Sem vložte zadání Vaší práce.

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Automatická detekce podezřelého síťového provozu pomocí blacklistů

Bc. Filip Šuster

Katedra počítačových systémů
Vedoucí práce: Ing. Tomáš Čejka, Ph.D.

10. ledna 2019

Poděkování

Tímto bych chtěl poděkovat vedoucímu práce Ing. Tomáši Čejkovi, Ph.D., jehož pravidelné konzultace a motivace při práci značně přispěly k výsledku mého snažení. Dále děkuji kolegům Ing. Václavu Bartošovi a Ing. Tomáši Jánskému, kteří mi byli nápomocni při optimalizaci provedení a řešení problémů. Děkuji své rodině a přítelkyni, měli se mnou svatou trpělivost a jejich podpora byla obdivuhodná. Můj dík rovněž patří přátelům a kolegům, kteří věřili v můj úspěch. V neposlední řadě děkuji Ludwigu van Beethovenovi za jeho skvostné symfonie, jejichž poslech mě při tvorbě práce inspiroval.

Prohlášení

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

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

V Praze dne 10. ledna 2019 .....

České vysoké učení technické v Praze
Fakulta informačních technologií
© 2019 Filip Šuster. Všechna práva vyhrazena.
Tato práce vznikla jako školní dílo na Českém vysokém učení technickém v Praze, Fakultě informačních technologií. Práce je chráněna právními předpisy a mezinárodními úmluvami o právu autorském a právech souvisejících s právem autorským. K jejímu užití, s výjimkou bezúplatných zákonných licencí a nad rámec oprávnění uvedených v Prohlášení na předchozí straně, je nezbytný souhlas autora.

Odkaz na tuto práci

Šuster, Filip. *Automatická detekce podezřelého síťového provozu pomocí blacklistů*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2019.

Abstrakt

Tato práce se zabývá implementací sady modulů pro detekci podezřelého síťového provozu pomocí veřejných blacklistů. Kromě základní detekce, která spočívá v nahlášení všech síťových toků, umožňují vytvořené moduly sledovat další provoz klientů, kteří s blacklistovanou entitou komunikovali.

Cílem práce je využít analýzu dodatečně zachycených informací o provozu podezřelých klientů k lepšímu a přesnějšímu rozhodování o podstatě/kontextu komunikace. Díky této analýze je možné odhalit, jestli základní detekce není jen falešný poplach. K zachytávání dodatečných informací v reálném čase byl vytvořen modul, tzv. *adaptivní filtr*, který patří k hlavním přínosům této práce. Práce se zaměřuje zejména na využití veřejně dostupných seznamů Command&Control serverů a analýzu provozu klientů, kteří s těmito servery komunikují.

Všechny vytvořené softwarové nástroje jsou součástí open source projektu NEMEA, který se používá k analýze provozu a detekci bezpečnostních událostí v národní akademické síti CESNET2.

Klíčová slova blacklist, C&C, botnet, síťový provoz, detekce, adaptivní filtr, NEMEA, C/C++, Python

Abstract

This thesis deals with implementation of a set of modules for detection of suspicious network traffic with the use of public blacklists. In addition to basic detection, which consists in reporting all network flows, the modules can be used to track additional traffic of clients who communicated with the blacklisted entity.

The aim of the thesis is to use the analysis of additionally captured information about the suspicious clients' traffic for better and more precise decision about the essence/context of communication. This analysis makes it possible to reveal whether the basic detection is not just a false alarm. To capture additional real-time information, a module called *adaptive filter*, which is one of the main benefits of this thesis, has been created. The work focuses mainly on the use of publicly accessible lists of Command&Control servers as well as on the analysis of the traffic of clients communicating with these servers.

All created software tools are part of the open-source NEMEA project, which is used to analyze traffic and detection of security incidents in the national academic network CESNET2.

Keywords blacklist, C&C, botnet, network traffic, detection, adaptive filter, NEMEA, C/C++, Python

Obsah

Úvod **1**

1 Analýza a rešerše **7**
1.1 Botnety a C&C servery 7
1.2 Veřejné blacklisty 13
1.3 Metody analýzy síťového provozu 16
1.4 Systém NEMEA 19

2 Návrh **27**
2.1 Sada modulů blacklistfilter 27

3 Implementace a konfigurace modulů **31**
3.1 Blacklist downloader 31
3.2 Detektory 35
3.3 Agregátory 38
3.4 Adaptivní filtr 39
3.5 Reportér 42

4 Testování a nasazení **43**
4.1 Testovací prostředí 43
4.2 Testování detekčních modulů 44
4.3 Testování agregačního modulu 45
4.4 Statistiky z reálného provozu 47

5 Vyhodnocení dat **49**

Závěr **53**

Literatura **55**

A Seznam použitých zkratek **59**

B Ukázka IDEA zprávy **61**

C Obsah přiloženého CD **63**

Seznam obrázků

0.1 Schéma detekce nakažených botů 3
0.2 Detekce klientů DNS 4
0.3 Horizontální sken sítě s blacklistovanou adresou 5

1.1 Centralizovaná architektura botnetu 8
1.2 Decentralizovaná architektura botnetu 9
1.3 Hybridní architektura botnetu 10
1.4 ISO/OSI model 16
1.5 Architektura monitorování provozu pomocí flows 18
1.6 Ukázka propojení NEMEA modulů 19
1.7 Podporované datové formáty systému NEMEA 20
1.8 Ukázka použití IFC rozhraní v NEMEA modulech 22
1.9 Topologie sítě CESNET2 23
1.10 Bezpečnostní systémy v síti CESNET2 24

2.1 Návrh sady modulů blacklistfilter 28

3.1 Vývojový diagram modulu Downloader 34
3.2 Ukázka prefix tree 37
3.3 Implementace Agregátoru 39
3.4 Diagram tříd modelu Adaptivní filtr 40

4.1 Propustnost detekčních modulů v závislosti na počtu blacklistovaných entit 45
4.2 Agregační modul — Graf spotřeby paměti v čase 47

Seznam tabulek

1.1 Datové typy UniRec 20

4.1 Propustnost detekčních modulů podle počtu použitých entit (pro různé sady blacklistů) 46
4.2 Využití RAM detekčních modulů 46
4.3 Statistiky detektorů z měření v reálném provozu 48

Úvod

Monitorování a analýza síťového provozu je nezbytnou součástí každé infrastruktury. Sledování provozu může vést ke zlepšování výkonu, stability a především bezpečnosti sítě. To platí dvojnásob pro poskytovatele internetu a síťových služeb. Uživatelé internetu podléhají čím dál sofistikovanějším podvodům a útokům a jejich obezřetnost nemusí být vždy dostatečná. Snadno si například stáhnou emailovou přílohu se škodlivým kódem v domnění, že se jedná o legitimní email od kolegy. Pokud uživatel nemá nainstalovaný antivirus, jeho zařízení se může snadno stát součástí botnetu (síť nakažených zařízení).

Navíc v době rozmachu IoT (Internet of Things) zařízení, které jsou často špatně zabezpečené, je mezi útočníky velmi populární tato zařízení zneužívat. Pokud útočník takto ovládá tisíce zařízení, může např. provádět Distributed Denial-of-Service (DDoS) útoky. Na webu dokonce existují servery, kde si může kdokoliv takový útok objednat, takové službě se přezdívá Booter (více v [1]). Je také běžné, že útočníci zneužívají infikovaná zařízení k těžení kryptoměn. V takovém případě postižený uživatel nemusí nic tušit, pouze může vnímat snížení výkonu počítače. Další z řady hrozeb představuje tzv. ransomware, který po spuštění na cílové stanici zašifruje důležité soubory a požaduje výkupné pro jejich odemčení. Ransomware zvaný WannaCry dokonce pronikal do systému bez zásahu uživatele, jednalo se tedy o počítačového „červa“, který se sám replikoval po síti na další zařízení. Zmíněné typy bezpečnostních hrozeb mají různou závažnost, avšak jejich detekce je velice žádoucí, pokud chceme udržet síť bezpečnou.

Cílem této práce je vytvoření sady modulů pro systém NEMEA [2], která bude automaticky zpracovávat podezřelý provoz pomocí veřejných blacklistů. Blacklisty obsahují zpravidla IP, URL či DNS záznamy. Modul se bude zaměřovat zejména na detekci komunikace Command&Control (C&C) serverů a jejich klientů — botů. Modul bude své detekční mechanismy přizpůsobovat aktuálnímu provozu, neboť bude dohledávat související komunikaci stanic, které se podílely na podezřelém provozu. Vznikne tak tzv. adaptivní filtr, který bude

schopný zachytávat podezřelý provoz potenciálně škodlivých/nakažených klientských stanic a ukládat tato data k pozdější analýze. Tímto způsobem bude možné lépe rozhodovat o tom, zda detekovaná komunikace znamená reálnou hrozbu. Důkladnější analýza může také přispět ke zpřesnění nahlášených alertů a administrátorům sítí tak poskytnout srozumitelnější informaci, jak incident řešit.

Modul byl vytvořen pro open-source systém NEMEA, který je určen k analýze informací o síťovém provozu v reálném čase a k detekci podezřelého provozu. Výsledky této práce se staly součástí oficiální distribuce systému NEMEA, který je vyvíjen a používán v národní akademické infrastruktuře CESNET2.

Tato práce navazuje na existující NEMEA moduly, které byly určeny k filtrování síťových toků podle blacklistů. Vzhledem k nevýhodám a omezením stávajícího řešení byly v rámci této diplomové práce moduly přepracovány a rozšířeny o funkcionalitu adaptivního filtrování, které bude popsáno v dalších kapitolách.

Kapitola 1 zasazuje práci do širšího kontextu, obsahuje popis botnetu, jeho architektur a jakých možných útoků je schopen. Dále rozebírá veřejné blacklisty a jejich typy. Vysvětluje metody analýzy síťového provozu a obsahuje popis systému NEMEA. Kapitola 2 obsahuje návrh sady modulů *blacklistfilter* a mechanismus adaptivního filtrování. Kapitola 3 popisuje implementaci a konfiguraci všech vytvořených modulů. Kapitola 4 obsahuje testování modulů a statistiky nasazení v reálném provozu. Kapitola 5 popisuje možnosti vyhodnocení zachycených dat adaptivním filtrem.

Motivace

Hlavní motivací práce je monitorování podezřelých klientů v síti a následné vyhodnocování jejich aktivit. Podezřelým se stává klient tehdy, pokud komunikuje s entitou[1], která je na některém z veřejných blacklistů. Tyto blacklisty jsou různých typů, ať už se jedná o phishing, komunikaci botnetu či spam. Přímočaré použití vytvořených blacklist modulů je následující — je zachycena komunikace klienta a blacklistované entity, tento provoz je agregován a poté nahlášen jako alert. Toto použití může být pro některé případy vhodné, např. když chce administrátor sítě sledovat, jestli některý z klientů nepoužívá nástroje pro těžbu kryptoměn. Jako blacklist se v tomto případě použije veřejně dostupný seznam tzv. *mining pool* serverů[2]. Další takový scénář může být, pokud administrátor chce v síti detekovat použití anonymizačních nástrojů ToR[3].

[1] V kontextu této práce je entita subjekt, který je identifikován IP, URL nebo DNS záznamem

[2] server který rozděluje práci (těžbu) mezi klienty

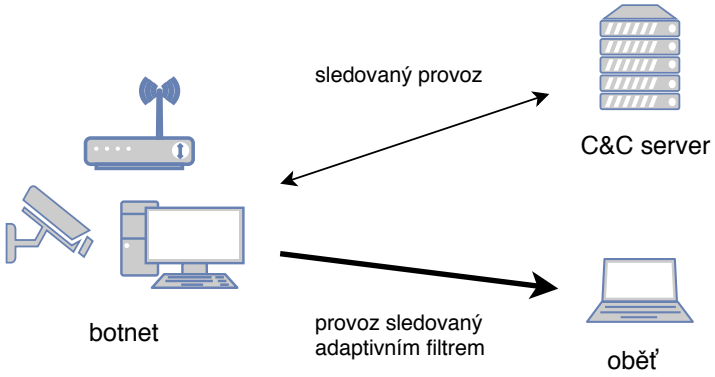
[3] The Onion Router - nástroje pro anonymní komunikaci na internetu

Existují ale případy, kdy zachycená komunikace klienta s blacklistovanou entitou nemusí znamenat hrozbu. Například pokud na IP adrese, která je na blacklistu C&C serverů, běží i legitimní služby. Útočník mohl napadnout legitimní stroj a provozuje na něm C&C server bez vědomí uživatele/administrátora. V případě, že zachytneme komunikaci klienta s legitimní službou na tomto serveru a incident nahlásíme, jedná se o falešný poplach (false-positive). Je samozřejmé, že cílem každého detektoru je minimalizace těchto falešných poplachů. Při zachycení podezřelé komunikace klienta s blacklistovanou entitou tedy nebudeme ihned nahlašovat incident, ale začneme sledovat komunikaci klienta. V této komunikaci se pak snažíme hledat dodatečné informace, které mohou potvrdit/vyvrátit přítomnost hrozby. Jednoduché heuristiky analýzy komunikace jsou popsány v Kapitole 5.

Dále uvádím několik scénářů, kdy sledování provozu podezřelého klienta může vést ke snížení hlášených false-positive alertů či ke zpřesnění nahlášeného alertu.

Detekce botnetu

Existuje několik veřejných blacklistů s C&C servery, např. *ZeusTracker* z projektu *abuse.ch*[4]. Pokud detekujeme komunikaci klienta s tímto serverem, začneme sledovat jeho aktivity. To provede mechanismus adaptivního filtru, který je popsán v Kapitole 2. Při následné analýze provozu můžeme například vidět vysoký objem dat směřující k jednomu cíli, což může znamenat, že se jedná o DDoS útok, jak ukazuje Obrázek 0.1. Pokud klient neprovádí konkrétně DDoS útok, ukazatelů, že se jedná o nakažený stroj, může být více (použité IRC porty, pravidelné HTTP požadavky apod.).

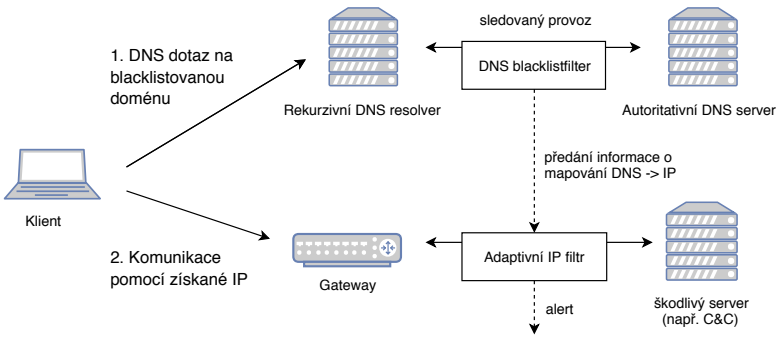


Obrázek 0.1: Schéma detekce nakažených botů

[4]https://zeustracker.abuse.ch

Detekce klientů DNS

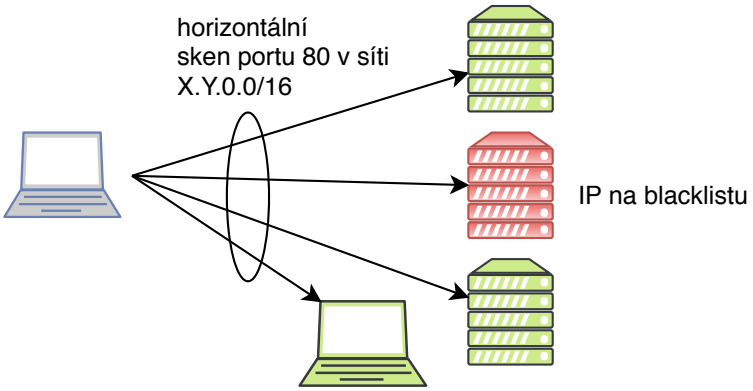
Detekce přístupu na blacklistované domény je problematická. Pokud totiž takový provoz detekujeme, jako zdroj (IP adresu) dotazu často vidíme rekurzivní DNS resolver a nikoliv samotného klienta. Sondy, které monitorují naši sledovanou síť, jsou totiž nasazeny na páteřních směrovačích (více o síti CESNET2 v Kapitole 1.4.3). Pokud tedy takovou komunikaci zachytíme, doptáme se na IP adresy blacklistované domény, a tyto adresy začneme sledovat adaptivním filtrem. Když poté detekujeme přístup na některou z těchto IP adres, je pravděpodobné, že zdrojem bude původní klient, který přistupoval na blacklistovanou doménu. Zohledňujeme časovou lokalitu těchto událostí, neboť v běžné komunikaci klient začne DNS dotazem a poté komunikuje s IP adresou, kterou dotazem získal. Situaci naznačuje Obrázek 0.2.



Obrázek 0.2: Detekce klientů DNS

Detekce skenování portů

Typickým příkladem false-positive je nahlášení klienta, který komunikoval s blacklistovanou entitou v důsledku skenování portů. Pokud například klient provádí horizontální sken rozsáhlé sítě, snadno přistoupí i na blacklistovanou IP adresu (Obrázek 0.3).



Obrázek 0.3: Horizontální sken sítě s blacklistovanou adresou

Kapitola 1

Analýza a rešerše

1.1 Botnety a C&C servery

Protože se v této práci zaměřuji zejména na detekci komunikace s Command&Control servery (C&C), je záhodno popsat, jak C&C servery a botnety fungují.

Podle [3] je bot zařízení nakažené škodlivým software (malware), které může provádět nějakou nekalou činnost. Síť takovýchto nakažených strojů se nazývá botnet. Botnet může mít v součtu obrovský výpočetní výkon a šířku pásma, pomocí něj může útočník například rozesílat spam, provádět DDoS útoky nebo krást uživatelská data formou keyloggeru[5] [4].

Například botnet zvaný Mirai zneužívá zejména špatně zabezpečené IP kamery. Autor škodlivého kódu se pochlubil infikováním až 380 tis. zařízení [5]. Kontroverzní projekt Insecam[6] poukazuje na špatně zabezpečené IP kamery tím, že zveřejňuje jejich stream na webu. Mnoho vlastníků IP kamer si totiž neuvědomuje, že ponechat výchozí jméno a heslo znamená bezpečnostní riziko.

Botnety jsou řízeny jedním nebo více C&C servery, které jsou pod kontrolou útočníka (botmaster). C&C servery rozdělují úkoly botům, distribuují jim další malware, nebo od botů sbírají ukradená data či hesla. Uživatelé většinou netuší, že jejich stroj je nakažený, což je v zájmu útočníka, aby mohl latentně zařízení zneužívat. Z tohoto důvodu se také botům přezdívá *zombies*. Podle [6] botnety původně sloužily i legitimním účelům přes Internet Relay Chat (IRC) protokol, ale právě jeho jednoduchost a nezabezpečenost umožnila útočníkům protokol lehce zneužít. Mezi nejčastější komunikační kanály mezi boty a C&C servery patří protokoly IRC, HTTP, POP3 a jiné. Existují ale i decentralizované (P2P) botnety, kde nefiguruje žádný centrální server.

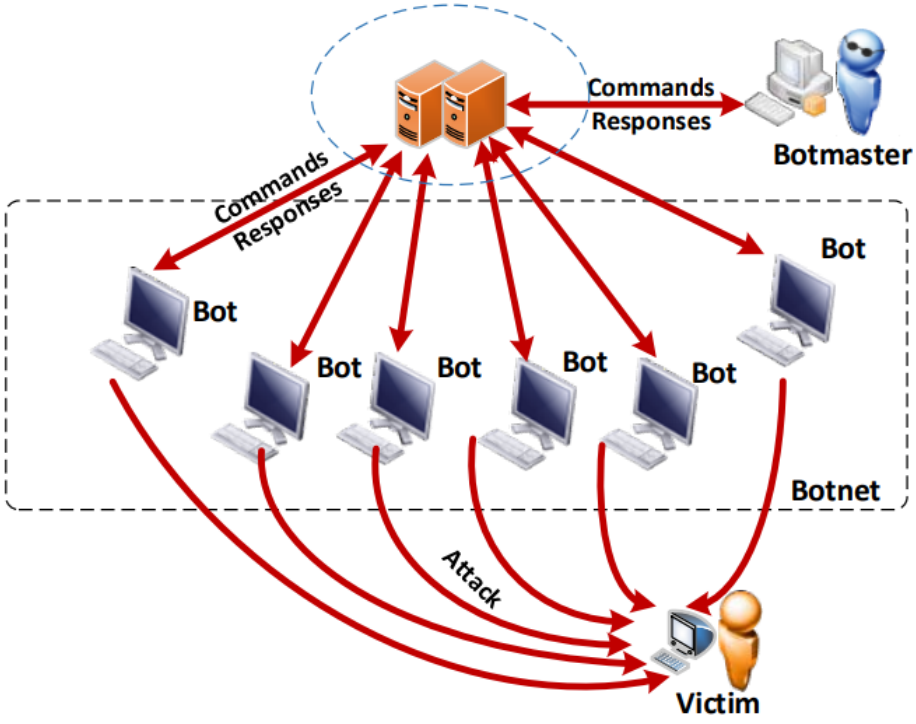
[5] software, který snímá stisky jednotlivých kláves
[6] http://www.insecam.org/

1.1.1 Architektury botnetu

Popis a schémata architektur botnetu vychází především z [7].

Centralizované botnety

První generace botnetů na internetu komunikovala metodou klient-server pomocí protokolu IRC. Botmaster skrze IRC kanál zadával příkazy botům pomocí *push*[7] mechanismu. Protože IRC protokol se dá snadno filtrovat pomocí firewallu nebo IDS, je častěji používán protokol HTTP. Síťový provoz HTTP botnetu bývá velmi podobný tomu legitimnímu. Na rozdíl od IRC, tyto botnety používají *pull* mechanismus, jednotliví boti se periodicky dotazují serveru na požadovanou akci. Nehledě na protokol, centralizované botnety jsou sice jednoduché na údržbu a provoz, jejich C&C server je však tzv. *single point of failure* (SPOF). Je-li server detekován a uveden mimo provoz, klienti se stávají neaktivními.

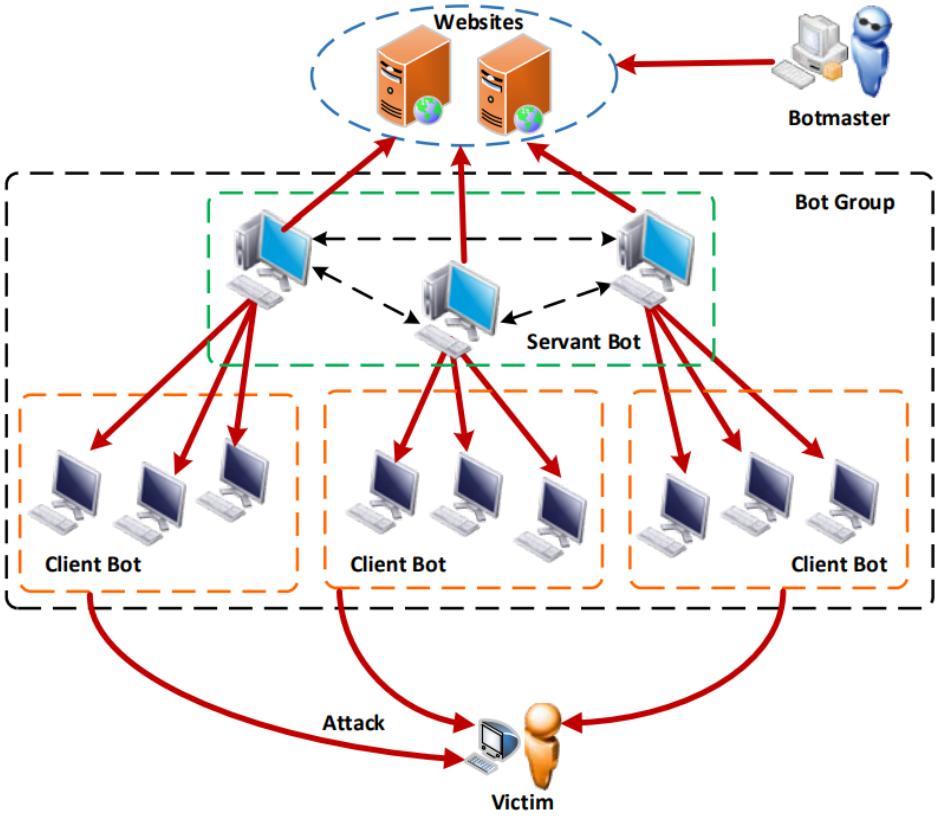


Obrázek 1.1: Centralizovaná architektura botnetu [7]

[7] metoda komunikace na internetu, kdy centrální server rozesílá zprávy přihlášeným klientům

Decentralizované botnety

U decentralizovaných (peer-to-peer) botnetů mizí problém se SPOF, každý bot se může chovat zároveň jako klient i C&C server. Distribuce příkazů probíhá tak, že botmaster zašle příkaz jednomu nebo několika botům, a ti ho mezi sebou dále šíří. Takový botnet je složitější na implementaci a provoz, zato je hůře detekovatelný a nedá se snadno zneškodnit.

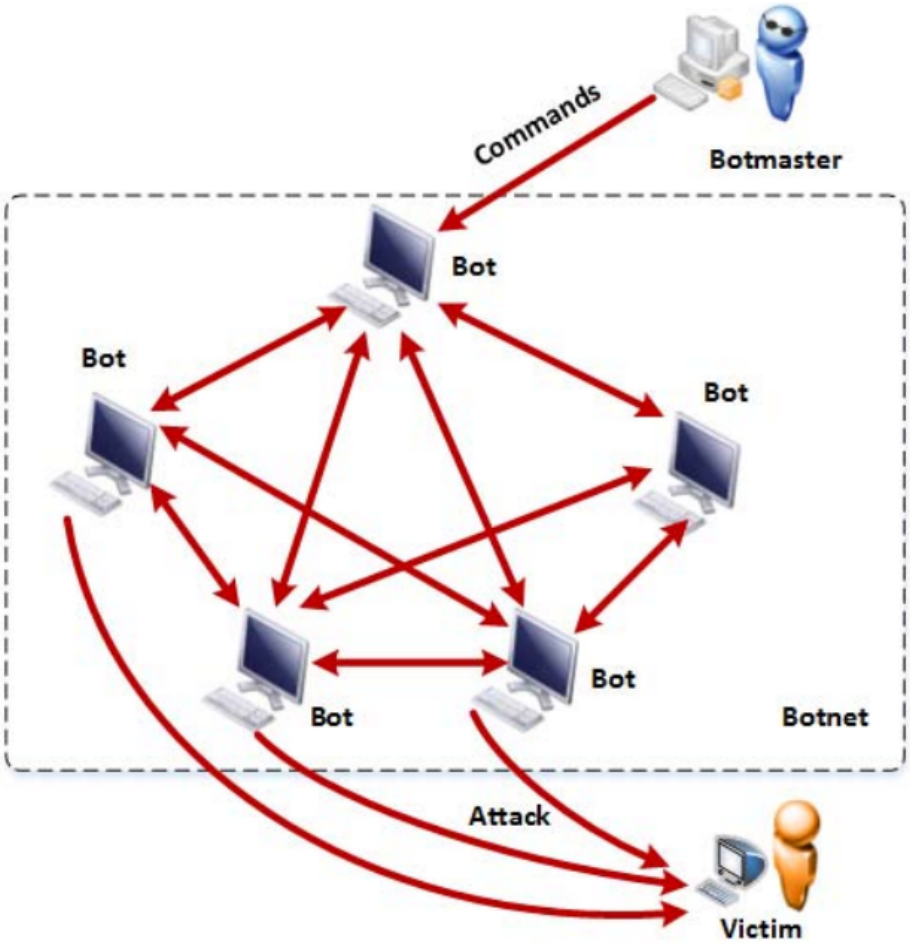


Obrázek 1.2: Decentralizovaná architektura botnetu [7]

Hybridní botnety

Hybridní model [8] kombinuje výhody modelů předchozích. Jeho detekce je velmi složitá. Architektura botnetu je rozdělena do tří skupin, botmaster, *servant bots* a *client bots*. Botmaster vystaví příkazy klientům na některou sociální síť, odkud ho servant boti stáhnou a rozdistribuují klient botům. Sa-

motní klient boti poté provádějí útoky. Servant boti jsou tedy jakási mezivrstva, která se chová současně jako klient a C&C server.



Obrázek 1.3: Hybridní architektura botnetu [7]

Centralizované botnety jsou jediné, které se dají detekovat pomocí blacklistů. Blacklistované jsou totiž pouze C&C servery. Architektury se dají také kombinovat, např. u botnetu Zeus, což je jeden z nejdéle existujících, si podle [9] bezpečnostní experti myslí, že decentralizovaná architektura slouží pouze jako záloha, pokud centralizovaná selže (C&C servery přestanou být dostupné).

1.1.2 Škodlivé aktivity botnetu

Níže jsou popsány některé scénáře škodlivých aktivit botnetu. Různorodost těchto útoků poukazuje na to, jak mocným nástrojem botnety jsou.

DDoS útoky

Distributed Denial-of-Service je útok, kdy rozsáhlá síť zařízení současně zahltí cílový stroj nebo síť excesivními požadavky. Proti tomuto útoku je obrana složitá, protože takový provoz se zdá být legitimní a nelze ho jednoduše blokovat. Na rozdíl od klasického (nedistribuovaného) DoSu, kterému lze snadno zabránit omezením vysokého počtu spojení z jedné IP adresy. Výsledkem útoku je nedostupnost služby, která může být kritická. Například v roce 2016 hackerská skupina New World Hacking takto odstavila z provozu server BBC a stránky D. Trumpa cca na 3 hodiny [10]. Typickými způsoby, jak tento útok provést, jsou např. TCP SYN flood nebo UDP flood[8].

Spam

Spam botnety jsou mocným nástrojem útočníků, skrze ně je možné provádět nejrůznější zlovolné kampaně. V dřívějších dobách stačilo útočníkovi nalézt nezapezpečený nebo otevřený SMTP[9] server a skrze něj rozesílat spam. V dnešní době existují sofistikované anti-spam nástroje, a otevřené SMTP servery jsou často blacklistované. Využití botnetu ke spamování je efektivní, neboť se těžko rozeznává legitimní a škodlivý provoz (stejně jako u DDoS). Nevyžádané reklamy jsou pouze špičkou ledovce, výrazně nebezpečnější je šíření malware pomocí phishingu. Útočník se snaží oklamat uživatele, aby si stáhl závadnou přílohu se škodlivým obsahem, což může být například ransomware. V minulém roce například dominoval botnet Necurs, který generoval až 97 % provozu všech spam botnetů [11], nejčastěji se takto šířil ransomware Globeimposter.

Odposlouchávání sítě

Boti mohou také pomocí programu pro sledování provozu sítě (packet sniffer) odposlouchávat data na nakažené stanici a odesílat je C&C serveru. Proto je nebezpečné posílat jména a hesla v otevřené podobě (plain text), např. vyplňováním formuláře na nešifrované webové stránce (HTTP), naopak je doporučeno všechen provoz šifrovat.

Keylogging

Pokud uživatel nakažené stanice šifruje všechna odchozí data, není to nic platné, pokud má nainstalovaný keylogger. Útočník takto může získat citlivá

[8] Zahlcení stroje, který poté není schopen navázat nové spojení
[9] Simple Mail Transfer Protocol

data, ať už to jsou přístupové údaje k bankovnictví, platební údaje kreditní karty nebo osobní údaje.

Nevyžádaná instalace keyloggeru se může šířit třeba pomocí spam kampaně, kdy uživatel spustí škodlivý VBScript[10], domnívaje se, že otevírá legitimní Word dokument. Poté je každý jeho stisk klávesy monitorován a dávkově odesílán ToR serveru, který kontroluje útočník [12].

Click fraud

Dalším příkladem využití botnetu je podvodné klikání (click fraud), většinou na reklamy na webu. Například když útočník někomu pronajme reklamní prostor na svém webu, tato třetí strana tam umístí své reklamy, a útočník na ně poté pomocí botnetu generuje klikání. Útočník z toho má příjem, neboť mu inzerent platí za každé kliknutí (pay per click). Může to být také nástroj poškozující konkurenci, kdy útočník generuje klikání na její reklamu, čímž jí způsobuje ztrátu a falešný pocit, že má její reklama dosah.

Příkladem může být payload[11] s názvem GhostVisitor botnetu Zeus, který po spuštění pomocí skrytého okna Internet Exploreru imituje chování uživatele podle jeho historie prohlížení. Navíc k tomu používá uživatelské cookies[12], čímž uživateli v podstatě krade identitu a klikání na reklamy je pak věrohodnější [13].

Těžba kryptoměn

Těžba kryptoměn je příklad zneužití výpočetního výkonu nakaženého stroje. Podle [14] v roce 2018 zažil tento fenomén obrovský boom oproti předchozím rokům. Je znám případ botnetu Smominru, kdy přes půl milionu zařízení vytěžilo bezmála 9000 kusů měny Monero, ta se s pomocí botnetu těží efektivněji než například Bitcoin [15]. Pro útočníka je to obecně jednoduchý způsob, jak si přivydělat, uživatel nakaženého stroje navíc většinou netuší, že mu na pozadí běží těžící program.

[10] Visual Basic script - skriptovací jazyk, používaný také pro makra v Microsoft Office

[11] Spustitelný soubor, který bot obdržel od C&C serveru

[12] v protokolu HTTP se tak označuje malé množství dat, která web server pošle prohlížeči, který je uloží na počítači uživatele

1.2 Veřejné blacklisty

Na internetu lze nalézt velké množství blacklistů různých typů. Jsou vytvářeny různými společnostmi, neziskovými organizacemi, jako je například Spamhaus[13]. Nebo jednotlivými nadšenci, kteří zkrátka chtějí učinit internet bezpečnějším, to je kupříkladu švýcarský projekt Abuse[14]. Generátor pravidel pro firewall zvaný FireHOL[15] zase na svých stránkách obsahuje více blacklistů z různých zdrojů, přidává k nim popis, kategorii apod.

Některé blacklisty, přestože mohou být užitečné, neobsahují žádné informace o tom, jak jsou vytvářeny, a tudíž potenciálně klesá jejich použitelnost. Často například tyto blacklisty uvádějí pouze to, že dané entity se chovaly škodlivě, ale žádnou konkrétní informaci neposkytují. Užitečnější jsou blacklisty, které jsou specifické pro danou hrozbu či malware. V následujícím textu si stručně rozebereme dostupné typy veřejných blacklistů.

1.2.1 Typy blacklistů

C&C servery

Jak už bylo řečeno v Sekci 1.1, blacklistování botnetů se týká takřka výhradně jejich C&C serverů. Existují nástroje, které aktivně skenují celý internet a hledají C&C servery. Jednoduše předstírají, že jsou nakaženými boty a pokud dostanou očekávanou odezvu, vědí, že se jedná o C&C server. Takovým nástrojem je například Malware Hunter[16] a je součástí projektu Shodan[17]. Další populární blacklisty pocházejí z webu Abuse. Například Feodo tracker blacklist sleduje C&C servery spojené s distribucí malwaru z rodiny Feodo (z Feodo vychází další vzorky malware např. Dridex nebo Heodo).

Tor exit nodes

„Tor je softwarový systém zajišťující anonymizaci uživatele při pohybu na internetu, k čemuž využívá model klient-server. Uživatel využívá klientskou část a jeho datový tok prochází nejprve sítí Tor složené ze serverových částí a teprve pak k cílovému počítači. Tím je možné skrýt informace o IP adrese uživatele a další faktory, které by ho mohly identifikovat. Uživatele Toru a jejich činnosti na internetu je obtížné vysledovat.“ [16]

Uživatelé sítě Tor často usilují o anonymitu na síti a nechtějí být kýmkoliv sledováni. To umožňuje například svobodu projevu (aktivistů, novinářů apod.) v nesvobodných zemích. Podle některých zdrojů však sítí Tor proudí mnoho škodlivého/nelegitimního provozu (podle serveru CloudFlare až 94 %

[13] https://www.spamhaus.org/
[14] https://abuse.ch/
[15] https://firehol.org/
[16] https://malware-hunter.shodan.io
[17] internetový vyhledávač zaměřený na různé typy zařízení a jejich služby/zranitelnosti

z celkového Tor provozu [17]). Administrátoři sítí mají dobrý důvod blokovat takový provoz, toho se dá docílit blacklistováním veřejně dostupného seznamu tzv. *Tor exit nodes*[18]. Tor exit node je takový uzel, který přeposílá data mezi sítí Tor a internetem. Pokud v síti existuje takový stroj, je možné (ba pravděpodobné), že skrze něj budou prováděny útoky. Sám provozovatel exit nodu poté může nést zodpovědnost za vyřizování stížností DMCA[19], hlášení zneužití (abuse) apod., přestože se sám na útocích nepodílí.

Phishing

„Phishing je podvodná technika používaná na internetu k získávání citlivých údajů (hesla, čísla kreditních karet apod.) v elektronické komunikaci. Principem phishingu je typicky rozesílání e-mailových zpráv nebo instant messaging, které často vyzývají adresáta k zadání osobních údajů na falešnou stránku, jejíž podoba je takřka identická s tou oficiální“. [18]

Existují dva známé blacklisty phishingových webových stránek, PhishTank[20] a OpenPhish[21]. Jedná se o komunitní stránky, kde uživatelé a kooperující bezpečnostní instituce nahlašují škodlivé URL adresy. Systémem hlasování je pak ověřováno, zda se opravdu jedná o phishing. Prohlížeč Google Chrome zase používá technologii Google Safe Browsing, kdy každá uživatelem navštívená URL adresa je zkontrolována na malware a phishing. Tato služba také poskytuje API[22] třetím stranám.

Samotné proaktivní skenování potenciálních phishing adres je podle [19] možné provádět různými technikami, hledání podobností pomocí strojového učení, používání vyhledávačů (phishingové stránky jsou méně indexovány než legitimní, protože jejich životnost bývá velmi krátká), apod.

Bogon adresy

Tzv. bogony jsou „falešné“ IP adresy ve veřejném internetu. Může se jednat o adresy, které nebyly alokované resp. přidělené autoritami IANA[23] resp. RIR[24]. Bogony také obsahují tzv. *martany* [20], to jsou třeba privátní adresy (např. rozsah 10.0.0.0/8), nebo testovací rozsah loopback (127.0.0.0/8), tyto adresy nemají co dělat ve veřejném internetu. Adresy IPv4 jsou již od roku 2011 plně vyčerpané (alokované), ne všechny jsou však aktuálně přidělené. Seznam takových adres se nazývá *fullbogon*. Blacklistování takových adres je jednou z metod tzv. *anti-spoofingu*. IP spoofing je technika skrytí identity, kdy

[18] https://check.torproject.org/exit-addresses
[19] Digital Millenium Copyright Act - ochrana autorských práv
[20] https://www.phishtank.com/
[21] https://openphish.com/
[22] Application Programming Interface
[23] Internet Assigned Numbers Authority
[24] Regional Internet Registry

útočník posílá IP pakety s falešnou zdrojovou adresou (toho se využívá např. v DDoS útocích).

Malware

Existují také blacklisty entit (jedná se většinou o domény), o kterých je známo, že distribuují nějaký malware. Například MalwareDomains[25] obsahuje domény s malwarem a spywarem[26]. Projekt malc0de[27] zase zobrazuje takové IP adresy a domény, na kterých byl za posledních 30 dní nalezen malware.

Kategorizační blacklisty

Web shallalist.de[28] nabízí seznamy domén podle jejich kategorie. Domény jsou zařazeny do více než 80 kategorií, jako například terorismus, zbraně, drogy, warez[29] apod. Může být v zájmu administrátora detekovat přístup na některý z těchto webů.

[25] http://www.malwaredomains.com/
[26] program, který bez vědomí uživatele odesílá data třetí straně
[27] http://malc0de.com/
[28] http://www.shallalist.de/
[29] nelegálně šířená autorská díla

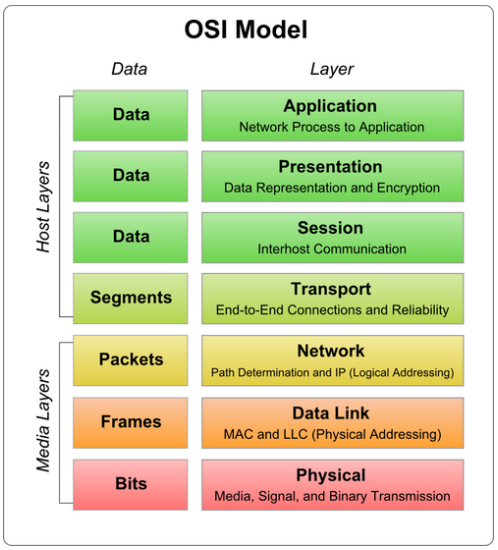
1.3 Metody analýzy síťového provozu

Součástí monitorování síťového provozu jsou *intrusion detection* resp. *intrusion prevention* systémy (IDS resp. IPS), které analyzují provoz a pomocí deterministických pravidel či statistických odchylek detekují potenciální problémy/útoky v síti. V případě IPS se těmto útokům systém navíc snaží aktivně zabránit. Tyto systémy pomáhají administrátorům mít síť pod kontrolou, starat se o její bezpečnost a reagovat na případné incidenty.

Vytvořená sada modulů používá pro analýzu podezřelého provozu datové toky. V této kapitole budou popsány výhody a nevýhody použití datových toků oproti paketové analýze.

1.3.1 Paketová analýza

Paketová analýza zpracovává každý IP paket zvlášť. Analýza zachovává kompletní informaci o provozu. Zpracování takto surových dat v reálném čase může být výpočetně náročné, zejména pak pokud detekční systém provozuje *deep packet inspection* (DPI), kdy zkoumá obsah jednotlivých vrstev ISO/OSI modelu (Obrázek 1.4) včetně aplikačních dat (L7). Takto se ovšem dá efektivně zabránit útokům na základě komunikačních vzorců právě v aplikační vrstvě. Například se takto dají detekovat útoky na webové aplikace pomocí inspekce HTTP protokolu. Monitorování se většinou provádí zrcadlením portů (port mirroring) na přepínači nebo směrovači dané sítě, kdy se všechen provoz přeposílá na monitorovací stanici.



Obrázek 1.4: ISO/OSI model [21]

1.3.2 Analýza datových toků

Monitorování rozsáhlých a páteřních sítí pomocí DPI často není možné kvůli výpočetní náročnosti. Pro velké sítě vznikla potřeba monitoringu pomocí metadat provozu, což přináší snazší pohled na globální dění v síti, ovšem za cenu ztráty informace o provozu (chybí např. některá aplikační data). Poskytovatelé internetu (ISP) využívají tento typ monitorování pro účtování ceny služeb v závislosti na množství přenesených dat, odhalování úzkých míst v síti, efektivnější plánování budoucího rozvoje sítí apod. [22].

Pojmu IP flow chybí exaktní definice, často se jedná spíše o abstraktní popis jako např. v RFC 2722 [23], kde je flow popsáno jako „Uměle vytvořený logický ekvivalent hovoru nebo spojení“. Jsou to ale typicky pakety, které mají společné alespoň následující atributy:

- zdrojová IP adresa
- cílová IP adresa
- zdrojový port
- cílový port
- L4 (transportní) protokol

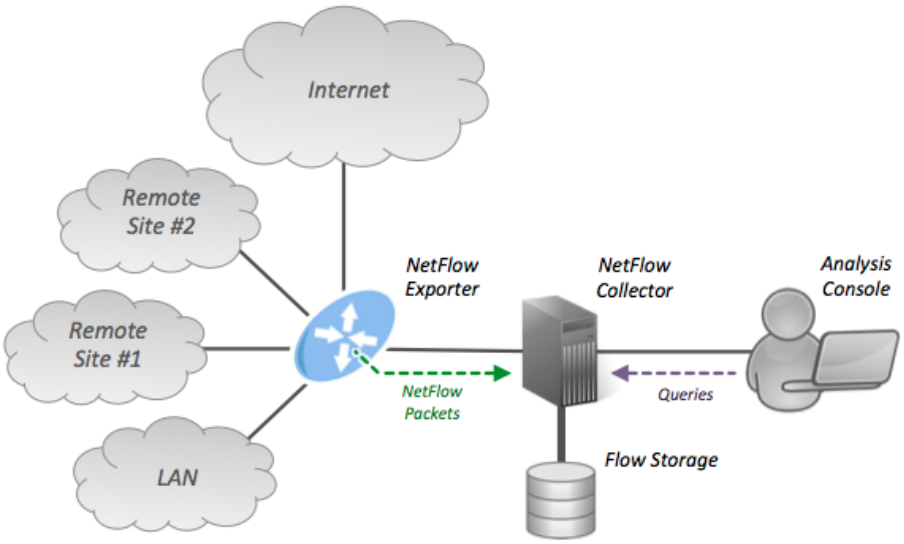
Pakety se stejnými atributy se agregují do jednoho (jednosměrného) flow záznamu (datový tok). Pro každý tok je zaznamenávána doba jeho vzniku, délka jeho trvání, počet přenesených paketů, bajtů a další údaje. Existují různé exportní protokoly pro vytváření flows, zejména pak NetFlow a IPFIX.

„NetFlow je otevřený protokol vyvinutý společností Cisco Systems, určený původně jako doplňková služba k Cisco směrovačům.“ [24]. Protokol prošel značným vývojem, nejčastěji se používají verze 5 a 9. Verze 5 zpracovává pouze IPv4 pakety a má fixní počet polí, verze 9 je flexibilnější, neboť využívá šablony pro tvorbu exportovaných dat. Z NetFlow verze 9 vychází protokol IPFIX (IP Flow Information Export), který je standardizován v RFC 7011 [25]. IPFIX je velmi univerzální a snadno rozšiřitelný, co se týče možnosti exportu dat. Lze nadefinovat export v podstatě libovolných dat z vrstev L2-L7, navíc IPFIX oproti NetFlow podporuje export dat proměnné délky (např. URL v HTTP). Flow exportéry s tzv. *application awareness* tak v podstatě kombinují DPI s tradiční flow analýzou.

Typická architektura monitorování provozu pomocí flows je zobrazena na obrázku níže. Figurují zde tři hlavní komponenty [26]:

- Flow exportér (monitorující stanice): Agreguje pakety do flows a exportuje je jednomu nebo více kolektorům (tomuto procesu se také říká *metering*. Nasazen je typicky na směrovači či přepínači

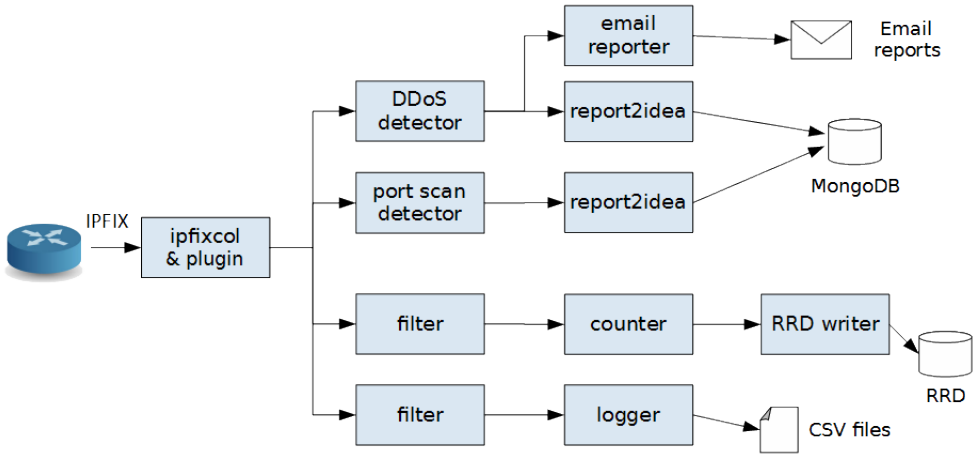
- Flow kolektor: Přijímá, zpracovává a ukládá flow data přijaté od exportérů
- Aplikace pro analýzu: Analyzuje a vyhodnocuje flow data (detekce útoků, počítání statistik apod.)



Obrázek 1.5: Architektura monitorování provozu pomocí flows [24]

1.4 Systém NEMEA

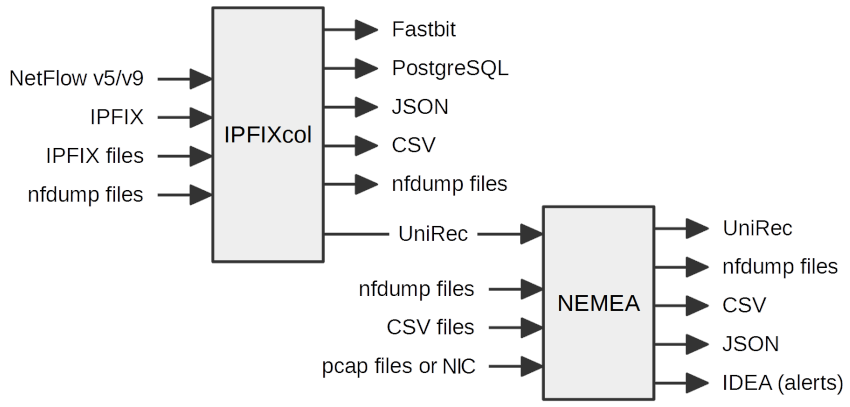
NEMEA (Network Measurements Analysis) je open-source systém pro monitorování sítě a detekci anomálií v reálném čase. Je to modulární systém založený na proudovém zpracování dat (není třeba data perzistentně ukládat). Systém sestává z nezávislých modulů, které jsou propojeny speciálním rozhraním a data si vyměňují pomocí jednotného formátu. Každý modul je samostatný proces. Moduly mají různou funkcionalitu, detekují podezřelé chování v síti, agregují nahlášené alerty, počítají statistiky o provozu apod. Celý systém pracuje nad flow daty. Díky flexibilitě vstupních/výstupních rozhraní může zpracovávat také zachycený provoz (offline). Obrázek 1.6 ukazuje možné propojení NEMEA modulů.



Obrázek 1.6: Ukázka propojení NEMEA modulů [27]

Obrázek 1.7 demonstruje, s jakými druhy dat dokáže systém NEMEA pracovat. IPFIXcol[30], což je implementace kolektoru IPFIX od spol. CESNET, přijímá flow formáty NetFlow v5/v9 a IPFIX. Data mohou být živá i offline. IPFIXcol pak exportuje data systému NEMEA pomocí formátu UniRec (kapitola 1.4.1). Tento formát podporuje i některé aplikační protokoly (HTTP, DNS a další), což je nezbytné pro vytvořené detekční moduly, které detekují podezřelou komunikaci podle URL a DNS záznamů.

[30] https://github.com/CESNET/ipfixcol



Obrázek 1.7: Podporované datové formáty systému NEMEA [27]

1.4.1 Formát UniRec

UniRec (*unified record*) představuje vlastní implementaci a reprezentaci binárních dat pro efektivní komunikaci modulů v systému NEMEA. Vychází z návrhu exportních protokolů flow (Netflow v9 a IPFIX). UniRec používá šablony a zprávy. Šablona udává definici políček, které jsou obsaženy v každé zprávě. Políčko je definováno jménem a typem, vývojáři si mohou definovat svá vlastní políčka (ty mohou mít i proměnnou délku). V Tabulce 1.1 lze vidět podporované datové typy formátu UniRec.

Název	Velikost (B)	Popis
int{8,16,32,64}	1, 2, 4, 8	celá čísla
uint{8,16,32,64}	1, 2, 4, 8	kladná čísla
char	1	ASCII znak
float	4	číslo s plovoucí řádovou čárkou
double	8	číslo s plovoucí řádovou čárkou (dvojitá přesnost)
ipaddr	16	IPv4/IPv6 adresa
macaddr	6	MAC adresa
time	8	časová značka
string	proměnlivá délka	řetězec tisknutelných znaků
bytes	proměnlivá délka	pole bytů

Tabulka 1.1: Datové typy UniRec

Příklad šablony, která obsahuje informace o HTTP spojení, je uveden níže. Hlavičky `HTTP_URL` a `HTTP_USER_AGENT` mají variabilní délku, ostatní hlavičky jsou fixní.

```
ipaddr SRC_IP, ipaddr DST_IP, uint16 SRC_PORT,
uint16 DST_PORT, uint8 PROTOCOL, uint8 TCP_FLAGS,
uint32 PACKETS, uint32 BYTES, uint16 HTTP_RSP_CODE,
string HTTP_URL, string HTTP_USER_AGENT
```

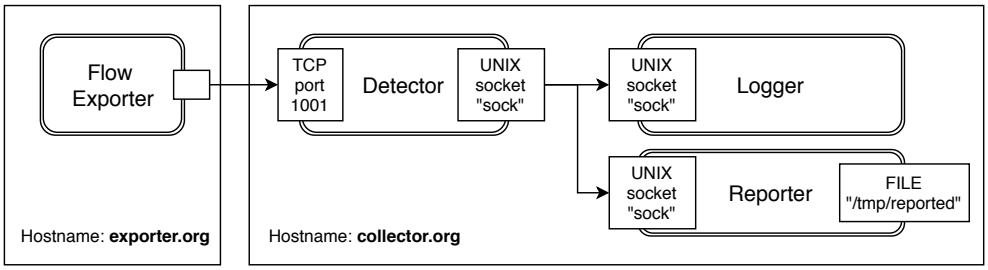
1.4.2 Knihovna Libtrap

Libtrap je knihovna implementující sadu funkcí pro výměnu dat mezi moduly. Knihovna pracuje s různými typy komunikačních rozhraní (IFC[31]). Při vývoji modulu si programátor stanoví, kolik vstupních a výstupních IFC použije. Uživatel modulu poté volí libovolné typy IFC pro odesílání a příjem dat (jako parametry při spuštění). Výměna dat je možná ve formátu UniRec, ale i ve formátu JSON (přesněji se jedná o binární serializovaný formát, v systému NEMEA existují interní moduly pro de/kódování do JSON). Typy komunikačních rozhraní (IFC) jsou následující:

- TCP – komunikace přes TCP soket
- TLS – komunikace přes TCP soket se šifrováním TLS (nutno zadat certifikát v `.pem` formátu)
- UNIX socket – komunikace přes unixový soket. Název soketu může být jakýkoliv řetězec použitelný pro soubor
- Blackhole (pouze výstup) – výstup je zahozen (obdoba přesměrování do souboru `/dev/null`)
- File – čtení/zápis do souboru, používá se přípona `.trapcap`

[31] https://nemea.liberouter.org/trap-ifcspec/

Propojení modulů může vypadat například takto:



Obrázek 1.8: Ukázka použití IFC rozhraní v NEMEA modulech

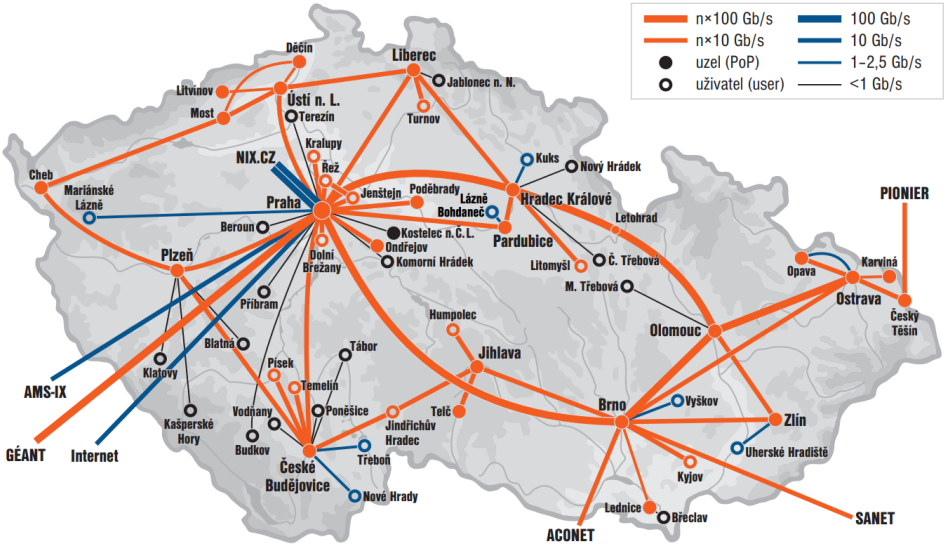
Flow exporter posílá data přes TCP soket na port 1001 modulu Detector. Všechny moduly kromě Flow exporteru běží na stroji s hostname collector.org, tudíž mohou komunikovat pomocí UNIX soketu. Logger a Reporter dostanou od Reporteru vždy stejná data. Výstup z Reporteru je ukládán do souboru. Spuštění modulů by vypadalo následovně:

```
:~$ flow_exporter -i t:collector.org:1001
:~$ detector -i t:exporter.org:1001,u:sock
:~$ logger -i u:sock
:~$ reporter -i u:sock,f:/tmp/reported
```

1.4.3 Síť CESNET2

Systém NEMEA je vyvíjen ve spolupráci se sdružením CESNET (český NREN[32] operátor) a českými vysokými školami, a slouží k analýze provozu v síti CESNET2. CESNET2 (Obrázek 1.9) je „vysokorychlostní počítačová síť, jejíž páteř propojuje okruhy s vysokými přenosovými rychlostmi největší univerzitní města České republiky a další oblasti. Klíčové je napojení na evropskou síť GÉANT, jejíž uzel se nachází přímo v prostorách sdružení CESNET“ [28]

[32]National Research and Education Network



Obrázek 1.9: Topologie sítě CESNET2 [28]

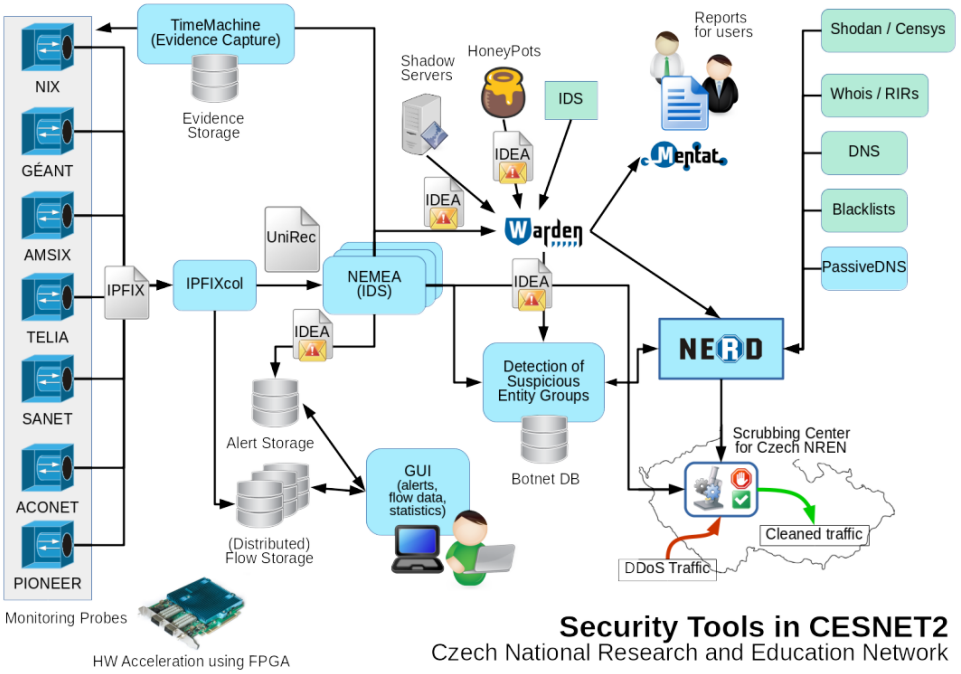
Systém NEMEA spolupracuje s dalšími bezpečnostními systémy v rámci sítě CESNET2. NERD[33] je systém pro evidenci reputací entit. Warden [34] je zase systém pro sdílení informací o detekovaných bezpečnostních událostech. Zdrojem dat jsou potom pro NEMEA systém výstupy monitorujících sond, které používají specializované hardwarově akcelerované COMBOv2 karty[35], pomocí nichž generují pokročilá flow data. Tyto sondy jsou nasazeny na páteřních směrovačích peeringových sítí jako je NIX, GÉANT a další. Schéma spolupráce bezpečnostních systémů je vidět na Obrázku 1.10.

1.4.4 Související systémy

NERD

Network Entity Reputation Database (NERD) je systém pro evidenci reputace síťových entit. Autor projekt popisuje jako „databáze škodlivých entit na internetu a vše co o nich víme“ [30]. V databázi jsou ukládány užitečné informace k IP adresám, které byly nahlášeny jako škodlivé/útočné. Například se jedná o název hostitele (hostname), informace z *whois* databáze, geolokace,

[33] https://nerd.cesnet.cz
[34] https://warden.cesnet.cz
[35] https://www.liberouter.org/technologies/cards/



Obrázek 1.10: Bezpečnostní systémy v síti CESNET2 [29]

přítomnost na blacklistech apod. Hlavním zdrojem dat pro NERD je systém Warden, NERD je ale stále ve vývoji a další zdroje jsou přidávány, například MISP[36] nebo informace z vyhledávače Shodan. Hlavním výstupem pro každou entitu je tzv. *reputační skóre*. Hodnota udává, jak velkou hrozbu entita představuje, nebo také jaká je pravděpodobnost, že entita bude v budoucnu znovu útočit. V současné době je vyvíjen model strojového učení pro výpočet tohoto skóre. NERD poskytuje přístup k datům třetím stranám pomocí REST API.

Warden

Warden je „systém pro sdílení informací o detekovaných bezpečnostních událostech, umožňuje jednoduše a efektivně týmu CERTS/CSIRT a dalším zapojeným bezpečnostním týmům rychlé sdílení a využití informací o detekovaných anomáliích, které byly zjištěny nasazenými nástroji v jimi monitorovaných sítích. Správci sítí do něj mohou na dobrovolné bázi zasílat data o detekovaných hrozbách (které se primárně nemusí týkat jejich sítě) a naopak si stahují data, která je zajímají a jsou užitečná pro zajištění zdraví a bezpečnosti sítě pod jejich správou.“ [31]

[36]Malware Information Sharing Platform

V současné době je Warden vyvíjen a provozován pro síť CESNET2, v budoucnu je však plánován rozvoj systému jako otevřeného projektu. Využití systému může být obrana vlastní sítě před vnějšími útočníky, kdy data ze systému mohou být zdrojem pro externí nástroje jako je například *fail2ban*[37]. Díky tomu je možné preventivně zabránit pokusům o prolomení zařízení v síti. Další využití je odhalování napadených strojů z vlastní sítě, kdy lze například využít data z honeypotů ostatních zapojených organizací. V nich lze hledat IP adresy strojů z vlastní sítě a vůči nim poté podniknout patřičná opatření.

[37] nástroj, který skenuje systémové logy a odepírá přístup podezřelým IP adresám

Kapitola 2

Návrh

2.1 Sada modulů blacklistfilter

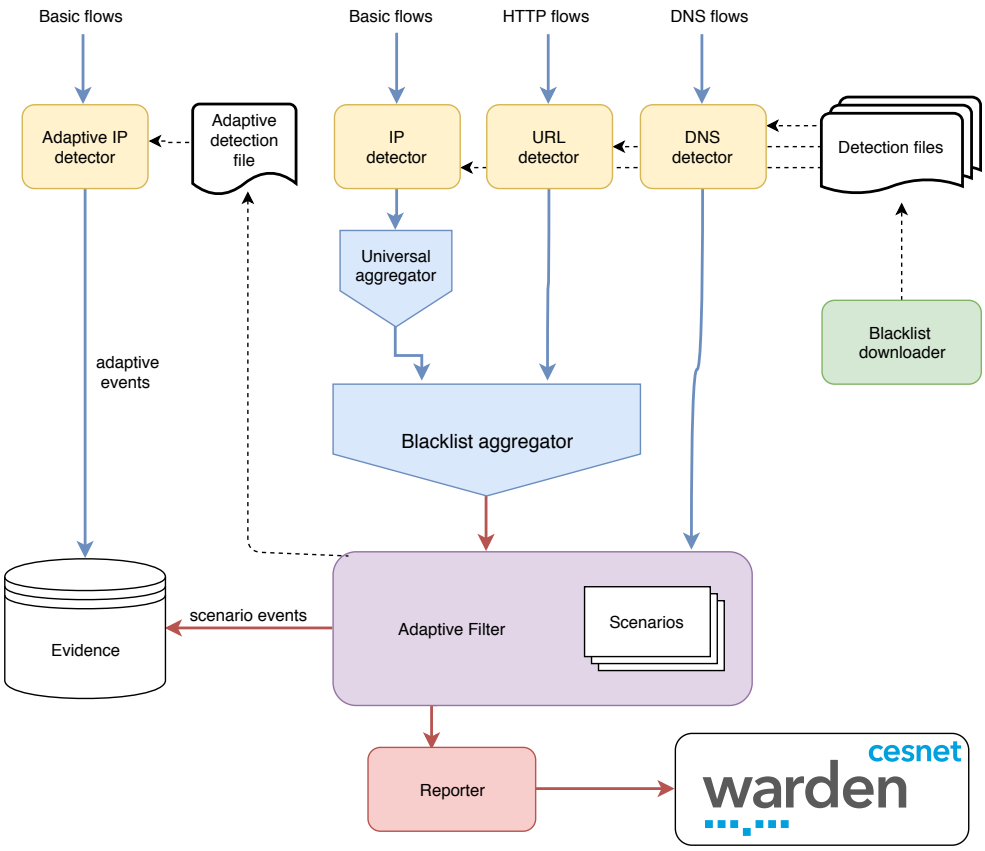
Sada modulů s názvem **blacklistfilter** byla navrhována s ohledem na modularitu systému. Bylo tedy cílem, aby se každý dílčí modul zaměřoval na jednu aktivitu (stahování blacklistů, detekce, agregace, apod.). Návrh a implementace prošla několika iteracemi.

Nejdříve byla snaha zprovoznit původní implementaci detektorů, na kterou jsem navázal. Trpěla však přílišnou komplexností, samotný detektor totiž současně stahoval blacklisty, detekoval a agregoval data. Cílem další iterace bylo celou sadu modulů psát v jazyce Python, což se zdálo implementačně jednodušší a flexibilnější řešení. Nakonec se toto řešení projevilo jako nedostatečné z hlediska výkonu. Finální návrh tedy jednak klade důraz na již zmíněnou modularitu, a také dbá na dostatečnou výkonnost/propustnost všech dílčích modulů. Původní detektor byl přepracován tak, aby prováděl pouze samotnou detekci. Na agregaci dat a stahování blacklistů byly vytvořeny samostatné moduly. Požadavky na výkon jsou kladeny zejména na detektory, které by měly zpracovávat alespoň 300 tisíc flows/s, proto jsou psány v C/C++. Všechny ostatní moduly (Agregátor, Blacklist downloader, ..) jsou psány v Pythonu, zde je důležitější spíše čitelnost a rozšiřitelnost kódu.

Na obrázku 2.1 můžeme vidět architekturu celé sady modulů blacklistfilter. Moduly můžeme rozdělit do několika skupin (na obrázku barevně rozlišené):

- Downloader: stará se o stahování blacklistů a jejich distribuci detektorům
- Detektory: mají za úkol přijímat data (flows) a detekovat, jestli je nějaká entita (IP adresa, URL, doména) na blacklistu
- Agregátory: agregují detekovaná data podle různých klíčů (záleží na konkrétním typu dat)

- Adaptivní filtr: sleduje příchozí data a podle definovaných scénářů vytváří (interní) detekční soubor, díky kterému se poté detekuje provoz klientů v síti. Adaptivní filtr rozhoduje, které incidenty budou ihned nahlášeny (zaslány Reportéru) a které se pouze uloží pro pozdější analýzu (do tzv. Evidence)
- Reportér: Zasílá alerty do systému Warden



Obrázek 2.1: Návrh sady modulů blacklistfilter

Plné čáry značí výměnu, transformaci nebo uložení zpráv (flow dat). Formáty dat jsou buď UniRec (modré čáry), nebo JSON (červené čáry). Přerušované čáry zase značí specifické akce (vytváření a čtení tzv. *detekčních souborů*). Uvedené moduly byly vyvinuty v rámci této práce s výjimkou modulu zvaného *Univerzální agregátor*, který byl vytvořen v rámci jiné závěrečné práce [32].

Důležitá myšlenka návrhu je, že všechny detekované události prochází Adaptivním filtrem. Ten rozhoduje podle předem definovaných scénářů, jakou akci pro danou událost provede. Flexibilní implementace umožňuje snadnou rozšiřitelnost o tyto scénáře. Adaptivní filtr může na základě scénářů provádět tyto akce:

1. Ihned nahlásí danou událost (zaslání Reportéru).
2. Událost nenahlásí, pouze pošle data do evidence na důkladnější analýzu (bez sledování dalšího provozu podezřelých klientů).
3. Událost nenahlásí a začne sledovat související provoz podezřelých klientů (pomocí Adaptivního IP detektoru). Tento související provoz i provoz původní potom ukládá do Evidence na analýzu.

Právě 3. bod je důležitý use-case pro vylepšení detekce komunikace botnetů (a dalších scénářů). Do Evidence se v tomto případě ukládají dva typy událostí: `scenario events` a `adaptive events`. `scenario events` jsou ty události, kde figuruje C&C server a seznam klientů, kteří s ním komunikovali. Adaptivní události jsou zachycené flows všech těchto klientů. Adaptivní filtr tedy vytvořil interní blacklist s těmito podezřelými klienty. Každá událost má jednoznačný identifikátor, díky kterému se tyto události při analýze spárují. Adaptivní události nejsou agregovány úmyslně, aby nebyla ztracena žádná informace o provozu klientů.

Kapitola 3

Implementace a konfigurace modulů

3.1 Blacklist downloader

Základem sady modulů blacklistfilter je stahování veřejných blacklistů, to provádí modul zvaný Downloader. Modul periodicky stahuje blacklisty, extrahuje z nich požadované záznamy (IP adresy, URL adresy, domény) a vytváří soubory pro jednotlivé detektory (detekční soubory). Pro stahování je použita standardní Python knihovna `requests`. Extrakce záznamů je prováděna zejména regulárními výrazy. Většina blacklistů na webu totiž poskytuje klasický textový formát s jednou entitou na řádce, který je jednoduše zpracovatelný. Některé blacklisty jsou ve formátu CSV, k těmto jsou poté v konfiguračním souboru dány sloupce, které se mají extrahovat. Modul také detailně loguje pomocí standardní knihovny `logging`.

Výstupem modulu jsou tyto soubory:

- *ip4.blist* — pro IP detektor (obsahuje IPv4 záznamy)
- *ip6.blist* — pro IP detektor (obsahuje IPv6 záznamy)
- *url.blist* — pro URL detektor (obsahuje URL i FQDN[38] záznamy)
- *dns.blist* — pro DNS detektor (obsahuje pouze FQDN záznamy)

IP detektor má tedy na vstupu dva soubory, ostatní detektory po jednom. V těchto detekčních souborech jsou záznamy ze všech blacklistů daného typu (IP, URL nebo DNS). Každý záznam v detekčním souboru má následující formát:

[38] Fully Qualified Domain Name

`<entita><separátor><blacklist_index>`

Níže uvádím ukázky detekčních souborů, na kterých jsou demonstrovány následující důležité vlastnosti:

- Entity jsou vždy seřazené (různý komparátor dle typu entit)
- `blacklist_index` je 64b bitmapa, kde jedničkový bit na pozici $2^{(x-1)}$ udává přítomnost entity na blacklistu s ID rovno `x` (entita může být maximálně na 64 blacklistech)
- IP blacklisty (v4 i v6) mohou obsahovat i adresní rozsahy
- IPv6 adresy jsou v nezkrácené podobě
- URL a DNS entity neobsahují `http(s)://` schéma ani `www` prefix
- Separátory jsou různé, podle typu detekčního souboru
- Internationalized Domain Names (IDN) jsou kódovány v Punnycode[39] (například háčkyčárky.cz ~ xn–hkyrky-ptac70bc.cz)

Ukázka `ip4.blist`:

```
...
5.187.5.171,5
5.187.10.1,256
5.188.10.0/23,8
104.31.85.1,64
216.176.184.21,1
...
```

Ukázka `ip6.blist`:

```
...
2602:ffa0:0000:0000:0000:0000:0000:0000/36,256
2605:5200:0000:0000:0000:0000:0000:0000/32,256
2607:d100:0000:0000:0000:0000:0000:0000/32,256
...
```

[39] Kódování z Unicode do ASCII

Ukázka `url.blist`:

```
...
52uo5k3t73ypjije.catfills.mobi\128
amvic.ru\3
vegantravelshow.com/images/secure/gate.php\64
yunali.gtacomputer.com/viewerf.html\2
...
```

Ukázka `dns.blist`:

```
...
account-chek-police.000webhostapp.com\1
agonyproducts.com\32
wauwadbubdubzxcvuxzvcuvxz.blogspot.com\2
xn--hkyrky-ptac70bc.cz\1
...
```

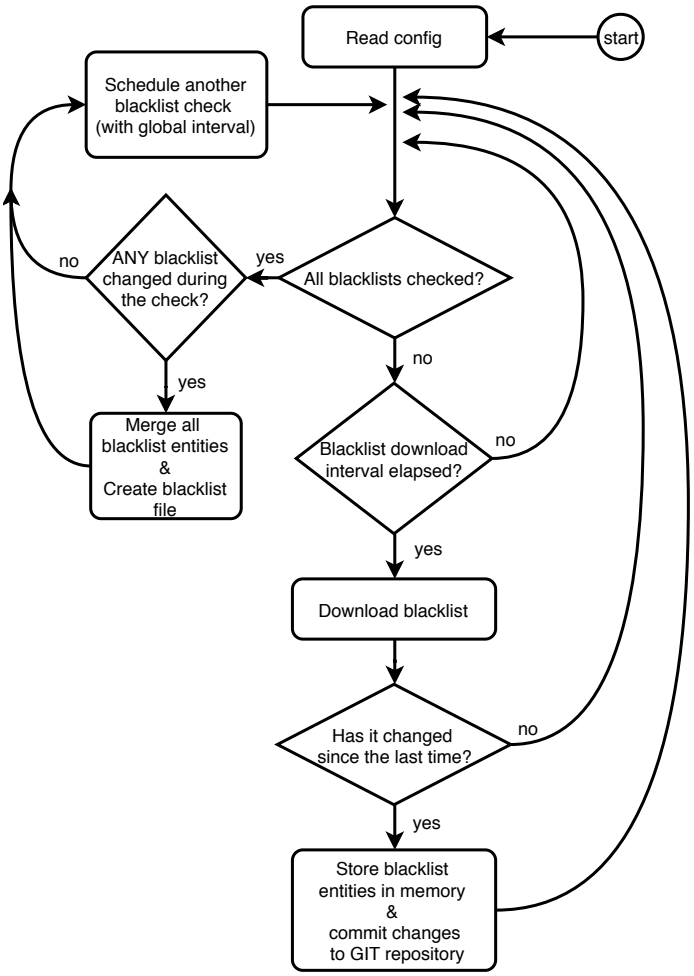
Konfigurační soubor Downloaderu je ve formátu XML. Obsahuje obecnou konfiguraci a seznam použitých blacklistů. Obecná konfigurace obsahuje cesty k detekčním souborů a globální `download_interval`. Ten udává, jak často se kontrolují všechny blacklisty (ve výchozím nastavení každých 10 minut). V konfiguračním souboru jsou pouze 2 typy blacklistů: `IP` a `URL/DNS` s následujícími atributy:

- `id` — ID blacklistu v rozmezí <1, 63> (tzn. maximální počet blacklistů pro daný typ je 64)
- `category` — kategorie blacklistu, formát odpovídá kategoriím formátu IDEA (například Intrusion.Botnet, více v Sekci 3.5)
- `method` — metoda obdržení blacklistu (zatím jediná hodnota `web`)
- `name` — název blacklistu (např. Zeus Tracker)
- `source` — zdrojová URL blacklistu
- `download_interval` — interval pro stažení tohoto blacklistu. Servery, které poskytují blacklisty, někdy dávají omezení, jak často smí klient blacklist stahovat, jinak hrozí klientovi *ban*.
- `ip_version` (pouze pro typ IP) — udává, jestli blacklist obsahuje IPv4 nebo IPv6 entity

- `detectors` (pouze pro typ `URL/DNS`) — udává, pro které detektory z množiny `{URL, DNS}` bude tento blacklist použit

Downloader také pomocí nástroje GIT verzuje všechny blacklisty, změny jsou tedy zaznamenány a je možné dohledat, kdy některá z entit byla odebrána/přidána na blacklist. To může být užitečná informace ve finálním reportu.

Obrázek 3.1 ukazuje celkovou funkcionalitu Downloaderu ve formě vývojového diagramu.



Obrázek 3.1: Vývojový diagram modulu Downloader

3.2 Detektory

Všechny detektory jsou psány v C/C++ a provádí *pattern matching*. Na vstupu mají data (flows) ve formátu UniRec, výstupem jsou vždy stejná data obohacená o informaci, na kterých blacklistech se entita nachází. Pokud entita není na blacklistu, detektor logicky na výstup nic neposílá.

Detektory obsahují hlavní smyčku, ve které čekají na vstupní data v blokujícím režimu. Při každém zpracování dat se zároveň kontroluje příznak, je-li potřeba znovu načíst detekční soubor. Tento příznak nastavuje dedikované vlákno (*blacklist watcher*), které sleduje změny v detekčním souboru, toho je dosaženo pomocí systémové knihovny *inotify*. Kód blacklist watcheru je staticky linkován ke každému detektoru. Vlákno používá systémové volání `inotify_add_watch` a sleduje příznak `IN_CLOSE_WRITE` pro svůj detekční soubor. Tento příznak je definován v dokumentaci[40] jako „File opened for writing was closed.“. Downloader, který při vytváření nového seznamu přepisuje detekční soubor, tento příznak nastaví. Takto je tedy řešena distribuce blacklistů detektorům. Nyní si rozebereme jednotlivé detektory a jejich implementaci pattern matchingu.

3.2.1 IP detektor

Vstupní UniRec šablona

```
SRC_IP,DST_IP,SRC_PORT,DST_PORT,PROTOCOL,
PACKETS,BYTES,TIME_FIRST,TIME_LAST
```

Výstupní UniRec šablona

```
SRC_IP,DST_IP,SRC_PORT,DST_PORT,PROTOCOL,
PACKETS,BYTES,TIME_FIRST,TIME_LAST,SRC_BLACKLIST,DST_BLACKLIST
```

`SRC_BLACKLIST` a `DST_BLACKLIST` jsou 64b čísla (bitmapy) a udávají, jestli zdrojová resp. cílová IP adresa je na některých blacklistech, přičemž druhá hodnota je vždy 0 (případ obou adres na blacklistu zanedbáváme). Principiálně totiž detekujeme podezřelé klienty uvnitř naší sítě s blacklistovanými entitami mimo naši síť. Pokud je některý stroj uvnitř naší sítě nalezen na veřejném blacklistu, detekuje to jiná bezpečnostní služba (tzv. IntelMQ feeds[41]). IP detektor ignoruje flows se zdrojovým resp. cílovým portem 53 (DNS dotazy),

[40] http://man7.org/linux/man-pages/man7/inotify.7.html
[41] https://github.com/certtools/intelmq

protože zdrojovou resp. cílovou IP adresou je vždy DNS resolver a nikoliv samotný klient (původce dotazu).

Implementace pattern matchingu

Vyhledávání IP adresy v seznamu adres a rozsahů je známý problém, který řeší např. routery ve svých směrovacích tabulkách. Ty pro to používají např. algoritmus Longest-Prefix Match (LPM). Navázal jsem však na původní implementaci existujícího modulu, kde byla použita metoda půlení intervalů, toto řešení se ukázalo jako dostačující.

Jak bylo již vysvětleno, detektor si načítá detekční soubor vždy, když vlákno *blacklist watcher* nastaví příznak (`RELOAD_BLACKLISTS`). Protože je tento soubor předzpracovaný a seřazený Downloaderem, může ho detektor pouze načíst (po řádcích) do svých struktur (`std::vector`). Při zpracovávání příchozích flows se potom provádí klasický `binary_search`, kdy je provedeno maskování hledané IP adresy s maskou aktuálně vybrané IP adresy/rozsahem ze seznamu (pro konkrétní IP adresu je maska `/32`). Pokud je (vymaskovaná) hledaná adresa shodná s aktuálně vybranou, adresa je na blacklistu, jinak pokračuje půlení intervalů. Výhoda tohoto řešení je, že vnitřně nerozlišuje mezi konkrétní IP adresou a rozsahem. Asymptotická složitost algoritmu je $O(log(n))$, kde n je počet záznamů v detekčním souboru.

3.2.2 URL a DNS detektor

URL a DNS detektor jsou si velice podobné, rozlišují je jen drobné rozdíly v implementaci. Vstupní a výstupní UniRec šablony obsahují stejná políčka jako IP detektor obohacená o políčka z aplikačních protokolů HTTP a DNS. Na výstupu mají oba detektory stejnou šablonu jako na vstupu, obohacenou o políčko `BLACKLIST` (opět 64b bitmapa indikující přítomnost entity na blacklistech). Narozdíl od IP detektoru je zde pouze jedna hodnota, detekované jsou pouze požadavky (request) klienta.

Vstupní UniRec šablona

`URL`

```
<IP fields>,
HTTP_REQUEST_HOST,HTTP_REQUEST_REFERER,HTTP_REQUEST_URL
```

`DNS`

```
<IP fields>,
DNS_ID,DNS_ANSWERS,DNS_NAME,DNS_QTYPE,DNS_RLENGTH,
DNS_RCODE,DNS_RDATA
```

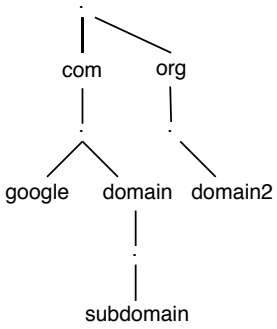
Výstupní UniRec šablona

```
<IP fields><URL/DNS fields>,BLACKLIST
```

Implementace pattern matchingu

Pro ukládání a vyhledávání doménových jmen a URL adres jsem použil prefixový strom, který je součástí základních knihoven systému NEMEA[42]. Implementace umožňuje zvolit prefix nebo suffix režim, separátor řetězců a další parametry.

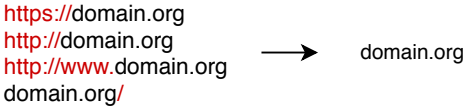
Doménová jména jsou ve stromu ukládána v režimu suffix od top-level-domain (TLD) a separátor je tečka. Entita s nejvyšším počtem subdomén tedy udává hloubku stromu. Příklad doménového stromu:



Obrázek 3.2: Ukázka prefix tree

URL adresy jsou ve stromu ukládány v režimu prefix a bez separátoru (každý znak je uložen v uzlu stromu). Blacklistované URL mají často společný prefix a liší se např. jen koncovkou souboru. Hloubka stromu je určena nejdelší nalezenou URL.

V obou případech detekce se hledá přesná shoda (*exact match*). Každý záznam je proto předzpracován tak, aby v detekci nehrály roli drobné změny v doménovém jméně/URL (to se týká detektorů i Downloaderu). Následující ukázka to demonstruje.



V případě šifrovaného HTTPS spojení jsou aplikační data URL detektoru skryta. Díky Server Name Indication (SNI) je však vidět alespoň hostitel

[42] https://github.com/CESNET/Nemea-Framework/tree/master/common/prefix_tree

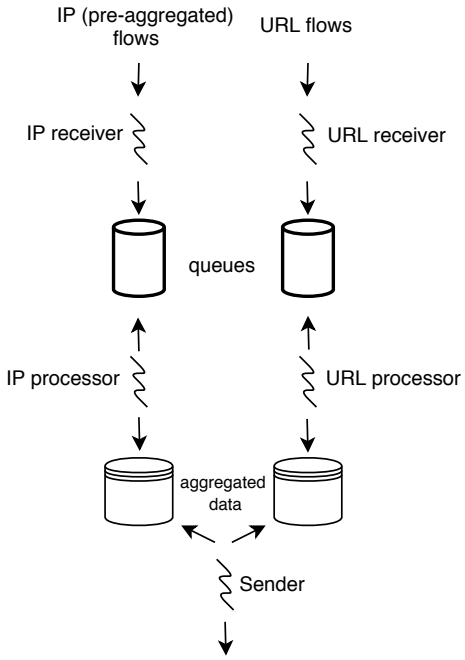
(`HTTP_HOST`). SNI je technika, která umožňuje hostovat více webových stránek na jedné IP adrese. V průběhu SSL-handshake (který samotný není šifrovaný) klient specifikuje cílového hostitele, aby mohl server vrátit příslušný certifikát. URL detektor tedy v tomto případě detekuje pouze FQDN záznamy ze svého detekčního souboru.

3.3 Agregátory

Přestože flows jsou již agregované pakety, je potřeba další agregace dat kvůli efektivnějšímu nahlašování událostí. Často totiž komunikace klienta s blacklistovanou entitou generuje více flows, a nahlašování jednotlivých flow by bylo redundantní. Agregují se pouze výstupy z IP a URL detektorů, DNS detektor zasílá data přímo do Adaptivního filtru (více v Sekci 3.4).

Agregace IP toků probíhá ve dvojí fázi. V první fázi je použit *Univerzální agregátor*, který je psán v C++ a data zpracovává velmi rychle (kolem 1M flows/s při použití 5 agregačních klíčů). Druhá fáze poté předzpracovaná data agreguje do tzv. *bi-flow* (obousměrné flow), to provádí Blacklist agregátor (dále jen Agregátor). Agregace URL probíhá pouze v Agregátoru.

Agregátor pracuje s 5 vlákny, dvě přijímací, dvě pracující a jedno odesílací. Přijímací vlákna musí být dvě, protože každé přijímá jinou šablonu UniRec dat. Vlákna pouze ukládají data do fronty, odkud je čtou pracující vlákna, které provádějí samotnou agregaci. Obecně je kladen důraz na to, aby přijímací vlákna prováděly minimum práce tak, aby stíhaly přijmout větší objem dat a zprávy se nemusely zahazovat. Pracující vlákna provádějí agregaci a výsledky ukládají do globálních struktur. Odesílací vlákno vždy za určitý časový interval agregovaná data odešle na výstup (Obrázek 3.3).



Obrázek 3.3: Implementace Agregátoru

3.4 Adaptivní filtr

Adaptivní filtr pracuje se 4 vlákny:

- *IP/URL receiver* — přijímá již agregovaná data v JSON formátu, obsahující IP a URL detekce
- *DNS receiver* — přijímá flows přímo z DNS detektoru
- *Controller* — čte data ze společné fronty od *receiverů*; pokud provoz odpovídá některému definovanému scénáři, vytvoří tzv. `scenario event` a uloží ji do globálního slovníku (`g_scenario_events`); odesílá také události Reportéru
- *Processor* — periodicky zpracovává `g_scenario_events`, vytváří detekční soubor pro Adaptivní IP detektor, posílá události do Evidence

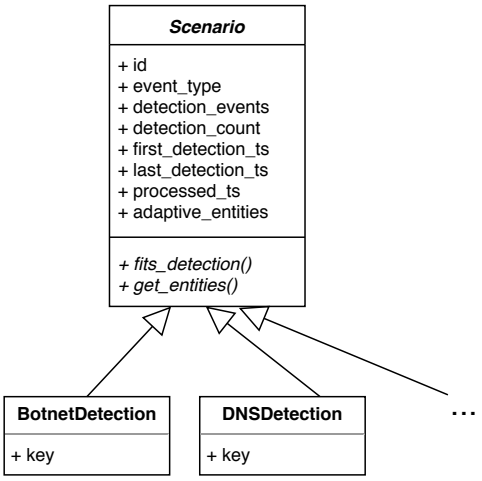
Scénáře jsou definovány v souboru `scenarios.py`. Abstraktní třída Scenario zapouzdřuje detekovanou událost. Konkrétní scénáře se definují jako třídy, které abstraktní třídu rozšiřují (Obrázek 3.4). Instanci takové třídy nazývám `scenario event`. Význam atributů je následující:

- `id` — ID ve formátu UUID (unikátní 32znakový řetězec)

- `event_type` — název (odvozen z názvu podtřídy třídy Scenario)
- `grouped_events` — seznam detekčních událostí se stejným klíčem
- `grouped_events_count` — počet detekčních událostí se stejným klíčem
- `first_detection_ts` — čas. značka první události (z `grouped_events`)
- `last_detection_ts` — čas. značka poslední události (z `grouped_events`)
- `processed_ts` — časová značka posledního zpracování vláknem *Processor*
- `adaptive_entities` — tzv. *adaptivní entity* k dané události, které se zapisují do detekčního souboru pro Adaptivní IP detektor

Podtřídy samy definují tyto metody a atributy:

- `fits_detection()` — zjišťuje, zda přijatá data odpovídají danému scénáři (třídní metoda)
- `get_entities()` — získává adaptivní entity
- `key` — klíč, který identifikuje instanci scénáře (`scenario event`). Například pro BotnetDetection je to IP adresa C&C serveru



Obrázek 3.4: Diagram tříd modelu Adaptivní filtr

Controller čte příchozí data (`detection_event`) a kontroluje, jestli detekce odpovídá některým scénářům. Pokud je nalezena shoda, kontroluje se, jestli v `g_scenario_events` už existuje událost s daným klíčem. Pokud ano, událost

se pouze aktualizuje. Pokud ne, vytvoří se nová. To demonstruje následující ukázka kódu.

```
# Try to fit the detection event to some scenario
for scenario_class in scenarios.Scenario.__subclasses__():
    if scenario_class.fits(detection_iface, detection_event):
        detected_scenario = scenario_class(detection_event)

if detected_scenario:
    # Scenario fits,
    # detected_scenario is an instance of some scenario subclass
    try:
        events_lock.acquire()
        # Do we know about this specific case of the scenario?
        scenario_event = g_scenario_events[detected_scenario.key]

        # Store the detection event and update metadata
        scenario_event.grouped_events.append(detection_event)
        scenario_event.grouped_events_cnt += 1
        scenario_event.last_detection_ts = time()

    except KeyError:
        # New scenario event
        g_scenario_events[detected_scenario.key] = detected_scenario
    finally:
        events_lock.release()
```

Processor periodicky zpracovává `g_scenario_events` a provádí tři úkony:

- Získává adaptivní entity jednotlivých událostí
- Vytváří detekční soubor pro Adaptivní IP detektor
- Exportuje `scenario_events` do Evidence

Processor má uložený aktuální seznam adaptivních entit všech událostí. Při průchodu událostmi znovu získává adaptivní entity, pokud se událost změnila. Na průchodu událostmi konci zkontroluje, jestli se změnil seznam všech entit. Pokud ano, vytvoří detekční soubor pro Adaptivní IP detektor a seznam entit si uloží. Každá událost se zpracovává následujícím způsobem:

- Pokud `last_detection_ts > processed_ts` (událost nebyla zpracována od poslední změny), znovu se získají adaptivní entity a aktualizuje se `processed_ts`
- Pokud `first_detection_ts + evidence_timeout > now` (uplynul *evidence timeout*, tj. čas, za který se má daná událost exportovat do Evidence), je událost označena k odeslání

Po průchodu událostmi provádí *Processor* tyto úkony:

- Události označené k odeslání jsou exportovány do Evidence a ze seznamu adaptivních entit jsou odebrány adaptivní entity těchto událostí

- Je vytvořen detekční soubor pro Adaptivní IP detektor, který obsahuje aktuální adaptivní entity

Adaptivní IP detektor funguje skoro identicky jako normální IP detektor (je to instance téhož programu). Rozdíl je pouze v tom, že používá speciální formát detekčního souboru. Ke každému záznamu v detekčním souboru je přidána položka `ADAPTIVE_IDS` (seznam oddělený separátorem). Jedná se o ID hodnoty přidružených `scenario events`. Pomocí těchto ID se poté párují `scenario events` a `adaptive events` v Evidence. Položka `ADAPTIVE_IDS` je tedy součástí výstupu. Detekční záznam má formát:

```
<entita><separátor><adaptive_bl_index><separátor><adaptive_ids>
```

`adaptive_bl_index` je pevně zvolená konstanta, pomocí které IP detektor rozezná, že má být spuštěn v adaptivním módu. Příklad:

```
...
10.12.91.158,999,a459ffe0-cdc3-430e-9bf4-25133975272b
10.12.157.180,999,a459ffe0-cdc3-430e-9bf4-25133975272b
10.12.161.237,999,770b96cc-7ea3-46ce-b315-92647a04c51a
...
```

3.5 Reportér

Modul Reportér přijímá agregované události (z IP a URL detektorů) a převádí je na formát Intrusion Detection Extensible Alert (IDEA[43]), což je standardní formát hlášení v systému Warden a dalších systémech sdílení incidentů. Jedná se o JSON formát s přesně definovanou strukturou záznamů. V příloze B je uvedena ukázka IDEA zprávy s popisem vybraných polí.

[43] https://idea.cesnet.cz/en/index

Kapitola 4

Testování a nasazení

Tato kapitola obsahuje testy dílčích modulů zejména na propustnost, zátěž CPU a paměti. Kromě dílčích testů byl pak proveden také *end-to-end* test, který ověřuje funkčnost celé sady modulů blacklistfilter.

4.1 Testovací prostředí

4.1.1 Testování na lokálním stroji

Testy propustnosti a agregace byly provedeny na mém osobním notebooku Lenovo Y570 (2012) s těmito parametry:

- Systém Ubuntu 16.04 LTS
- Procesor Intel Core i7-2670QM CPU 2.20GHz x 8 (Sandy Bridge)
- 8 GB RAM

4.1.2 Kolektor NEMEA

Kolektor sbírá data z reálného provozu sítě CESNET2 a je na něm nasazen systém NEMEA. Testovací verze kolektoru[44] je určená pro testování vytvořených modulů. Kolektor byl jednak použit k end-to-end testování, a také posloužil jako zdroj dat reálného provozu. Konfigurace stroje:

- Scientific Linux 7.3
- Intel(R) Xeon(R) CPU E5-2670 @ 2.60GHz
- 16 GB RAM

[44] collector-nemea-test.liberouter.org

4.2 Testování detekčních modulů

Pro testovací účely byly nasbírány vzorky (flows) reálného provozu a uloženy do následující souborů:

- `http_10M.trapcap` — 10 milionů flows provozu HTTP
- `dns_10M.trapcap` — 10 milionů flows provozu DNS
- `agg_ipblacklist.trapcap` — výstup z Univerzálního agregátoru (agregující data z IP detektoru) s 3 616 475 flows

Následně byly vytvořeny speciální sady blacklistů. Každá sada obsahuje právě 9 blacklistů s různým počtem entit od 10 do 10 mil.

- Sada IP — náhodně generované IP adresy z rovnoměrného rozdělení (důležité pro korektní testování `binary_search`)
- Náhodná sada DNS — náhodně generovaná doménová jména (délka všech domén přesně 14 znaků)
- Náhodná sada URL — stejné jako náhodná sada DNS, ale ke každé entitě přidán náhodný řetězec (PATH) o délce 67 znaků (dle [33] naměřený medián délky URL)
- Sada Alexa DNS — milion nejnavštěvovanějších webových stránek podle serveru Alexa[45]
- Sada Alexa URL — stejné jako sada Alexa DNS, ke každé entitě opět přidán náhodný řetězec o délce 67 znaků

S těmito sadami blacklistů byly provedeny testy propustnosti detekčních modulů. Vstupem byly soubory s 10 mil. flows (`{dns,http}_10M.trapcap`). Měření probíhalo pomocí následujícího příkazu, který čte vstupní data ze souboru a přeposílá je na UNIX soket detekčního modulu.

```
traffic_repeater -i f:dns_10M.trapcap,u:in
ipblacklistfilter -i u:in,f:out -4 ip4_100.blist
```

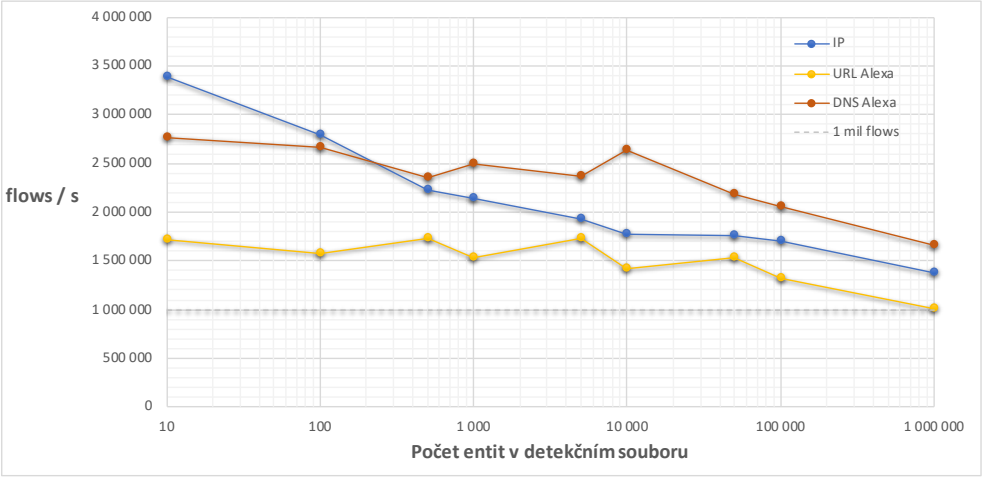
Každé měření bylo provedeno 10krát a byl vypočten průměr. Jak je vidět v grafu na Obrázku 4.1, i pro milion entit v blacklistu je propustnost modulů více než 1 mil. flows/s, což je pro síť CESNET2 dostačující (je požadováno minimálně 300 tis. flows/s). Náhodné sady blacklistů neobsahují na výstupu žádné detekce (procházení prefixovým stromem končí v prvních úrovních stromu).

[45] http://s3.amazonaws.com/alexa-static/top-1m.csv.zip

Naopak sady Alexa obsahují reálné domény a tudíž je propustnost mírně nižší. Propustnost je tedy závislá na zvolených datech, ale rozdíl je zanedbatelný. Kompletní měření je vidět v Tabulce 4.1. Náhodné sady byly použity také pro měření spotřeby paměti, uvedeno v Tabulce 4.2. Spotřeba paměti je konstantní po celou dobu běhu modulu a závisí pouze na počtu záznamů v detekčních souborech (objem načtených dat).

Byly napsány také testy na ověření funkčnosti detektorů. U URL a DNS detektorů byly testovány drobné rozdíly v názvu entit (WWW prefix, malá/-velká písmena apod.) a jejich detekce. U IP detektoru byl ověřován vstup a výstup následujícím způsobem (pro IPv4 i IPv6):

- Detekce první a poslední adresy v detekčním souboru (test funkčnosti `binary_search`)
- Detekce všech adres v detekčním souboru
- Detekce podsítí s různými prefixy



Obrázek 4.1: Propustnost detekčních modulů v závislosti na počtu blacklistovaných entit

4.3 Testování agregačního modulu

Testování propustnosti agregačního modulu není tak přímočaré jako u detekčních modulů. Výstup agregace je závislý na zvoleném časovém okně, takže nelze jednoduše měřit vstup a výstup. Snadno měřitelná je však schopnost přijímacích vláken ukládat příchozí data do fronty. Opět pomocí nástroje

Vstupní data	Počet entit	Čas zpracování (s)				
		IP	URL		DNS	
			Random	Alexa	Random	Alexa
10M flows	10	2,954	5,982	5,819	3,999	3,612
	100	3,581	6,104	6,342	3,748	3,748
	500	4,497	6,140	5,810	3,827	4,253
	1 000	4,681	6,235	6,536	3,598	3,998
	5 000	5,191	5,946	5,755	3,424	4,214
	10 000	5,621	5,751	7,036	3,088	3,796
	50 000	5,676	5,585	6,500	3,004	4,582
	100 000	5,873	7,081	7,609	3,152	4,880
	1 000 000	7,263	7,964	9,921	3,588	6,050

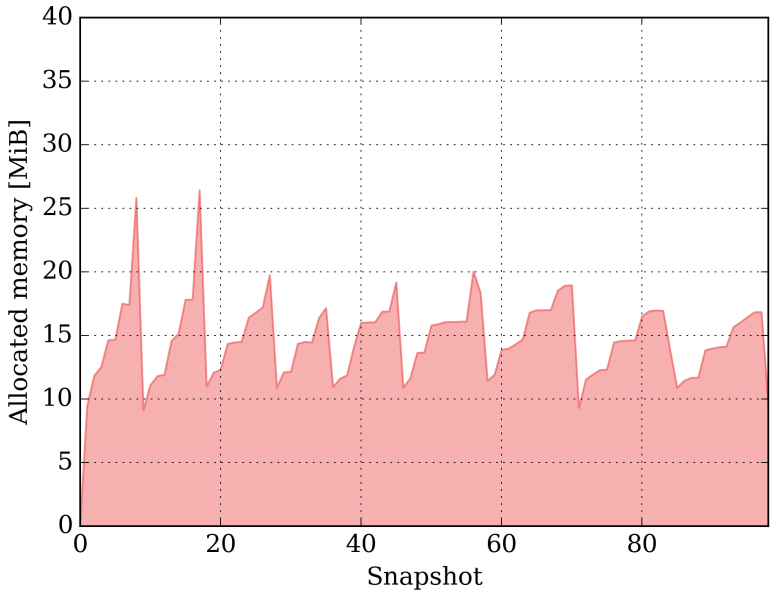
Tabulka 4.1: Propustnost detekčních modulů podle počtu použitých entit (pro různé sady blacklistů)

	Využití RAM (MB)		
Počet entit	IP	URL Random	DNS Random
10	5,7	1,4	1,4
100	5,7	1,4	1,4
500	5,8	1,9	1,9
1 000	6,0	2,4	2,4
5 000	6,1	6,8	6,8
10 000	6,3	12,5	11,9
50 000	11,5	56,6	52,1
100 000	15,2	114,0	103,5
1 000 000	61,8	1164,0	1037,0

Tabulka 4.2: Využití RAM detekčních modulů

`traffic_repeater` bylo naměřeno 57 tis. flows/s (průměr z 10 iterací). Ačkoliv je to oproti detekčním modulům výrazně méně, je to dostačující hodnota. Agregační modul totiž přijímá mnohem menší objem dat než detekční moduly, které zpracovávají celý provoz sítě CESNET2. Poměr výstup/vstup je pro IP detektor nasazený na testovacím kolektoru cca 0.003 (s cca 20 tis. blacklistovanými entitami).

Dále bylo testováno, jestli se přijatá data stíhají zpracovávat/agregovat. Agregační modul byl spuštěn po dobu 45 minut (doba zpracování souboru `agg_ipblacklist.trapcap` s nástrojem *massif*) s časovým agregačním oknem 5 minut. Pomocí nástroje massif byla změřena spotřeba paměti. Na Obrázku 4.2 je vidět, že spotřeba kolísá cyklicky (9 cyklů odpovídá 45/5), ale dlouhodobě neroste. Maxima (26 MB) dosahuje těsně před odesláním agrego-



Obrázek 4.2: Agregační modul — Graf spotřeby paměti v čase

vaných dat. Z toho můžeme usuzovat, že pro maximální možný tok přijatých dat (57 tis. flows/s) se data stíhají zpracovávat/agregovat. Je ovšem otázka, jak znatelný vliv má na měření samotné použití nástroje massif, které program zpomaluje.

4.4 Statistiky z reálného provozu

Bylo provedeno testování celé sady modulů (end-to-end test) na testovacím kolektoru NEMEA, který monitoruje síť CESNET2. Testování trvalo 4 h a 10 min. Bylo použito 10 IP a 8 URL/DNS blacklistů (různých kategorií). V tabulce 4.3 je statistika detektorů, počet příchozích a detekovaných flows, a průměrná spotřeba CPU (změřená příkazem `ps -p <pid> -o %cpu`). Spotřeba CPU ostatních modulů (Agregátor, Adaptivní filtr, ..) byla zanedbatelná.

Celkem bylo vytvořeno:

- 21 333 událostí nahlášených Reportérem
- 3 940 událostí uložené do Evidence pro analýzu

  Bylo tedy vytvořeno 85 nahlášených událostí (alertů) za minutu, což je relativně vysoké číslo. V takové záplavě hlášení potřebují administrátoři sítí, kterým jsou posílány varovné e-maily, věřit, že se nejedná o false-positive, a že se mají alertem zabývat. To je hlavní motivace pro ukládání dat do Evidence pro důkladnější analýzu.

Detektor	Počet blacklist. entit	Vstupních flows	Výstupních flows	Detekční poměr	Prům. spotřeba CPU
IP	17 793	1 071 038 838	1 692 436	0,0015	3,1 %
URL	45 317	112 077 023	18	—	0,8 %
DNS	30 879	313 462 422	17 738	0,00005	1,5 %

Tabulka 4.3: Statistiky detektorů z měření v reálném provozu

Kapitola 5

Vyhodnocení dat

Vytvořený Adaptivní filtr ukládá detekované události do systému Evidence, kde je možné provádět další analýzu. Evidence je úložiště zachyceného provozu podezřelých klientů a sestává ze 2 souborů:

- `scenario_events` — obsahuje události generované klasickými detektory (např. komunikace několika klientů s C&C), ve formátu JSON
- `adaptive_events` — obsahuje události generované Adaptivním IP detektorem (veškerá komunikace těchto klientů), ve formátu UniRec

Níže uvádím ukázky obou souborů, jednotlivé události jsou párovány pomocí identifikátoru UUID.

scenario_event


```
{
  "event_type": "BotnetDetection",
  "id": "d23824f8-62c9-42cb-a931-d3022e68a11f",
  "key": "80.78.250.26",

  "first_detection_ts": 1544620961.7213767,
  "last_detection_ts": 1544621177.7711003,

  "grouped_events_cnt": 4,
  "grouped_events": [
    {
      "type": "ip",
      "blacklist_id": 1,
      "source": "80.78.250.26",
      "source_ports": [
        443
      ],
      "targets": [
```

```
        "1.2.3.4",
        "5.6.7.8"
      ],
      "protocol": 6,
      "src_sent_flows": 4,
      "src_sent_bytes": 292510,
      "src_sent_packets": 203

      "tgt_sent_flows": 3,
      "tgt_sent_bytes": 3284,
      "tgt_sent_packets": 63,

      "ts_first": 1544620900.178,
      "ts_last": 1544620901.165,

      "agg_win_minutes": 0.2,
    },
    ...
    ...
}
```

adaptive_event

```
...
string ADAPTIVE_IDS     "d23824f8-62c9-42cb-a931-d3022e68a11f"
ipaddr DST_IP           9.9.9.9
ipaddr SRC_IP           1.2.3.4
uint64 BYTES            339
uint64 DST_BLACKLIST    0
uint64 SRC_BLACKLIST    999
time TIME_FIRST         2018-12-12T13:27:14.152
time TIME_LAST          2018-12-12T13:27:20.061
uint32 PACKETS          6
uint16 DST_PORT         443
uint16 SRC_PORT         52943
uint8 PROTOCOL          6

string ADAPTIVE_IDS     "d23824f8-62c9-42cb-a931-d3022e68a11f"
ipaddr DST_IP           1.2.3.4
ipaddr SRC_IP           9.9.9.9
uint64 BYTES            78
uint64 DST_BLACKLIST    999
uint64 SRC_BLACKLIST    0
time TIME_FIRST         2018-12-12T13:26:15.282
time TIME_LAST          2018-12-12T13:26:15.282
uint32 PACKETS          1
uint16 DST_PORT         52943
uint16 SRC_PORT         443
uint8 PROTOCOL          6
```

Interpretace `scenario_event`: Klienti s adresami `1.2.3.4` a `5.6.7.8` komunikovali (ve stejném agregačním časovém okně) s C&C serverem s adresou `80.78.250.26` a připojili se k TCP portu 443. C&C je na blacklistu s indexem 1 (v konfiguračním souboru). C&C server odeslal klientům celkem 4 flows (`src_sent_flows`), klienti v součtu odeslali 3 flows (`tgt_sent_flows`). `grouped_events_cnt` udává počet seskupených událostí ve stejném agregačním časovém okně. Celá událost je vytvořena po vypršení intervalu, který je v konfiguraci adaptivního filteru označen jako `evidence_timeout`.

Interpretace `adaptive_event`:

- První zpráva: Podezřelý klient s adresou `1.2.3.4` se připojil k (neznámé) adrese `9.9.9.9` na TCP port 443, přičemž odeslal celkem 6 paketů (z portu 52 943)

- Druhá zpráva: Neznámá adresa `9.9.9.9` odeslala podezřelému klientovi 1 paket (v rámci stejného spojení).

Výhodou je, že v adaptivních událostech máme klasické flows, a tedy více informací (žádná agregace), včetně obou směrů toku dat. Tato data se následně analyzují nástrojem zvaným Evaluátor. Evaluátor je další Python skript, který načítá oba soubory z Evidence a používá jednoduché heuristiky, aby ověřil/vyvrátil podezření uvalené na klienta. Skript je spouštěn manuálně a výstupem je textový report o podezřelých klientech, který je nutné podrobit manuální analýze. Lze na něm ale demonstrovat ověření koncepce řešení. Skript pro každou událost a podezřelého klienta vyhodnocuje následující parametry provozu:

- Počet flows obsahující komunikaci klienta s původním C&C serverem

- Počet jiných C&C na aktuálních blacklistech, se kterými klient komunikoval

- Počet flows klienta obsahující IRC porty

- Informace o klientovi získané ze systému NERD

Následující výpis je příkladem výstupu skriptu:

```
...
-------- SUSPICIOUS CLIENT REPORT --------
event ID:            fb356830-4daa-4dd3-805e-db143a5967be
Suspicious client:   1.2.3.4
CC Server:           198.54.117.200
Flows w/ IRC ports:  15
NERD INFO:           [{'cat': 'ReconScanning',
                       'node': 'cz.cesnet.nemea.vportscan',
                       'date': '2019-01-06', 'n': 6}]
-------- SUSPICIOUS CLIENT REPORT --------
event ID:            d23824f8-62c9-42cb-a931-d3022e68a11f
CC Server:           80.78.250.26
Suspicious client:   5.6.7.8
Flows w/ this CC:    4
Flows w/ other CCs:  5
...
```

Reporty (`SUSPICIOUS CLIENT REPORT`) výše jsou tedy souhrnem nalezených parametrů komunikace podezřelého klienta.

V prvním reportu lze vidět, že byly nalezeny IRC porty v komunikaci klienta, ale zároveň je k tomuto klientovi událost v systému NERD od detektoru *cz.cesnet.nemea.vportscan*. Klient tedy prováděl vertikální sken portů a je pravděpodobné, že IRC porty pouze skenoval, navíc nebyly nalezeny žádné flows komunikace s původním C&C serverem. Detekci klienta lze tedy označit za falešný poplach.

V druhém reportu je naopak vidět komunikace s původním C&C serverem (4 flows) a některými ostatními C&C servery, které se vyskytují na aktuálních blacklistech (5 flows). V systému NERD není o klientovi žádná informace. Detekci lze buď podrobit důkladnější analýze, nebo nahlásit do systému Warden.

Důkladnější analýza může probíhat hledáním vzorců komunikace klienta s C&C serverem/servery. Podle [34] lze například hledat HTTP provoz, kde se opakuje počet přenesených bajtů, nebo kde je časová pravidelnost komunikace. Může se pak jednat o tzv. *Beacon* pakety, které nakažení klienti pravidelně posílají C&C serveru jako indikátor toho, že jsou aktivní (heart-beat).

Takový provoz byl v Evidence skutečně nalezen, podezřelý klient pravidelně posílal C&C serveru 336 bajtů a dostával odpověď se 124 bajty. Navíc se v jeho komunikaci vyskytovaly IRC porty. C&C server byl navíc nalezen na dalších blacklistech jako mining server. Tento klient byl nahlášen jako podezřelý bot/miner týmu CESNET-CERTS[46](CESNET Computer Security Incident Response Team).

[46]https://csirt.cesnet.cz/cs/index

Závěr

Komunikace zařízení v počítačových sítích je založená na posílání síťových paketů. V praxi se provoz reprezentuje síťovými toky mezi dvojicí zařízení, což je úspornější řešení než paketová analýza. Síťové toky umožňují bezpečnostní analýzu, díky které se dá identifikovat škodlivý provoz či podezřelé chování zařízení.

K účelu identifikace škodlivých adres v internetu existují veřejně dostupné seznamy adres, tzv. *blacklists*. Výskyt adresy vlastního zařízení na nějakém ze seznamů je důvodem ke hledání potenciálních problémů, ale v rámci této práce je důležitějším případem spíše komunikace s cizí adresou, která se objevuje na blacklistu. Tato práce se zabývá identifikací případů, kdy je potřeba komunikaci s cizími adresami hlásit jako samostatnou bezpečnostní událost a kdy je naopak takové hlášení nežádoucí, protože by vedlo k chybné interpretaci provozu a podstaty události.

Cílem práce bylo vylepšit existující základní řešení detekce podezřelé komunikace v open-source systému NEMEA. Toto řešení původně hlásilo každý síťový tok, který libovolná adresa z blacklistu vygenerovala nebo přijala, přičemž tato hlášení byla agregována podle zdrojových a cílových adres. Nově navržené a vytvořené řešení tuto funkcionalitu značně rozšiřuje, a to o zachytávání souvisejícího provozu a jeho následnou analýzu. Na základě kontextu, který zachycená data vytváří, je možné přesněji rozhodnout, zda-li není pozorovaný síťový tok součástí jiné události. V případě, že pozorovaná komunikace nesouvisí s účelem blacklistu, není žádoucí posílat bezpečnostní hlášení a nová sada modulů proto taková hlášení za základě analýzy ani nevytvoří.

Text této práce popisuje konkrétní scénáře, při kterých je irelevantní hlásit správcům komunikaci zařízení s podezřelými adresami. Příkladem častého typu provozu, může být horizontální skenování, při kterém se z jednoho zdroje generují síťové toky k různým (často náhodně vybraným cílům). V případě těchto skenů by bylo zavádějící hlásit zdroj skenu jako stroj nakažený malwarem pouze na základě pozorovaného provozu na C&C server, který je jinak s malwarem spojován. Problém těchto potenciálně falešných hlášení řeší vy-

tvořená sada NEMEA modulů na základě zachytávání kontextu komunikace a následnou analýzou těchto dodatečných dat.

Hlavním přínosem této práce je sada modulů pro systém NEMEA, která zahrnuje tzv. adaptivní filtr a modul pro analýzu zachycených dat. Na základě vzniklého návrhu, který byl také prezentován na mezinárodní konferenci [35], byly moduly vytvořeny, otestovány a nasazeny v pilotním provozu na kolektoru sdružení CESNET. Nasazení vzniklé nové verze modulů využívajících blacklisty vedlo ke zmenšení množství hlášených bezpečnostních událostí (díky použití pokročilejší agregace), ale také ke zlepšení jejich kvality (nehlásí se false-positive události spojené např. se síťovými skeny).

Budoucí práce

Cíle této závěrečné práce byly naplněny, neboť došlo k výraznému vylepšení využití veřejně dostupných blacklistů k detekci podezřelého chování síťových zařízení. Na druhou stranu tato diplomová práce umožňuje další pokračující výzkum hlavně v oblasti vyhodnocování nasbíraných dat pomocí adaptivního filtru a dalšího vylepšování přesnosti hlášených bezpečnostních událostí.

Vzhledem k tomu, že vytvořená sada modulů umožňuje snadnou rozšiřitelnost pomocí scénářů (popsaných v textu práce), nabízí se jako další pokračování této práce vylepšování stávajících scénářů a vývoj dalších.

Vytvořený nástroj na stahování obsahu blacklistů ukládá také historii obsahu. V rámci navazující práce se nabízí myšlenka tuto historii využít a zohledňovat dobu přidání adresy na blacklist. Důvodem je fakt, že dlouhá doba od přidání adresy na blacklist může poskytnout dost času na opravu problému (např. vyčištění nakaženého stroje), a proto již výskyt na blacklistu nemusí být aktuální. Tento předpoklad však musí být důkladně ověřen a proto nápad na zohlednění historie blacklistu je potenciálním směrem dalšího výzkumu.

Na mou diplomovou práci bude navazovat bakalářská práce studenta Jana Suchara, který v době psaní této práce projevil zájem o spolupráci na tomto tématu.

Literatura

[1] Santanna, J.; Rijswijk-Deij, R.; Hofstede, R.; aj.: Booters - An analysis of DDoS-as-a-service attacks. 06 2015, s. 243–251, [cit. 2018-12-20].

[2] Cejka, T.; Bartos, V.; Svepes, M.; aj.: NEMEA: A Framework for Network Traffic Analysis. In *12th International Conference on Network and Service Management (CNSM 2016)*, 2016, doi:10.1109/CNSM.2016.7818417. Dostupné z: `http://dx.doi.org/10.1109/CNSM.2016.7818417`

[3] What is a Botnet? [online], [cit. 2018-12-25]. Dostupné z: `https://www.techopedia.com/definition/384/botnet`

[4] Goel, S.; Baykal, A.; Pon, D.: Botnets: the anatomy of a case. *Journal of Information Systems Security*, ročník 1, č. 3, 2006: s. 1–12.

[5] OVH: The DDoS that didn't break the camel's VAC*. [online], [cit. 2018-12-25]. Dostupné z: `https://www.ovh.com/world/news/articles/a2367.the-ddos-that-didnt-break-the-camels-vac`

[6] Canavan, J.: The evolution of malicious IRC bots. 01 2005. Dostupné z: `https://www.symantec.com/avcenter/reference/the.evolution.of.malicious.irc.bots.pdf`

[7] Limarunothai, R.; Munlin, M. A.: Trends and Challenges of Botnet Architectures and Detection Techniques. *Journal of Information Science and Technology*, 2015, [cit. 2018-12-20].

[8] Cao, L.; Qiu, X.: ASP2P: An advanced botnet based on social networks over hybrid P2P. In *2013 22nd Wireless and Optical Communication Conference*, May 2013, ISSN 2379-1268, s. 677–682, doi:10.1109/WOCC.2013.6676460.

[9] Botnets, how do they work? Architectures and case studies – Part 2. Apr 2013, [cit. 2018-12-20]. Dostupné z: `https:`

//resources.infosecinstitute.com/botnets-how-do-they-work-architectures-and-case-studies-part-2

[10] Čandrlić, G.: 15 Most Dangerous DDoS Attacks That Ever Happened. [online], 2016, [cit. 2018-12-25]. Dostupné z: https://www.globaldots.com/15-most-dangerous-ddos-attacks-that-ever-happened

[11] Davis, G.; Roccia, T.; Enterprise, M.: Necurs Botnet Leads the World in Sending Spam Traffic. Mar 2018. Dostupné z: https://securingtomorrow.mcafee.com/other-blogs/mcafee-labs/necurs-botnet-leads-the-world-in-sending-spam-traffic/

[12] Abrams, L.: SPAM Campaign Underway that uses Encrypted Word Docs to Install Ursnif. Apr 2017. Dostupné z: https://www.bleepingcomputer.com/news/security/spam-campaign-underway-that-uses-encrypted-word-docs-to-install-ursnif/

[13] How to Defraud Display Advertisers with Zeus. 2013. Dostupné z: http://www.spider.io/blog/2013/11/how-to-defraud-display-advertisers-with-zeus/

[14] McAfee Labs Threats Report. Technická zpráva, McAfee, 2018, [cit. 2018-12-25]. Dostupné z: https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-dec-2018.pdf

[15] Seth, S.: What is Botnet Mining? Apr 2018. Dostupné z: https://www.investopedia.com/tech/what-botnet-mining/

[16] Wikipedie: Tor (software) — Wikipedie: Otevřená encyklopedie. 2018, [cit. 2018-12-25]. Dostupné z: https://cs.wikipedia.org/w/index.php?title=Tor_(software)&oldid=16788337

[17] 94% of Tor Traffic is Malicious, According To CloudFlare. May 2018. Dostupné z: https://darkwebnews.com/anonymity-tools/tor/tor-malicious-cloudflare/

[18] Wikipedie: Phishing — Wikipedie: Otevřená encyklopedie. 2018, [cit. 2018-12-25]. Dostupné z: https://cs.wikipedia.org/w/index.php?title=Phishing&oldid=16700526

[19] Varshney, G.; Misra, M.; K. Atrey, P.: A survey and classification of web phishing detection schemes. *Security and Communication Networks*, ročník 9, 10 2016, doi:10.1002/sec.1674.

[20] What is Martian Address? - Definition from Techopedia. Dostupné z: https://www.techopedia.com/definition/25075/martian-address

[21] The OSI Model – What It Is; Why It Matters; Why It Doesn't Matter. [cit. 2018-12-25]. Dostupné z: http://www.tech-faq.com/osi-model.html

[22] Introduction to Cisco IOS NetFlow - A Technical Overview. May 2012. Dostupné z: https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html

[23] Brownlee, N.; Mills, C.; Ruth, G.: RFC 2722: Traffic flow measurement: Architecture. Technická zpráva, IETF, 1999. Dostupné z: www.ietf.org/rfc/rfc2722.txt

[24] Wikipedie: NetFlow — Wikipedie: Otevřená encyklopedie. 2016, [cit. 2018-12-25]. Dostupné z: https://cs.wikipedia.org/w/index.php?title=NetFlow&oldid=13565400

[25] Claise, B.; Trammell, B.; Aitken, P.: Specification of the IP flow information export (IPFIX) protocol for the exchange of flow information. Technická zpráva, IETF, 2013. Dostupné z: https://tools.ietf.org/html/rfc7011

[26] Hofstede, R.; Čeleda, P.; Trammell, B.; aj.: Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX. *IEEE Communications Surveys Tutorials*, ročník 16, č. 4, Fourthquarter 2014: s. 2037–2064, ISSN 1553-877X, doi:10.1109/COMST.2014.2321898.

[27] CESNET, z. s. p. o: General Information; NEMEA: System for network traffic analysis and anomaly detection. [online], [cit. 2018-12-25]. Dostupné z: https://nemea.liberouter.org/

[28] CESNET, z. s. p. o: Topologie sítě CESNET2. [online], [cit. 2018-12-25]. Dostupné z: https://www.cesnet.cz/sluzby/pripojeni/topologie/

[29] Čejka T.: Network Monitoring and Anomaly Detection. *Síťová bezpečnost*. [online], [cit. 2019-01-08]. Dostupné z: https://edux.fit.cvut.cz/courses/MI-SIB.16/_media/tutorials/cejkat-network-monitoring-detection.pdf

[30] Bartos, V.: NERD: Úvod pro vývojáře. [cit. 2018-12-25]. Dostupné z: https://github.com/CESNET/NERD/raw/master/doc/NERD_intro_for_developers_(CZ).pdf

[31] CESNET, z. s. p. o: Warden. [cit. 2019-01-08]. Dostupné z: https://warden.cesnet.cz//en/index

[32] Slabihoudek, M.: *Universal module for data aggregation in the NEMEA system*. Master's thesis, CTU in Prague, Prague, 2018.

[33] Kelvin: Average length of a URL (Part 2). [online], 2010. Dostupné z: http://www.supermind.org/blog/740/average-length-of-a-url-part-2

[34] How to Leverage Log Services to Analyze C&C Traffic. Aug 2018. Dostupné z: https://securityintelligence.com/how-to-leverage-log-services-to-analyze-cc-traffic/

[35] Šuster, F.; Čejka, T.: Stream-wise adaptive blacklist filter based on flow data. In *Proceedings of the 6th Prague Embedded Systems Workshop*, ČVUT v Praze, Fakulta informačních technologií, 2018, ISBN 978-80-01-06456-6, s. 38–39.

[36] CESNET, z. s. p. o: IDEA format definition. [online], 2015. Dostupné z: https://idea.cesnet.cz/en/definition

Příloha A

Seznam použitých zkratek

API Application Programming Interface

C&C Command-and-Control

CERTS/CSIRT Cyber Security Response Team

CPU Central Processing Unit

DDoS Distributed Denial-of-Service

DNS Domain Name System

DPI Deep Packet Inspection

FQDN Fully Qualified Domain Name

HTTP Hypertext Transfer Protocol

IANA Internet Assigned Numbers Authority

IDEA Intrusion Detection Extensible Alert

IDS Intrusion Detection System

IFC Interface Communication

IoT Internet of Things

IP Internet Protocol

IPFIX IP Flow Information Export

IPS Intrusion Prevention System

IRC Internet Relay Chat

ISP Internet Service Provider

JSON JavaScript Object Notation

NREN National Research and Education Network

P2P Peer-to-peer

POP3 Post Office Protocol

RAM Random Access Memory

RIR Regional Internet Registrar

SMTP Simple Mail Transfer Protocol

SNI Server Name Indication

SPOF Single Point of Failure

TCP Transmission Control Protocol

TLD Top-level Domain

TLS Transport Layer Security

ToR The Onion Router

UDP User Datagram Protocol

UniRec Unified Record

URL Uniform Resource Locator

UUID Universally Unique Identifier

XML Extensible Markup Language

Příloha B

Ukázka IDEA zprávy


```
{
  "ID": "3a78c339-df09-4ff8-b36c-02a4fd27734e",
  "Category": [
    "Suspicious.Miner"
  ],
  "Node": [
    {
      "AggrWin": "00:05:00",
      "SW": [
        "Nemea",
        "blacklistfilter"
      ],
      "Type": [
        "Flow",
        "Blacklist"
      ],
      "Name": "undefined"
    }
  ],
  "EventTime": "2019-01-07T14:52:02Z",
  "Description": "51.15.65.182 (listed: Cryptoioc Miners)
                  communicated with 147.228.42.175.",
  "Format": "IDEA0",
  "CeaseTime": "2019-01-07T14:52:54Z",
  "CreateTime": "2019-01-07T14:55:56Z",
  "Note": "IP '51.15.65.182' was found on blacklist:
           Cryptoioc Miners",
  "Source": [
    {
      "InFlowCount": 1,
      "Proto": [
        "tcp"
      ],
```

```
      "OutPacketsCount": 0,
      "OutFlowCount": 0,
      "OutByteCount": 0,
      "InByteCount": 480,
      "InPacketsCount": 7,
      "IP4": [
        "51.15.65.182"
      ],
      "Type": [
        "PoolServer"
      ],
      "Port": [
        14433
      ]
    },
    {
      "Type": [
        "Miner"
      ],
      "IP4": [
        "147.228.42.175"
      ],
      "Proto": [
        "tcp"
      ]
    }
  ],
  "ByteCount": 480,
  "FlowCount": 1,
  "PacketCount": 7,
  "DetectTime": "2019-01-07T14:52:54Z",
  "Ref": [
    "https://cryptoioc.ch/downloads/csv"
  ],
}
```


Popis vybraných polí

- `Category` — kategorie detekované události (je uvedena vždy v konfiguraci blacklistu)
- `Node` — popis detektoru, který událost nahlásil
- `Source` — tzv. *Source of trouble*, tedy entita/entity, které jsou spojeny s nahlášenou událostí (nejedná se tedy o „zdroj“ komunikace). V poli Source je tedy uvedena jednak entita z monitorované sítě, a jednak veřejná entita, která byla nalezena na blacklistu. Administrátorům sítí jsou tyto události odesílány emailem, aby mohli událost investigovat.
- definici ostatních polí lze nalézt v [36]

Příloha C

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF