje diverzitu jak AS, tak IXP. Nicméně žádný z těchto návrhů na vylepšení algoritmu výběru cesty zatím Tor nezahrnul do návrhu vlastní sítě.

4.5.3 Vliv prostředků útočníka na získání vhodné pozice pro korelaci

Johnson et al. se zabývají [29] touto problematikou a odpovídají na otázku jak moc lokalita provozu a chování uživatelů ovlivňují úspěšnost útočníka a tím pádem snižují úroveň poskytované anonymity uživatelům.

Definují dva typy útočníků, první typ je běžný útočník, který má pod kontrolou určité množství kompromitovaných OR s dostupnou přenosovou kapacitou a druhý typ je útočník na úrovni vyšších síťových celků jako je AS nebo IXP. Dále definují třídy uživatelů na základě toho, jakým způsobem Tor využívají (prohlížení webu, BitTorrent, ...).

Zavádějí také nové metriky, které říkají, s jakou pravděpodobností se vyskytne útočník ve vhodné pozici pro korelaci během určité doby a také jaká je pravděpodobnost, že bude kompromitována nějaká část všech datových toků z celkového počtu pro různé třídy uživatelů.

Implementovali simulátor výběru cesty sítí Tor nazvaný TorPS, který využívá data dřívějších konsensů adresářových serverů sítě Tor. Každému datovému toku v rámci testování je přidělen okruh za stejných podmínek, v jakých byla síť Tor v době platnosti použitého konsensu.

Studie využívá faktu [29], že v Německu je Tor využíván hlavně pro přístup k destinacím, které jsou taktéž v Německu, tím je pro reálného útočníka přístup k datovému toku na vstupu i výstupu sítě lehčí. Dá se očekávat, že podobné to bude i v jiných zemích.

Úspěšnost běžného útočníka a doba trvání útoků závisí na zdrojích, které má k dispozici. Pokud se jedná o OR, je důležitý jejich počet, přenosová kapacita, kterou nabízejí a velmi důležitý je také status (*guard, exit, stable*). Pro útočníka provádějícího korelaci jsou nejzajímavější směrovače se statusem *guard* nebo *exit*. Primárním cílem útočníka je zvýšit pravděpodobnost, že právě jeho síťové zdroje budou vybrány při vytváření virtuálních okruhů, které má zájem pozorovat.

Útočník na úrovni AS nebo IXP se nesnaží, aby byl vybrán nějaký z jeho kompromitovaných směrovačů, ale využívá své pozice, ze které může přímo sledovat síťový provoz na lince klient – strážce i na lince výstupní uzel – cílová destinace. Z jejich výsledků vyplývá, že čím více AS, IXP má útočník pod kontrolou, tím se snižuje doba potřebná, aby kompromitoval první datový tok.

IXP mají výrazně nižší pravděpodobnost odhalení datových toků uživatele, ale na druhou stranu je provedení korelačního útoku u IXP výrazně jednodušší než u AS. Zatímco AS pokrývají velkou geografickou oblast a síťový provoz nemusí jít těmi stejnými směrovači při cestě k cíli a zpět, tak IXP jsou geograficky koncentrovány. Takže ve výsledku IXP sice představují pro uživatele menší hrozbu, co se týče úspěšnosti korelace, ale provést pomocí nich korelaci je podstatně jednodušší.

Z výsledků vyplývá, že 80 % všech typů uživatelů Toru může být deanonymizováno útočníkem na úrovni OR, v průběhu šesti měsíců. Útočníkovi na úrovni AS se může podařit identifikovat až 100 % uživatelů v souvisejících lokacích již během tří měsíců (95 % pro IXP). Také ukazují, že útočník, který má pod kontrolou dvě AS, snižuje medián doby potřebné k odhalení identity o celý řád. Z tří měsíců pro běžného uživatele (prohlížení webu) na jeden den a z tří měsíců na stěží jeden měsíc pro BitTorrent uživatele.

Kapitola 5

Současné korelační metody

V této kapitole jsou popsány existující korelační metody. Některé metody byly ověřeny pouze v simulaci nebo v laboratorním nasazení sítě Tor, jiné přímo v reálném provozu a nakonec jsou i metody, které byly popsány pouze teoreticky.

Všechny uvedené korelační metody byly navrženy pro sítě s nízkým zpožděním doručení dat, některé konkrétně pro Tor, jiné jsou obecnější a lze je aplikovat i na jiné anonymizující systémy.

Uvádím i metody, které byly prezentovány již před delší dobou a Tor na ně zareagoval nasazením různých protiopatření, nicméně i když momentálně nepředstavují již tak velkou hrozbu, stále mohou být základem pro metody lepší a sofistikovanější.

Nejdříve definuji požadované vlastnosti korelační metody, viz sekci 5.1 a ve zbytku kapitoly popisuji jednotlivé korelační metody.

5.1 Definice požadovaných vlastností metody

Wang et al. definovali [48] matematicky korelační metodu pro určení závislosti dvou datových toků. Od korelační metody je požadováno, aby co nejpřesněji určila závislost dvou datových toků. Označím-li Tor_{in} jako množinu všech datových toků na vstupu sítě Tor, Tor_{out} jako množinu všech datových toků na výstupu sítě Tor, tak ideální korelační funkce je definována jako $CF : Tor_{in} \times Tor_{out} \rightarrow \{0, 1\}$.

$$CF = \begin{cases} 1 & \Leftrightarrow \text{ datové toky } \textbf{jsou} \text{ v korelaci (shodné)} \\ 0 & \Leftrightarrow \text{ datové toky } \textbf{nejsou} \text{ v korelaci} \end{cases}$$

Charakteristiku datového toku lze modelovat metrickou funkcí $M : F \times P \rightarrow Z$, kde F je doména datových toků, P je vybraná doména parametrů datových toků a Z je doména korelační metody. Na základě metriky lze postavit ohodnocenou korelační funkci $CVF : Z \times Z \rightarrow \langle 0, 1 \rangle$, kde výsledkem je reálné číslo mezi 0 a 1. Touto funkcí CVF, viz vzorec 5.1, lze aproximovat CF zavedením prahové hodnoty δ takové, že $0 \leq \delta \leq 1$.

$$CVF = (M(f_i, p), M(f_j, p)) \geq \delta \Rightarrow f_i \text{ je v korelaci s } f_j \tag{5.1}$$

K tomu je potřeba nalézt M, p, CVF a δ takové, které splňují podmínky formule 5.2.

$$\forall f_i, f_j \in F : CVF(M(f_i, p), M(f_j, p)) \geq \delta \Leftrightarrow f_i \text{ je v korelaci s } f_j \tag{5.2}$$

Klíčem je nalézt takové unikátní charakteristiky datového toku, které jsou invariantní při průchodu sítí Tor, neovlivněné šifrováním a unikátní pro každý datový tok. Na těchto je následně možné postavit spolehlivou korelační metodu.

Důležité je, aby výběr unikátních vlastností datových toků zajistil, že bude míra správných pozitivních výsledků (*true positives*) vysoká a míra falešných pozitivních výsledků (*false positives*) nízká.

5.2 Počítání paketů pro samostatný datový tok

Serjantov et al. [40] analyzovali možnost korelace pomocí počítání paketů, kdy na vstupní a výstupní lince mixu lze pozorovat osamělý datový tok. Využívají počítání paketů v rámci subintervalů, které jsou na sobě vzájemně nezávislé. Pokud je míra výskytu paketů vstupního datového toku téměř stejná jako u výstupního datového toku, je mezi nimi závislost. Diskutovali také další rysy síťového provozu, které lze využít ke sledování datového toku.

Předpoklady metody:

- Na každé vstupní a výstupní lince mixu se nachází pouze jediný datový tok a počet paketů datového toku zůstává při průchodu mixem stejný.
- Útočník má přístup k linkám na cestě anonymizovaného spojení.
- Interval pro počítání paketů je mnohem větší než zpoždění na mixu.
- Počty paketů v intervalu na vstupní a výstupní lince nebudou totožné, protože některé pakety byly v mixu ještě před začátkem intervalu a jiné tam zůstanou po skončení intervalu.
- Interval je mnohem menší než střední doba mezi vytvořením nového spojení na páru sledovaných linek.

Autoři uvádějí pravděpodobný počet osamělých datových toků v rámci anonymizujícího systému na základě průměrného počtu datových toků, které uživatel iniciuje.

Úspěšnost metody klesá v případě, kdy se na linkách sledovaného spojení objeví i jiné datové toky nebo vycpávkový síťový provoz.

Metoda je použitelná i na *end-to-end* korelaci, v tomto případě je ale náchylná na fluktuace ve zpoždění. Délka intervalu pro počítání paketů musí být větší než celkové zpoždění na všech mixech po cestě a zpoždění na linkách mezi nimi.

5.3 Časová okna a křížový korelační koeficient

Levine et al. [32] předpokládá kompromitované mixy. V této metodě má útočník přístup k informaci o časování na úrovni výskytu jednotlivých buněk Toru. Ke konstrukci vektorů pro korelaci nepoužívají mezipaketové rozestupy, protože tento způsob je velice citlivý na ztrátu paketů. Místo toho datový tok z rozdělí na nepřekrývající se časová okna o velikosti w (sekund) a počítají výskyt paketů v těchto oknech. Takto vytvořený vektor časové série je doplněný nulami, aby byla reprezentace datových toků časově synchronizována.

Nad vektory časové série x a y definují křížový korelační koeficient při zpoždění d, viz vzorec 5.3, kde z_i je i-tý prvek časové série $\{z_i\}_w$ a $\bar{z}$ je průměrná hodnota ze všech z_i

v $\{z_i\}_w$. V jejich další analýze použili hodnotu zpoždění 0 a velikost okna 10 sekund. Tyto výchozí hodnoty se podle nich ukázaly být efektivní.

$$r(d) = \frac{\sum_i (y_i - \bar{y})(x_{i+d} - \bar{x})}{\sqrt{\sum_i (y_i - \bar{y})^2} \sqrt{\sum_i (x_{i+d} - \bar{x})^2}} \tag{5.3}$$

Pokud platí $|r(d)| > t$, kde t je vhodně zvolená prahová hodnota, útočník může říci, že datové toky jsou v korelaci. Vysoká prahová hodnota zvyšuje míru *false negatives* a snižuje míru *false positives*, zatímco nízká prahová hodnota ovlivňuje výše zmíněné přesně naopak. Standardní metrikou pro tuto situaci je *crossover error rate* (Levin et al. ji nazývají jako jako *equal error rate*), při které je míra *false positives* stejná jako míra *false negatives*.

DeFabbia-Kane tuto metodu upravil [20] a uvažoval korelační formuli, která je normalizovaným skalárním součinem dvou vektorů. Hodnota skalárního součinu je vyšší čím více paralelní vektory v prostoru jsou, takže vyšší výsledná hodnota tohoto koeficientu znamená vyšší pravděpodobnost, že dva vektory jsou v korelaci. Problematické je, že hodnota skalárního součinu je proměnlivá s délkou vektorů, je proto užitečné ji normalizovat. K tomu využil vztahu mezi skalárním součinem a úhlem mezi dvěma vektory viz vzorec 5.4, kde a, b jsou vektory definované jako $a = y - \bar{y}$ a $b = x - \bar{x}$, $|a|$ je délka vektoru a a θ je úhel mezi nimi.

$$a \cdot b = |a|\,|b|\,cos(\theta) \tag{5.4}$$

Úhel θ izoloval, viz vzorec 5.5. Definiční obor funkce $arccos$ je $\langle -1, 1 \rangle$. Uvedený argument této funkce je křížový korelační koeficient, který spadá do jejího definičního oboru. Závěrem je, že vektory jsou paralelní, když je hodnota argumentu 1 a antiparalelní, když je hodnota argumentu -1.

$$\theta = arccos\left(\frac{a \cdot b}{|a|\,|b|}\right) \tag{5.5}$$

V některých případech je výsledek této korelační metody nedefinovaný, konkrétně když jsou všechny prvky časové série rovny jejich průměrné hodnotě, pak dojde k dělení nulou.

5.4 Daneziho algoritmus

Danezi navrhuje [19] koeficient míry pravděpodobnosti, že se vstupní datový tok nachází v agregátu výstupních datových toků. K výpočtu koeficientu je nutné mít informaci o ostatních výstupních datových tocích.

Metoda využívá modelu zpoždění celé sítě a funguje na principu, kdy útočník pozoruje datový tok např. odpověď webového serveru zpět k uživateli, který je autorem požadavku. Tento datový tok může být reprezentován jako funkce objemu datového toku v čase. Tato funkce je konvoluována s exponenciální funkcí zpoždění, výsledkem je šablona, která předpovídá, jak bude datový tok vypadat v anonymizující síti. Všechny linky v síti jsou pak porovnány s tímto odhadem a je určena míra podobnosti s danou šablonou datového toku. Podle toho lze uzly klasifikovat jako potenciální první, druhý a třetí uzel na cestě daného spojení.

Problémem tohoto útoku je, že útočník musí pozorovat všechny uzly, síťové linky a musí být schopen zaznamenávat metadata o provozu ve velice nízké granularitě. Nicméně je tento útok robustní (když je méně dat, je jich jen část nebo nejsou požadované granularity,

útok trvá déle) a i útočník s omezenými zdroji je stále schopen sledovat náhodně nějakou komunikaci.

Danezi předpokládá, že síť Tor může být modelována jako síť mixů se zpožděním, které je předvídatelné. Výstupní tok považuje za kompozici vstupního toku, na který je aplikována funkce zpoždění průchodu sítí a toku běžného síťového provozu na pozadí, který se objevuje nezávisle na vstupním toku.

U dvou výstupních toků v časovém intervalu $[0, T]$ lze pozorovat zprávy v časech $X_{1\cdots n}$ ve výstupním toku uzlu X a $Y_{1\cdots n}$ ve výstupním toku uzlu Y. Pro odhad, který z výstupních toků obsahuje vstupní tok, jsou uvažovány hypotézy H_0 a H_1 vzhledem k pozorovaným zprávám na výstupech v uvedených časech. První hypotéza říká, že vstupní tok je proložen v prvním výstupním toku a druhá, že ve druhém. Každý výstupní tok je aproximován rozložením výskytu počtu paketů na výstupu uzlu v daném časovém intervalu. Pro výstupní tok uzlu X, viz výraz 5.6 (obdobně pro Y):

$$C_x(t) = \frac{\lambda_f (d * f)(t) + (\lambda_x - \lambda_f) U(t)}{\lambda_x} \tag{5.6}$$

Sledovaný vstupní tok popisuje funkce $f(t)$, předpokladem je, že všechny zprávy popsané funkcí $f(t)$ náleží jednomu vstupnímu toku. Zpoždění průchodu paketu sítí reprezentuje funkce $d(x)$, ta se může v průběhu vývoje sítě měnit a může být empiricky odhadnuta. λ_x je střední hodnota počtu výskytů odchozích paketů za jednotku času ve výstupním toku, λ_f je střední hodnota počtu výskytů příchozích paketů za jednotku času v daném intervalu ve vstupním toku, $U(t)$ je uniformní rozložení ostatního (šumového) síťového provozu v intervalu $[0, T]$, u je hodnota $U(t)$ v čase t, $C_x(t)$ je odhadované rozložení paketů na výstupu uzlu X v daném intervalu a $(d * f)(t)$ je konvoluce funkce vstupního signálu a funkce zpoždění udávající pravděpodobnost, že zpráva zpožděná funkcí $d(x)$ je výstupem toku $f(t)$ v čase t, viz výraz 5.7.

$$(d * f)(t) = \int d(x) f(t - x) dx \tag{5.7}$$

Výpočtem koeficientu pravděpodobnosti podle vzorce uvedeného v jejich článku se rozhodne, která z hypotéz H_0 a H_1 je pravdivá. Je-li podmínka splněna platí hypotéza H_0, v opačném případě H_1.

Daneziho algoritmus v základní podobě porovnává pouze dva výstupní toky, v praxi je porovnávaný počet škálovatelný. K výběru toho nejvhodnějšího kandidáta může být použita stromová vyhledávací struktura nebo jiný algoritmus výběru maxima.

Johnson et al. experimentálně tuto metodu ověřili [30]. Při realizaci experimentu byl pro funkci zpoždění použit Gaussovský model a autoři experimentu předpokládají, že vzhledem k počtu paketů procházejících jednou linkou a jejich rozložení zpoždění lze podobné rozložení zpoždění očekávat i u ostatních linek, ale s jiným měřítkem. Síťový provoz na pozadí byl generován Poissonovým rozložením, které lépe aproximuje skutečný výskyt šumového provozu. Výsledky experimentu ukázaly, že Daneziho algoritmus funguje. V experimentu úspěšně detekoval všech 24 výstupních toků. Danezi klade důraz na přesný model funkce zpoždění, ale z výsledků experimentů vyplývá, že je postačující i méně přesný model zpoždění.

V původním návrhu algoritmu [19] je řečeno, že algoritmus nepočítá s uzly, které by měnily své chování (zpoždění) v závislosti na míře příchozích paketů a počtu paketů ukládaných do fronty. Každopádně by to nemělo být překážkou, funkce zpoždění by pak byla navíc závislá právě na těchto dalších proměnných.

K úspěšnému provedení útoku musí mít útočník možnost pozorovat samostatný vstupní tok, aby mohl vybudovat model, který později použije pro detekci. Pokud bude útočník znát vztahy mezi pakety na pozorovaných linkách (kompromitovaný uzel), může toho využít ve svůj prospěch a statistickým testem vyloučit vycpávkový provoz. Útok je dostatečně efektivní na to, aby mohl být aplikován globálním pasivním útočníkem na rozsáhlé sítě. Má-li navíc útočník přístup ke kompromitovaným uzlům a nebo možnost tvarovat průchozí provoz, bude útok o to efektivnější. Pokud se anonymizující síť nebrání dostatečným objemem vycpávkového provozu a změnou přidaného zpoždění, je vůči tomuto útoku velice zranitelná a poskytovaná úroveň anonymity je diskutabilní.

5.5 Vzájemná informace a frekvenční analýza datových toků

Zhu et al. navrhují dvě třídy korelačních metod [51], jmenovitě metody postavené na časové doméně a metody postavené na frekvenční doméně. První kategorie metod využívá ke korelaci datových toků statistické informace o míře výskytu počtu paketů v čase a druhá kategorie metod využívá ke korelaci datových toků spektrální funkci počtu výskytu paketů v čase, tedy jaké jsou frekvence počtu výskytů paketů v rámci časového intervalu.

Prezentované metody spadají do kategorie útoků, kdy se útočník snaží nalézt vstupní datový tok v agregovaném výstupním datovém toku a určit odpovídající výstupní linku mixu.

Na rozdíl od Daneziho útoku [19] není potřebná informace o tom, jakou část agregátu tvoří další výstupní datové toky.

Zaměřují se především na anonymizující sítě, které explicitně implementují různé strategie dávkového přeposílání. Tor do této skupiny nepatří, spoléhá na vnitřní mechanismus řízení zahlcení, který implicitně zkresluje charakteristiku síťového provozu, nicméně zahrnují do testování navrhovaných metod i ty strategie, které s dávkovým přeposíláním nepracují.

Autoři korelační metody testují [51] pro 7 různých strategií dávkového přeposílání na jediném mixu, ale diskutují [49] také zranitelnost větších anonymizujících sítích tzv. *continuous mixes* vůči těmto metodám. Autoři ukazují [50], jak lze jejich práci rozšířit v kontextu distribuovaných anonymizujících sítí. Obě metody pak vyžadují pozici globálního útočníka, musí mít možnost pozorovat všechny vstupní a výstupní uzly sítě a to je při dnešní velikosti sítě Tor nepravděpodobné. Zmiňují i tu možnost, že pokud útočník uvažuje část anonymizující sítě (např. Tor) jako jeden atomický celek, může uvedené korelační metody použít přímo bez dalších modifikací.

Předpokladem je, že útočník má k dispozici charakteristiku vstupního datového toku, zná topologii anonymizující sítě a použité strategie na mixech, které ale nemusí znát detailně. To je v kontrastu s Daneziho útokem [19], který spoléhá na nezávislost časování paketů mezi datovými toky (což není případem většiny mixů). Útočník nemůže přímo korelovat konkrétní paket vstupního datového toku s jiným paketem ve výstupním datovém toku a to z toho důvodu, že časování paketu je zkresleno dávkovým zpracováním a korelace na základě obsahu paketu nebo jeho velikosti je nemožná kvůli šifrování respektive zarovnání velikosti paketu. A anonymizující síť neposílá vycpávkový provoz (tento předpoklad Tor splňuje).

Korelace probíhá následovně:

1. Útočník sbírá informace o všech paketech na vstupní i výstupní lince. Pro každý paket zaznamenává čas jeho příchodu. Časy příchodu paketů pro r vzorků v rámci

vzorkovacího intervalu tvoří sérii $A_i = (a_{i,1}, \ldots, a_{i,r})$, ve které $a_{i,k}$ je čas příchodu k-tého paketu na lince i. Vzorkovací interval je zvolen tak, aby obsáhl větší počet dávek. Obdobně lze vyjádřit časy příchodu paketů v sérii $B_j = (b_{j,1}, \ldots, b_{j,s})$ pro výstupní linku j.

2. S notací uvedenou v předchozím kroce dále útočník analyzuje časové série $A_i s$ a $B_i s$ za účelem zjištění vzájemné závislosti. Aby byla analýza efektivní, musí být tyto série převedeny na tzv. vektory vzorů. Efektivní transformace závisí na dávkové strategii, kterou mix implementuje. Po transformaci získává útočník vektor vzoru $X_i = (x_{i,1}, \ldots, x_{i,q})$ respektive $Y_j = (y_{j,1}, \ldots, y_{j,q})$. Tyto dva vektory mají na rozdíl od předchozích sérií stejnou délku.

3. Definuje se funkce vzdálenosti $d(X_i, Y_j)$ pomocí, které se určí podobnost vstupního a výstupního datového toku, které jsou reprezentovány vektory vzorů. Definice funkce vzdálenosti je pro korelační analýzu klíčová.

4. Vlastní korelační analýza vybere tu výstupní linku na mixu, jejíž výstupní provoz je dle funkce vzdálenosti nejvíce podobný vektoru vzoru X_i pro sledovaný vstupní datový tok.

Autoři definují dvě metody pro extrakci vektorů vzorů. Cílem těchto metod je rozdělit vzorkovací interval na více subintervalů a vypočítat průměrnou míru množství síťového provozu v každém takovém subintervalu jako hodnotu výsledného vektoru vzorů. Metody se liší v tom, jakým způsobem rozdělují vzorkovací interval.

Získané vektory jsou korelovány pomocí funkcí vzdálenosti. Autoři definují dva typy těchto funkcí: první typ je založen na porovnávání hodnoty vzájemné informace (*mutual information*) (MI), která je teoretickou mírou závislosti dvou náhodných proměnných. Druhý typ je založen na frekvenční analýze.

Používají FFT nebo vlnovou transformaci na vzorcích X_i a Y_j, aby získali frekvenční spektrum X_i^F a Y_j^F. Poté aplikují metodu filtru shody na získaná spektra. Využívají faktu, že frekvenční složky vstupního datového toku jsou přeneseny i na agregovaný výstupní datový tok. Filtr shody je optimální filtr pro detekci signálu zanořeného v šumu. Optimální v tom smyslu, že poskytuje pro daný vstupní signál na výstupu maximální hodnotu odstupu signál–šum (SNR). Konkrétně přímou aplikací teorie filtrů shody definují funkci vzdálenosti $d(X_i, Y_j)$ jako inverzní detektor filtru shody $M(X_i^F, Y_j^F)$, viz vzorec 5.8, kde $\left\langle X_i^F, Y_j^F \right\rangle$ je skalární součin (*inner product*) X_i^F a Y_j^F.

$$d(X_i, Y_j) = \frac{1}{M(X_i^F, Y_j^F)} = \frac{1}{\frac{\left\langle X_i^F, Y_j^F \right\rangle}{\left\| Y_j^F \right\|}} \tag{5.8}$$

$$\left\| Y_j^F \right\| = \sqrt{\left\langle Y_j^F, Y_j^F \right\rangle}$$

Metody byly ověřeny experimentálně v laboratorních podmínkách. Výsledky ukazují, že pro všechny strategie na mixu stoupá míra správné detekce (přibližuje se ke 100 %) při dostatečném množství nasbíraných dat.

Byla ověřena i efektivita metod na větším počtu mixů v kaskádě a z výsledků vyplynulo, že úspěšnost detekce je větší u delších kaskád, protože jsou zde zřetelnější charakteristiky časování v síťovém provozu.

Na závěr autoři upozorňují na to, že je velice důležité vhodně zvolit velikost vzorkovacího intervalu. V experimentech volili velikost vzorkovacího intervalu v závislosti na RTT hodnotě pro datový tok FTP. Pro aplikačně limitované datové toky jako je SSH nebo webový provoz by měl být zvolen interval v závislosti na dynamice těchto datových toků.

5.6 Pravděpodobnost korelace datových toků s využitím Bayesovi formule

Murdoch et al. navrhli [36] metodu z pozice pasivního útočníka, která využívá Bayesovi formule. Každý datový tok p modelují Poissonovým rozdělením s časem začátku s, dobou trvání l a průměrným počtem paketů za sekundu r. Protože tento datový tok je přenášen sítí Tor a není tedy přímo pozorovatelný, uvažují místo něj datové toky x a y (na vstupu a výstupu sítě Tor), které jsou pro útočníka přímo pozorovatelné. Ty modelují nezávislými Poissonovými procesy na základě paramterů datového toku p.

Pomocí Bayesovi formule odvozují pravděpodobnost $P(T_k)$, viz vzorec 5.9, že x a y_k jsou součástí toho samého datového toku.

$$P(T_k|y_{1..n}, x) = \frac{P(y_{1..n}|T_k, x)}{\sum_i P(y_{1..n}|T_i, x)P(T_i|x)} \tag{5.9}$$

Spíše než absolutní pravděpodobnosti chtějí pouze relativní, a proto ignorují všechny faktory závislé na k, odtud odvozují vzorec pro výslednou pravděpodobnost, viz 5.10. $\Gamma(n) = (n-1)!$, n_y je celkový počet paketů v datovém toku y, $n_{xy_k} = n_x + n_{y_k}$, $l_x = x_{max} - x_{min}$ je pozorovaná délka toku x a $l_{xy_k} = max(x_{max}, y_{max} - min(x_{min}, y_{min})$ je celková pozorovaná délka x a y.

$$P(T_k|y_{1..n}, x) = \frac{P(x, y_k|T_k)}{P(y_k)} \sim \frac{\Gamma(n_{xy_k})}{2^{n_{xy_k}}\Gamma(n_{y_k})} \cdot \frac{n_{y_k}(n_{y_k}-1)}{n_{xy_k}(n_{xy_k}-1)} \cdot \frac{l_{y_k}^{n_{y_k}-1}}{l_{xy_k}^{n_{xy_k}-1}} \tag{5.10}$$

Formule se skládá ze tří částí, které jsou mezi sebou vynásobeny a výsledkem je pravděpodobnost. První část je založena na míře výskytu pozorovaných paketů v x a y, druhá část faktor korekce pro míru výskytu paketů a třetí část vyjadřuje závislost toků vzhledem k jejich délce, tím je zajištěno, že budou uvažovány pouze datové toky, které se vyskytly v blízkém čase a měly podobnou dobu trvání.

Tato korelační metoda byla původně zamýšlena k použití pro útočníky, kteří mají pod kontrolou IXP, vně sítě Tor a kteří mohou pozorovat pouze několik málo vzorků (1 z 2000 paketů pro každý datový tok) daného datového toku. Nicméně metoda je použitelná i uvnitř sítě Tor pro útočníky, kteří mají pod kontrolou nějaké OR.

Výsledky této metody jsou relativní a nejsou normalizovány, nelze říci, kdy je výsledek *dostatečně dobrý* na to, aby bylo možné říci, že jsou dva datové toky v korelaci. To je problematické v případech, kdy nemá útočník možnost pozorovat všechny datové toky v síti Tor.

5.7 Čas vytvoření okruhu a počet přenesených buněk

DeFabbia-Kane [20] zkoumal jakými faktory ovlivňuje Tor přenášené datové toky. Zjistil, že zpoždění okruhů je proměnlivé a čas vstupu datového toku do sítě Tor je kratší než čas

jeho výstupu ze sítě Tor. První z těchto faktorů může ovlivnit korelační algoritmy, které spoléhají na časování individuálních paketů, viz Levine et al. [32] a druhý může ovlivnit algoritmy, které spoléhají na celkové časování okruhu, viz Murdoch et al. [36].

Navrhuje novou korelační metodu, která je odolná vůči dvěma výše zmíněným faktorům. Bere v úvahu čas vytvoření okruhu a celkový počet buněk Toru přenesených daným okruhem. Korelační koeficient je definován vzorcem 5.11.

$$c = \frac{T - abs(x_{start} - y_{start})}{T} \cdot \frac{|x| + |y| - abs(|x| + |y|)}{|x| + |y|} \tag{5.11}$$

Z autorova měření vyplynulo, že čas vytvoření okruhu je relativně unikátní, a proto ho použil při definici vlastního korelačního koeficientu. Rozdíl časů vytvoření okruhu mezi datovými toky x a y vyjádřil procentuálně z pevně dané hodnoty T, která je větší než jím pozorované zpoždění v síti Tor (20 s).

Z pozice útočníka, který má k dispozici kompromitované uzly sítě Tor, má k dispozici informaci o počtu buněk přenesených okruhem. Tor zaručuje doručení všech zaslaných buněk, a proto budou počty stejné u datových toků, které jsou v korelaci.

Korelační koeficient má obor hodnot v rozmezí $\langle \infty, 1 \rangle$, ale hodnoty pod 0 značí, že čas vytvoření okruhu byl pro datové toky velice rozdílný, takže uvažuje pouze rozmezí $\langle 0, 1 \rangle$. Tento výsledek je pro dva korelované datové toky absolutní.

5.8 Korelace vektorů mezipaketových rozestupů

Wang et al. navrhli [48] korelační metodu, která nebyla přímo navržená pro anonymizační sítě, ale pracuje v podobných podmínkách. Z této metody vycházím při návrhu vlastní korelační metody, bude popsána blíže v kapitole o návrhu.

Metoda byla navržena pro korelaci datových spojení tvořící sérii *stepping stones*. V jednom spojení se jedná např. o protokol SSH a ve druhém např. o Telnet. Cílem metody bylo určit, zda tato dvě spojení patří do stejné série datových toků.

Princip metody je založen na postupném vyhledávání korelačních bodů uvnitř porovnávaných datových toků. Využívá k tomu časového okna, ve kterém je pro vyskytující se pakety extrahován vektor mezipaketových rozestupů. Tento vektor je následně vyhledáván ve druhém datovém toku, porovnání závislosti vektorů jsou definovány korelační funkce, které porovnávají podobnost těchto vektorů. Postupným pohybem časového okna dochází vyhodnocení lokálních podobností datových toků.

V základní podobě tato metoda není přímo použitelná na anonymizační sítě, protože autoři předpokládají, že se v korelovaných datových tocích promítnou pakety v poměru 1:1. Při testování narazili na skutečnost, že tomu tak být nemusí a extrahované vektory mezipaketových rozestupů jsou tímto zkreslené. V případech, kdy byl tento předpoklad splněn, byla metoda při určení závislosti datových toků velice efektivní.

5.9 Další metody

Murdoch et al. [35] navrhli metodu, ve které umožňují útočníkům s omezeným přístupem k síti, odhadnout, které uzly přeposílají anonymní datové toky. Metoda využívá časových signatur anonymních datových toků, pomocí kterých lze provádět sledování v síti Tor. Útočník, který nemůže pozorovat tyto signatury přímo na síti, místo toho zasílá vlastní datový

tok a pozoruje zpoždění na sledovaných uzlech sítě. Užívá faktu, že objem dat jednoho datového toku ovlivňuje zpoždění dalších datových toků přenášených tím samým uzlem (OR). Datové toky procházející tím stejným OR se vzájemně ovlivňují, protože spotřebovávají společné zdroje jako je procesorový čas nebo přenosová kapacita. Aby si ověřil, že některý sledovaný cílový datový tok je přenášen určitým uzlem, přesměruje datový tok, který může pozorovat přes ten daný uzel a pozoruje změny zpoždění. Předpokladem je, že útočník má pod kontrolou nějaký OR. Pomocí toho pak vytváří spojení, které prochází přes OR, jehož zátěž je cílem měření. Součástí spojení je sondovací síťový provoz, který měří latenci spojení a tím odhaduje zátěž na měřeném OR. Útočník může pozorovat spojení vstupující nebo vystupující do/z sítě Tor a výše popsanou techniku použít k určení, kterými uzly datový tok spojení prochází. Server, ke kterému uživatel přistupuje, může také zanést velmi specifické vzory síťového provozu, vytvořit šablonu a korelovat ji s objemem dat na OR. Jejich způsob analýzy síťového provozu může být také použit pro korelaci datových toků. Technika dovoluje provést analýzu na celé síti a tím aproximuje možnosti globálního pasivního útočníka.

Overlier a Syverson navrhli [37] metodu, jejíž principem je, že útočník působí v síti Tor zároveň jako klient a OR. Takto se může dostat do pozice, kdy jeho OR zprostředkuje spojení (okruh) mezi skrytým serverem (HS) a OR, které slouží jako *rendezvous point* (RP). Metoda rozšiřuje korelační metodu založenou na počítání paketů [40]. Navíc používají časovou informaci o tom, kdy která buňka byla přijata, odeslána a směr každé individuální buňky. Cílem útočníka je dostat pod kontrolu vstupní uzel okruhu směrem od HS k RP. Útočník provádí časovou korelaci na uzlu mezi HS a RP, aby si ověřil, že je součástí tohoto okruhu a aby přesně určil jeho pozici v okruhu. Pokud se mu to podaří, ví, že je součástí okruhu na některé ze tří pozic. Protože zná IP adresu RP, může odhadnout, zda je výstupním uzlem okruhu (pak může okruh jednoduše opustit a útok opakovat). Má-li jeho OR neznámou IP adresu na obou stranách okruhu, ví, že je buď strážcem nebo prostředním uzlem okruhu. Pomocí statistických metod a časování může svou pozici přesně určit a jakmile se dozví, že je strážcem, zná i IP adresu HS. Díky implementaci strážců do návrhu Toru byla šance na úspěch tohoto útoku výrazně snížena. Stejně tak adresářové servery nepřidělují uzlům v síti váhy na základě jimi reportované dostupné přenosové kapacitě, ale provádí ověřující měření, zda informace souhlasí.

Chakravarty et al. představili [14] metodu, ve které prezentovali nástroj *LinkWidth*, který umožňuje odhadnout dostupnou přenosovou kapacitu libovolné síťové linky bez přímého přístupu k této lince i bez nutnosti spolupráce s vhodně umístěným spolupracujícím uzlem. Útočník může nástroje využít k detekci indukovaných fluktuací síťového provozu na sledovaném okruhu. Fluktuace může vytvářet přímo server (pokud spolupracuje s útočníkem), směrovač nebo jiný uzel blízký serveru. Nebo lze využít cíleného DoS útoku na vhodně zvolenou linku, směrovač či samotný server. Předpokládají, že má útočník přístup pouze k několika uzlům na krajích sítě s dostatečně velkou přenosovou kapacitou k odhalení Tor uzlů příslušejících danému okruhu. V navržené metodě útočník neustále monitoruje přenosovou kapacitu linek všech Tor uzlů. Když anonymní uživatel kontaktuje server poskytující nějakou službu, datový provoz představující odpověď od serveru je uměle modulován (buď přímo serverem nebo směrovačem na zpětné cestě okruhu) a útočník může takto upravený provoz detekovat.

Bauer et al. [9] rozšiřují výše zmíněný útok na skryté servery [37]. Funguje pouze na uživatele, kteří si ještě nevybrali jejich preferované strážce. Útočník nasadí několik uzlů, které hlásí vysoké přenosové kapacity a doby provozu, ale ve skutečnosti jimi nedisponují. S velkou pravděpodobností bude útočník schopen úspěšně kompromitovat vstupní a vý-

stupní uzel okruhu. Jedná se o metodu korelace, která dokáže odhalit komunikaci klienta s jeho cílem ještě předtím, než se zašlou nějaká užitečná data. To je výhodné, protože kompromitované uzly, které nedisponují velkou přenosovou kapacitou, nemusí přenášet velké množství datových paketů. Dnes tato metoda již není efektivní, protože stejně jako v případě metody pro odhalení skrytých serverů stojí na faktu, že adresářové servery neověřují reportovanou přenosovou kapacitu od uzlů sítě Tor.

Blond et al. [12, 11] navrhli metodu, která využívá vlastností přenášeného aplikačního protokolu k odhalení IP adresy zdroje nebo k trasování Tor uživatele. Zároveň využívají skutečnosti, že odhalením jediného datového toku okruhu jsou automaticky všechny další přenášené datové toky asociovány s prozrazenou identitou. Útočník musí mít pod kontrolou jeden výstupní Tor směrovač. Útok spočívá v modifikaci odpovědí trackeru, k tomu je potřeba mít pod kontrolou veřejně dostupného BitTorrent klienta, který bude přijímat příchozí TCP spojení. Podmínkou pro tento útok je, že přenos sdíleného obsahu probíhá mimo síť Tor.

Craven et al. [18] představili metodu, ve které využívá algoritmů pro generování skrytých Markovských modelů (HMMs) z časové informace (mezipaketových rozestupů) datových toků a nástrojů pro efektivní detekci vzorů užitím těchto modelů. Tato metoda dává absolutní výsledek pro libovolné dvě dvojice porovnávaných datových toků. Otestována byla pouze laboratorně a na určitém typu síťového provozu (ping), kdy byly datové toky na základě vytvořených modelů správně korelovány. Autoři uznávají, že nasazení metody na reálné síti Tor bude vyžadovat další výzkum.

Kapitola 6

Návrh vlastní korelační metody

V této kapitole je popsán návrh vlastní korelační metody. Nejdříve je nastíněna myšlenka funkcionality metody, viz sekci 6.1. Ve zbývajících částech této kapitoly jsou rozebrány jednotlivé fáze, tak jak je metoda při korelaci datových toků postupně aplikuje. Explicitně uvádím, ze kterých existujících korelačních metod jsem vycházel, jakým způsobem jsem převzanou funkcionalitu modifikoval, aby byla navržená metoda použitelná v kontextu dříve definovaného pasivního útočníka.

6.1 Princip metody

Korelační metodu jsem postavil na tvrzení autorů Toru, že je zranitelný vůči časové a objemové korelaci dat. [23, 22, 4] V návrhu uvažuji dříve definovaného útočníka s cílem realizovat *end-to-end* korelaci, viz sekci 4.1. Nelze použít informace v hlavičkách protokolů, která jsou unikátní pro TCP spojení, protože užitečná data jsou přenášena dvěma různými TCP spojeními, na lince mezi klientem a strážcem jako *payload* TLS a na lince mezi výstupním uzlem a destinací jako *payload* TCP.

Z toho důvodu nelze použít mezipaketové rozestupy, protože přenášená užitečná data se v datových tocích přesně nepromítnou na korespondující pakety. Navíc díky proměnlivému zpoždění a rozptylu nejsou tyto rozestupy při průchodu sítí Tor zachovány, viz podsekci 3.7.4

I když nejsou užitečná data pro útočníka na lince mezi klientem a strážcem přímo viditelná, může pozorovat objem přenášených dat v čase. Pokud z tohoto objemu dokáže odfiltrovat režii TLS a Toru, je informace o objemu dat v čase dostatečně unikátní pro identifikaci datového toku.

Datový tok na lince mezi výstupním uzlem a destinací (X) považuji na úrovni TCP i IP za atomický[1] na rozdíl od datového toku na lince mezi klientem a strážcem (Y), který může přenášet (multiplexovat) i jiná zašifrovaná TCP spojení uživatele, dále označovaná jako agregát datových toků Y.

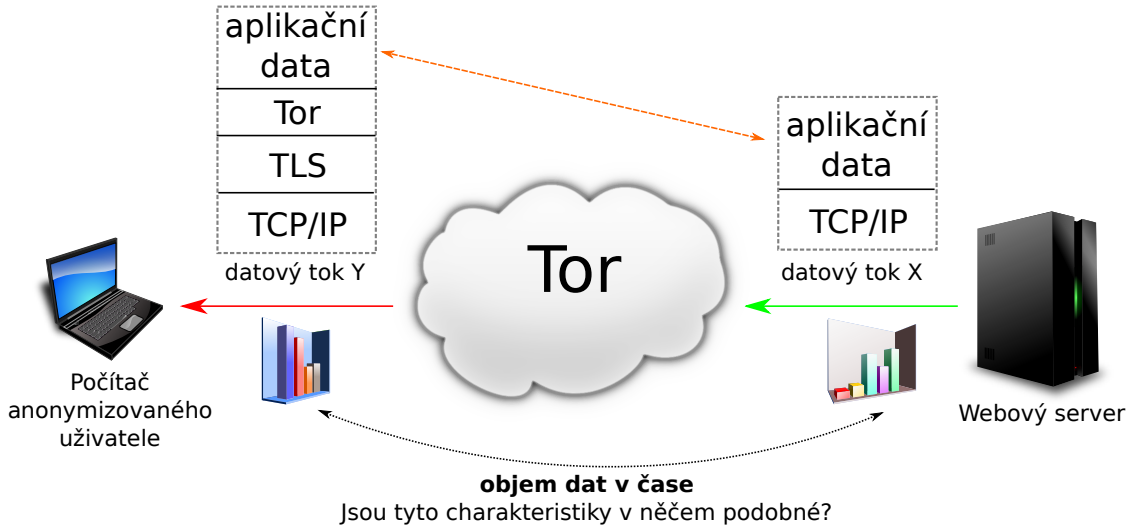
Vzhledem k atomičnosti datového toku X se jeho signatura promítne do datového toku Y buď celá nebo vůbec. Metoda koreluje jednosměrné datové toky a směr datových toků určuje, jestli bude vyhledávat signaturu datového toku X v čase $+\Delta t$ nebo v čase $-\Delta t$.

Návrh jsem přizpůsobil pro korelaci datových toků přenášejících HTTP ve směru odpovědi webového serveru, kdy X předchází v čase Y, viz obrázek 6.1. Klient může přes

[1] Za atomický označuji takový datový tok, který obsahuje pouze komunikaci mezi klientem (respektive OR) a danou destinací.

Tor přenášet libovolný TCP datový tok, ale ze statistik vyplynulo, viz podsekci 3.7.2, že nejčastěji je to právě HTTP. Tento typ datových toků je vhodný zejména kvůli výskytu nárazových objemových špiček, které mohou datový tok lépe identifikovat než např. druhý nejčastěji přenášený protokol BitTorrent. Tento přístup sledování odpovědi webového serveru byl použit i v Daneziho metodě, viz sekci5.4.

Navržená vlastní metoda porovnává objem datových toků X a Y v čase a využívá k tomu časových oken, tento způsob byl použit v řadě dříve popsaných korelačních metod, viz sekce 5.2, 5.3, 5.5. Zmiňované metody rozdělují datový tok na nepřekrývající se časová okna o zvolené velikosti a počítají výskyt paketů v těchto oknech. Ve výsledku porovnávají dva vektory, které představují charakteristiku celého datového toku. Takový přístup kvůli vysokému stupni agregace zkresluje unikátní charakteristiku datových toků a není citlivý na lokální podobnosti.



Obrázek 6.1: Ilustrace korelace datových toků přenášejících odpověď webového serveru.

Vycházel jsem z metody, která taktéž pracuje s časovými okny, ale zaměřuje se na lokální podobnost datových toků, popsána je v sekci 5.8. Z této metody jsem převzal způsob hledání korelačních bodů, který spočívá v nalezení takových pozic časového okna v X a Y, při kterých je interpretovaný vzor v těchto oknech v korelaci. Tento postup jsem vylepšil aplikací filtru shody, viz sekci 6.4.

Původní metoda z paketů nacházejících se uvnitř okna extrahovala vektor mezipaketových rozestupů. Navržená metoda časové okno dělí na předem určený počet kontejnerů a extrahuje normalizovaný vektor představující průměrný počet přenášených buněk Toru v každém kontejneru za jednotku času, viz sekci 6.3.

Normalizované vektory z X a Y jsou vstupem korelačních funkcí, viz sekci 6.6, které vyjádřují pomocí hodnoty korelačního koeficientu podobnost signatur těchto vektorů.

Posuvem časového okna dochází k vyhledávání lokálních podobností X a Y. Vyhodnocená pozice okna, kde míra podobnosti signatur je nad předem zvolenou prahovou hodnotou, je označena jako korelační bod.

Fáze korelační metody:

1. Předzpracování datových toků X a Y.

2. Hledání korelačních bodů (podobných signatur datového toku X a Y).
3. Vyhodnocení nalezených korelačních bodů – absolutní a relativní korelační koeficient.

Jednotlivé fáze budou blíže popsány v následujících částech textu.

6.2 Předzpracování datových toků

Před vlastní korelací musí datové toky projít fází předzpracování.

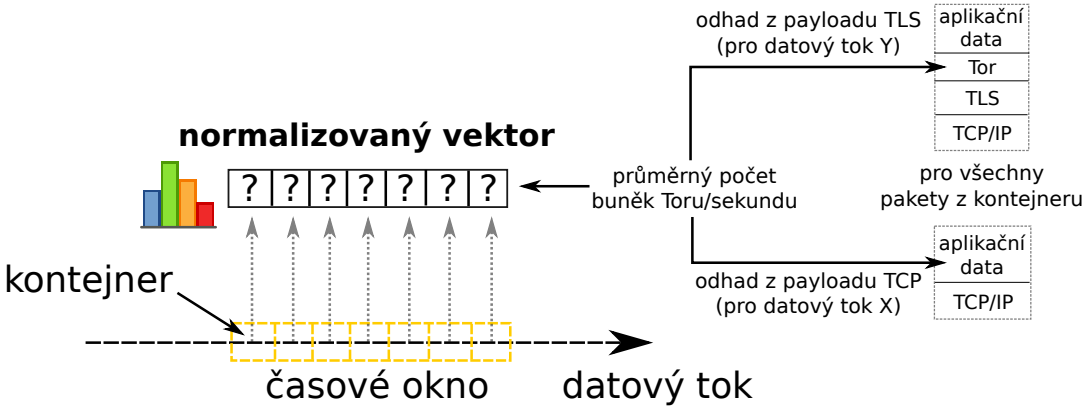
1. Extrakce jednosměrných datových toků na úrovni IP na lince ve směru destinace – výstupní uzel Toru (X) a strážce – klient (Y).
2. Odstranění duplicitních paketů TCP spojení.
3. Odfiltrování režijních paketů (TCP, TLS, Tor), které nejsou z hlediska korelace objemu užitečných dat žádané.

6.3 Extrakce normalizovaných datových vektorů

Proces extrakce datového vektoru z aktuální pozice okna v datovém toku je klíčovou fází metody, protože určuje způsob, jakým jsou informace o datovém toku uvnitř okna interpretovány.

Efektivita metody závisí na vhodném zvolení takových parametrů datového toku, které jsou co nejvíce unikátní pro každý datový tok a jsou invariantní při průchodu sítí Tor.

Navrhl jsem způsob extrakce, který časové okno dělí na zvolený počet menších kontejnerů. Výsledný vektor vznikne odhadem průměrného počtu přenášených buněk Toru za jednotku času v těchto kontejnerech. Proces extrakce je uveden na obrázku 6.2. Normalizaci na úrovni Tor buněk jsem zvolil, protože Tor zarovnává obsah užitečných dat právě na velikost buňky. Při správném odhadu počtu buněk je zkreslení způsobené zarovnáváním eliminováno.



Obrázek 6.2: Ilustrace extrakce normalizovaného vektoru pro danou pozici časového okna v datovém toku X nebo Y.

Odhad počtu buněk v paketech v datovém toku na lince mezi klientem a strážcem je přesnější, protože užitečná data přenášená protokolem TLS jsou skutečně Tor buňky.

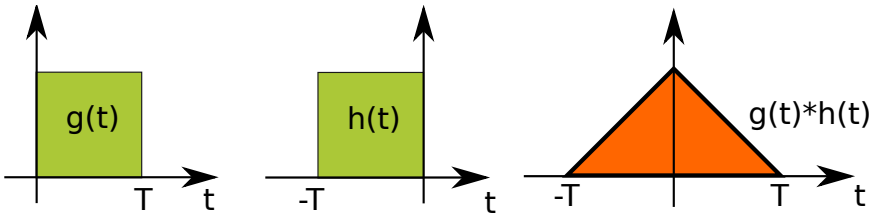
Zatímco odhad na straně mezi výstupním uzlem a destinací je pouze aproximací toho, jak Tor užitečná data přenášená protokolem TCP zapouzdřuje do jednotlivých buněk. Vycházel jsem z faktu, že Tor se musí snažit o co nejnižší zpoždění, a proto nemůže výstupní uzel přijatá data od destinace dlouze akumulovat, ale okamžitě je přeposílá. Kvůli tomu navržený algoritmus provádí odhad na úrovni jednotlivých paketů a nikoliv z objemu přeneseného více pakety v časovém intervalu.

Důležitými body jsou přiřazení průměrného počtu buněk do správného kontejneru a odhad počtu buněk pro daný paket. Pro datový tok X je uvažován *payload* nad vrstvou TCP a pro datový tok Y *payload* nad vrstvou TLS.

6.4 Aplikace filtru shody (matched filter)

Vzhledem k tomu, že Tor má při přenosu datových toků nezanedbatelné zpoždění, nelze očekávat, že signatura extrahovaného vektoru okna X v čase t bude nalezena v okně Y v tom stejném čase. Minimální a maximální zpoždění v síti Tor pak určuje interval v Y, ve kterém má smysl hledat odpovídající signaturu. Cílem je nalézt časový offset Δt okna v Y vůči oknu v X, při kterém je podobnost extrahovaných vektorů maximální.

Za tímto účelem jsem aplikoval z teorie signálů tzv. filtr shody[2] (*matched filter*) [45], který určuje podobnost tvaru signálů pro časový posuv. Pro vstupní signál $g(t)$ a impulsní odezvu $h(t) = g(T - t)$, je výstup filtru uveden na obrázku 6.3.



Obrázek 6.3: Výstup filtru shody pro vstupní signál $g(t)$ a implusní odezvu $h(t)$. Filtr pracuje s operací konvoluce signálů.

Z filtru shody vychází křížová korelace signálů (*cross-correlation*), která pracuje na stejném principu, ale je definována tak, že nepoužívá časově obráceného signálu a tím je jednodušší na implementaci. Pro spojité signály je výsledek korelace definován ve vzorci 6.1, kde f a g jsou porovnávané signály a f^* je komplexně sdružený signál f. Vztah mezi křížovou korelací a konvolucí použitou u filtru shody je následující $(f \star g)(t) = f^*(-t) * g(t)$.

$$(f \star g)(\tau) = \int f^*(\tau)\, g(t+\tau)\, \mathrm{d}\tau \tag{6.1}$$

Pro použití v této práci je potřebná definice pro diskrétní signály, která je uvedena ve vzorci 6.2. V lineární algebře odpovídají dílčí operace skalárním součinům.

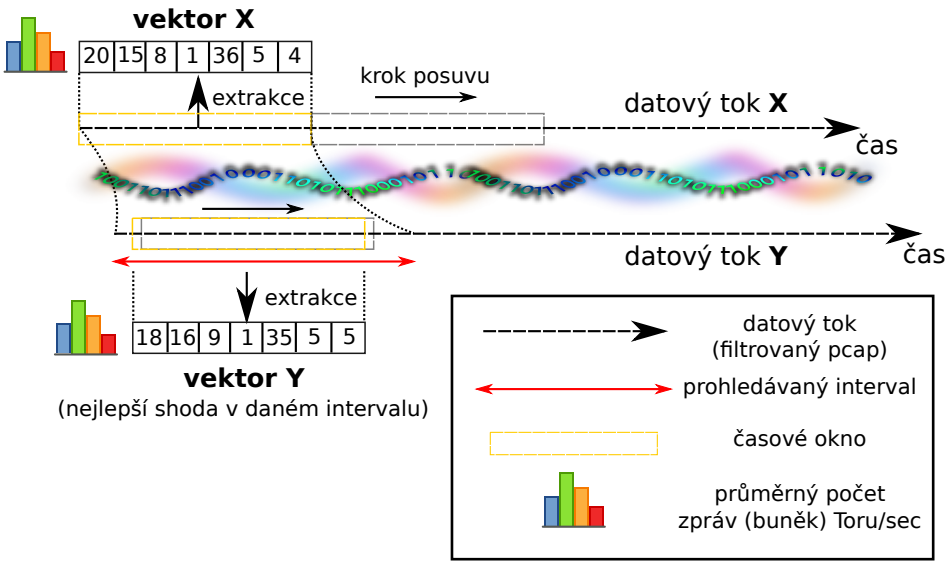
$$(f \star g)[n] = \sum_{m=-\infty}^{\infty} f^*[m]\, g[m+n] \tag{6.2}$$

[2]Filtr shody je optimální lineární filtr, který maximalizuje poměr signál–šum (SNR) v přítomnosti přídavného stochastického šumu. Běžně je používán v radarech, kde je vyslán známý signál, který je následně porovnáván s přijatým odraženým signálem na podobné vzory. Z odhadu vzájemného časového posuvu signálů lze vypočítat vzdálenost objektu, od kterého se původní signál odrazil.

Výsledek nabývá maxima pro diskrétní časový posuv n, kde jsou si signály f a g nejvíce podobné. V návrhu je křížová korelace aplikována tak, že pro hledanou signaturu v okně X (signál f) je postupným posuvem okna o předem určený konstantní krok prohledáván časový interval v Y. S tím, že při každém posuvu okna v Y je potřeba extrahovat příslušný datový vektor (signál g). Výsledný diskrétní časový posuv n je hledaný offset okna v Y vůči pozici okna v X, při kterém jsou si vektory nejvíce podobné. Zároveň lze n interpretovat jako odhad zpoždění.

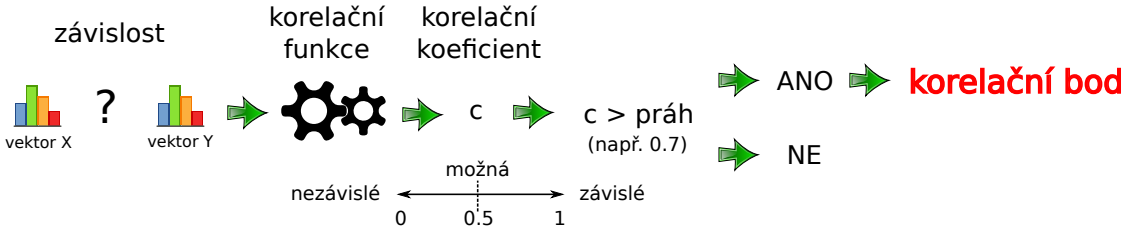
6.5 Hledání korelačních bodů

Princip metody spočívá v systematickém vyhledávání korelačních bodů datových toků X a Y, viz obrázek 6.4. Vyhledávání probíhá postupným posuvem okna v datovém toku X a vyhledáváním nejlepší shody té samé signatury v datovém toku Y, zde je aplikován filtr shody.



Obrázek 6.4: Ilustrace metody hledání korelačních bodů.

Extrahované normalizované vektory pro danou pozici okna v X a pozici nejlepší shody v Y jsou vstupem korelační funkce, která určí jejich závislost, viz obrázek 6.5. Pokud je výsledná hodnota korelačního koeficientu vyšší než zvolená prahová hodnota, je tato pozice oken v X a Y označena jako korelační bod.



Obrázek 6.5: Ilustrace vyhodnocení potenciálního korelačního bodu.

Čím více korelačních bodů je pro oba datové toky nalezeno, tím je větší pravděpodobnost, že je mezi nimi závislost.

Postup metody je následující:

1. Nastaví počáteční pozici okna v X na $[wx_{start} = x_1, wx_{end} = x_1 + w_{size}]$, kde wx_{start} je čas začátku okna, wx_{end} čas konce okna, x_i čas výskytu i-tého paketu v X a w_{size} velikost okna.

2. Extrahuje datový vektor V_x pro aktuální pozici okna v X $[wx_{start}, wx_{end}]$, viz sekci 6.3.

3. Určí časový interval $\langle m_{start}, m_{end} \rangle$, ve kterém se může nacházet signatura okna v X v toku Y.

$$m_{start} = wx_{start} + d_{min}$$
$$m_{end} = wx_{start} + w_{size} + d_{max}$$

   Kde wx_{start} je počáteční pozice okna v X, w_{size} velikost okna, d_{min} minimální zpoždění v síti Tor a d_{max} maximální zpoždění v síti Tor.

4. Aplikací filtru shody (*matched filter*), viz sekci 6.4, je určen offset okna v určeném časovém intervalu v Y, kde jsou signatury extrahovaných vektorů nejvíce podobné. Nastaví počáteční pozici okna v Y na $[wy_{start} = m_{start}, wy_{end} = m_{start} + w_{size}]$ a opakuje následující:

   (a) Extrahuje datový vektor V_y pro aktuální pozici okna v Y (wy_{start}, wy_{end}).

   (b) Vypočítá skalární součin V_x a V_y. Je-li tato hodnota větší nebo rovna než doposud vypočtené, uloží vektor jako dosud nejlepší shodu V_{ymax}

   (c) Posune okno v Y o předem zvolený krok $[wy_{start} \mathrel{+}= innerstep, wy_{end} \mathrel{+}= innerstep]$ a je-li splněna podmínka $wy_{end} \leq m_{end}$, pokračuje bodem (4a).

5. Pro vektor V_x a vektor V_{ymax} vypočítá pomocí zvolené korelační funkce, viz sekci 6.6, hodnotu korelačního koeficientu cpv v intervalu $\langle 0, 1 \rangle$.

6. Je-li $cpv \geq \delta_{cp}$, kde δ_{cp} je prahová hodnota korelační funkce, pak aktuální pozici okna v X a Y označí jako korelační bod. Každý vyhodnocený bod uloží jako trojici $[wx_{start}, \Delta t, cpv]$, kde wx_{start} je začátek okna v X, Δt je časový offset okna v Y zjištěný filtrem shody a cpv je výsledek korelační funkce.

7. Posune okno v X o předem zvolený krok $[wx_{start} \mathrel{+}= outerstep, wx_{end} \mathrel{+}= outerstep]$ a je-li splněna podmínka $wx_{end} \leq x_{last}$, pokračuje bodem (2).

Uvedené řešení má výhodu v tom, že způsob extrakce i korelace vektorů je jednoduše nahraditelný, díky tomu je navržená metoda dále rozšiřitelná.

6.6 Korelace normalizovaných vektorů

Pro porovnání podobnosti normalizovaných vektorů slouží korelační funkce (*correlation point function*). První tři funkce (*minimax, ndp1, ndp2*) jsem převzal z dříve popsané metody, viz sekci 5.8. Ty jsou použitelné při porovnání přímé podobnosti vektorů, tj. v případě, kdy datové toky X, Y přenáší samostatný datový tok. Korelační funkci *cap* jsem navrhl,

je použitelná v případě, kdy nelze odlišit více přenášených datových toků v rámci datového toku Y a kvůli tomu je objemová charakteristika toku Y v čase navýšena o objem agregovaných datových toků.

6.6.1 Poměr minimálních a maximálních členů (minimax)

Jednoduchou metrikou pro kvantitativní vyjádření podobnosti mezi dvěma vektory je poměr mezi sumou minimálních a sumou maximálních prvků vektorů, viz vzorec 6.3.

$$CPF_{minimax}(V_x, V_y) = \frac{\sum_{i=1}^{n} min\,(x_i, y_i)}{\sum_{i=1}^{n} max\,(x_i, y_i)} \tag{6.3}$$

6.6.2 Normalizovaný skalární součin verze 1 (ndp1)

Tato metrika je založena na lineární korelaci dvou diskrétních signálů (*matched filter*), který dosahuje maxima v bodě, kde jsou si signály nejvíce podobné. Nicméně výsledek lineární korelace není nutně v rozmezí $\langle 0, 1\rangle$. Pokud jsou diskrétní signály nahrazeny dvěma vektory, odpovídající operací k lineární korelaci dvou signálů je skalární součin (*inner-product, dot-product*) dvou vektorů v n-rozměrném prostoru.

V lineární algebře je skalární součin dvou vektorů roven kosinu úhlu mezi těmito vektory vynásobený délkou obou vektorů, viz vzorec 6.4, v tomto tvaru byla metrika použita v metodě, viz sekci 5.3. Kosinus úhlu mezi vektory je použit přímo jako výsledek korelační funkce v rozsahu $\langle -1, 1\rangle$. Toto řešení má nevýhodu v tom, že pro libovolný vektor V_x a pro libovolný vektor $V_y = a \times V_x + b$ bude výsledná hodnota 1.

$$V_x \cdot V_y = |V_x|\,|V_y|\,cos(\theta) \tag{6.4}$$

V metodě, viz sekci 5.8, uvedený vztah dále upravují a výsledný tvar je uveden ve vzorci 6.5. Výsledek je v rozmezí $< 0, 1 >$ a hodnoty 1 nabývá pouze tehdy, když $V_x = V_y$.

$$CPF_{ndp1}(V_x, V_y) = \frac{\sum_{i=1}^{n} x_i \times y_i}{max\left(\sum_{i=1}^{n} x_i^2, \sum_{i=1}^{n} y_i^2\right)} \tag{6.5}$$

6.6.3 Normalizovaný skalární součin verze 2 (ndp2)

Dalším způsobem jak normalizovat skalární součin dvou vektorů je vyjádřen vzorcem 6.6.

$$CPF_{ndp2}(V_x, V_y) = \frac{\sum_{i=1}^{n} x_i \times y_i}{\sum_{i=1}^{n} \left[max\,(x_i, y_i)\right]^2} \tag{6.6}$$

6.6.4 Objemová kapacita prohledávaného vektoru (cap)

Tato korelační funkce se vymyká původnímu záměru určit podobnost dvou vektorů, ale je cílena na zjištění, zda jsou prvky prvního vektoru menší nebo rovny prvkům druhého vektoru. Tento scénář se vyskytuje v případě, kdy je signatura hledaného datového toku skryta kvůli agregaci s dalším síťovým provozem. Tato funkce by měla být brána spíše jako odhad, že se v daném čase vyskytl takový objem přenášených dat, který by *pojmul* objem vyhledávané signatury. Funkce je definována ve vzorci 6.7.

$$CPF_{cap}(V_x, V_y) = \frac{\sum_{i=1}^{n} (flags(x_i, y_i))}{n} \tag{6.7}$$

$$flags(x,y) = \begin{cases} 1 & \Leftrightarrow x \leq y \\ 0 & \Leftrightarrow x > y \end{cases}$$

U této korelační funkce je potřeba počítat s faktem, že bude mít vyšší míru *false positives*. Při korelaci hledaného datového toku s datovým tokem, který bude mít konstantní hodnotu objemu dat v čase danou nejvyšší objemovou *špičkou* hledaného datového toku, by byl zcela jistě výsledek korelace nesprávně pozitivní. Tuto negativní vlastnost funkce *cap* by bylo možné částečně eliminovat tím, že by se pokusila odfiltrovat konstantní síťový provoz rozdílem sousedních prvků uvnitř vektorů. Stále by to neřešilo problém, když objem aditivního síťového provozu nebude konstantní.

6.7 Vyhodnocení podobnosti datových toků

Nalezené korelační body jsou dobrým indikátorem lokální podobnosti korelovaných datových toků, ale samy o sobě neříkají nic o celkové podobnosti. Podobnost na úrovni celých datových toků lze vyjádřit dalším zpracováním dříve vyhodnocených bodů[3].

Absolutní korelační koeficient: Tento koeficient při vyhodnocení míry celkové podobnosti uvažuje **všechny** vyhodnocené body. Kvůli tomu bude jeho hodnota vždy nižší, protože jsou zahrnuty i ty body, které nevykazují příliš vysokou lokální podobnost datových toků. Na druhou stranu je dobrým ukazatelem celkové závislosti datových toků.

Relativní korelační koeficient: V tomto případě jsou uvažovány **pouze** body korelační, tedy ty, pro které byl výsledek korelační funkce nad prahovou hodnotou δ_{cp}.

Výsledky tohoto koeficientu mohou být užitečné, když je cílem získat představu o podobnosti datových toků na určité hladině významnosti. Počet nalezených korelačních bodů a průměrná hodnota korelačního koeficientu jsou úzce spjaté a při určení závislosti datových toků je nutno tyto dvě výsledné hodnoty konfrontovat. Vysoký počet korelačních bodů při nízké prahové hodnotě korelační funkce ani nízký počet korelačních bodů při vysoké hodnotě korelační funkce nemusí vypovídat o vysoké podobnosti datových toků. Vždy je potřeba vzít v úvahu celkový počet vyhodnocených bodů datového toku a počet nalezených korelačních bodů.

Průměrný korelační koeficient: Je definován jako průměr výsledků korelačních funkcí pro vyhodnocené body, viz vzorec 6.8, kde cpv_i je korelační hodnota bodu a n je počet uvažovaných bodů. Podle kontextu (absolutní, relativní) koeficient je počítán buď ze všech vyhodnocených bodů nebo pouze z korelačních bodů.

$$corr_{avg} = \frac{\sum_{i=1}^{n} cpv_i}{n} \tag{6.8}$$

Míra závislosti korelačních bodů: U korelačních bodů má smysl uvažovat závislost těchto bodů a odhadnutého zpoždění (offset okna v Y vůči X). Předpokladem je, že tento offset by měl být podobný pro korelační body náležící tomu samému datovému toku, pokud se podmínky pro přenos daným okruhem příliš nemění.

Nalezené korelační body je možné reprezentovat jako vektory $C_x =< x_1, \ldots, x_n >$ a $C_y =< y_1 = x_1 + \Delta t_1, \ldots, y_n = x_n + \Delta t_n >$, prvky C_x jsou časy začátku okna v X a

[3] Vyhodnocenými body jsou myšleny i ty body (pozice okna), které nebyly označeny jako body korelační.

prvky C_y jsou odpovídající časy začátku okna v Y. Tyto vektory by pak měly být lineárně závislé. Lineární závislost se dá vyjádřit statistickým korelačním koeficientem nazývaným jako Pearsonův korelační koeficient, viz vzorec 6.9.

$$\rho\left(C_x, C_y\right) = \frac{\sum_{i=1}^{n}\left(x_i - \bar{C_x}\right)\left(y_i - \bar{C_y}\right)}{\sqrt{\sum_{i=1}^{n}\left(x_i - \bar{C_x}\right)^2}\sqrt{\sum_{i=1}^{n}\left(y_i - \bar{C_y}\right)^2}} \tag{6.9}$$

Kapitola 7

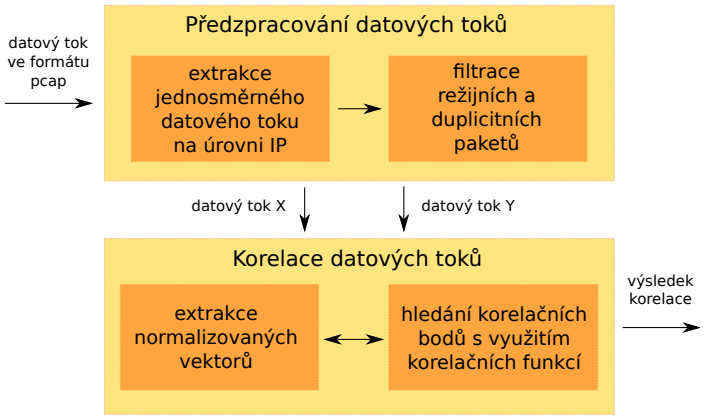
Implementace softwarového nástroje

V této kapitole popisuji implementaci navržené korelační metody. Implementaci jsem rozdělil do dvou logických celků, viz blokové schéma v sekci 7.1. Nejdříve jsem vytvořil nástroj pro extrakci a filtrování datových toků, viz sekci 7.2. Dále jsem vytvořil nástroj pro extrakci normalizovaných datových vektorů, viz sekci 7.3. Na závěr jsem vytvořil nástroj pro korelaci, viz sekci 7.4.

Nástroje jsem implementoval ve skriptovacím jazyce Python[1] s využitím knihovny Scapy[2], která objektově reprezentuje datové toky zachycené ve formátu *pcap*.

7.1 Blokové schéma programu

Na obrázku 7.1 je zachycena funkcionalita implementovaného nástroje. Bloky schématu jsou realizovány sadou tříd, viz implementační diagram tříd v příloze B.1.



Obrázek 7.1: Způsob implementace jednotlivých fází korelační metody znázorněný blokovým schématem programu.

[1] https://www.python.org/
[2] http://www.secdev.org/projects/scapy/

7.2 Extrakce a filtrování datových toků

Korelační metoda pracuje s jednosměrnými datovými toky na úrovni síťové vrstvy (IPv4). Implementoval jsem skript *flowparser*, který využívá třídu `FlowParser`. Tato třída obsahuje metodu pro extrakci jednosměrného datového toku na základě zdrojové a cílové IP adresy. Využívá k tomu přímo průchod objektovou reprezentací paketů v datovém toku.

Součástí třídy jsou i metody pro filtraci režijních paketů TCP a TLS, které jsou pro korelaci nežádoucí.

Filtrované režijní pakety:

1. Režijní pakety TCP[3] (v případě X i Y) – SYN, SYN/ACK, pouze ACK, FIN a FIN/ACK.
2. Režijní pakety TLS[4] (v případě Y) – všechny záznamy kromě Application Data.
3. Režijní pakety Toru [6] (v případě Y) – pakety, které přenášejí pouze 1 Tor buňku.

Filtrace režijních paketů TCP je přímočará, stačí pouze zkontrolovat příznaky segmentu. U příznaku A (ACK) je kontrolováno zda je délka užitečných dat segmentu TCP nulová, jinak by bylo nežádoucí takový paket odstranit.

Knihovna Scapy v současné stabilní verzi neposkytuje *dissector* vrstvy TLS, použil jsem proto vývojovou verzi 2.2.0. Uzly Toru (OR) jsou pro účely vytvoření spojení TLS dostupné na libovolném TCP portu, ale nejčastěji se jedná o porty (443, 993, 9001)[5]. Wireshark běžně rozpoznává spojení TCP na portu 443, kdy se jedná o HTTPS a na portu 993, který je používán v souvislosti se zabezpečením přenosu protokolu IMAP. Pomocí vestavěné funkce `bind_layers` jsem ve Scapy navázal dissector TLS na výše zmíněné porty. Takže při načtení datového toku interpretuje Scapy komunikaci na těchto TCP portech jako spojení TLS. Může se stát, že spojení TLS bude inicializováno na jiných TCP portech, pak by bylo nutné tyto porty manuálně dospecifikovat. V opačném případě Scapy spojení TLS nerozpozná, ale to v dalším zpracování nevadí, pouze neproběhne filtrace režijních paketů protokolu TLS.

Protokol TLS přenáší užitečná data v záznamech různých typů určených identifikačním číslem a délkou *payloadu*, užitečná data jsou přenášena v záznamech typu (`Application Data, 23`), všechny ostatní typy záznamů jsou odfiltrovány. Nedochází však k odstranění úplně všech nežádoucích záznamů z toho důvodu, že záznamy mohou být větší než maximální velikost MTU a jsou rozděleny do více paketů, tedy i segmentů. Bez zpětného sestavení vrstvy TLS (zašifrované) je interpretace záznamů komplikovaná a v případě zaslání většího počtu záznamů se stává, že je rozpoznán pouze první záznam v sérii příslušných segmentů.

Filtrace režijních paketů Toru využívá faktu, že buňka Toru má standardizovanou velikost 512 B. Velikost užitečných dat síťové vrstvy pro paket přenášející 1 Tor buňku bývá často 597 B, ale kvůli proměnlivým velikostem hlaviček protokolu IP a TCP se může měnit. V případě, že by tato velikost nebyla konstantní, by bylo možné identifikovat přenos jediné buňky Toru testem ne na konkrétní velikost *payloadu* IP, ale na velikost v určitém intervalu (vzhledem k pozorovaném rozdělení velikostí paketů).

[3] http://www.ietf.org/rfc/rfc793.txt
[4] http://tools.ietf.org/html/rfc5246
[5] http://torstatus.blutmagie.de/

Další metoda zajišťuje odstranění duplicitních paketů. Každý takový paket je zdrojem zkreslení charakteristiky datového toku a pro další korelaci nežádoucí. Metoda volá externí proces programu `editcap`, který postupně prochází datový tok a eliminuje všechny duplicitní výskyty toho samého paketu. Používá k tomu klouzajícího okénka, ve kterém se může vyskytovat specifikovaný počet paketů nebo může být ohraničeno časovým intervalem.

7.3 Extrakce normalizovaných vektorů

Nástroj pro extrakci normalizovaných vektorů jsem implementoval skriptem *vectorextractor*, který využívá třídu `VectorExtractor`. Ta má jednu stěžejní metodu, která implementuje způsob extrakce normalizovaného vektoru.

Při odhadu počtu buněk Toru přenášených v jednom paketu jsem potřeboval znát velikost užitečných dat přenášených nad vrstvou TCP, respektive TLS. Vytvořil jsem skript *flowtools* s funkcemi, které z metadat v hlavičkách protokolů určí velikost *payloadu* pro zvolenou vrstvu.

Výpočet užitečných dat přenášených daným protokolem:

- U protokolu IP značí položka *len* v hlavičce celkovou velikost paketu [B] včetně hlavičky. Velikost hlavičky je dána položkou *ihl*, hodnotou je počet 32bitových dvojslov.

  Velikost užitečných dat je $payload_{IP} = len - 4 * ihl$.

- U protokolu TCP značí offset dat od začátku hlavičky, tedy velikost hlavičky položka *dataofs*. Její hodnotou je opět počet 32bitových dvojslov.

  Velikost užitečných dat je $payload_{TCP} = payload_{IP} - dataofs$.

- U protokolu TLS mají záznamy typu **Application Data** standardizovanou velikost hlavičky 5 B, která určuje typ záznamu a velikost přenášených dat. Situaci komplikuje fakt, že do této velikosti je započítán i kontrolní součet MAC (SHA1 - 20 B, MDP5 - 16 B) a zarovnání specifikované v posledním bajtu záznamu (max. 15 B). Odhadem jsem určil, že režie TLS představuje přibližně 40 B. Skutečná hodnota se může lišit, ale rozdíl nebude výrazný.

  Velikost užitečných dat je $payload_{TLS} = payload_{TCP} - overhead_{TLS}$.

Vlastní odhad počtu buněk Toru v paketu nad vrstvou TCP je určen jako podíl hodnot $payload_{TCP}$ a 498 B. Kde 498 B je velikost užitečných dat přenášených buňkou Toru. Odhad nad vrstvou TLS je určen jako podíl hodnot $payload_{TLS}$ a standardní velikosti Tor buňky 512 B. Výsledný odhad je v obou případech zaokrouhlen na celé číslo.

7.4 Korelace datových toků

Navrženou korelační metodu jsem implementoval ve skriptu *vectorcorrelator*, který využívá interně třídu `VectorCorrelator`. Metoda třídy `find_correlation_points` implementuje navržený postup hledání korelačních bodů, viz sekci 6.5. Ta při postupném průchodu datovým tokem a pohybu okna volá metodu `evaluate_correlation_point`, která je implementací aplikace filtru shody v kontextu hledání nejlepší signatury v prohledávaném

datovém toku, viz sekci 6.4. K tomu používá další metody třídy, které implementují korelační funkce definované v sekci 6.6. Celkové vyhodnocení korelace má na starosti metoda `overall_correlation_value`. Zdůraznil bych možnost ve skriptu nastavit zřetelnost hledané signatury datového toku X, momentálně je implementována varianta, kdy nejsou uvažovány vektory, které po extrakci obsahují samé nuly. Tato „prázdná“ místa v datovém toku nejsou unikátním ukazatelem jejich závislosti.

Vstupem skriptu jsou datové toky X a Y, tak jak byly předem definované. Před vlastní korelací musí projít předzpracováním, k tomu je použit skript *flowparser*. Extrahuje požadovaný jednosměrný datový tok, odstraní duplicitní pakety a nežádoucí režijní pakety.

Výstupem korelačního skriptu je celkový počet vyhodnocených bodů, počet nalezených korelačních bodů a absolutní i relativní korelační koeficient. Na základě absolutního korelačního koeficientu a nastavené prahové hodnoty skript rozhodne, zda datové toky jsou či nejsou v korelaci. Výsledná hodnota korelace určená Pearsonovým korelačním koeficientem založená na odhadu zpoždění je sice implementována, ale považuji ji až za sekundární ukazatel závislosti.

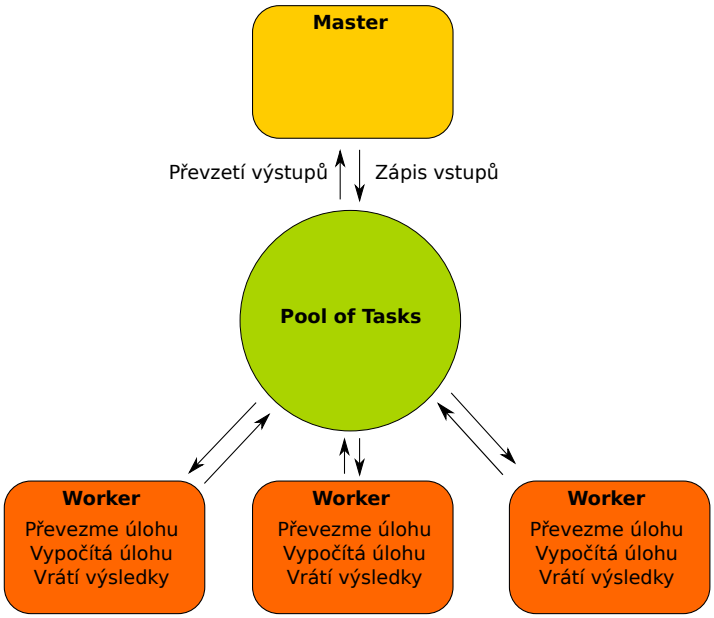
Parametry skriptu jsou:

- Předzpracované datové toky **X** a **Y** ve formátu *pcap*.
- Prahová hodnota pro uznání korelačního bodu (potřebná pro relativní korelační koeficient).
- Prahová hodnota pro celkový výsledek korelace (potřebná pro absolutní korelační koeficient).
- Korelační funkce (*minimax, ndp1, ndp2, cap*) použitá pro vyhodnocení potenciálního korelačního bodu.
- Způsob celkového vyhodnocení korelace (průměrný korelační koeficient, statistický korelační koeficient).
- Velikost časového okna.
- Krok posuvu časového okna.
- Velikost kontejneru uvnitř časového okna.
- Krok posuvu časového okna při aplikaci filtru shody.
- Minimální a maximální uvažované zpoždění v síti Tor.
- Úroveň paralelismu (počet paralelních procesů)[6].

Skript lze spouštět přímo specifikací parametrů na příkazové řádce a nebo s využitím konfigurace uložené v *csv* souboru. Je-li specifikován konfigurační soubor (výchozí *defaults.csv*), skript iterativně provádí korelaci datových toků X a Y pro každý řádek specifikace parametrů. Výstup skriptu v tomto případě nejde na standardní výstup, ale také do *csv* souboru. Tímto způsobem lze pohodlně automaticky korelovat datové toky s různým nastavením metody a uživatel nemusí individuálně spouštět každý běh skriptu.

[6]Není-li zadán počet paralelních procesů, vytvoří se jich tolik, kolik je na platformě dostupných jader procesorů.

Paralelní zpracování implementuje vzor **Master – Worker**, viz obrázek 7.2. Řídicí proces (*master*) rozdělí vyhodnocované pozice okna v datovém toku X mezi další procesy (*workers*), kteří pro zadanou pozici vykonávají metodu `evaluate_correlation_point`. Řídicí proces od nich posbírá výsledky a provede celkové vyhodnocení. Tento paralelní vzor jsem chtěl původně implementovat pomocí vláken, jenže Python neumožňuje při použití vláken běh více než jednoho vlákna z interních důvodů. Výsledná verze využívá paralelismu na úrovni procesů a tam dochází ke kopii adresového prostoru procesu, proto může mít program při korelaci časově delších a objemných datových toků vyšší paměťovou náročnost.[7]



Obrázek 7.2: Ilustrace vzoru paralelního programování Master – Worker.

[7] V případě, že operační systém (např. Linux) v rámci správy paměti používá metodu COW (*copy-on-write*), jsou paměťové nároky nižší, protože jednotlivé stránky procesu jsou kopírovány až při pokusu o zápis, do té doby jsou mezi procesy sdílené. To je výhodné pro současnou implementaci, vytvořené procesy totiž pouze čtou sdílená data, kterými jsou datové toky X a Y v objektové reprezentaci Scapy, ale nezapisují do nich.

Kapitola 8

Testování korelační metody a vyhodnocení úspěšnosti

V této kapitole se zabývám testováním navržené korelační metody a vyhodnocením úspěšnosti při nasazení na reálných datech ze sítě Tor. Nejdříve jsem určil cíle testování, na základě kterých jsem navrhl sadu experimentů. Každý z těchto experimentů ověřuje chování metody ve specifických případech. Z těchto výsledků je možné vycházet při studování chování metody u případů složitějších.

8.1 Cíle testování

Cílem navržených experimentů je vyhodnotit chování korelační metody v následujících případech:

1. Efektivita metody při určení závislosti datových toků. Správné určení závislosti datových toků, které **jsou** v korelaci (*true positives*). Správné určení závislosti datových toků, které **nejsou** v korelaci (*true negatives*). Při nesprávném určení závislosti jsou doplňkem (*false positives* a *false negatives*).

2. Efektivita metody při detekci hledaného datového toku v agregovaném datovém toku v síti Tor.

3. Vliv geografické vzdálenosti okruhu (zpoždění, rozptyl) na výsledek korelace.

4. Vliv objemu datového toku v čase na výsledek korelace.

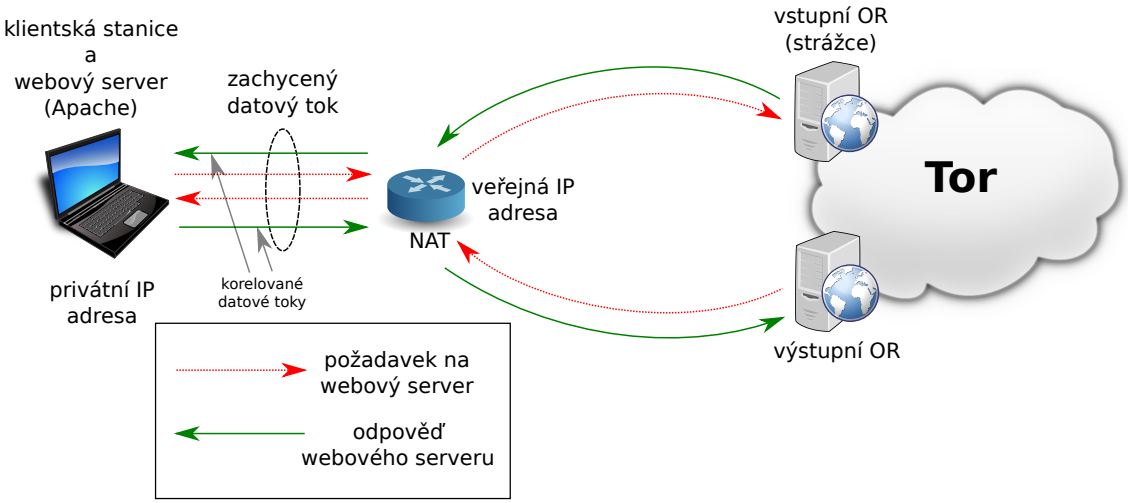
5. Doba běhu metody v závislosti na velikosti vstupu a nastavení parametrů.

8.2 Metodika testování

U každého z experimentů uvádím jeho vstupy (použité okruhy, datové toky, nastavení metody, nastavení instance Toru) a jeho výstupy – výsledky korelační metody relevantní po daný experiment.

Pro účely testování a na základě zvolené granularity datových toků, viz podsekci 8.2.1, jsem uvažoval datové toky s dobou trvání 10 minut, které prochází přes předem vybrané okruhy.

Abych měl přístup k datovému toku na lince klient – strážce i výstupní uzel – destinace, generoval jsem HTTP požadavky na vlastní webový server, viz obrázek 8.1. Ty byly po dobu 10ti minut generovány v časových intervalech určených exponenciálním rozdělením s parametrem λ (konkrétní hodnota zvolena v rámci experimentu).



Obrázek 8.1: Ilustrace topologie použité při testování.

8.2.1 Granularita datových toků

Korelace probíhá ve všech případech na úrovni datových toků IP, na této úrovni lze zachytit všechna TCP spojení klienta pro testovaný okruh a tím pádem je k dispozici větší objem dat ke korelaci. Minimální doba trvání datového toku, kterou mohu dále uvažovat, je pak dána výchozí dobou životnosti okruhu tj. 10 minut.

Důležitým faktem je, že při rotaci okruhů je velká pravděpodobnost, že dojde ke změně výstupního uzlu a tím pádem se odliší i datový tok na úrovni IP. Naopak je málo pravděpodobné, že by více klientů současně přistupovalo ke stejné destinaci přes ten samý výstupní uzel.[1]

8.2.2 Zvolené okruhy pro testování

Zvolené okruhy, viz tabulka 8.1, jsem vybíral na základě dostatečné inzerované přenosové kapacity a geografické lokace. Strážce i výstupní OR jsem zvolil na pevno, fyzická vzdálenost okruhu je určena výběrem prostředního OR. K výběru jsem použil nástroje Compass[2] a Atlas[3]. První z nich umí vyhledávat OR podle celé řady kritérií. Druhý z nich pak zobrazí informace o konkrétním OR.

Dostupná přenosová kapacita okruhů a podmínky na síti jsou v čase proměnlivé, z toho důvodu by se mohly výsledky při opakování testování do budoucna mírně lišit.

[1] Současný algoritmus výběru cesty nebere v potaz fyzickou vzdálenost OR, přistupuje-li více uživatelů přes Tor ke stejné populární destinaci, s velkou pravděpodobností bude mít jejich okruh jiný výstupní OR.

[2] https://compass.torproject.org

[3] https://atlas.torproject.org

Okruh	strážce (*guard*)	OR (*middle*)	výstupní uzel (*exit*)
č.1	devaley, ČR	fluxe4, Německo	sergeanthorstonion1, ČR
č.2	devaley, ČR	thevillage1, Velká Británie	sergeanthorstonion1, ČR
č.3	devaley, ČR	wannabe, USA	sergeanthorstonion1, ČR
č.4	devaley, ČR	Fushigi, Japonsko	sergeanthorstonion1, ČR

Tabulka 8.1: Virtuální okruhy použité v experimentech.

8.2.3 Výchozí nastavení metody

Při testování jsem použil výchozí nastavení metody uvedené v tabulce 8.2. Pokud je v experimentu použito nastavení odlišné, je to explicitně uvedeno.

Parametr	Hodnota	Komentář
Velikost časového okna	15 s	-
Velikost kontejneru	1 s	Dle naměřených hodnot rozptylu zpoždění [20].
Krok posuvu časového okna	10 s	-
Krok při aplikaci filtru shody	0.1 s	-
Minimální zpoždění	0 s	-
Maximální zpoždění	3 s	Dle naměřených hodnot zpoždění [21, 10, 38].
Počet jader procesoru	dostupné	-

Tabulka 8.2: Výchozí nastavení metody.

8.2.4 Nastavení instance Toru

Při testování jsem použil nastavení instance Toru uvedené v tabulce 8.3.

Parametr	Hodnota	Komentář
MaxCircuitDirtiness	86400 s	Životnost okruhu.
MaxCircuitPeriod	30000 s	Perioda, kdy OP zvažuje konstrukci nového okruhu.
MaxClientCircuitsPending	1	Počet okruhů, které jsou v čase k dispozici.
CircuitIdleTime	86400 s	Časovač uzavření nepoužitého (čistého) okruhu.
CircuitStreamTimeout	86400 s	Časovač odpojení datového toku od okruhu (při chybě).
LeaveStreamsUnattached	1	Příchozí datové jsou přiřazovány okruhům manuálně.
DisablePredictedCircuits	1	Deaktivuje predikci konstrukce okruhů.

Tabulka 8.3: Nastavení instance Toru pro účely experimentování.

8.3 Použité nástroje při testování

Při realizaci experimentů jsem potřeboval získat data ze sítě Tor. Použil jsem k tomu následující nástroje.

8.3.1 Generování síťového toku

Vytvořil jsem skript *flowgenerator*, který pomocí nástroje `wget` generuje HTTP dotazy na zadanou adresu webového serveru. Použil jsem adaptér `torsocks`, který tuneluje všechny požadavky na TCP spojení daného programu přes síť Tor.

Tímto způsobem je simulováno chování uživatele. Dotazy jsou generovány v časových intervalech určených exponenciálním rozdělením s parametrem λ. Kromě klasických HTTP dotazů je možné generovat i HTTPS dotazy, ale ty jsem při testování neuvažoval.

8.3.2 Zachytávání síťového toku

Datové toky jsem zachytával pomocí vytvořeného skriptu *flowsniffer*, ten pro zvolené síťové rozhraní a dobu zachytávání ukládá datový tok do souboru ve formátu *pcap*. Interně pracuje s funkcí `sniff` definovanou v knihovně Scapy. Formát pcap jsem zvolil z toho důvodu, že metoda vyžaduje přesnou časovou informaci o výskytu paketu v rámci datového toku. To formát pcap splňuje, poskytuje časová razítka pro každý zaznamenaný paket na úrovni jádra operačního systému.

8.3.3 Řízení lokální instance Toru

Ve výchozím nastavení si instance Toru u klienta řídí správu okruhů automaticky. Požadavkům na TCP spojení přiděluje dopředu predikované okruhy a nebo vytvoří nový, který je schopný uspokojit požadavek. Při testování to znamená, že se neustále mění IP adresy datových toků. Z toho důvodu jsem v rámci testování automatickou správu okruhů vyřadil a řídil vše manuálně.

Za tímto účelem jsem vytvořil skript *torcontroller*, který využívá knihovny **Stem**[4] napsané v jazyce Python. Tato knihovna implementuje Tor Control Procotol [1], přes který je možné lokálně či vzdáleně komunikovat s danou instancí Toru.

Uvedený skript jsem použil k vytváření testovacích okruhů a aplikaci příslušného nastavení, viz podsekci 8.2.4.

8.4 Experiment č.1 (true positives, false negatives)

Cílem prvního experimentu bylo zjistit, jak efektivní je metoda při rozlišení datových toků, které jsou v korelaci. Vstupem byl datový tok, viz tabulka 8.4, který představoval komunikaci mezi klientem a vlastním webovým serverem přes síť Tor. Koreloval jsem úsek (destinace – výstupní uzel) s úsekem (strážce – klient).

V první části experimentu jsem se zaměřil na absolutní korelační koeficient, viz graf na obrázku 8.2. Z výsledků vyplývá, že nejlépe si při rozpoznání závislosti datových toků vede korelační funkce *ndp1*. Nižší míru závislosti, ale stále správně určila korelační funkce *minimax*. S rostoucí velikostí časového okna stoupá i míra závislosti datových toků, tento výsledek si lze vysvětlit tím, že s rostoucí velikostí časového okna (při konstantním kroku)

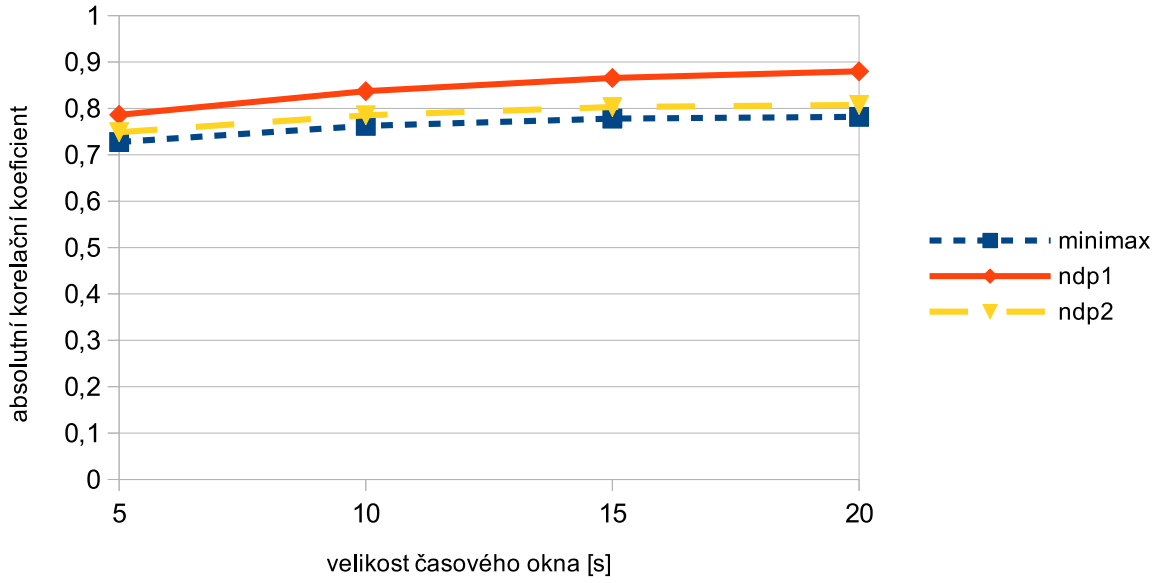
[4] `https://stem.torproject.org`

datový tok	okruh	velikost stránky	parametr λ	destinace
č.1	č.1	250 KB	10 s	vlastní webserver

Tabulka 8.4: Datové toky použité v experimentu.

se zvyšuje i počet bodů, které při vyhodnocení metoda bere v úvahu[5]. Pro velikost časového okna (5, 10, 15, 20) byl celkový počet vyhodnocených bodů (57, 81, 95, 102) z celkového počtu 120[6].

Prahová hodnota 0.7 se jeví jako dobrá hranice pro rozlišení datových toků, které jsou zcela jistě v korelaci. Výsledky všech korelačních funkcí se nachází právě nad touto prahovou hodnotou.



Obrázek 8.2: Absolutní korelační koeficient v závislosti na velikosti časového okna pro datové toky, které **jsou** v korelaci.

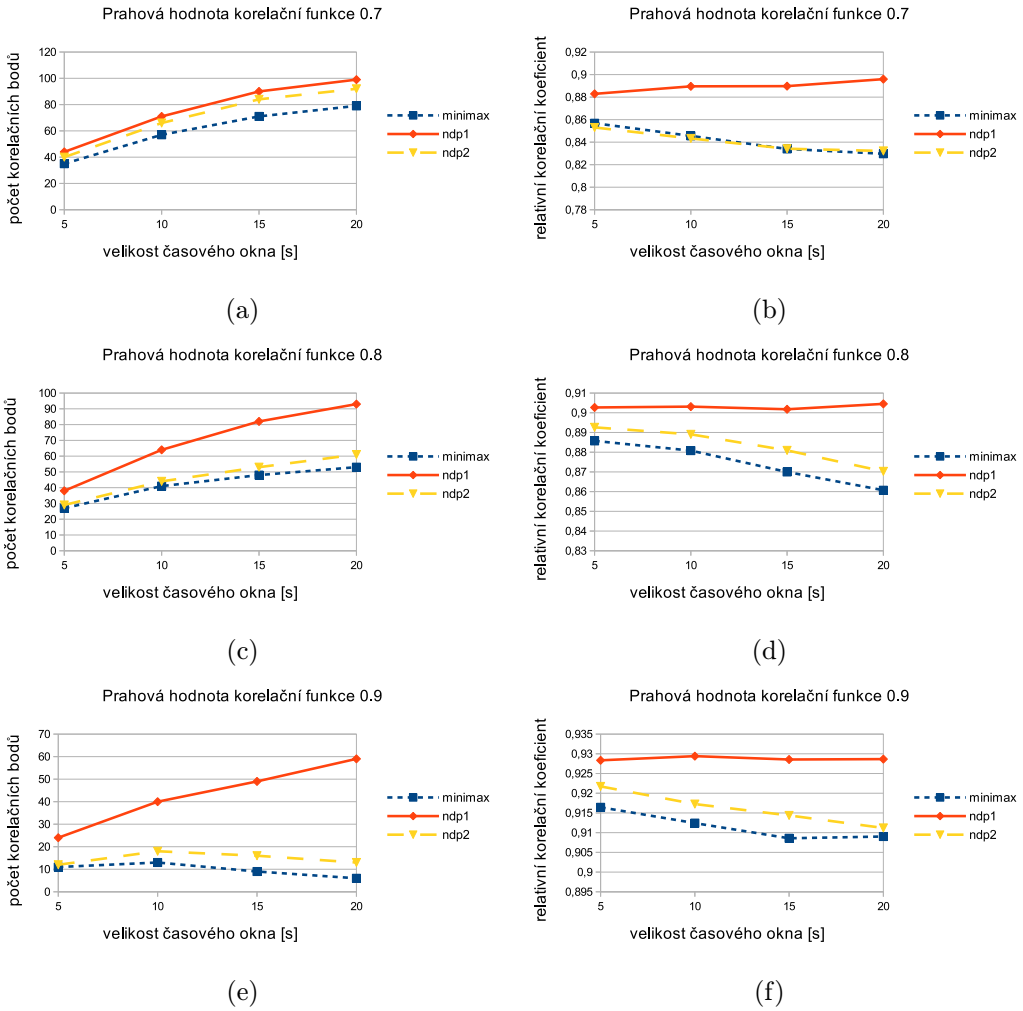
Druhou částí experimentu bylo vyhodnocení počtu nalezených korelačních bodů a relativního korelačního koeficientu v závislosti na velikosti časového okna při zvolené prahové hodnotě korelační funkce. Výsledky jsou uvedeny v grafech na obrázku 8.3. Lze vidět, že s rostoucí prahovou hodnotou korelační funkce se počet nalezených korelačních bodů snižuje pro všechny velikosti časového okna.

Až na případ 8.3e se s rostoucí velikostí časového okna zvyšuje počet nalezených korelačních bodů. Relativní korelační koeficient korelační funkce *ndp1* vykazuje poměrně konstantní hodnotu, zatímco pro funkce *minimax* a *ndp2* s rostoucí velikostí časového okna

[5]S rostoucí velikosti časového okna je více pravděpodobné, že v extrahovaném vektoru nebudou samé nuly – takové vektory nejsou při zpracování dále uvažovány.

[6]Datový tok s dobou trvání 10 minut, při velikosti časového okna 5 s a kroku 5 s dává 120 pozic. Pro jiné velikosti časového okna byl použit také krok 5 s.

klesá. Při vyšší velikosti okna je méně pravděpodobné, že bude nalezen „stejně podobný“ vzor a to lépe vystihují funkce *minimax* a *ndp2*.



Obrázek 8.3: Počet nalezených korelačních bodů a relativní korelační koeficient v závislosti na intervalu hlavního okna při prahové hodnotě korelační funkce 0.7 (a,b), 0.8 (c,d) a 0.9 (e,f).

Závěrem prvního experimentu je, že všechny korelační funkce byly úspěšné při určení závislosti dvou datových toků, které jsou v korelaci (*true positives*). Neobjevily se žádné případy *false negatives*. Byla určena horní prahová hodnota pro absolutní korelační koeficient (0.7), kdy při hodnotách nad toutou prahovou hodnotou lze říci, že mezi datovými toky je zcela jistě závislost. Nejlepších výsledků při detekci dosahují korelační funkce *ndp2* a *minimax*, které s rostoucí velikostí časového okna odráží skutečnou závislost datových toků lépe, přestože má funkce *ndp1* vyšší hodnoty korelačních koeficientu.

8.5 Experiment č.2 (true negatives, false positives)

Cílem druhého experimentu bylo určit efektivitu metody při rozlišení datových toků, které nejsou v korelaci. Vstupem byly dva datové toky, viz tabulka 8.5, koreloval jsem úsek

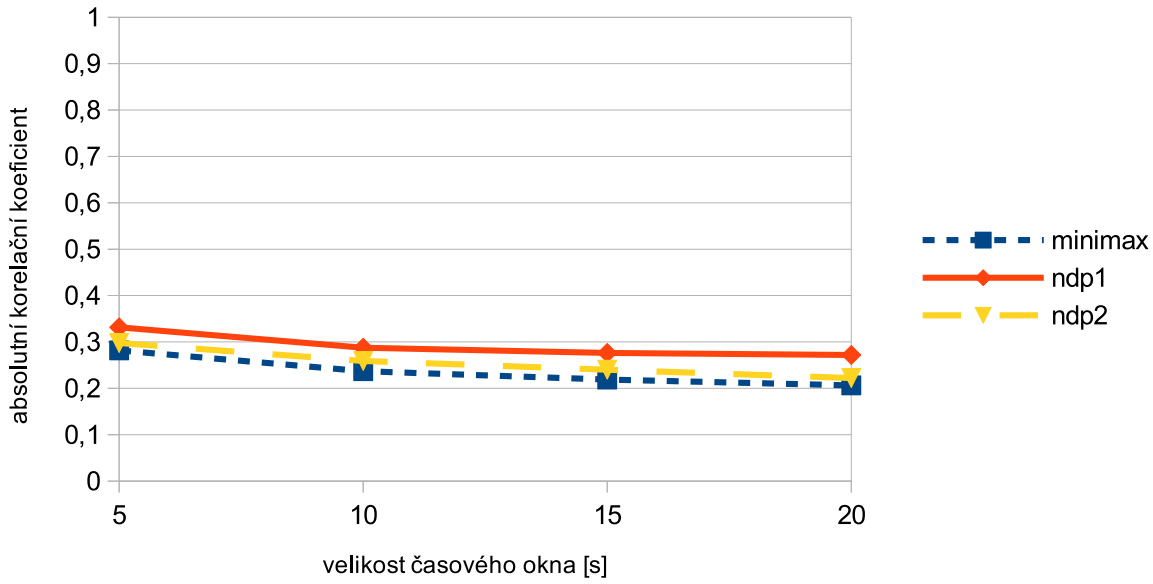
(destinace – výstupní uzel) datového toku č.1 s úsekem (strážce – klient) datového toku č.2.

datový tok	okruh	velikost stránky	parametr λ	destinace
č.1	č.1	250 KB	10 s	vlastní webserver
č.2	č.1	750 KB	15 s	www.seznam.cz

Tabulka 8.5: Datové toky použité v experimentu.

Obdobně jako v předchozím experimentu jsem se nejdříve zaměřil na absolutní korelační koeficient v závislosti na velikosti časového okna, který je uveden v grafu na obrázku 8.4. Opět s rostoucí velikostí hlavního okna je určení závislosti přesnější, to je dáno vyšším počtem vyhodnocených bodů. Nejlépe si vede korelační funkce *minimax*, která je citlivější na lokální detaily korelovaných datových vektorů. Korelační funkce *ndp1* je naopak ve vyhodnocení závislosti mírnější, má tendenci míru závislosti zvyšovat.

Všechny korelační funkce pro velikost časového okna nad 10 sekund dávají výsledky pod prahovou hodnotou 0.3. Tu by bylo možné uvažovat při rozlišení datových toků, které zcela jistě nejsou v korelaci.

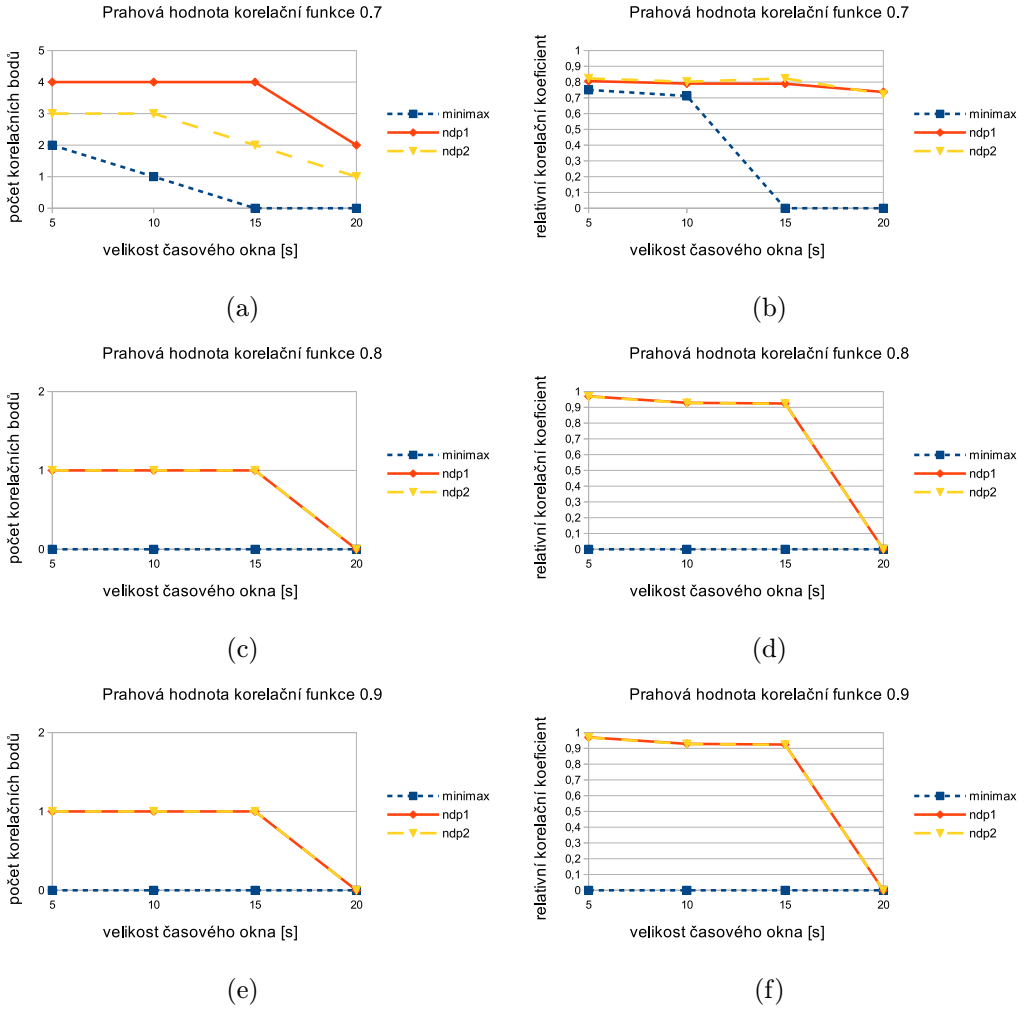


Obrázek 8.4: Absolutní korelační koeficient v závislosti na velikosti časového okna pro datové toky, které **nejsou** v korelaci.

Výsledky relativního korelačního koeficientu, viz grafy na obrázku 8.5, mohou být na první pohled znepokojující, protože byly nalezeny nějaké korelační body. Ale při další interpretaci lze s velkou pravděpodobností říci, že datové toky v korelaci nejsou, protože nalezený počet korelačních bodů je pro všechny prahové hodnoty příliš nízký. Pravděpodobnost, že se vyskytne v datovém toku na určitém místě podobný vzor, je sice malá, ale je potřeba s ní počítat.

Na základě principu korelační metody lze již dopředu říci, že čím více jsou v rámci okna charakteristiky datového toku agregovány (menší počet kontejnerů pro stejný časový interval), tím více nesprávně identifikovaných korelačních bodů se může vyskytnout. To je způsobeno tím, že se ztrácí unikátnost charakteristik datových toků (co se týče objemu v čase).

Z dosažených výsledků lze vyvodit, že ideální velikostí okna pro zachování unikátních charakteristik datových toků je velikost 15 s a více. Nejlépe si vede korelační funkce *minimax*, která pro prahové hodnoty korelační funkce (0.8 a 0.9) nenalezne vůbec žádný korelační bod.



Obrázek 8.5: Počet nalezených korelačních bodů a relativní korelační koeficient v závislosti na velikosti časového okna při prahové hodnotě korelační funkce 0.7 (a,b), 0.8 (c,d) a 0.9 (e,f).

Závěrem druhého experimentu je, že všechny korelační funkce byly úspěšné při určení závislosti dvou datových toků, které nejsou v korelaci (*true negatives*). A neobjevily se žádné *false positives* výsledky. Byla určena dolní prahová hodnota pro absolutní korelační koeficient (0.3) a velikosti časového okna 10 s a více, kdy při hodnotách pod touto prahovou hodnotou lze říci, že mezi datovými toky zcela jistě není závislost. Nejlepších výsledků dosahují všechny korelační funkce při velikosti hlavního okna 20 s – mají menší počet nalezených

korelačních bodů a to je žádoucí. Nejlépe rozliší datové toky, které nejsou v korelaci, funkce *minimax*.

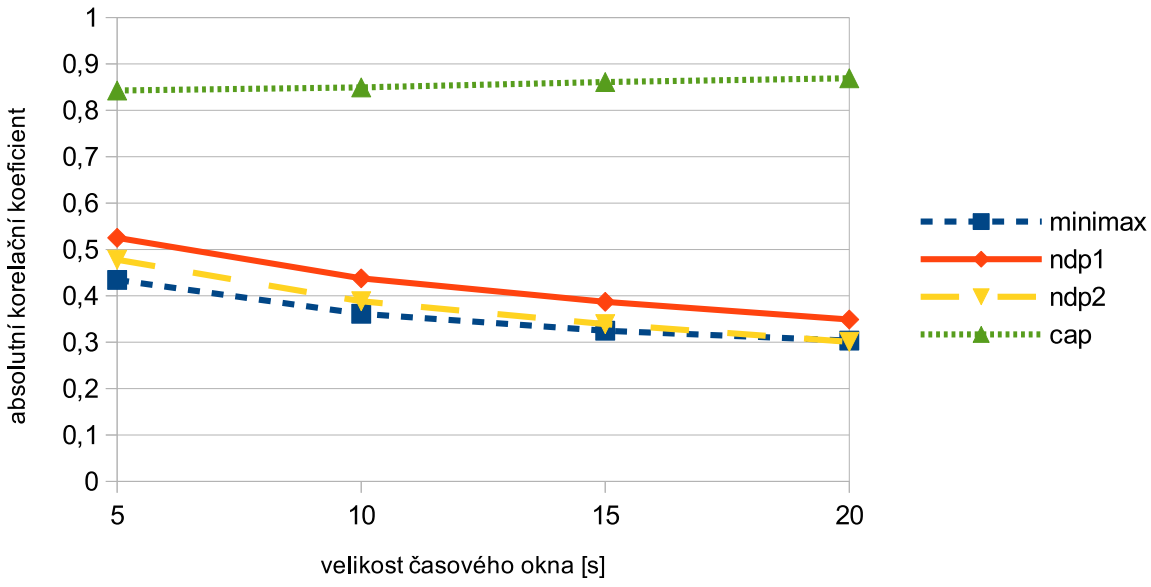
8.6 Experiment č.3 (agregovaný datový tok)

Cílem třetího experimentu bylo určit efektivitu metody při detekci hledaného datového toku, který je agregován s dalším síťovým provozem procházejícím také sítí Tor. Jedná se o případ, kdy klient zároveň inicuje více TCP spojení přes síť Tor a pro tato spojení je vybrán buď ten stejný okruh a nebo stejný strážce pro okruh jiný. V takovém případě nelze kvůli použití TLS na lince (klient – strážce) rozlišit požadavky na TCP spojení v rámci datového toku IP.

Vstupem byly dva datové toky, viz tabulka 8.6, koreloval jsem úsek (destinace – výstupní uzel) datového toku č.1 s úsekem (strážce – klient), který je v tomto případě společný pro oba datové toky.

datový tok	okruh	velikost stránky	parametr λ	destinace
č.1	č.1	250 KB	10 s	vlastní webserver
č.2	č.1	750 KB	15 s	www.seznam.cz

Tabulka 8.6: Datové toky použité v experimentu.



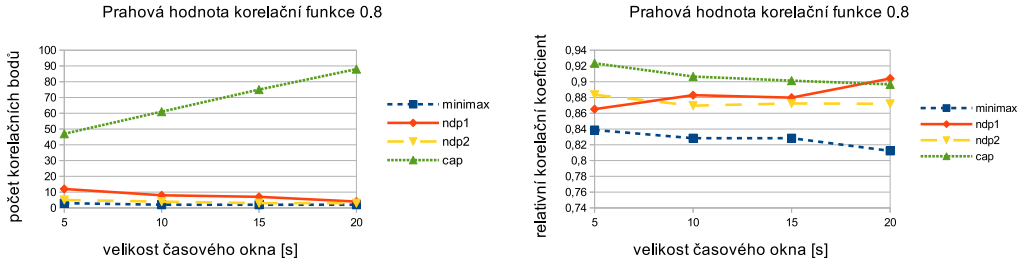
Obrázek 8.6: Absolutní korelační koeficient v závislosti na velikosti časového okna pro vybrané korelační funkce.

V tomto experimentu byla poprvé použita korelační funkce *cap*, která vyhodnocuje objemovou kapacitu korelovaných vektorů v čase (zda se v prohledávaném datovém toku

vyskytl vzor, který je objemově schopen pojmout aktuální vzor hledaného datového toku). V předchozích experimentech nebyla uvažována, protože není určena pro korelaci datových toků, které přenášejí samostatný datový tok.

V grafu na obrázku 8.6 jsou výsledky absolutního korelačního koeficientu pro jednotlivé korelační funkce. Funkce *ndp1, ndp2, minimax* porovnávají přímou podobnost dvou vektorů a za podmínek tohoto experimentu si nevedou příliš dobře. Vzhledem ke stanovené dolní hranici absolutního korelačního koeficientu (0.3) jsou jejich výsledky spíše neutrální vzhledem k určení závislosti datových toků při velikosti hlavního okna 5 s a s rostoucí velikostí hlavního okna tíhnou výsledky k závěru, že mezi datovými toky spíše není závislost. Naopak nově představená korelační funkce *cap* určila závislost mezi datovými toky úspěšně.

Počet nalezených korelačních bodů, viz graf na obrázku 8.7a, pro prahovou hodnotu korelační funkce 0.8 je pro dříve představené funkce *ndp1, ndp2, minimax* velice nízký. Těchto pár nalezených korelačních bodů je výsledkem toho, že hledaný datový tok se v agregátu na některých místech ne zcela „překrývá“ s dalším síťovým provozem a tím pádem zůstává jeho původní charakteristika zachována. Podstatně lépe je na tom korelační funkce *cap*, počet nalezených korelačních bodů stejně jako v předchozích experimentech s rostoucí velikostí hlavního okna stoupá.



(a) (b)

Obrázek 8.7: Počet nalezených korelačních bodů (a) a relativní korelační koeficient (b) v závislosti na velikosti časového okna při prahové hodnotě korelační funkce 0.8.

Závěrem třetího experimentu je, že korelační funkce navržené pro vyhodnocení závislosti dvou datových toků přenášejících samostatný datový tok nejsou příliš efektivní v případě, kdy je tento hledaný datový tok agregován s dalším síťovým provozem, jenž nelze odlišit. V tomto případě se naopak osvědčila nově představená korelační funkce *cap*, její výsledky jsou uspokojivé, ale je nutno počítat se zvýšenou mírou případů *false positives*, která je dána podstatou funkce, viz definice funkce v sekci 6.6.4.

8.7 Experiment č.4 (geografická vzdálenost uzlů okruhu)

Cílem čtvrtého experimentu bylo zjistit, jaký vliv má na metodu geografická vzdálenost uzlů, které tvoří okruh.

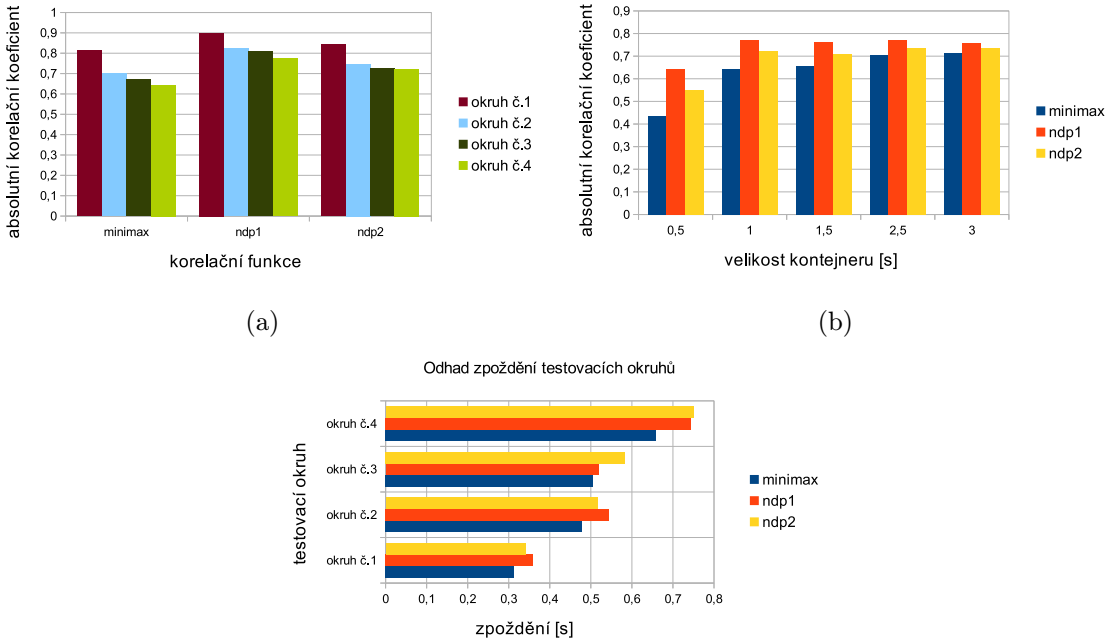
Pro každý z testovacích okruhů č.1 až č.4, viz tabulka 8.7, jsem přes tyto okruhy generoval datový tok, v rámci kterého jsem se dotazoval na vlastní webový server. Okruhy jsou číslovány podle vzdálenosti od geograficky nejkratšího po nejdelší.

V grafu na obrázku 8.8a jsou hodnoty absolutního korelačního koeficientu pro jednotlivé okruhy. Odlišnosti mezi jednotlivými korelačními funkcemi potvrzují opět výsledky experimentu č.1. Prvním dílčím výsledkem tohoto experimentu je, že s rostoucí geografickou

datový tok	okruh	velikost stránky	parametr λ	destinace
č.1	č.1	250 KB	15 s	vlastní webserver
č.2	č.2	250 KB	15 s	vlastní webserver
č.3	č.3	250 KB	15 s	vlastní webserver
č.4	č.4	250 KB	15 s	vlastní webserver

Tabulka 8.7: Datové toky použité v experimentu.

vzdáleností míra závislosti klesá. Tento výsledek podle mě souvisí s faktem, že s rostoucí vzdáleností je rozptyl zpoždění více proměnlivý. To se nepříznivě projeví na výsledku korelace, protože jednotlivé pakety přeskakují mezi kontejnery uvnitř časového okna.



(a) (b)

(c)

Obrázek 8.8: (a) Absolutní korelační koeficient při použití testovacích okruhů s rostoucí fyzickou vzdáleností. (b) Absolutní korelační koeficient při použití různé velikosti kontejneru pro testovací okruh č.4 (nejdelší). (c) Odhad zpoždění testovacích okruhů jako vedlejší produkt filtru shody.

Předchozí zjištění jsem podložil dalším dílčím experimentem, ve kterém zjišťuji pro testovací okruh č.4 (je nejdelší), jaký vliv má na výsledek korelace velikost kontejneru uvnitř časového okna. Při větší velikosti kontejneru jsou korelované datové vektory kratší, charakteristiky datových toků jsou v nich více agregovány a pakety tak mají méně možností, kdy mohou mezi jednotlivými kontejnery přeskočit. Výsledky tohoto testu jsou v grafu na obrázku 8.8b. Na velikosti vedlejšího okna opravdu záleží, 0.5 s je málo, pakety pak mezi kontejnery často přeskakují, ale pro velikost kontejneru 1 s a více se výsledky stabilizují.

Ideální hodnotou se jeví velikost 1 s, jednak dává toto nastavení dobré výsledky a zároveň je to vhodná úroveň granularity pro zachování unikátních charakteristik datového toku.

S fyzickou vzdáleností souvisí i nastavení parametrů metody týkajících se minimálního a maximálního uvažovaného zpoždění v síti Tor. Protože tyto parametry udávají interval prohledávání v cílovém datovém toku (ten, ve kterém je vyhledáván vzor), je důležité, aby nebyla hodnota maximálního zpoždění nastavena na nižší hodnotu, než jaké může ve skutečnosti být. Díky použití funkce *match filter* je zpětnou vazbou korelační funkce také odhadované zpoždění pro každý vyhodnocený bod v prohledávaném datovém toku. Průměrem všech těchto hodnot lze získat celkový odhad zpoždění datového toku.

Odhadnutá doba zpoždění pro všechny testovací okruhy je uvedena v grafu na obrázku 8.8c. Pro jednotlivé korelační funkce se výsledky mírně liší, to je dáno tím, že odhad zpoždění je počítán pouze z nalezených korelačních bodů.

Závěrem čtvrtého experimentu je, že s rostoucí fyzickou vzdáleností okruhů jsou charakteristiky datového toku zkresleny více proměnlivým rozptylem zpoždění. To ovlivňuje výsledek korelace pro všechny uvedené korelační funkce, ale tento vliv lze částečně eliminovat volbou vhodné velikosti kontejneru uvnitř časového okna.

8.8 Experiment č.5 (objem datového toku v čase)

Cílem pátého experimentu bylo zjistit, jaký vliv má na výsledek korelace objem datového toku v čase. Generoval jsem 5 různých datových toků, viz tabulka 8.8, v rámci kterých jsem se dotazoval na webové stránky na vlastním serveru o různých velikostech.

dat. tok	okruh	velikost stránky	celková velikost	parametr λ	destinace
č.1	č.1	250 B	50 KB	10 s	webserver
č.2	č.1	10 KB	500 KB	10 s	webserver
č.3	č.1	250 KB	10 MB	10 s	webserver
č.4	č.1	500 KB	20 MB	10 s	webserver

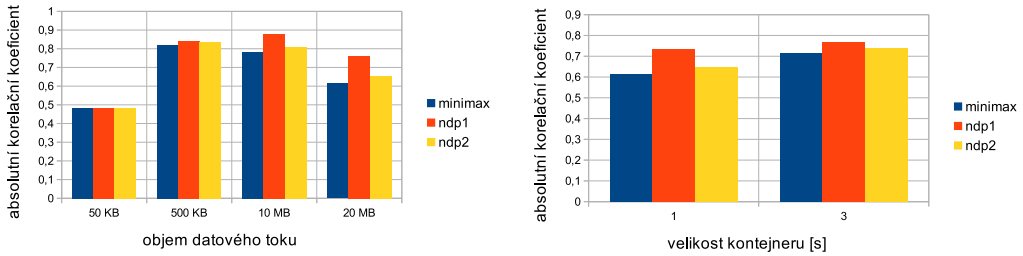
Tabulka 8.8: Datové toky použité v experimentu.

Velikost stránky 250 B považuji za okrajový případ, stránky o této velikosti se vyskytují velice zřídka[7], ale je to specifický případ, kdy je celá stránka přenášena právě v 1 Tor buňce a kvůli tomu ji nelze odlišit od řídicích buněk Toru pevné velikosti. A proto jsem v tomto případě vyřadil odfiltrování paketů, které přenášejí pouze 1 Tor buňku.

V grafu na obrázku 8.9a je uveden absolutní korelační koeficient v závislosti na velikosti dotazovaných webových stránek (obecně objemu datového toku v čase). Při velikosti stránky 250 B (celkem 50 KB) jsou výsledky rozporuplné, je to dáno tím, že užitečný obsah datového toku je příliš nevýrazný ve srovnání s režií Toru. Tento výsledek byl očekávaný, protože metoda vyžaduje zřetelnější charakteristiku datového toku. U velikostí stránky 10 KB (celkem 500 KB) a 250 KB (celkem 10 MB) jsou výsledky již v normě a odpovídají závěrům dřívějších experimentů. Naopak při velikosti stránky 500 KB (celkem 20 MB) jsou výsledky o něco horší, protože při větším objemu dat za jednotku času je pravděpodobnější, že kvůli rozptylu přeskočí do vedlejšího kontejneru v časovém okně více paketů.

Abych tento předpoklad ověřil, provedl jsem pro datový tok o celkovém objemu 20 MB opakovaně experiment při velikosti kontejneru 3 s a výsledek je uveden v grafu na obrázku 8.9b. Hodnota koeficientu se při větší velikosti kontejneru zvýšila a celkový výsledek je opět v normě společně s dalšími velikostmi stránky.

[7] http://www.webperformancetoday.com/2013/06/05/web-page-growth-2010-2013/



(a) (b)

Obrázek 8.9: (a) Absolutní korelační koeficient v závislosti na objemu datového toku v čase (dáno velikostí přenášené stránky). (b) Absolutní korelační koeficient v závislosti na velikosti kontejneru uvnitř časového okna.

Závěrem pátého experimentu je, že metoda je při zpracování datových toků s vyšším počtem paketů za jednotku času méně efektivní v případě, že je hodnota parametru udávajícího velikost kontejneru uvnitř okna příliš nízká.

8.9 Experiment č.6 (doba zpracování)

V šestém experimentu jsem se zaměřil na dobu zpracování dat navrženou metodou. Z návrhu vyplývá, že časová i paměťová složitost metody je lineární – $\mathcal{O}(n)$ v závislosti na počtu paketů a délce dat.

Celková doba zpracování je dána nejen délkou (dobou trvání) datového toku, ale průměrným počtem přenesených paketů za jednotku času. Tyto je pak možné brát jako vstup metody při určení její časové i paměťové složitosti. Konstantu časové složitosti určuje nastavení parametrů korelační metody.

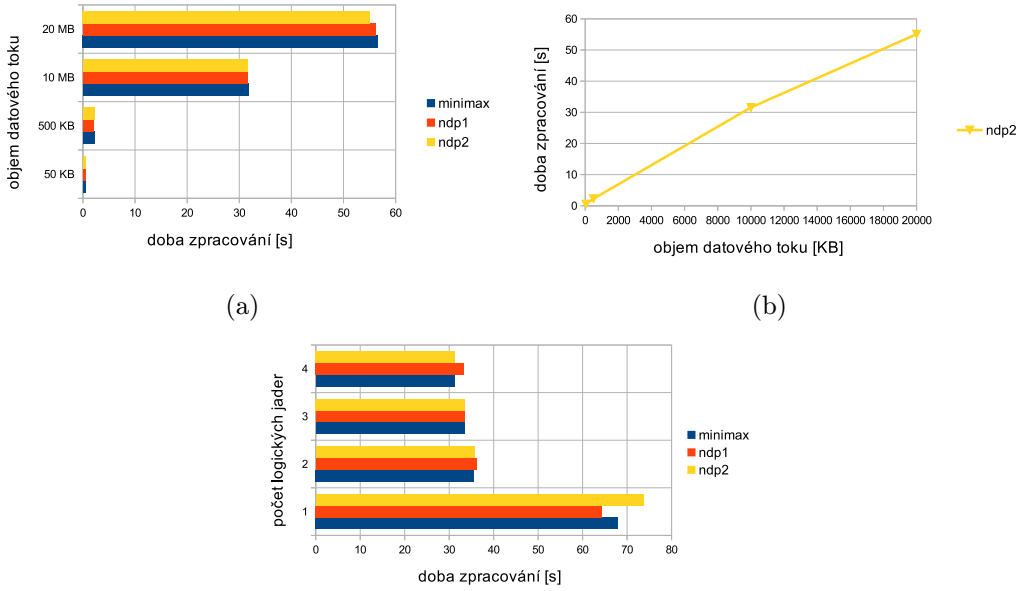
Při testu časové složitosti jsem použil datové toky z předchozího experimentu, viz tabulka 8.8. Celkový objem datového toku tvoří všechny požadavky na webovou stránku o dané velikost.

Celková doba korelace pro tyto datové toky je uvedena v grafu na obrázku 8.10a. S rostoucím objemem datového toku, tedy se zvyšujícím se průměrným počtem paketů za jednotku času, se doba zpracování lineárně zvyšuje, nejlépe je to vidět při porovnání datového toku přenášejícího stránky o velikostech 250 KB (celkem 10 MB) a 500 KB (celkem 20 MB).

V grafu na obrázku 8.10b jsem se pokusil zachytit linearitu doby zpracování s rostoucím objemem datového toku konkrétně pro korelační funkci *ndp2*.

Implementace metody je paralelní, v grafu na obrázku 8.10c uvádím dobu zpracování v závislosti na počtu použitých logických jader procesoru. Testoval jsem na procesoru, který má k dispozici pouze 2 fyzická jádra s technologií HT (HyperThreading), celkově 4 logická jádra. Z výsledků je vidět, že při použití n fyzických jader dojde téměř k n-násobnému zrychlení. Konkrétně pro procesor použitý při testování je vidět téměř dvojnásobné zrychlení při použití jednoho a dvou fyzických jader.

Závěrem šestého experimentu je, že naměřená doba zpracování skutečně odpovídá teoretické složitosti navrženého algoritmu. Metoda je dobře paralelizovatelná a při vyšším stupni paralelizace je možné dosáhnout konstantní časové složitosti $\mathcal{O}(1)$.



Obrázek 8.10: (a) Doba zpracování v závislosti na objemu datového toku při použití 4 logických jader procesoru. (b) Doba zpracování v závislosti na objemu datového toku pro korelační funkci *ndp2* (znázornění linearity) – osa x má logaritmické měřítko. (c) Doba zpracování datového toku o objemu 20 MB v závislosti na počtu použitých logických jader procesoru.

8.10 Celkové vyhodnocení experimentů

Efektivita metody při rozlišení závislosti datových toků je dobrá. Funkce *ndp2* dobře rozlišuje datové toky, které jsou, respektive nejsou v korelaci. Funkce *ndp1* má tendenci závislost uměle zvyšovat a funkce *minimax* naopak snižovat. Přesto všechny uvedené funkce správně rozpoznaly závislost mezi datovými toky a nenastal žádný případ *false positives* ani *false negatives*.

Při detekci hledaného datového toku v agregovaném datovém toku si korelační funkce navržené pro detekci samostatného datového toku dávají nejednoznačné výsledky, kdy není možné říci, zda závislost mezi datovými toky je či není. Ale korelační funkce *cap* navržená speciálně pro tento případ má výsledky uspokojivé. S tím, že je potřeba počítat s vyšší mírou *false positives* výsledků.

Výsledek korelace může být ovlivněn i geografickou vzdáleností okruhů, vliv proměnlivého rozptylu zpoždění je možné částečně eliminovat vhodnou velikostí kontejneru uvnitř časového okna. Při větší velikosti kontejneru nedochází tak často k jevu, že pakety mezi kontejnery přeskakují. Stejný závěr platí i pro vliv objemu datového toku v čase na výsledek korelace.

Doba zpracování v závislosti na objemu (počtu paketů za jednotku času) a délce datového toku je skutečně lineární. Metoda je dobře paralelizovatelná.

Kapitola 9

Závěr

V rámci semestrálního projektu jsem nastudoval problém anonymizace na Internetu a anonymizační techniky, které se tento problém snaží řešit. V kapitole 2 jsem se blíže zaměřil na sítě Mix-Net a Onion Routing, z nichž Tor přejímá významnou část mechanismů anonymizace. Podrobně jsem nastudoval principy sítě Tor a popsal je v kapitole 3, s tím, že jsem se snažil zachytit všechny významné změny, kterými za dobu jeho letitého vývoje prošel. Zaměřil jsem se především na způsob realizace anonymizovaného spojení, konstrukci virtuálních okruhů a jakým způsobem se tato anonymizovaná spojení projeví na úrovni IP a TCP datových toků. Popis sítě Tor jsem dále rozvedl a podrobně zdokumentoval v technické zprávě [17], která se zabývá fungováním sítě Tor na úrovni specifikace protokolu.

V diplomové práci jsem tuto část rozšířil o stručnou statistiku sítě Tor, ve které jsou zachyceny užitečné informace pro realizaci korelačního útoku. Jedná se o současnou velikost sítě Tor, její přenosové kapacity, geografické rozložení uživatelů a zpoždění přenosu datových toků.

Ze semestrálního projektu jsem převzal částečně kapitolu 4, kde jsem shrnul využití analýzy síťového provozu v kontextu korelačního útoku na síť Tor, principy korelačního útoku, podmínky pro jeho realizaci v síti Tor a lokalitu síťového provozu. Závěrem této části práce je, že korelační útok na síť Tor je díky požadavku na nízké zpoždění doručení dat možný, což je podloženo řadou existujících korelačních metod. Ze studií zabývajících se lokalitou provozu vyplynulo, že se útočník může dostat do pozice, aby mohl korelační útok realizovat. Uživatelé Toru často využívají Tor k přístupu k té části Internetu, která se taktéž nachází v jejich zemi.

V rámci diplomové práce jsem navázal kapitolou 5, ve které jsem popsal existující korelační metody. Některé pasáže jsem převzal ze semestrálního projektu, ale většina prošla radikálními úpravami. Žádná z nastudovaných metod nebyla přímo použitelná v kontextu útočníka zákonných odposlechů, velká část z nich koreluje datové toky uvnitř anonymizační sítě a nebo vyžaduje aktivního útočníka, ale některé principy těchto metod mě inspirovaly při návrhu vlastní korelační metody, kterou jsem popsal v kapitole 6.

Vlastní korelační metodu jsem postavil na hledání lokálních podobností charakteristik datových toků z hlediska objemu přenášených dat v čase. Převzal jsem způsob hledání korelačních bodů a vylepšil ho použitím filtru shody, který hledá nejlepší shodu signatur v časovém intervalu určeném minimálním a maximálním zpožděním sítě Tor. Při porovnávání lokálních charakteristik datových toků jsem převzal několik funkcí pro porovnání podobnosti extrahovaných vektorů a navrhl jednu vlastní, která se specializuje na případ, kdy je objemová charakteristika datového toku zkreslena agregací s jiným datovým tokem.

Metoda se specializuje na korelaci datových toků ve směru od cíle komunikace zpět

k uživateli, ale s drobnými modifikacemi by byla použitelná i pro směr opačný. Volba směru korelovaných datových toků vychází ze specializace metody na korelaci datových toků odpovědi webového serveru.

Metodu jsem implementoval několika nástroji, implementační podrobnosti jsem shrnul v kapitole 7. Výsledný program využívá paralelního zpracování.

Navrženou korelační metodu jsem otestoval na reálných datech ze sítě Tor, experimenty popsal v kapitole 8. Abych mohl navrženou korelační metodu ověřit na reálných datech ze sítě Tor, implementoval jsem potřebné nástroje pro generování, zachytávání datových toků a řízení lokální instance Toru. Při testování metody jsem se zaměřil na efektivitu a úspěšnost metody. Z výsledků testování vyplynulo, že navržená metoda je efektivní při rozlišení datových toků, které jsou respektive nejsou v korelaci. Dokáže si poradit i v případě, kdy je charakteristika datového toku zkreslena jiným datovým tokem. Vhodným nastavením parametrů dosahuje dobrých výsledků i pro datové toky přenášené fyzickými cestami s rostoucí geografickou vzdáleností. V rámci testování jsem se zaměřil i na dobu zpracování v závislosti na velikosti vstupu (průměrný objem datového toku v čase). Časová složitost metody je lineární, ale při paralelním zpracování může metoda při dostatečném počtu fyzických jader dosáhnout až konstantní časové složitosti.

Současné řešení by se dalo vylepšit lepším odhadem počtu buněk Toru přenášených uvnitř TLS záznamů. Také je zde prostor pro návrh korelační funkce s nižší mírou *false positives*, která správně rozliší datové toky i pro případ, kdy je charakteristika korelovaného datového toku zkreslena agregací s jinými datovými toky přenášenými tím stejným virtuálním okruhem. Přínosem by bylo i ověření efektivity metody při použití maskování síťového provozu Toru za jiný protokol pomocí *pluggable transports*.

Uvedené řešení diplomové práce má výhodu v tom, že navržená korelační metoda je dobře rozšiřitelná. Způsob extrakce korelovaných vektorů z metadat datových toků i vlastní korelační funkce jsou nahraditelné. Bylo by tedy možné vylepšit řešení tím, že budou datové toky korelovány na základě jiných charakteristik než je objem dat v čase. Použitím unikátnějších charakteristik datových toků by mohly být výsledky korelace ještě o něco přesnější.

Literatura

[1] Tor Control Protocol Specification. [online], [cit. 2014-04-05].
URL https://gitweb.torproject.org/torspec.git?a=blob_plain;hb=HEAD;f=control-spec.txt

[2] Tor Directory Protocol Specification. [online], [cit. 2014-04-05].
URL
https://gitweb.torproject.org/torspec.git/blob_plain/HEAD:/dir-spec.txt

[3] Tor Path Specification. [online], [cit. 2014-04-21].
URL https://gitweb.torproject.org/torspec.git?a=blob_plain;hb=HEAD;f=path-spec.txt

[4] Tor Project : Anonymity Online. [online], [cit. 2013-10-27].
URL https://www.torproject.org/

[5] Tor Project : Volunteers. [online], [cit. 2014-05-26].
URL https://www.torproject.org/getinvolved/volunteer.html.en

[6] Tor Protocol Specification. [online], [cit. 2014-04-05].
URL https:
//gitweb.torproject.org/torspec.git?a=blob_plain;hb=HEAD;f=tor-spec.txt

[7] Akhoondi, M.; Yu, C.; Madhyastha, H. V.: LASTor: A low-latency AS-aware Tor client. In *Security and Privacy (SP), 2012 IEEE Symposium on*, IEEE, 2012, s. 476–490.

[8] Back, A.; Möller, U.; Stiglic, A.: Traffic analysis attacks and trade-offs in anonymity providing systems. In *Information Hiding*, Springer, 2001, s. 245–257.

[9] Bauer, K.; McCoy, D.; Grunwald, D.; aj.: Low-resource routing attacks against tor. In *Proceedings of the 2007 ACM workshop on Privacy in electronic society*, ACM, 2007, s. 11–20.

[10] Bauer, K. S.: *Improving Security and Performance in Low Latency Anonymity Networks*. Dizertační práce, University of Colorado, 2011.

[11] Blond, S. L.; Manils, P.; Abdelberi, C.; aj.: One bad apple spoils the bunch: exploiting P2P applications to trace and profile Tor users. *arXiv preprint arXiv:1103.1518*, 2011.

[12] Blond, S. L.; Manils, P.; Chaabane, A.; aj.: De-anonymizing bittorrent users on tor. *arXiv preprint arXiv:1004.1267*, 2010.

[13] Chaabane, A.; Manils, P.; Kaafar, M. A.: Digging into anonymous traffic: A deep analysis of the tor anonymizing network. In *Network and System Security (NSS), 2010 4th International Conference on*, IEEE, 2010, s. 167–174.

[14] Chakravarty, S.; Stavrou, A.; Keromytis, A. D.: Identifying Proxy Nodes in a Tor Anonymization Circuit. In *Proceedings of the 2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*, SITIS '08, Washington, DC, USA: IEEE Computer Society, 2008, ISBN 978-0-7695-3493-0, s. 633–639, doi:10.1109/SITIS.2008.93.

[15] Chaum, D. L.: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Commun. ACM*, ročník 24, č. 2, Únor 1981: s. 84–90, ISSN 0001-0782, doi:10.1145/358549.358563.

[16] Christin, N.: Traveling the Silk Road: A measurement analysis of a large anonymous online marketplace. In *Proceedings of the 22nd international conference on World Wide Web*, International World Wide Web Conferences Steering Committee, 2013, s. 213–224.

[17] Coufal, Z.; Polčák, L.: Anonymizační síť Tor. Technická zpráva, FIT-TR-2014-02, Brno: Fakulta informačních technologií VUT v Brně, 2014.
URL http://www.fit.vutbr.cz/research/view_pub.php?id=10626

[18] Craven, R. M.: *Traffic analysis of anonymity systems*. Dizertační práce, Clemson University, 2010.

[19] Danezis, G.: The traffic analysis of continuous-time mixes. In *Privacy Enhancing Technologies*, Springer, 2005, s. 35–50.

[20] DeFabbia-Kane, S.: *Analyzing the Effectiveness of Passive Correlation Attacks on the Tor Anonymity Network*. Dizertační práce, Wesleyan University, 2011.

[21] Dhungel, P.; Steiner, M.; Rimac, I.; aj.: Waiting for anonymity: Understanding delays in the Tor overlay. In *Peer-to-Peer Computing (P2P), 2010 IEEE Tenth International Conference on*, IEEE, 2010, s. 1–4.

[22] Dingledine, R.; Mathewson, N.; Murdoch, S.; aj.: Tor: The Second-Generation Onion Router (2014 DRAFT v1), 2014.
URL http://www.cl.cam.ac.uk/~sjm217/papers/tor14design.pdf

[23] Dingledine, R.; Mathewson, N.; Syverson, P.: Tor: The Second-generation Onion Router. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, Berkeley, CA, USA: USENIX Association, 2004, s. 21–21.

[24] Edman, M.; Syverson, P.: AS-awareness in Tor path selection. In *Proceedings of the 16th ACM conference on Computer and communications security*, ACM, 2009, s. 380–389.

[25] Feamster, N.; Dingledine, R.: Location diversity in anonymity networks. In *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, ACM, 2004, s. 66–76.

[26] Goldberg, I. A.: *A pseudonymous communications infrastructure for the internet.* Dizertační práce, University of California, 2000.

[27] Goldschlag, D. M.; Reed, M. G.; Syverson, P. F.: Hiding routing information. In *Information Hiding*, Springer, 1996, s. 137–150.

[28] Hopper, N.; Vasserman, E. Y.; Chan-Tin, E.: How much anonymity does network latency leak? *ACM Transactions on Information and System Security (TISSEC)*, ročník 13, č. 2, 2010: str. 13.

[29] Johnson, A.; Wacek, C.; Jansen, R.; aj.: Users Get Routed: Traffic Correlation on Tor by Realistic Adversaries. In *Proceedings of the 20th ACM conference on Computer and Communications Security (CCS 2013)*, November 2013.

[30] Johnson, N.; McLaughlin, S.; Thompson, J.: Path tracing in TOR networks. In *18th European signal processing conference (EUSIPCO-2010)*, 2010, s. 1856–1860.

[31] Juen, J. P. J.: *Protecting Anonymity in the Presence of Autonomous System and Internet Exchange Level Adversaries.* Diplomová práce, University of Illinois, 2012.

[32] Levine, B. N.; Reiter, M. K.; Wang, C.; aj.: Timing attacks in low-latency mix systems. In *Financial Cryptography*, Springer, 2004, s. 251–265.

[33] Li, B.; Erdin, E.; Gunes, M. H.; aj.: An Overview of Anonymity Technology Usage. *Computer Communications*, 2013.

[34] McCoy, D.; Bauer, K.; Grunwald, D.; aj.: Shining light in dark places: Understanding the Tor network. In *Privacy Enhancing Technologies*, Springer, 2008, s. 63–76.

[35] Murdoch, S. J.; Danezis, G.: Low-cost traffic analysis of Tor. In *Security and Privacy, 2005 IEEE Symposium on*, IEEE, 2005, s. 183–195.

[36] Murdoch, S. J.; Zieliński, P.: Sampled traffic analysis by internet-exchange-level adversaries. In *Privacy Enhancing Technologies*, Springer, 2007, s. 167–183.

[37] Overlier, L.; Syverson, P.: Locating hidden servers. In *Security and Privacy, 2006 IEEE Symposium on*, IEEE, 2006, s. 15–pp.

[38] Panchenko, A.; Lanze, F.; Engel, T.: Improving performance and anonymity in the Tor network. In *Performance Computing and Communications Conference (IPCCC), 2012 IEEE 31st International*, IEEE, 2012, s. 1–10.

[39] Polčák, L.; Hranický, R.: Útoky na systémy pro zákonné odposlechy. Technická zpráva, FIT-TR-2012-008, Brno: Fakulta informačních technologií VUT v Brně, 2012. URL http://www.fit.vutbr.cz/research/view_pub.php?id=10201

[40] Serjantov, A.; Sewell, P.: Passive attack analysis for connection-based anonymity systems. In *Computer Security–ESORICS 2003*, Springer, 2003, s. 116–131.

[41] Shmatikov, V.; Wang, M.-H.: Timing analysis in low-latency mix networks: Attacks and defenses. In *Computer Security–ESORICS 2006*, Springer, 2006, s. 18–33.

[42] Song, D. X.; Wagner, D.; Tian, X.: Timing Analysis of Keystrokes and Timing Attacks on SSH. In *USENIX Security Symposium*, ročník 2001, 2001.

[43] Syverson, P.: Practical Vulnerabilities of the Tor Anonymity Network. *Advances in Cyber Security: Technology, Operation, and Experiences*, 2013: str. 60.

[44] Syverson, P.; Tsudik, G.; Reed, M.; aj.: Towards an analysis of onion routing security. In *Designing Privacy Enhancing Technologies*, Springer, 2001, s. 96–114.

[45] Turin, G.: An introduction to matched filters. *Information Theory, IRE Transactions on*, ročník 6, č. 3, 1960: s. 311–329.

[46] Vasudevan, A.: *An Empirical Study of an Anonymity Metric for Data Networks.* Dizertační práce, 2013.

[47] Wacek, C.; Tan, H.; Bauer, K.; aj.: An Empirical Evaluation of Relay Selection in Tor. In *Proceedings of the Network and Distributed Security Symposium (NDSS)(February 2013)*, 2013.

[48] Wang, X.; Reeves, D. S.; Wu, S. F.: Inter-packet delay based correlation for tracing encrypted connections through stepping stones. In *Computer Security–ESORICS 2002*, Springer, 2002, s. 244–263.

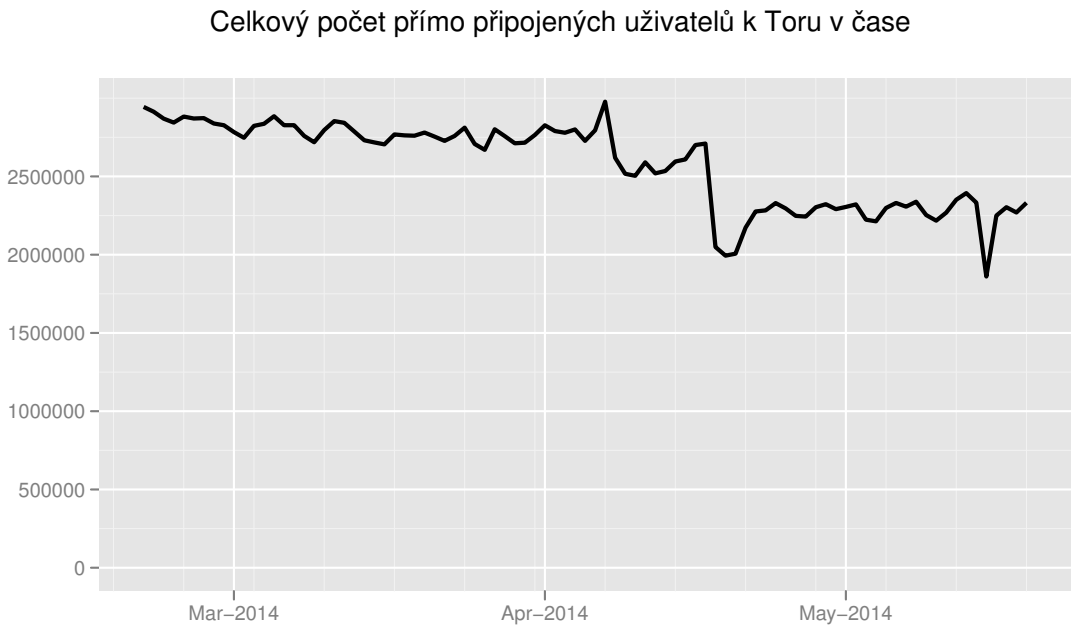
[49] Zhu, Y.; Fu, X.; Bettati, R.: On the effectiveness of continuous-time mixes under flow correlation attacks. Technická zpráva, Technical Report TR2005-2-6, Texas A&M Univ. Computer Science, 2005.

[50] Zhu, Y.; Fu, X.; Bettati, R.; aj.: Anonymity analysis of mix networks against flow-correlation attacks. In *in Proceedings of IEEE Global Communications Conference*, Citeseer, 2005.

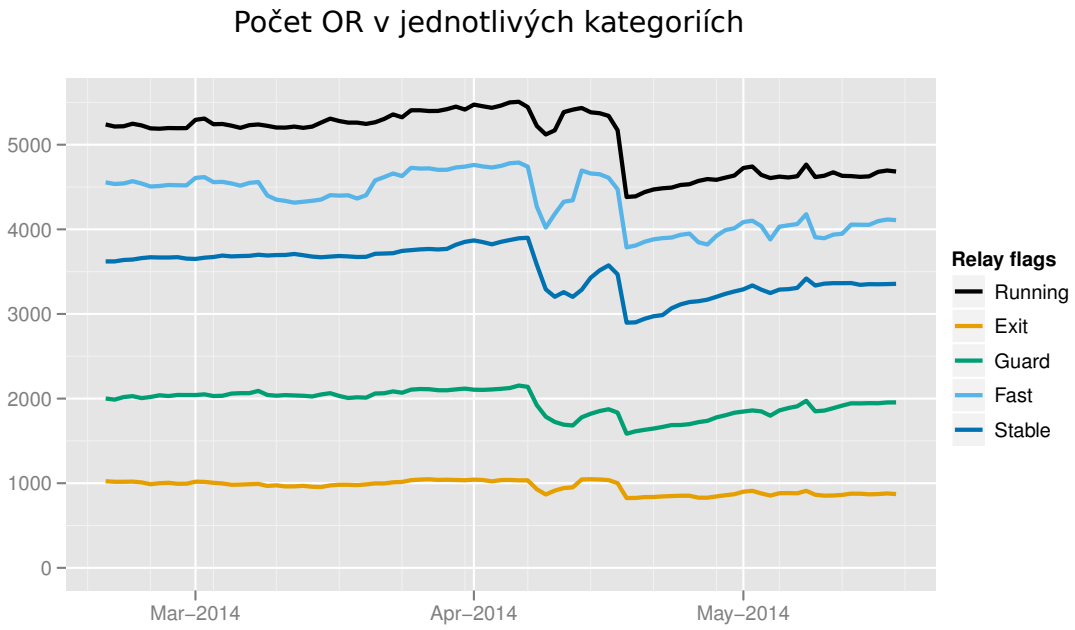
[51] Zhu, Y.; Fu, X.; Graham, B.; aj.: Correlation-Based Traffic Analysis Attacks on Anonymity Networks. *Parallel and Distributed Systems, IEEE Transactions on*, ročník 21, č. 7, July 2010: s. 954–967, ISSN 1045-9219, doi:10.1109/TPDS.2009.146.

Příloha A

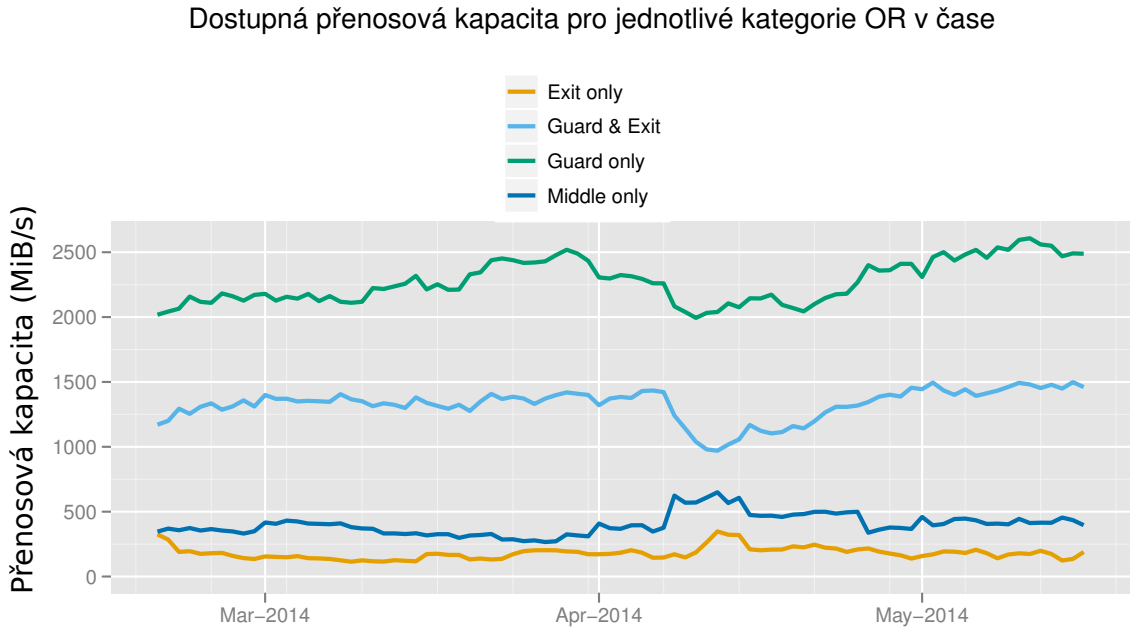
Statistiky sítě Tor



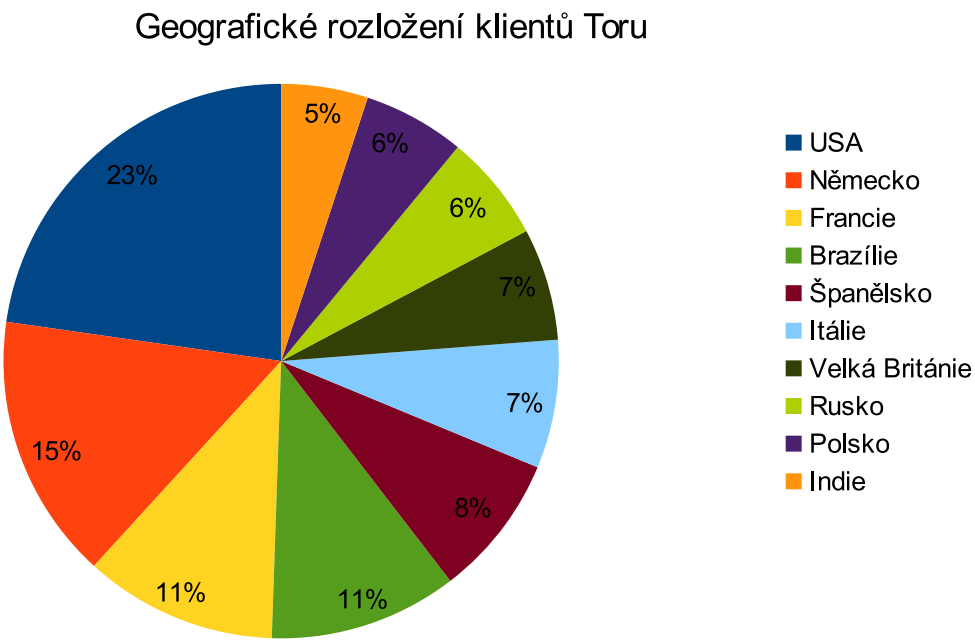
Obrázek A.1: Počet přímo připojených uživatelů k síti Tor (ne přes mosty) (březen 2014 – květen 2014). Převzato z http://metrics.torproject.org.



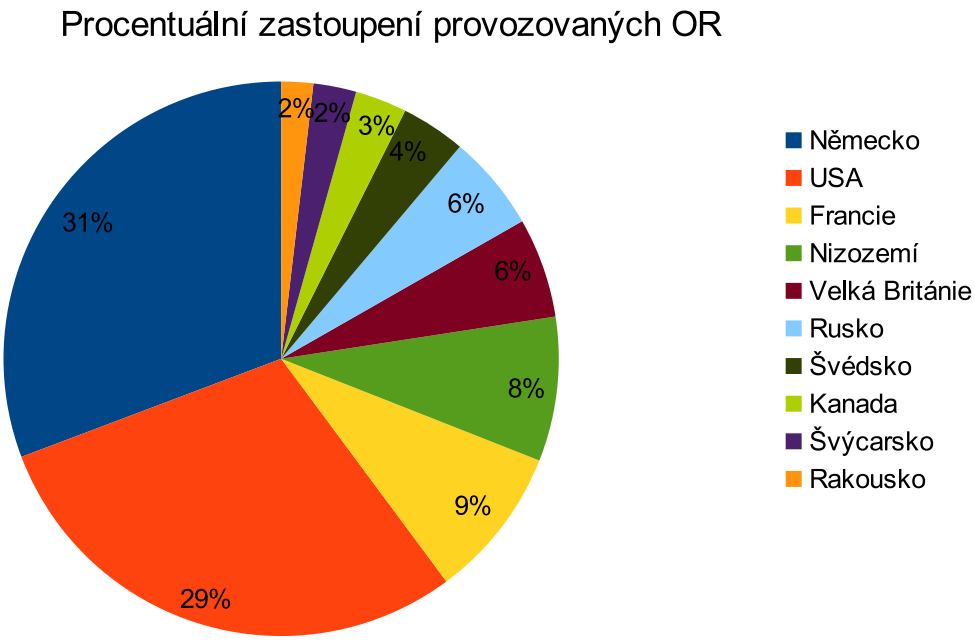
Obrázek A.2: Celkový počet OR v kategoriích dle přidělených statusů (březen 2014 – květen 2014). Převzato z http://metrics.torproject.org.



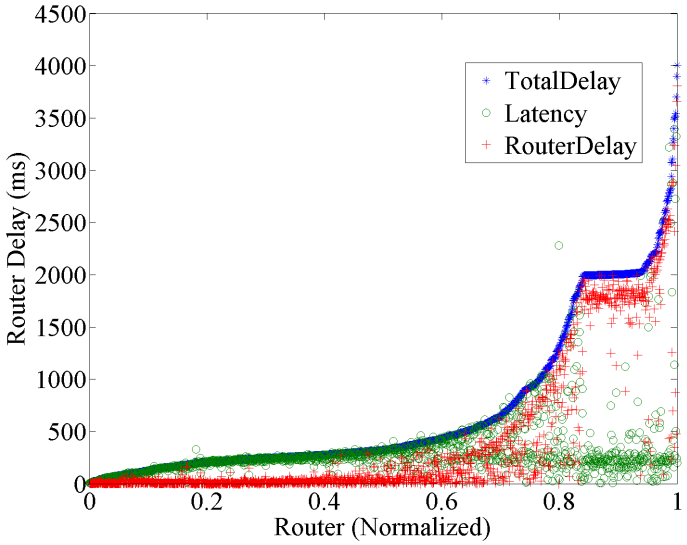
Obrázek A.3: Využití přenosové kapacity sítě Tor v rámci jednotlivých kategorií OR (březen 2014 – květen 2014). Převzato z http://metrics.torproject.org.



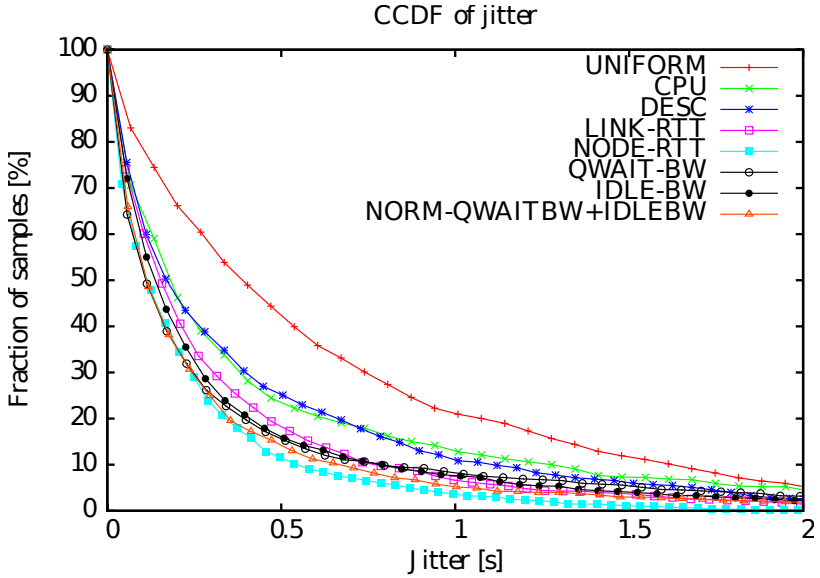
Obrázek A.4: Geografické rozložení klientů Toru pro 10 nejčetnějších zemí.



Obrázek A.5: Procentuální zastoupení provozovaných OR pro 10 nejčetnějších zemí.



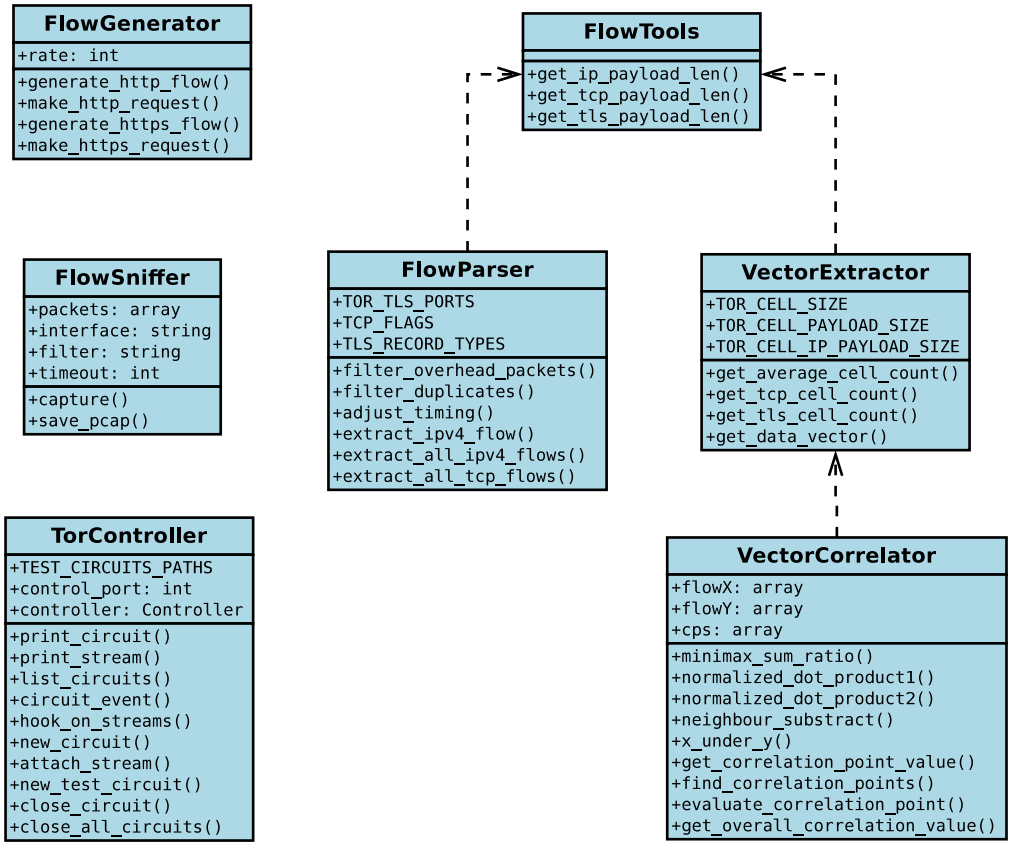
Obrázek A.6: Distributivní rozdělení zpoždění v síti Tor. Celkové zpoždění okruhu (*total delay*) je součtem zpoždění nosné infrastruktury (*latency*) a zpoždění zpracování na úrovni Toru (*router delay*). [21]



Obrázek A.7: Komplementární kumulativní distributivní funkce rozptylu zpoždění v síti Tor. Ukazuje, jaký byl rozptyl zpoždění pro určité procento testovaných okruhů pro různé metriky algoritmu výběru cesty. V současné implementaci Tor využívá metriku IDLE-BW, kdy algoritmus výběru cesty volí OR na základě jejich dostupné přenosové kapacity. [38]

Příloha B

Implementační diagram tříd



Obrázek B.1: Diagram tříd na úrovni implementace. Obsahuje třídy vlastní korelační metody i třídy navržené pro testování. Kvůli zachování přehlednosti neuvádím parametry metod.