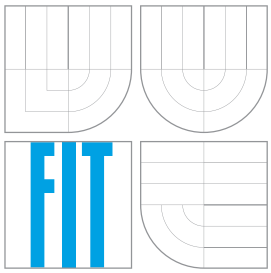
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

DETEKCE A AUTOMATICKÁ ANALÝZA SKENOVÁNÍ SÍTÍ

DETECTION AND AUTOMATIC ANALYSIS OF NETWORK SCANS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — ALEŠ PROCHÁZKA
AUTHOR

VEDOUCÍ PRÁCE — Ing. PAVEL KROBOT
SUPERVISOR

BRNO 2016

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2015/2016

Zadání bakalářské práce

Řešitel: **Procházka Aleš**
Obor: Informační technologie
Téma: **Detekce a automatická analýza skenování sítí**
Detection and Automatic Analysis of Network Scans
Kategorie: Počítačové sítě

Pokyny:

1. Nastudujte problematiku síťových skenů a způsobů jejich detekce spolu s různými možnostmi analýzy cílů útoků či útočníků.
2. Seznamte s projektem Nemea vyvíjeným sdružením CESNET.
3. Navrhněte metodu pro detekci skenovacích útoků a jejich analýzu.
4. Implementujte navržené řešení jako modul do systému Nemea a ověřte je na datech z reálné sítě.
5. Zhodnoťte dosažené výsledky.

Literatura:

- Dle pokynů vedoucího.

Pro udělení zápočtu za první semestr je požadováno:

- Splnění bodů 1 až 3 zadání.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Krobot Pavel, Ing.**, UPSY FIT VUT
Konzultant: Bartoš Václav, Ing., UPSY FIT VUT
Datum zadání: 1. listopadu 2015
Datum odevzdání: 18. května 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.

doc. Ing. Zdeněk Kotásek, CSc.
vedoucí ústavu

Abstrakt

Tato bakalářská práce se věnuje monitorování počítačových sítí s využitím toků. Je zde popsán framework Nemea, který lze využít k sestavení komplexního systému pro detekci síťových útoků a jehož součástí je modul vyvíjený v rámci této práce. Dále je popsána problematika skenování portů a různé metody, jimiž lze porty skenovat. Modul je navržen pro detekci horizontálního skenování. Základní myšlenkou metody je porovnání unikátního počtu cílových IP adres, na nichž se bylo doptáváno na daný port, se zadaným prahem v určitém časovém okně. V praktické části je představena implementace tohoto modulu a jsou prezentovány výsledky experimentů nad reálnými daty ze sítě Cesnet.

Abstract

This bachelor thesis is focused on a computer network monitoring that utilizes flows. Firstly, there is a framework Nemea described, which can be used to build a complex system for network attack detection, and whose module is developed within the thesis. Secondly, port scanning is explained and different methods that can be used to scan ports are defined. The module is designed to detect horizontal scanning. The idea behind this method is to compare a unique number of destination IP addresses, which were asked for with a specific port, with a given threshold in a specific time window. Finally, in the practical part of the thesis the implementation of the module is described and results of the experiments on real data from Cesnet are presented.

Klíčová slova

Nemea, NetFlow, skenování portů, detekce skenování portů, horizontální skenování

Keywords

Nemea, NetFlow, port scanning, portscan detection, horizontal scans

Citace

PROCHÁZKA, Aleš. *Detekce a automatická analýza skenování sítí.* Brno, 2016. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Krobot Pavel.

Detekce a automatická analýza skenování sítí

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Pavla Krobota. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Aleš Procházka
17. května 2016

Poděkování

Rád bych poděkoval svému vedoucímu Ing. Pavlu Krobotovi za vstřícný přístup, cenné rady a hlavně trpělivost při tvorbě této bakalářské práce.

Obsah

1 Úvod **3**

2 Monitorování sítě **4**
2.1 IP tok 4
2.2 NetFlow 5
2.3 IPFIX 5

3 Skenování portů **7**
3.1 Port 7
3.2 Princip skenování 8
3.3 Rozdělení skenování 8
3.4 Navázání TCP spojení 9
3.5 Metody skenování 10

4 Nemea **13**
4.1 Moduly 13
4.2 Rozhraní 13
4.3 Knihovna TRAP 14
4.4 Záznamy UniRec 14

5 Návrh modulu **16**
5.1 Datové struktury 16
5.2 Sběr dat 17
5.3 Vyhodnocení – detekce 17
5.4 Analýza 17

6 Implementace modulu **19**
6.1 Struktura programu 19
6.2 Implementace datových struktur 20
6.3 Vstup programu 21
6.4 Implementace sběru dat 21
6.5 Výpočet časového okna 22
6.6 Implementace detekce 22
6.7 Implementace analýzy 22
6.8 Výstup programu 24

7 Vyhodnocení a výsledky **25**
7.1 Data z Nemea kolektoru 25
7.2 Zhodnocení experimentů 29
7.3 Kontrola paměti 29

8 Závěr **31**

Literatura **32**

Přílohy **34**
Seznam příloh 35

A Náhled heat-mapy **36**

B Obsah CD **38**

Kapitola 1

Úvod

S přibývajícím počtem zařízení připojených do sítě Internet roste i četnost útoků na tato zařízení. Proto je důležité řešit otázku bezpečnosti internetového připojení. Většině síťových útoků předchází fáze průzkumu zvaná skenování portů. Touto technikou může potencionální útočník zjistit určité zranitelnosti systému a využít jich ve svůj prospěch při síťovém útoku. Z tohoto důvodu je vhodné detekovat skenování portů a zabezpečit tak dané zařízení, čímž se dá předejít takovým nepříjemnostem, jakým můžou být různé viry či trojské koně v systému.

V kapitole 2 bude popsáno, co je to monitorování sítě, co je to IP tok a jak pracuje technologie Netflow. V kapitole 3 se čtenář dozví, co je to port, princip skenování, základní rozdělení a vybrané metody, které se používají pro skenování portů. Kapitola 4 se zmiňuje o frameworku Nemea, jehož součástí je modul vyvíjený v rámci této práce. Kapitola 5 popisuje návrh modulu pro detekci horizontálního skenování, zejména návrh struktur pro ukládání záznamů, jednotlivé fáze běhu modulu a návrh způsobu analýzy. Kapitola 6 popisuje implementaci navrženého modulu. Předposlední kapitola 7 se zabývá vyhodnocením a výsledky při testování modulu na reálných datech v síti Cesnet a poslední kapitola popisuje výsledky experimentů a navrhuje další postup vývoje modulu.

Kapitola 2

Monitorování sítě

V dnešní době je monitorování plně funkční sítě tím nejdůležitějším prvkem sítě. Monitorováním sítě se myslí systém, jenž nepřetržitě monitoruje síť. Takový systém je schopen detekovat a ohlásit selhání zařízení nebo připojení. Obvykle měří využití procesoru hostitelů, využití šířky pásma linek a další aspekty provozu. Monitorování tak napomáhá při řešení bezpečnosti sítě a dovoluje detekovat různé síťové anomálie [11].

V následující podkapitole 2.1 bude popsáno, co je to IP tok. Dále možnost jeho zachytávání za pomoci protokolu NetFlow. V poslední podkapitole 2.3 bude popsán standard vycházející z NetFlow, zvaný IPFIX.

2.1 IP tok

IP tok je definován jako jednosměrná posloupnost IP paketů se stejnými vlastnostmi procházející pozorovaným bodem v určitém časovém období. Stejnými vlastnostmi se rozumí zdrojová a cílová IP adresa, zdrojový a cílový port a číslo protokolu. Jelikož je tok jednosměrný, v TCP spojení musí být dva toky (jeden od zdroje k cíli a druhý z cíle ke zdroji) [12].

Tok vzniká zachycením prvního paketu, pro něhož nebyl rozpoznán žádný existující tok. S příchodem dalších paketů patřících do stejného toku se záznam aktualizuje přidáním počtu paketů v toku, počtu oktetů a délky trvání toku [12]. Informace o toku se ukládá do vyrovnávací paměti (*angl. cache*) [5].

K uzavření toku může dojít několika způsoby:

- Dlouho neaktivní tok (*angl. inactive timeout*) – žádný nový paket nebyl v určitém čase přijat pro tok (výchozí hodnota časového limitu je 15 vteřin).
- Trvání toku překročilo aktivní časovač (*angl. active timer*) – výchozí hodnota časovače je 30 minut.
- TCP příznak naznačuje ukončení TCP spojení, čímž bude ukončen i tok (například příznaky FIN nebo RST).
- Zaplnění vyrovnávací paměti (čemuž by se mělo předcházet).

2.2 NetFlow

NetFlow je otevřený protokol vyvinutý společností Cisco[1] pro sledování IP toků. Toto sledování umožňuje podrobný přehled o provozu sítě v reálném čase, zároveň tak výrazně nenarušuje soukromí uživatelů, jak by narušovalo sledování obsahu. Technologie NetFlow je součástí operačního systému Cisco IOS, jež je instalován na síťových prvcích (směrovače, přepínače) vyvíjených firmou Cisco. To s sebou přináší výhodu v podobě nepotřebnosti dokupovat externí sondy [4].

Nabízí se hned několik možností využití:

- **Monitorování** – téměř v reálném čase lze získávat informace o využití směrovačů v rámci aktuální propustnosti a o procházejících tocích.
- **Plánování sítě** – NetFlow technologie může být využita k získávání dat v dlouhém časovém měřítku. Vyhodnocení těchto dat naznačí, které síťové prvky jsou velmi zatěžovány a měly by být optimalizovány.
- **Bezpečnostní analýza** – data z toků jsou využita pro detekci virů, červů a útoků odepření služby (*angl. denial of service*) v reálném čase. Změny chování sítě ukazují anomálie jasně prokázané v NetFlow datech.
- **Účtování** – NetFlow technologie také umožňuje internetovým poskytovatelům účtovat zákazníky na základě množství přenesených dat.

Architektura

Architektura NetFlow se typicky skládá z několika NetFlow exportérů a jednoho NetFlow kolektoru. Nicméně je možné se setkat i s případy, kdy jednotlivé exportéry zasílají data více kolektorům a ty zase přijímají záznamy od více exportérů (tzv. propojení M:N).

Exportér je síťové nebo programové zařízení monitorující provoz na sledované síti. Vytváří záznamy o tocích z jednotlivých přijatých a zpracovaných paketů. Záznamy jsou ukládány do vyrovnávací paměti. U každého záznamu jednou dojde k expiraci (okolnosti expirace již byly popsány v podkapitole 2.1) a následnému exportu do příslušného kolektoru. Jednotlivé údaje jsou přenášeny pomocí UDP protokolu.

Kolektor je zařízení přijímající jednotlivé NetFlow pakety z exportérů a ukládající je do databáze nebo do binárních souborů na disk. Podle typu kolektoru je možné nad daty provádět další operace (např. vizualizace). Jelikož je pro přenos dat využíván UDP protokol, není zajištěn spolehlivý přenos a může dojít ke ztrátě dat. Nicméně kolektor nemá možnost si zažádat o opakované odeslání ztracených dat.

2.3 IPFIX

IPFIX (Internet Protocol Flow Information Export) je protokol vyvíjený organizací IETF (Internet Engineering Task Force). Základem pro standard IPFIX bylo NetFlow verze 9. Oproti NetFlow lze u IPFIX zvolit, jaké položky budou tok identifikovat (je možné použít

[1] Firma zabývající se vývojem síťových prvků (např. směrovače, přepínače, IP telefony, atd.). Viz http://www.cisco.com/

klasickou pětici, jako u NetFlow, ale lze vybrat i další atributy) [6]. IPFIX je tak více flexibilní oproti NetFlow verze 9, navíc je NetFlow standard proprietární, kdežto IPFIX je otevřený standard. U IPFIX mohou být jednotlivé záznamy z exportérů přenášeny také za pomoci protokolů TCP a SCTP (Stream Control Transmission Protocol)[2].

[2]Transportní protokol navržený IETF, primárně určený pro přenos telefonní signalizace po IP

Kapitola 3

Skenování portů

Skenování portů je metoda vzdáleného testování síťových portů ke zjištění, v jakém stavu se port nachází. Účelem této praktiky je nalezení otevřených portů. Otevřený port je stav, kdy spuštěna služba naslouchá a přijímá připojení na daném portu[10]. Tyto získané informace o vzdálené stanici může útočník dále využít ve svůj prospěch, například v aplikování dalších typů útoků na dané služby.

Ke skenování portů je k dispozici mnoho technik. Ty nejpoužívanější a nejznámější jsou uvedeny v podkapitole 3.5.

3.1 Port

Síťový port je speciální číslo sloužící při komunikaci pomocí protokolů TCP a UDP k rozeznání aplikace v rámci počítače. Existuje 65536 portů, které lze rozdělit do tří skupin podle jejich rozsahu [17]:

- **Dobře známé porty** (*angl. well known ports*) – porty v rozsahu 0 až 1023, jsou to tzv. rezervované porty. Jsou kontrolovány a přidělovány organizací IANA[1]. Na většině systémech mohou být užívány jen systémovými procesy nebo programy spuštěnými s dostatečným oprávněním. Jsou to například porty: 21 (FTP), 22 (SSH), 23 (telnet), 53 (DNS), 80 (HTTP), 110 (POP3).
- **Registrované porty** – rozsah portů 1024 až 49151. IANA nepřiděluje tyto porty, pouze registruje jejich použití. Ve většině případů mohou být využívány procesy běžných uživatelů, tzn. nejsou potřeba žádná speciální práva. Například: 1194 (OpenVPN), 3306 (MySQL).
- **Dynamické a soukromé porty** – rozsah 49152 až 65535, vyhrazené pro soukromé využití, nejsou pevně přiděleny žádné aplikaci.

Porty také mohou být, z pohledu klienta, v jednom ze tří stavů, které určují, zda pomocí daného portu může dojít ke komunikaci:

- **Otevřený port** – na daném portu se vyskytuje určitá služba, jež může být použita pro komunikaci.

[1] Organizace, která dohlíží celosvětově na přidělování IP adres, správu kořenových zón DNS a další náležitosti internetových protokolů. Více viz. http://www.iana.org/

- **Zavřený port** – na daném portu (pokud je stanice aktivní) není přístupná žádná služba ke komunikaci.
- **Filtrovaný port** – nedorazila žádná odpověď. Důvodem může být filtrování požadavku nebo odpovědi firewallem. Jedná se spíše o podskupinu zavřeného portu.

3.2 Princip skenování

Velká část síťových služeb funguje na principu klient – server, kde serverová část běží nepřetržitě a nezávisle na kontaktu s uživateli. Klienti se připojují k serverům a zasílají své dotazy (požadavky), na které servery odpovídají. Aby se mohl klient k serveru připojit, musí server naslouchat na nějakém portu a port musí být na serveru v otevřeném stavu. Komunikace probíhá typicky za pomoci protokolů TCP a UDP.

V obou případech lze simulovat klienta, který se tváří že má o službu zájem. Využívá se k tomu speciální software (např. nmap[2]) [16]. Pomocí takových aplikací ze sebe lze vytvořit klienta pokoušejícího se připojit na port vzdáleného serveru. Podle úspěchu či neúspěchu spojení se pozná, zda port je v otevřeném nebo zavřeném stavu. Jednotlivé metody, které se k tomu používají budou popsány v podkapitole 3.5.

3.3 Rozdělení skenování

Skenování portů se dělí do tří základních typů založených na vzoru cílových destinací a portů, které objevila osoba provádějící skenování [9]. Následuje popis jednotlivých typů.

Vertikální skenování

Vertikální skenování je skenování, které se zaměřuje na porty na jednom hostiteli, tzn. mění se pouze cílový port a cílová IP adresa zůstává stejná. Útočník se tak snaží zjistit dostupné služby na jednom konkrétním cíli. Tento typ patří mezi lehce detekovatelné, jelikož je potřeba pouze mechanismus lokální detekce. Pro odhalení tedy postačuje mít spuštěný detekční program na dané stanici.

Horizontální skenování

Horizontální skenování je skenování, kdy se útočník zaměří na stejný port na několika koncových zařízeních, tzn. cílový port se nemění, ale cílová IP adresa ano. Nejčastěji si je útočník vědom určité zranitelnosti a doufá v nalezení zranitelného stroje. Velmi často se jedná o skenování rozsahu konkrétní sítě. V tomto typu skenování již nepostačuje mechanismus lokální detekce, a to právě z důvodu skenování nějakého rozsahu sítě. K detekci by mohl být využit detekční program na zařízení před sledovanou sítí, tedy na takovém, ke kterému jsou připojena zařízení, jejichž toky budou sledovány.

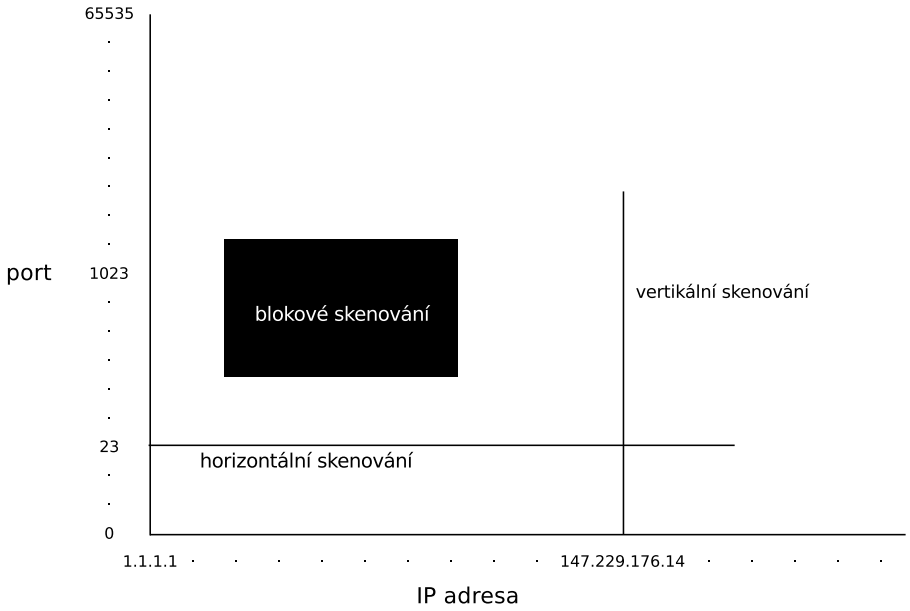
Blokové skenování

Blokové skenování je kombinace předchozích dvou typů, kdy je skenováno určité rozmezí portů nebo kompletní rozsah portů na určitých stanicích. Cílem je následně si vybrat některou podmnožinu poskytovaných služeb na několika stanicích k dalšímu možnému útoku.

[2] viz. https://nmap.org/

Detekce by mohla probíhat stejně jako v případě horizontálního skenování.

Pro lepší představu a pochopení rozdílů mezi typy skenování slouží následující obrázek 3.1. Než budou představeny jednotlivé metody využívané ke skenování portů, je potřeba vědět, jak probíhá navázání spojení. Je to nutné zejména pro podmnožinu skenování využívající TCP protokol. Tomuto tématu se věnuje následující podkapitola 3.4



Obrázek 3.1: Rozdělení skenování

3.4 Navázání TCP spojení

TCP je na rozdíl od UDP spojově orientovaný protokol, tzn. přenosu dat musí předcházet navázání spojení. TCP je označován za spolehlivý protokol, jelikož příjemce oznamuje odesílateli, že doručení selhalo. UDP je nespolehlivý protokol, protože takový potvrzovací mechanismus neobsahuje, ale to (nepotvrzování přijatých dat) ho činí protokolem rychlejším než TCP.

K ustavení TCP spojení se využívá třícestný handshake (*angl. three-way handshake*). Během zahajování spojení se obě strany dohodnou na sekvenčním čísle v TCP hlavičce. Toto číslo se používá pro číslování oktetů. Jelikož jsou číslovány všechny oktety, nabízelo by se potvrzování každého oktetu. Nicméně potvrzovací mechanismus je kumulativní, aby se nemuselo potvrzovat každý oktet zvlášť, což by mělo za následek větší zatížení sítě a ještě větší zpomalení přenosu. Potvrzení sekvenčního čísla X znamená potvrzení všech oktetů do čísla o jedno menší než X. Číslo prvního oktetu je určeno při ustavení spojení (vygenerováno náhodně) a následně se zvyšuje podle počtu oktetů, jež byly odeslány po ustavení spojení [13].

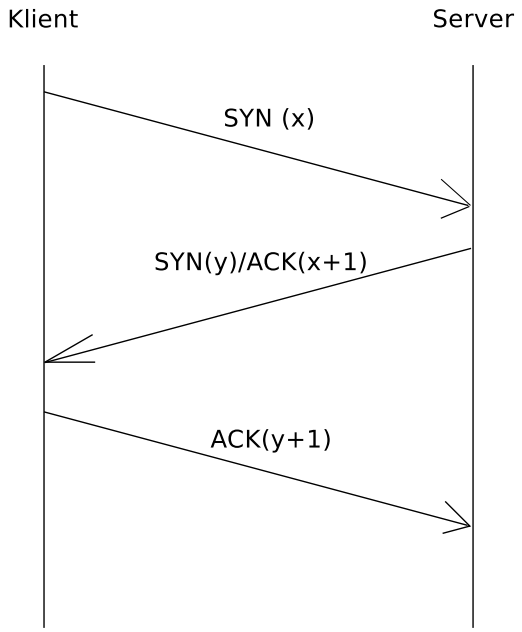
TCP segment obsahuje bitové pole příznaků, pomocí kterých nastavuje a řídí dané spojení. Jedná se o některé z 8 bitových hodnot – CWR (Congestion Window Reduced), ECE (ECN-Echo), URG (Urgent), ACK (Acknowledgement), PSH (Push), RST (Reset), SYN

(Synchronize), FIN.

Navázání spojení probíhá ve třech krocích [3]:

1. klient pošle paket s nastaveným příznakem SYN s náhodným číslem sekvence (x), číslo odpovědi 0
2. druhá strana si uloží číslo sekvence (x) a odpoví SYN/ACK, jako číslo sekvence nastaví své číslo (y) a do čísla odpovědi vloží (x+1) – další očekávanou hodnotu
3. klient odpoví ACK, číslo sekvence (x+1), číslo odpovědi (y+1)

Pro lepší pochopení a názornost slouží následující schéma na obrázku 3.2.



Obrázek 3.2: Schéma ustavení TCP spojení

Již víme jak probíhá navázání spojení, můžeme tedy přejít k vysvětlení jednotlivých metod skenování popsaných v následující podkapitole 3.5.

3.5 Metody skenování

Díky složitosti rodiny protokolů TCP/IP je možné využít několika metod ke skenování. V následujících podsekcích budou popsány vybrané metody. Čerpáno bylo ze zdrojů [7, 8, 10, 15].

TCP connect() skenování

Tato metoda je standardní cesta k vytvoření TCP spojení. Systémové volání `connect()`[3] je používáno k provedení *three-way handshake* a navázání spojení s cílovým zařízením. Ustavení spojení na portu indikuje, že odpovídající port je otevřen. Pokud by byl port uzavřený, spojení se nezdaří a byl by vrácen chybový kód. Výhodou metody je nepotřebnost speciálních práv pro uživatele. Nicméně z pohledu útočníka je to nevýhodné v odhalitelnosti jeho IP adresy, neboť si servery mohou do speciálních souborů ukládat informace o připojení, přičemž tyto záznamy mohou pomoci odhalit útočníka. Další možná detekce je analýza síťového provozu, kdy zdroj zasílá mnoho SYN a ACK paketů, ale žádná další data po ustavení spojení nezasílá.

TCP SYN skenování

V porovnání s předchozí metodou zde nedochází k ustavení TCP spojení. Pouze proběhne první krok *three-way handshake*, tj. zaslání SYN paketu konkrétní službě cílového zařízení. Pokud přijde odpověď ve formě SYN+ACK paketu, útočník vidí, že na patřičném portu naslouchá služba a následně nepotvrzuje přijetí paketu.

Servery si ukládají informace pouze při úspěšném navázaní spojení, což se v této metodě neděje. Detekce je možná za pomoci síťové analýzy, když ze zdroje přichází velké množství SYN paketů, přičemž dále nepotvrdí přijetí SYN+ACK paketu paketem s nastaveným příznakem ACK.

TCP FIN skenování

FIN skenování taktéž nenavazuje legitimní TCP spojení. Cílené oběti je zasílán FIN paket. Pokud oběť odpoví RST paketem, značí to uzavřený port. Otevřené ale i filtrované porty jednoduše ignorují tuto zprávu [8]. Odhalení a detekce je možná stejně jako u předchozí metody, tedy systémovým nástrojem.

TCP ACK skenování

Toto skenování nikdy neurčí otevřené porty. Většinou se užívá k určení filtrovaných portů. Při této metodě je nastaven jen ACK příznak. Při skenování systémů, jež nejsou chráněny firewallem, otevřené a zavřené porty odpoví RST paketem. Označeny jsou jako nefiltrované z důvodu nejasnosti, jestli se jedná o otevřený nebo zavřený port. Neodpovídající porty nebo porty zasílající chybové zprávy ICMP jsou označeny jako filtrované.

TCP Window skenování

Window skenování používá nesrovnalosti v TCP/IP stacku některých systémů k získání informací o cílovém zařízení. Principem je tato metoda podobná TCP ACK skenování. Na základě velikosti TCP okna[4] se snaží odvodit, zda je port otevřený či uzavřený. Na některých systémech, otevřený port používá kladnou velikost okna, zatímco okno velikosti 0 značí uzavřený port. Je poměrně nespolehlivé při zjišťování otevřenosti nebo uzavřenosti portu, neboť záleží na implementačních detailech TCP stacku na daném systému.

[3] Vytvoření nového TCP spojení

[4] Okno má určitou velikost, která představuje objem dat přenesený do potvrzení.

TCP Christmas Tree skenování

Christmas Tree (známé také jako XMAS) skenování posílá paket s nastavenými příznaky URG, FIN a PUSH. Pokud je port otevřený nebo filtrovaný, tuto zprávu ignoruje, kdežto uzavřené porty odpoví paketem RST.

TCP null skenování

Null skenování je obdobné jako XMAS popsané výše. Rozdíl je v tom, že při této technice nejsou nastaveny žádné bity příznaků, tj. TCP hlavička příznaků je 0. Pokud je port uzavřený, odešle odpověď RST.

UDP ICMP skenování

UDP ICMP skenování je technika zaměřená na skenování UDP portu. Není funkční ve všech případech, kvůli nespojově orientovanému protokolu UDP. Využívá *ICMP port unreachable* zprávy k detekci naslouchajících portů. Typický skener portů zašle prázdný paket na vybraný port cílového zařízení a čeká na odpověď. Přijetí zprávy *ICMP port unreachable* naznačuje, že vybraný port je uzavřený. Žádná odpověď většinou znamená otevřený nebo filtrovaný port.

UDP recvfrom() a write() skenování

Tato metoda využívá dvou systémových volání (`recvfrom()`[5] a `write()`[6]) a znalost BSD schránek (*angl. socket*)[7] k detekci otevřených a zavřených UDP portů na cílovém zařízení. Pro práci s ICMP sokety jsou potřeba administrátorská práva. Toto lze obejít pomocí již zmíněných dvou funkcí. Druhé volání funkce `write()` pro přístup k zavřenému portu skončí chybou. Volání `recvfrom()` při neblokujících UDP operacích vrátí chybu 13 (zkus znovu), pokud ICMP chyba nebyla přijata, a chybu 111 (spojení zamítnuto), pokud ICMP chyba přijata byla. Stejně jako v předchozím případě není tato technika příliš spolehlivá.

[5] Příjem zprávy z BSD schránek
[6] Zápis do schránky.
[7] Mechanismus pro přístup k službám protokolů transportní vrstvy.

Kapitola 4

Nemea

Nemea (Network Measurements Analysis) je framework vyvíjený sdružením CESNET, který umožňuje sestavení systému pro automatickou analýzu záznamů toků získaných od procesů monitorování sítě v reálném čase. Systém sestává z oddělených bloků zvaných moduly. Jednotlivé moduly jsou propojeny pomocí rozhraní. Komplexní systém pro analýzu síťového provozu v reálném čase je uspořádán propojením několika modulů. Informace o Nemea frameworku jsou čerpány z technické zprávy [1]

4.1 Moduly

Každý modul je samostatná aplikace běžící jako oddělený systémový proces. I když se zdá toto řešení méně efektivní než ucelený systém, běžící v rámci jednoho procesu, nabízí hned několik výhod. Například je jednoduché spouštění a zastavování procesů nezávisle na sobě a sledování jejich využití zdrojů. Také je možné psát moduly v různých programovacích jazycích, dokonce i v interpretovaných (např. Python), jen musí být napsána „obálka“ nad knihovnou TRAP (knihovně se věnuje podkapitola 4.3). V neposlední řadě stojí za zmínku, že tento způsob řešení modulu jako samostatného procesu nevyvolá pád celého systému, když dojde v nějakém modulu k chybě.

Moduly mohou být do systému přidány a odstraněny dynamicky. Když je nový modul spuštěn, snaží se připojit na specifikované rozhraní jiného modulu. Pokud není spojení možné, periodicky se snaží o připojení, dokud není navázáno. Stejné je to i v případě, kdy je spojení ztraceno.

4.2 Rozhraní

Všechna rozhraní jsou jednosměrná a přenáší data ve formě jednotlivých záznamů. Veškerá data zaslaná přes jedno rozhraní musí být stejného formátu, tzn. je potřeba, aby stejný formát byl na obou koncích (rozhraních) daného spojení.

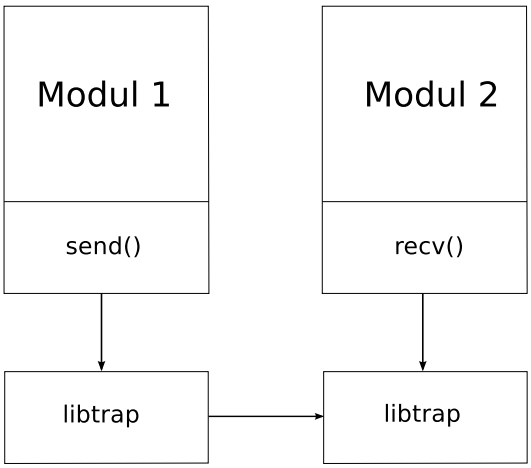
Formát užívaný daným rozhraním je specifikován dynamicky při připojení modulu do systému. To umožňuje přidat nové pole do flow záznamu bez nutnosti měnit kód modulů zpracovávajících tyto záznamy. Protokol, který určuje definici těchto formátů a vytváření a používání záznamů se nazývá UniRec. Více informací o tomto protokolu bude uevedno v kapitole 4.4.

Implementaci rozhraní pro komunikaci využívané systémem Nemea poskytuje *Traffic*

Analysis Platform (TRAP) – sdílena knihovna zvaná libtrap. Knihovna TRAP bude popsána v kapitole 4.3

4.3 Knihovna TRAP

Nemea moduly využívají společné komunikační rozhraní, které je reprezentováno sdíleným objektem zvaným *libtrap*, jež je linkovaný každým modulem. Koncept komunikace dvou modulů je vyobrazen na obrázku 4.1.



Obrázek 4.1: Princip komunikace dvou modulů. Převzato z [1].

Zdrojový modul zasílá data výstupním rozhraním hned jak je to možné. Cílový modul čte data ze vstupu v nekonečné smyčce, ale čtení je blokováno, dokud nedorazí nějaká data.

4.4 Záznamy UniRec

Unified Record (UniRec) je specifický datový formát zpráv zasílaných přes TRAP rozhraní. Moduly si musí vyměňovat různé typy dat a množství může být velmi velké. Z toho vyplývají tři požadavky, které jsou reflektovány v UniRec:

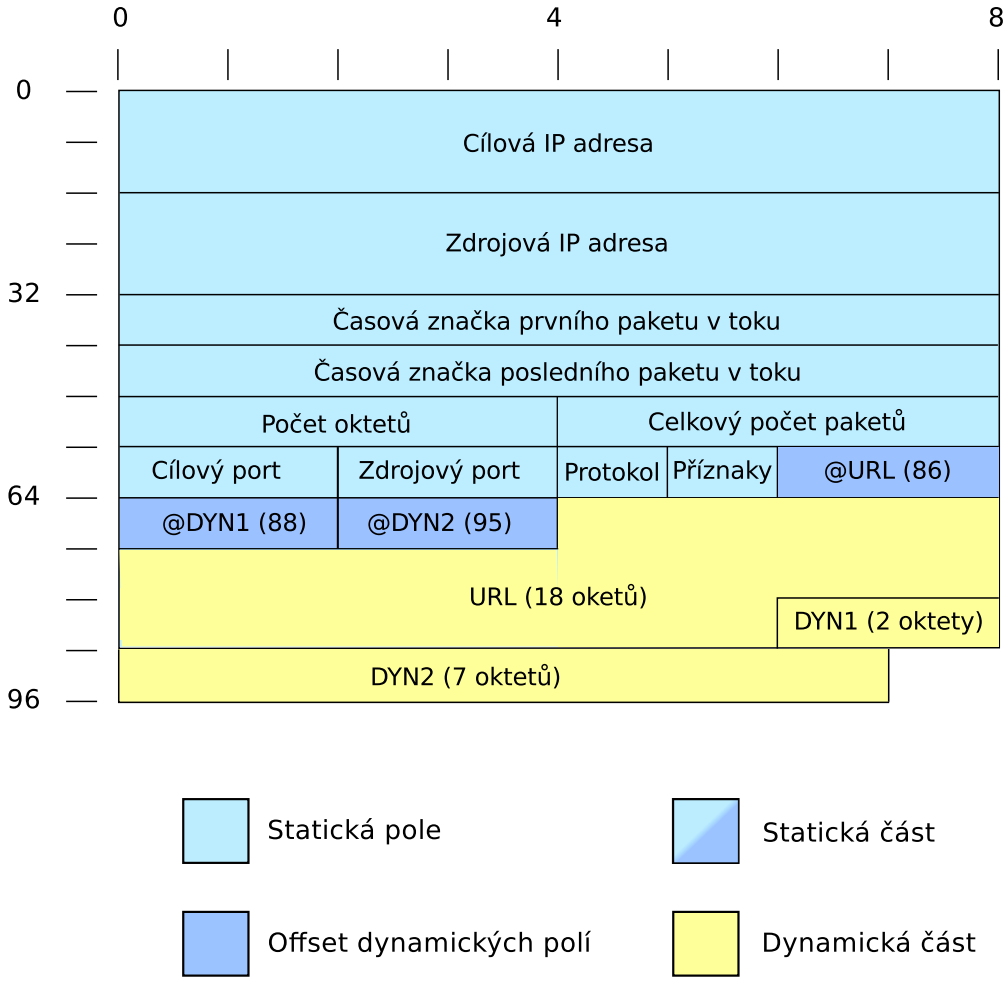
- Formát musí být flexibilní (mělo by být možné dynamicky vytvářet nové struktury zpráv).
- Formát by měl být paměťově efektivní (nízká paměťová náročnost).
- Manipulace s daty musí být rychlá.

Každý záznam UniRec se skládá z několika polí (každé má své jméno a typ). Sada polí se nazývá šablona a je určena výčtem jmen všech jejích polí. Seznam všech dostupných polí (společně s jejich typy a významem) je uveden v globálním konfiguračním souboru. Všechna TRAP rozhraní užívají přesně jednu šablonu. Při spojení dvou modulů musí oba používat stejný formát UniRec zprávy. Předloha každého rozhraní je definována během inicializace modulu poskytnutím řetězce se jmény všech položek v záznamu. Zatímco formát výstupního rozhraní je obvykle dán funkcí modulu, mnoho modulů umožňuje určit očekávaný vstupní formát pomocí parametrů příkazového řádku. Takto nastavený modul je možné napojit

na jiný, který má na výstupu nadmnožinu polí spouštěného modulu. Není možné nastavit formát zcela libovolně (pole, s nímž má modul pracovat, musí být přítomna).

Záznamy jsou velmi podobné jednoduchým strukturám z jazyka C - jednotlivé položky jsou uloženy ihned za sebou bez dalších dat nebo zarovnání. Záznam UniRec se skládá ze statické části, kdy všechny položky mají pevně danou velikost, a dynamické části, kde položky mají velikost proměnnou a jsou uloženy na konci záznamu za statickou částí. Nejprve jsou však uloženy jejich offsety z důvodu efektivního přístupu k dynamickým položkám.

Příklad, jak UniRec záznam může vypadat je vidět na obrázku 4.2



Obrázek 4.2: Formát zprávy UniRec. Převzato z [1].

Kapitola 5

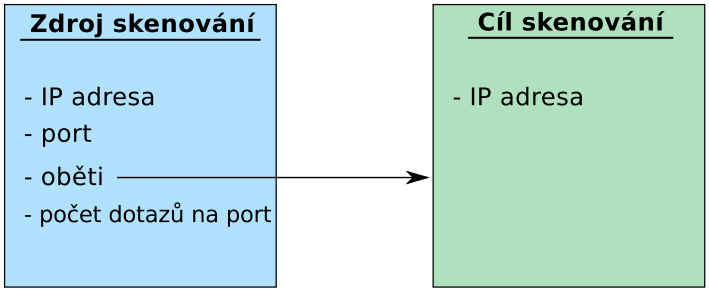
Návrh modulu

V této kapitole bude představen návrh modulu pro detekci horizontálního skenování, jenž je součástí modulárního systému Nemea (popsaného v kapitole 4). Základní myšlenkou metody je porovnání celkového počtu unikátních adres, na kterých se zdroj doptával na konkrétní port, se zadaným prahem (*angl. threshold*) v určitém časovém okně. Modul využívá dvou datových struktur. Struktury budou popsány v podkapitole 5.1. Běh programu probíhá ve dvou fázích – sběr dat a vyhodnocení nasbíraných dat. Následovat bude popis datových struktur a poté detailnější popis jednotlivých fází.

5.1 Datové struktury

Před zahájením detekce skenovaní portů je nutné nasbírat vzorek dat a vhodně ho uložit. Jelikož nejsou potřeba všechny informace z UniRec záznamů, se kterými pracuje systém Nemea, je vhodné ze záznamu vyjmout a uložit jen důležité položky. K uložení dat jsou navrhnuty dvě struktury.

První struktura identifikuje možného útočníka pomocí IP adresy (zdrojová IP adresa) a jím tázaného portu (cílový port). Při každém novém nalezeném dotazu ze stejné IP adresy na stejný cílový port je inkrementováno počítadlo dotazů pro daný pár (tedy zdrojová IP adresa a cílový port). Jelikož je požadavkem na aplikaci i sledování cílů skenování, je potřeba si IP adresy cílů uchovávat (cílová IP adresa). K tomu slouží struktura druhá. Grafické znázornění struktur je na obrázku 5.1. Podrobná implementace bude představena v podkapitole 6.2.



Obrázek 5.1: Navržené datové struktury

5.2 Sběr dat

V tomto kroku probíhá sběr a ukládání dat z jednotlivých přijatých záznamů o tocích. Z každého toku jsou získána data o cílové a zdrojové IP adrese, cílovém portu, použitém protokolu, celkovém počtu paketů a časová značka posledního paketu v toku. K ukládání do připravených struktur dochází za určitých podmínek:

- V toku je použit protokol TCP nebo UDP (podpora UDP se volí parametrem).
- V toku je počet paketů roven nebo menší než tři.

Druhá odrážka vychází ze znalostí popsaných v podkapitole 3.4, tedy že jen k ustavení legitimního TCP spojení jsou potřeba dva pakety. Za normálního provozu není obvyklé, aby tok měl tři nebo méně paketů (např. aby bylo navázáno spojení a následně ihned ukončeno).

Při splnění podmínek tedy dochází k uložení dat. Z posbíraných dat se do struktur ukládají informace o zdrojové IP adrese, cílovém portu a cílové IP adrese. Pokud se potencionální útočník a jím zkoumaný port (tedy zdrojová IP adresa a cílový port) v první struktuře již nachází, je inkrementováno počítadlo unikátních IP adres cílů skenování, na kterých se zdroj dotazoval na daný port. Následně potencionální oběť (cílová IP adresa) přidána do druhé struktury (pokud se tam již nachází, zmíněné počítadlo je dekrementováno). Sběr dat je ohraničen časovým oknem.

Časové okno udává dobu v sekundách, po kterou má probíhat sběr dat. Po jeho uplynutí proběhne nad sesbíranými daty detekce skenování. Po ukončení detekce se spustí nové časové okno. Tento proces se stále opakuje, dokud není odchycen signál `SIGINT` nebo `SIGTERM`. Hodnota časového okna je ovlivnitelná parametrem při spouštění modulu. Výchozí hodnota je nastavena na 5 minut.

Pro lepší názornost, jak probíhá sběr dat slouží vývojový diagram na obrázku 5.2.

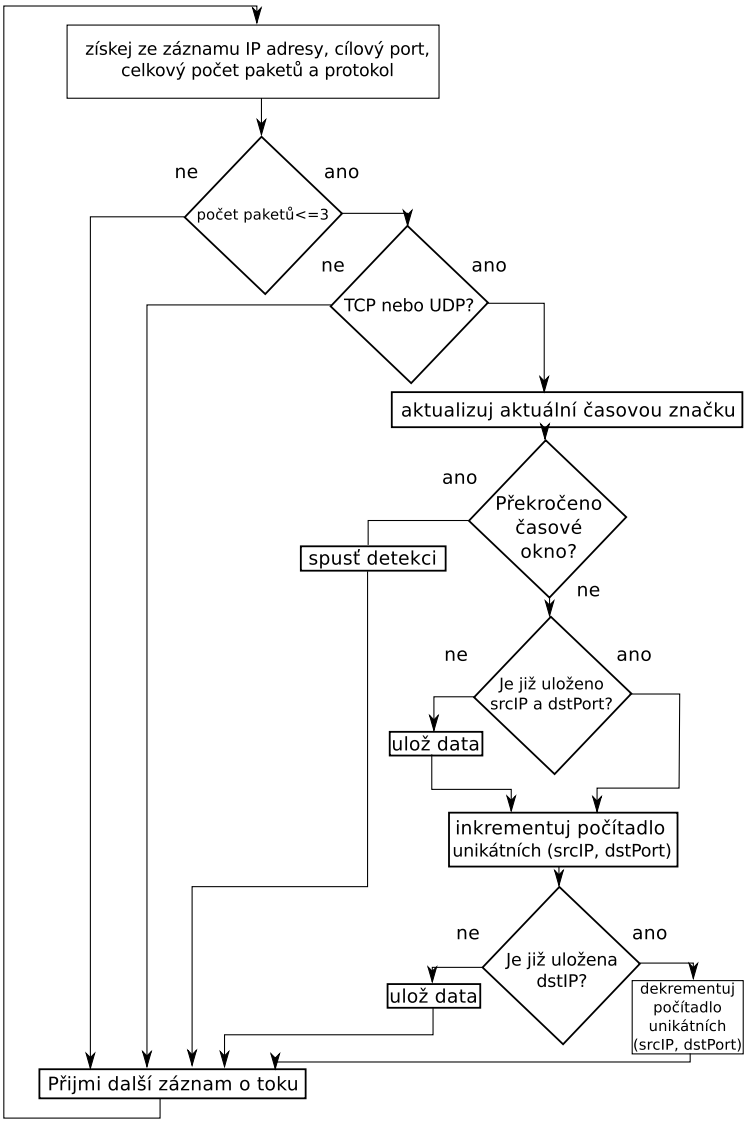
5.3 Vyhodnocení – detekce

Vyhodnocení nasbíraných dat probíhá poté, co časová značka záznamu již nebude v rámci aktuálního časového okna. Procházejí se postupně všechna uložená data a počítadlo dotazovaných unikátních adres je porovnáváno s daným prahem. Hodnota prahu je ovlivnitelná parametrem při spouštění modulu. Výchozí hodnota prahu je nastavena na 100. Pokud počet dotazů na port překročil daný práh, je tento záznam označen za skenování.

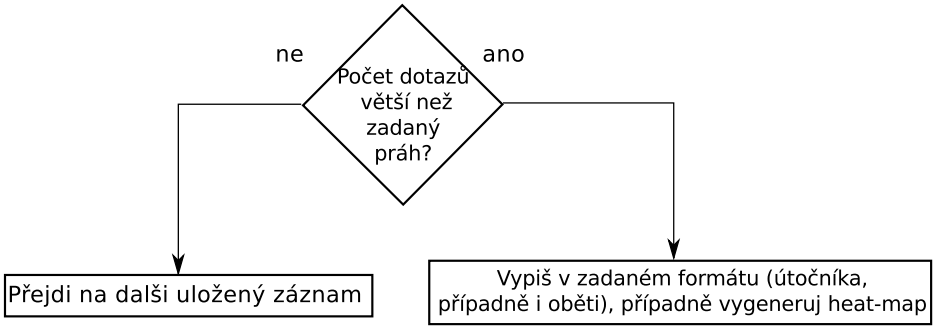
Podle navolených parametrů při spouštění modulu má detekce určité výstupy. Na výběr je mezi dvěma textovými výstupy, k nímž může být vygenerována tzv. *heat-mapa* a jeden výstup do souboru se statistikou nejvíce skenovaných portů a výskytu jednotlivých hodnot v posledním oktetu IP adresy. O výstupech pojednává podkapitola 6.8. Průběh vyhodnocování nasbíraných dat je vyobrazen na následujícím obrázku 5.3.

5.4 Analýza

Analýza probíhá zároveň s vyhodnocením nasbíraných dat. Jedním z hlavních výstupů analýzy je tzv. *heat-mapa*. Další výstupy jsou zejména statistiky. A to hlavně statistika nejvíce skenovaných portů, nejčastěji skenovaných portů a statistika četnosti výskytů jednotlivých hodnot v posledním bytu cílové adresy. Více o analýze bude popsáno v podkapitole 6.7.



Obrázek 5.2: Vývojový diagram sběru dat



Obrázek 5.3: Vývojový diagram vyhodnocení dat

Kapitola 6

Implementace modulu

V této kapitole bude popsána implementace modulu, jehož návrh byl popsán v předchozí kapitole. Jako implementační jazyk byl zvolen jazyk C. Hlavním důvodem bylo, že celý modulární systém Nemea je napsán v jazyce C/C++.

Nejdříve bude popsána struktura programu, dále jak jsou implementovány již dříve popsané struktury. V podkapitole 6.8 budou ukázány možné výstupy a jaké jsou očekávány vstupy. Následovat budou dvě podkapitoly, kde budou vysvětleny implementace sběru a vyhodnocení dat.

6.1 Struktura programu

K tvorbě aplikace bylo využito jednoduchého ukázkového programu[1], který přijímal data ze vstupu, provedl operaci a ihned odeslal na výstup. Tento zdrojový kód byl využit jako šablona, kde byly nachystané inicializace všech důležitých komponent a mohl být tedy rovnou psán kód algoritmu detekce.

Nyní budou popsány jednotlivé prvky. Nejdříve je modul popsán pomocí struktury `trap_module_info_t` (obsahuje například název modulu, slovní popis modulu, počet vstupních a výstupních rozhraní). Následně jsou pomocí makra definovány parametry, jež mohou být použity při spouštění modulu. Pomocí makra `TRAP_DEFAULT_SIGNAL_HANDLER` je nastaveno odchytávání signálů `SIGINT` a `SIGTERM`, při kterých bude provedeno přerušení hlavní smyčky, uvolnění zdrojů a ukončení programu.

Po zavolaní hlavní funkce programu `main()` proběhne inicializace knihovny TRAP makrem `TRAP_DEFAULT_INITIALIZATION`. Makro `TRAP_GETOPT` zpracovává parametry. Pomocí parametrů lze nastavit tyto vlastnosti:

- Délka časového okna (parametr `-w`).
- Velikost prahu (parametr `-t`).
- Podpora protokolu UDP (parametr `-u`).
- Formát výstupu (parametry `-b`, `-c` nebo `-s`). Výstupům se věnuje podkapitola 6.8.
- Možnost generování heat-mapy (parametry `-m` a `-y`). Heat-mapa bude popsána v podkapitole 6.7.

[1] Dostupného v Nemea pod názvem `example_module`

V další části je vytvořena vstupní UniRec šablona, která je reprezentována strukturou `ur_template_t`. Tato struktura obsahuje informace o tom, kde v UniRec najít konkrétní položky. Samotná šablona se vytvoří voláním funkce `ur_create_input_template()`, jíž se předávají názvy jednotlivých položek dané UniRec šablony. Více o očekávaném vstupu bude probráno v podkapitole 6.3.

Nyní již následuje nekonečná smyčka, jež tvoří základ programu – jeden průchod cyklem znamená jedno načtení toku. V rámci této smyčky jsou čtena data ze vstupního rozhraní pomocí makra `TRAP_RECEIVE`. Pokud čtení proběhlo v pořádku, je následně zkontrolována velikost přijatých dat. Když i tato kontrola proběhne v pořádku, může dojít ke zpracování přijatých dat.

6.2 Implementace datových struktur

Pro ukládání dat z přijatých UniRec záznamů byl použit abstraktní datový typ zvaný binární vyhledávací strom[2]. Binární vyhledávací strom je uspořádaný binární strom, pro jehož každý uzel platí, že klíče všech uzlů levého podstromu jsou menší než klíč v uzlu a klíče všech uzlů pravého podstromu jsou větší než klíč v uzlu [2]. Modul konkrétně využívá dva takové stromy. Hlavní (primární) binární vyhledávací strom je typu `scanTree` a uzel tohoto stromu je tvořen strukturou popsanou v podkapitole 5.1, jež obsahuje tyto položky (nejdříve je uveden datový typ a následně název proměnné):

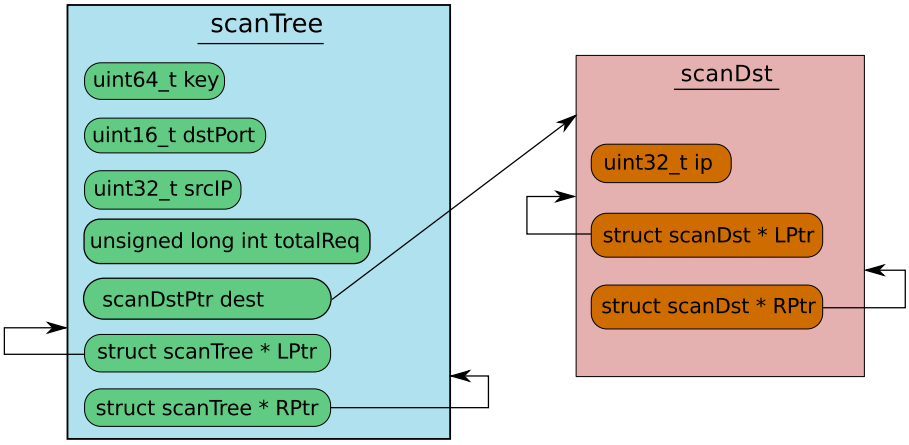
- `uint64_t key` – vyhledávací klíč,
- `uint16_t dstPort` – cílový port (zdrojem dotazovaný port),
- `uint32_t srcIP` – IP adresa zdroje skenování,
- `unsigned long int totalReq` – počítadlo unikátních cílových adres, ne kterých byl dotazován konkrétní port,
- `scanDstPtr dest` – ukazatel na vrchol druhého stromu, který obsahuje všechny IP adresy cílů skenování,
- `scanTree *LPtr` – ukazatel na levého potomka,
- `scanTree *RPtr` – ukazatel na pravého potomka.

Jelikož jsou jednotlivé horizontální skenovací útoky identifikovatelné dvojicí zdrojová IP adresa a cílový port, je klíč do tohoto stromu složen právě z této dvojice. Nejprve se do proměnné vloží IP adresa, ta se operací levý posun (angl. *left shift*) posune o 32 bitů doleva a následně se vloží hodnota cílového portu. Druhý strom, který je součástí toho primárního, je datového typu `scanDst` a jeho uzel se skládá z těchto položek:

- `uint32_t ip` – klíč a zároveň užitečná hodnota (IP adresa cíle skenování),
- `scanDst *LPtr` – ukazatel na levého potomka,
- `scanDst *RPtr` – ukazatel na pravého potomka.

Pro lepší představivost slouží grafické znázornění na následujícím obrázku 6.1.

[2] Binární vyhledávací strom byl využit hlavně kvůli rychlému vyhledávání.



Obrázek 6.1: Grafické znázornění implementovaných struktur

6.3 Vstup programu

Na vstupu modul očekává UniRec záznamy v základním obecném formátu využívaném v systému Nemea[3], který obsahuje tyto položky: zdrojová IP adresa, cílová IP adresa, zdrojový port, cílový port, protokol, celkový počet paketů, celkový počet oktetů, časová značka prvního paketu v toku, časová značka posledního paketu v toku, TCP příznaky, bitové pole (označující, zda byl tok zachycen na odpovídající lince), bitové pole pro určení příchozího nebo odchozího toku, typ IP služby a dobu platnosti dat. Důvod použití tohoto formátu spočívá v tom, že dané kolektory zasílají ostatním modulům záznamy právě v tomto formátu.

6.4 Implementace sběru dat

Sběr dat probíhá v nekonečné smyčce, kde jeden průchod cyklem znamená jeden načtený záznam ze vstupu. Pomocí funkce `ur_get()` jsou z přijatého UniRec záznamu extrahovány jednotlivé položky.

Po uložení načtených položek do proměnných se kontroluje, zda se jedná o IP adresy verze 4[4]. Následně proběhne kontrola protokolu a počtu paketů v toku (jak bylo popsáno v podkapitole 5.2). Pokud časová značka toku spadá do aktuálního časového okna, jsou jednotlivé položky uloženy do struktur. Jedná se o položky:

- zdrojová IP adresa a cílový port (uloženy budou do struktury `scanTree`)
- cílová IP adresa (uložena do vnořené struktury `scanDst`)

V rámci funkce pro přidávání prvků je kontrolováno, zda se tato data již ve strukturách nenacházejí, pokud už jsou ve strukturách obsažené, znovu se nepřidají. Čili záznamy v hlavní struktuře jsou unikátní. Pokud je přidáván nový cíl skenování, inkrementuje se počítadlo unikátních adres cílů skenování, na kterých byl dotazován daný port.

[3]Dle UniRec specifikace COLLECTOR_FLOW
[4]Implementovaný modul ve své první verzi, vyvinuté v rámci této práce, sleduje pouze IP adresy verze 4

6.5 Výpočet časového okna

Jak již bylo zmíněno, velikost časového okna je nastavitelná parametrem programu a jeho výchozí hodnota je 5 minut. Funkčnost časového okna je zajištěna pomocí porovnání časových značek jednotlivých záznamů o tocích.

Při prvním průchodu nekonečnou smyčkou je zaznamenána koncová časová značka prvního toku a je uložena do proměnné `startTime`. Při každém průchodu cyklem je zaznamenávána koncová časová značka do proměnné `actTime`, ale jen v případě, že nově načtená je vyšší než načtená z minulého průchodu, tzn. nevracíme se v čase. Tento mechanismus je důležitý proto, jelikož není zaručeno, že časové značky jednotlivých příchozích záznamů budou s neklesajícími hodnotami. Výpočet časového okna je dán vztahem znázorněným v rovnici 6.1

$$(actTime - startTime) < timeWindowSize \tag{6.1}$$

kde `timeWindowSize` je velikost časového okna ve vteřinách. Pokud podmínka platí, jsou přidávány záznamy do struktur. Jakmile přijde záznam, který do časového okna již nespadá, je spuštěna detekce nad nasbíranými daty. Po detekci se do `timeWindowSize` přičte velikost okna a probíhá nové časové okno.

6.6 Implementace detekce

Detekce probíhá ve funkci `detectAttacks()`, jejíž zavolání proběhne po skončení časového okna. V této funkci jsou postupně procházeny všechny záznamy struktury `scanTree` a porovnávány s daným prahem. Ten je nastavitelný pomocí parametru. Jeho výchozí hodnota je 100[5]. S prahem je porovnáván celkový počet unikátních cílových adres skenování, na kterých se zdroj dotazoval na daný port. Pokud je tato hodnota vyšší nebo rovna danému prahu, je tento záznam ve struktuře označen za skenování. Po porovnání a případném výstupu (výstupy budou popsány v následující podkapitole) je položka struktury smazána a zpracuje se položka další. Po zpracování všech dat je struktura obsahující informace o tocích prázdná. Modul dále pokračuje opětovným naplněním této struktury v nové fázi sběru dat.

6.7 Implementace analýzy

Jak již bylo řečeno v návrhu analýzy (v podkapitole 5.4), výstupem analýzy jsou tzv. *heat-mapa* a statistiky. Zvolit generování mapy nebo statistik je možné pomocí parametrů, které budou popsány v podkapitole 6.8.

Heat-mapa

Heat-mapa zobrazuje četnost výskytu jednotlivých cílů skenování (jejich IP adres). Ukázka, jak může vypadat heat-mapa, je zobrazena na obrázku A.1 v přílohách. Je to obrázek o velkém rozlišení (4096 x 4096 pixelů), kde každý pixel představuje jednotlivé sítě s prefixem

[5] Práh určuje maximální přípustný počet unikátních cílových adres skenování, na kterých se bylo dotazováno na daný port. Při překročení prahu je pak zdroj označen za zdroj skenování

/24. Každému pixelu může být přiřazena jedna z 256 barev. Barva pixelu značí počet hostů v rámci této sítě. Na tuto mapu byly pro lepší orientaci přidány popisky jednotlivých sítí[6].

Pro vykreslení heat-mapy je použit nástroj zvaný *ipv4-heatmap* vyvinutý společností *The Measurement Factory*[7].

Při generování heat-mapy je vytvářen dočasný soubor, kam jsou zapsány jednotlivé IP adresy cílů. Tento soubor je vytvářen kvůli tomu, že nástroj *ipv4-heatmap* potřebuje data na standardní vstup. Po uložení hodnot do souboru se pomocí volání funkce `cmd()` spustí program *ipv4-heatmap* a na jeho vstup je přiveden daný dočasný soubor, který je po vykreslení mapy vymazán.

Důležité! Pro běh tohoto nástroje je nutné mít nainstalovanou knihovnu `libGD`[8], s jejíž pomocí se vykresluje výsledný obrázek.

Statistika posledního oktetu IP adresy

V této analýze jsou zkoumány koncovky, nebo-li poslední oktety IP adres cílů skenování. K počítání koncovek slouží pole celých čísel `lastByteOfIP[256]`, kde indexy pole představují danou koncovku IP adresy. Tedy číslo v poli na indexu 20 představuje počet skenovaných IP adres ve tvaru X.X.X.20. Statistika je tisknuta ve tvaru `hodnota posledního oktetu: počet skenovaných adres s touto koncovkou`. Tisknutí této statistiky probíhá do souboru na konci časového okna. Jméno souboru vzniká spojením předpony `Stat_lastByte_` a aktuálního času, kdy soubor vzniká, tzn. pokud byl soubor generován 11. května 2016 v 22:09:15, název souboru bude `Stat_lastByte_2016-5-11T22:9:15`.

Statistika nejvíce a nejčastěji skenovaných portů

Tato analýza je zaměřena na 20 nejvíce skenovaných portů (ve smyslu vyskytujících se v nejvíce tocích) a na 20 nejčastěji skenovaných portů (nejvíce jednotlivců/zdrojů skenovalo tento port). K uchování informací o portu a pozdějšímu určení nejvíce a nejčastěji skenovaných portů slouží struktura `portStats`, která obsahuje položky:

- `uint16_t port` – číslo portu,
- `unsigned long int requests` – celkový počet dotazů na port ze všech zdrojů,
- `unsigned long int numOfSources` – celkový počet zdrojů, kteří se ptali na tento port,
- `portStats *LPtr` – ukazatel na levého potomka,
- `portStats *RPtr` – ukazatel na pravého potomka.

I tato statistika je tisknuta na konci časového okna. Tisk probíhá do souboru, jehož jméno vzniká obdobně jako v předchozím případě, jen místo předpony `Stat_lastByte_` je užita předpona `Stat_ports_`. Nejdříve je do souboru tisknuta statistika nejvíce skenovaných portů ve tvaru `číslo portu: počet tázaných cílů na tento port` a poté statistika nejčastěji skenovaných portů ve tvaru `číslo portu: počet zdrojů, kteří skenovali tento port`.

[6] Jelikož jsem měl k dispozici pouze anonymizovaná data, nemělo to pro mě moc smysl. Je to spíše myšleno pro budoucí použití.

[7] Nástroj je šířený pod licencí GNU GPLv2. Pro více informací viz. http://maps.measurement-factory.com

[8] viz. http://libgd.github.io/

6.8 Výstup programu

Jak již bylo řečeno v úvodu této kapitoly, výstupy jsou ovlivnitelné pomocí parametrů. V následujícím textu budou popsány významy parametrů a formát výstupu.

Parametr `-b` znamená stručný výstup. Tisk údajů probíhá na standardní výstup. Při tomto výstupu jsou data tisknuta ve formátu – `IP adresa zdroje skenování,port,počet tázaných adres`. Tento parametr nelze kombinovat s parametrem `-c`

Parametr `-c` znamená kompletní výstup. Údaje jsou také vypisovány na standardní výstup a jsou tisknuty ve formátu – `IP adresa zdroje skenování,port,IP adresa cíle skenování`. Při tomto způsobu tisknutí je na výstupu mnoho dat (např. pokud zdroj skenování zkoušel nějaký port na tisíci hostech, bude na výstupu tisíc řádků jen o tomto skenování). Tento parametr nelze kombinovat s parametrem `-b`.

Parametr `-s` generuje statistiku nejvíce skenovaných portů a četnosti výskytů jednotlivých hodnot v posledním oktetu IP adres obětí tak, jak bylo popsáno v předchozí podkapitole.

Pomocí parametru `-m` bude vygenerována heat-mapa. Heat-mapa se generuje na konci časového okna. Název souboru vzniká podobně jako u statistik, jen předpona bude `Map_` a následovat bude datum vytvoření souboru s heat-mapou.

Pomocí parametru `-y` bude vygenerován výřez heat-mapy. U této volby je povinný argument, který specifikuje jaký síťový blok se má vykreslit. Argument se zadává ve formátu `IP adresa/prefix`. Při zadání **0.0.0.0/0** se vykreslí celá mapa, totožná s tou, která se vykreslí při zadání parametru `-m`.

Kapitola 7

Vyhodnocení a výsledky

V této kapitole bude představeno testování navrženého a implementovaného modulu. Testování modulu probíhalo na základě experimentů s různými datovými sadami. Data byla sbírána v reálném čase přímo z přístupného kolektoru systému Nemea. Tyto záznamy nebyly nejdříve ukládány do souborů, ale přímo posílány na vstupní rozhraní modulu. V následujícím textu budou popsány experimenty a výsledky nad jednotlivými sadami.

7.1 Data z Nemea kolektoru

Testování probíhalo na kolektoru `collector-nemea:7700`, z něhož jsou přijímána reálná data ze sond umístěných v síti organizace Cesnet. Pro příjem dat z kolektoru je pouze nutné připojit na vstupní rozhraní modulu daný kolektor.

Z důvodu soukromí jsou IP adresy všech záznamů anonymizovány. To však na testování nemá žádný vliv, jelikož jsou anonymizovány pouze IP adresy, čili čísla portů zůstávají nezměněna. Pouze nebudou určeny skutečné zdroje a cíle skenování.

Nyní budou následovat výsledky některých experimentů s daty z kolektoru. Modul byl několikrát spouštěn s různými hodnotami parametrů pro nastavení časového okna a prahu. Zároveň s implementovaným modulem byl spouštěn modul `logger`, pomocí něhož byly informace o tocích zaznamenávány do souboru. Díky tomu mohl být analyzován výstup z testovaného modulu. Experimentování by se dalo rozdělit do několika etap v závislosti na parametrech. Ještě je nutné zmínit, že první tři experimenty proběhly, když modul podporoval jen TCP protokol, poslední je i s podporou UDP protokolu.

5minutové okno s prahem 50

Při tomto experimentu bylo časové okno ponecháno ve výchozím nastavení (výchozí hodnota časového okna je 5 minut) a práh byl nastaven na 50 (přepínačem `-t 50`). Výstup implementovaného modulu naznačoval celkem 713 zdrojů, jež skenovaly 108 portů. Graf na obrázku 7.1 ukazuje 10 nejvíce skenovaných portů. Jak je vidět, nejvíce skenovaným portem byl port číslo 1433. Z informací na internetu vyplývá, že port používá Microsoft SQL Server a zranitelnosti využívají různí červi[1]. Služby přiřazené portům z grafu 7.1 jsou uvedeny v tabulce 7.1. Služby a jejich porty byly zjištěny z [14].

Z dalších výstupů modulu byly podrobné analýze podrobeny:

- Prvních 20 nejaktivnějších zdrojů skenování.

[1] Viz. http://www.speedguide.net/port.php?port=1433

- Posledních 20 zdrojů (tzn. ti, co měli hodnotu počítadla skenovaných cílů blízko prahu).
- Namátkově několik záznamů, jež se nacházely mezi výše zmíněnými.

Přehled přiřazených portů běžným službám	
port	služba
23	telnet
22	ssh
6 379	redis
3 306	MySQL
443	HTTPS
3 389	MS WBT Server
1 723	PPTP
37 015	Nepřiřazeno
80	HTTP
1 433	Microsoft SQL server
10 020	Nepřiřazeno
8 000	iRDMI
8 080	HTTP alternativa
445	Microsoft DS

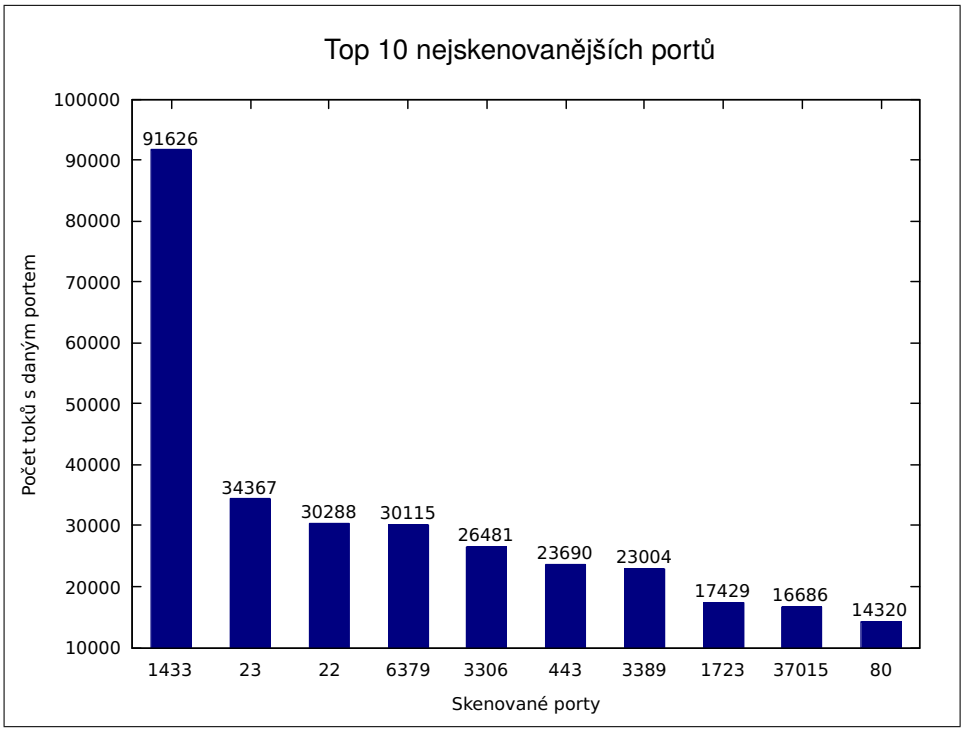
Tabulka 7.1: Tabulka služeb a jejich portů

Z této analýzy bylo zjištěno použití metody TCP SYN skenování (popsané v podkapitole 3.5) v drtivé většině zjištěných skenování. Druhou nejčastější metodou bylo TCP ACK skenování, popsané taktéž v podkapitole 3.5. Je nutné podotknout objevení několika falešných hlášení. A to zejména u označených zdrojů s počítadlem unikátních cílových adres blízko nastavenému prahu (tedy kolem 50 dotazů na port na těchto cílových adresách). Při bližším zkoumání těchto toků jsem si všiml, že se většinou jednalo o porty 443 a 80. Tyto porty jsou využívány webovými protokoly (HTTP a HTTPS). Vysvětlením, proč se za takový krátký okamžik vyskytlo mnoho toků od jednoho zdroje na mnoho cílů s těmito porty, by mohlo být, že daný zdroj je zařízení s aktivním NAT. Čili reálně toky šly od zařízení za NATem, jen pro veřejnou síť se prezentovaly veřejnou IP adresou prvku s překladem adres.

V tabulce 7.2 je přehled deseti nejaktivnějších zdrojů skenování. Místo jejich IP adres jsou zapsána jen písmena. Tabulka zobrazuje na jaký port se jednotlivé zdroje dotazovali a počet unikátních IP adres, na kterých byl dotazován daný port.

5minutové okno s prahem 100

Při tomto experimentu bylo časové okno a práh nechány ve výchozích nastaveních, tedy časové okno bylo 5 minut a práh 100. Datová sada byla jiná, než v předešlém experimentu. Nemohly tedy být srovnány výstupy a porovnány, co a jak se změnilo. Testování probíhalo jako v předešlém případě. Nyní bylo označeno 373 zdrojů skenujících celkem 72 portů. Na obrázku 7.2 je opět znázorněn graf s deseti nejvíce skenovanými porty. V tomto případě vyšel nejvíce skenovaný port číslo 1723. Tento port je využívaný protokolem Point-to-Point



Obrázek 7.1: Top 10 nejvíce skenovaných portů pro časové okno 5 minut a práh 50

Tunneling Protocol k realizaci virtuální privátní sítě[2]. Služby přiřazené daným portům jsou uvedeny v tabulce 7.1.

Po analýze dalších výstupů, stejně jako v předchozím případě, bylo zjištěno převážně TCP SYN skenování a zbývající skenování bylo prováděno metodou TCP ACK skenování. Nicméně v tomto experimentu bylo zjištěno velmi málo falešných hlášení, proto byla hodnota prahu 100 zvolena jako výchozí hodnota.

10minutové okno s prahem 50

V tomto experimentu bylo časové okno zvýšeno na 10 minut (přepínačem `-w 600`) a práh nastaven opět na 50 (pomocí přepínače `-t 50`). Graf s deseti nejvíce skenovanými porty v tomto experiment je na obrázku 7.3. Přiřazené služby portům jsou uvedeny v tabulce 7.1. Tentokrát se stal nejvíce skenovaným portem port číslo 23, jemuž připadá služba telnet. Z toho se dá vyvodit, že se daný zdroj skenování snažil zjistit, zda na stanicích běží služba telnet a zneužít ji k průniku do systému daného cílového stroje.

Opět se zde vyskytuje hlavně TCP SYN skenování. Díky nízko zvolenému prahu se vyskytlo i několik falešných hlášení, stejně jako tomu bylo u 5minutového experimentu se stejným zvoleným prahem.

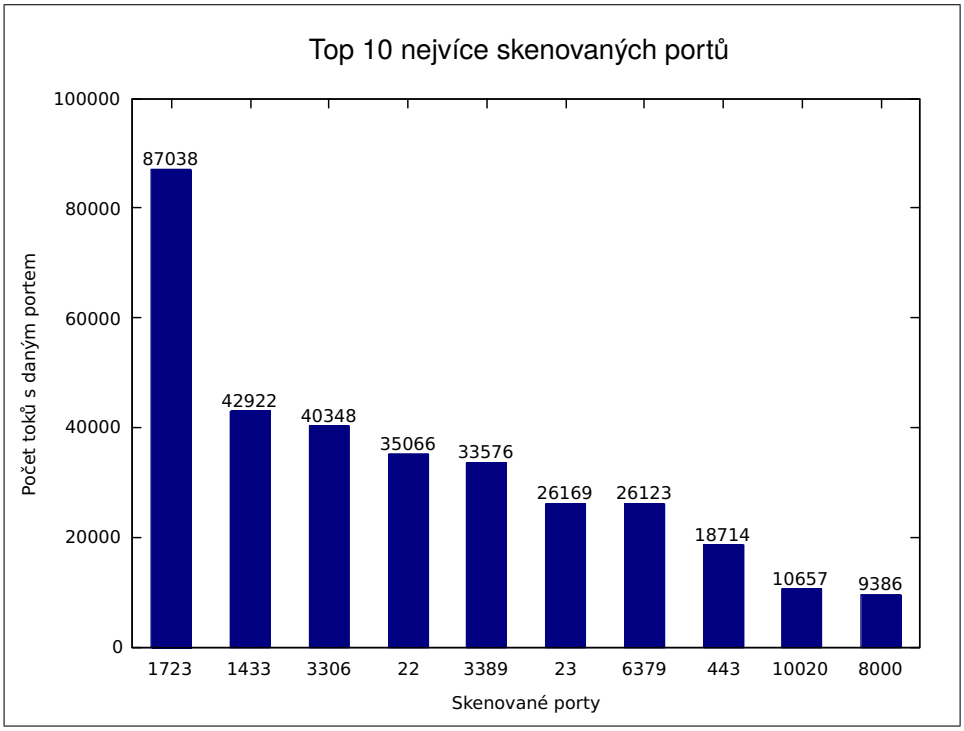
10minutové okno s prahem 100

Tento experiment měl nastaveno časové okno na 10 minut a hodnota prahu byla nastavena na 100. Je nutné podotknout, že tento experiment byl prováděn jiný den, než experimenty

[2]Viz. http://www.speedguide.net/port.php?port=1723

Nejaktivnější zdroje skenování		
zdroj	skenovaný port	celkový počet dotazů na port
A	6 379	28 200
B	1 433	26 615
C	1 433	26 543
D	3 306	25 659
E	22	22 480
F	1 433	19 206
G	1 723	16 783
H	37 015	16 686
I	8 000	12 040
J	9 200	11 500

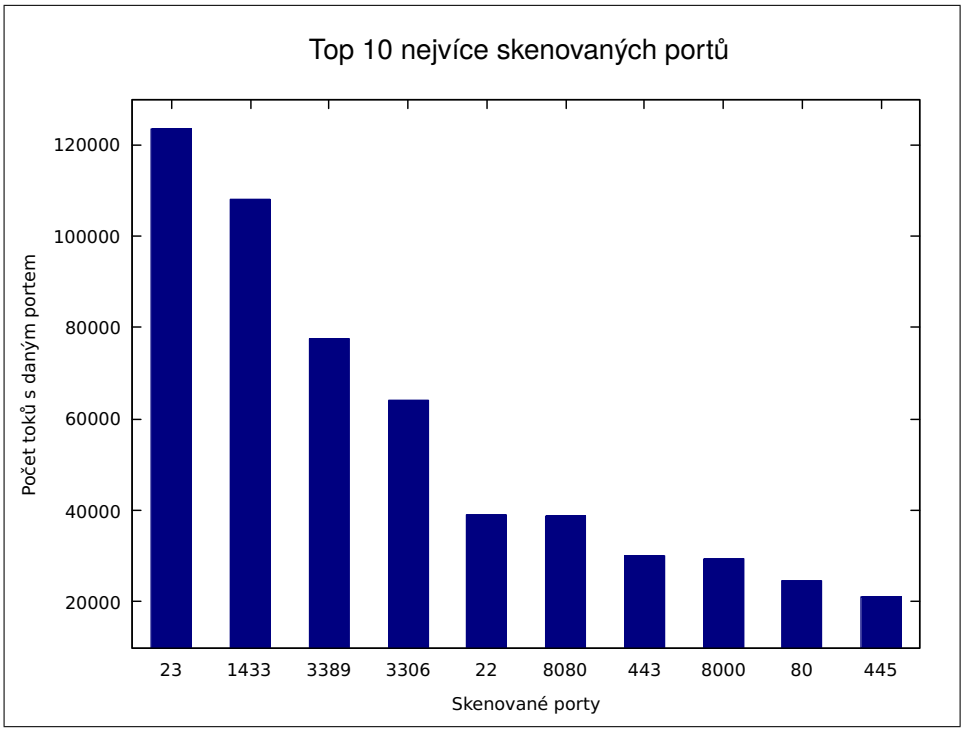
Tabulka 7.2: Tabulka nejaktivnějších zdrojů



Obrázek 7.2: Top 10 nejvíce skenovaných portů pro časové okno 5 minut a práh 100

předešlé. A modul byl upraven i na podporování protokolu UDP. To se promítlo i v grafu na obrázku 7.4, kde je složení skenovaných portů celkem jiné. Služby odpovídajících portů, které nejsou v tabulce 7.1, jsou doplněny do tabulky 7.3.

V tomto experimentu bylo zaznamenáno mnoho toků o jednom paketu služby DNS, což mohlo být zapříčiněno tím, že ostatní pakety šly jinou linkou, než kde odchytávala sonda. Tím vznikly jedno-paketové toky, které byly následné prohlášeny za skenování.



Obrázek 7.3: Top 10 nejvíce skenovaných portů pro časové okno 10 minut a práh 50

Přehled přiřazených portů běžným službám	
port	služba
53	DNS
123	NTP
1 900	SSDP
5 353	Multicast DNS

Tabulka 7.3: Tabulka služeb a jejich portů

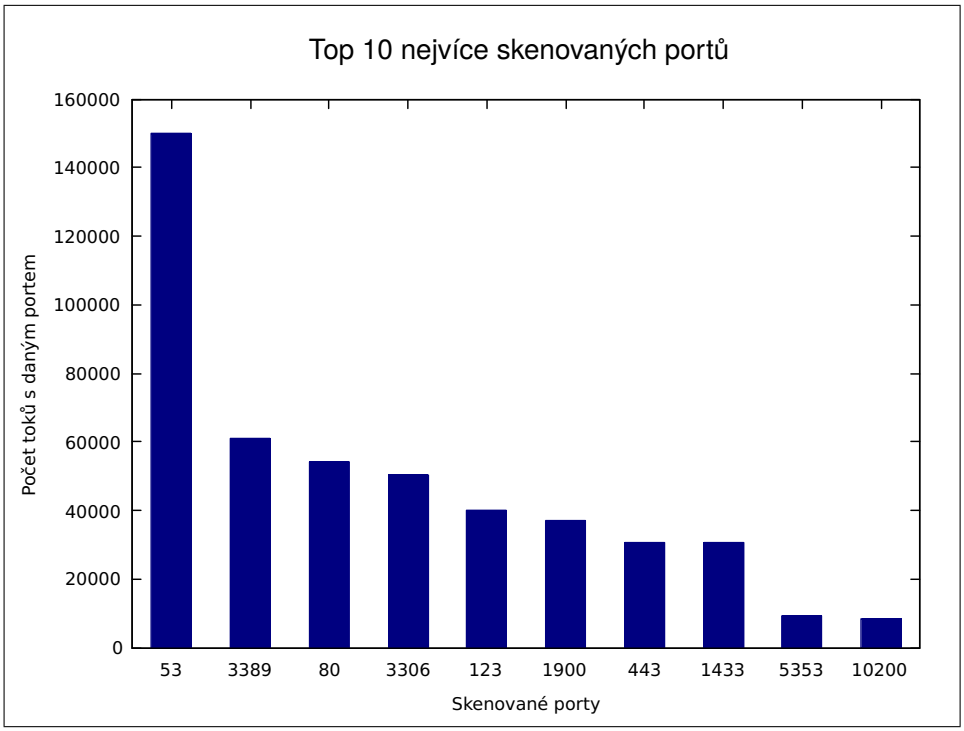
7.2 Zhodnocení experimentů

Z proběhlých experimentů je patrné, že množství falešných hlášení je závislé na zvolené hodnotě prahu. Bylo vypozorováno, že nejčastější používanou metodou ke skenování je metoda TCP SYN skenování popsaná v podkapitole 3.5. Druhou byla metoda TCP ACK skenování.

7.3 Kontrola paměti

Pro účely kontroly správného uvolňování alokované paměti byl použit nástroj `valgrind`[3]. Pomocí tohoto nástroje bylo zjištěno, že probíhá uvolňování všech alokovaných zdrojů a nedochází tak k únikům paměti (tzv. memory-leak).

[3]Více informací viz. http://valgrind.org/



Obrázek 7.4: Top 10 nejvíce skenovaných portů pro časové okno 10 minut a práh 100

Po dobu běhu modulu byl také spuštěn interaktivní prohlížeč procesů `htop`[4]. Pomocí něhož bylo sledováno využití zdrojů. Mohl jsem si tak všimnout celkem velkých paměťových nároků, kdy po přidání podpory protokolu UDP dokáže program během 5 minut zabrat až 300MB paměti. Z tohoto důvodu bylo implementováno zapínání podpory protokolu UDP pomocí parametru `-u`.

[4]Více informací viz. http://hisham.hm/htop/

Kapitola 8

Závěr

Cílem této bakalářské práce bylo vytvoření modulu pro systém analýzy síťových dat Nemea pro detekci horizontálního skenování v síti a následné automatické analýzy. Hlavní myšlenkou metody detekce je porovnání unikátního počtu cílových adres, na kterých byl dotazován daný port, s hodnotou prahu v určitém časovém okně. Při překročení tohoto prahu je pak záznam označen za skenování. Analýzou je pak generování heat-mapy či různých statistik.

V kapitole 2 bylo nastíněno, co je to monitorování sítě, IP tok a jak pracuje technologie NetFlow. Kapitola 3 vysvětlila pojem skenování portů, co je to port, rozdělení skenování a popsala vybrané metody uplatňované při skenování portů. Kapitola 4 popisovala systém Nemea, pro který byl modul implementován. V kapitolách 5 a 6 byly popsány návrh a následná implementace metody pro detekci a analýzu skenování. V předposlední kapitole byly uvedeny výsledky experimentů na základě reálných dat ze sítě Cesnet.

Z výsledků je patrné, že záleží na určení hodnoty prahu. Čím je hodnota nižší, tím víc bude falešných hlášení. Nicméně, když bude hodnota prahu vysoká, některá skenování portů nemusí být odhalena.

Zajímavým rozšířením modulu by byla například stavová detekce pro TCP protokol, což by zpřesnilo určení skenování. Dalším rozšířením, které by mohlo být součástí toho prvního, by mohlo být ukládání IP adres serverů, které odpověděly kladně (tedy v záznamu by byly nastaveny příznaky SYN+ACK). Tím by se získal seznam aktivních serverů, aniž by museli být ještě jednou skenovány. Ještě by do budoucna bylo dobré implementovat podporu IPv6.

Literatura

[1] Bartoš, V.; Žádník, M.; Čejka, T.: Nemea: Framework for stream-wise analysis of network traffic. Technická zpráva, CESNET, Prosinec 2013.

[2] Black, P. E.: binary search tree [online]. [2011-12-12], [cit. 2016-05-15].
URL http://www.fastar.org/dads/HTML/binarySearchTree.html

[3] Bouška, P.: TCP/IP – navázání a ukončení spojení [online]. [2007-09-13], [cit. 2016-04-26].
URL
http://www.samuraj-cz.com/clanek/tcpip-navazani-a-ukonceni-spojeni/

[4] CISCO: *Cisco IOS Netflow Data Sheet* [online]. [cit. 2016-05-06].
URL http://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/product_data_sheet0900aecd80173f71.html

[5] CISCO: *NetFlow Configuration Guide, Cisco IOS XE Release 3S* [online]. [cit. 2016-05-06].
URL http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/netflow/configuration/xe-3s/nf-xe-3s-book/cfg-nflow-data-expt-xe.html

[6] Claise, B.; Trammell, B.; Aitken, P.: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information [online]. RFC 7011, září 2013, [cit. 2016-05-06].
URL https://tools.ietf.org/html/rfc7011

[7] Danielyan, E.: Identifying port scanning techniques. Listopad 2001, copyright - Copyright Element K Journals Nov 2001; Poslední aktualizace - 2014-05-18; CODEN - INSOFC.

[8] Gadge, J.; Patil, A. A.: Port scan detection. In *2008 16th IEEE International Conference on Networks*, Prosinec 2008, ISSN 1531-2216, s. 1–6, doi:10.1109/ICON.2008.4772622.

[9] Lee, C. B.; Roedel, C.; Silenok, E.: Detection and characterization of port scan attacks. *Univeristy of California, Department of Computer Science and Engineering*, 2003.

[10] Lyon, G. F.: *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. USA: Insecure, 2009, ISBN 0979958717, 9780979958717.

[11] Matthew, A.; Khan, H.; Abdullah, N.: *Network Monitoring*. Diplomová práce, School of Information Science, Computer and Electrical Engineering Halmstad University, 2012.

[12] Muraleedharan, N.: Analysis of TCP flow data for traffic anomaly and scan detection. In *2008 16th IEEE International Conference on Networks*, Prosinec 2008, ISSN 1531-2216, s. 1–4, doi:10.1109/ICON.2008.4772645.

[13] Postel, J.: Transmission control protocol [online]. RFC 793, září 1981, [cit. 2016-04-26].
URL https://tools.ietf.org/html/rfc793

[14] Touch, J.; Lear, E.; Mankin, A.; aj.: Service Name and Transport Protocol Port Number Registry [online]. [2016-05-13], [cit. 2016-05-15].
URL http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

[15] de Vivo, M.; Carrasco, E.; Isern, G.; aj.: A Review of Port Scanning Techniques. *SIGCOMM Comput. Commun. Rev.*, ročník 29, č. 2, Duben 1999: s. 41–48, ISSN 0146-4833, doi:10.1145/505733.505737.

[16] Wikipedia: *Skenování portů* [online]. [cit. 2016-04-26].
URL https://cs.wikipedia.org/wiki/Skenování_portů

[17] Wikipedia: *Síťový port* [online]. [cit. 2016-04-26].
URL https://cs.wikipedia.org/wiki/Síťový_port

Přílohy

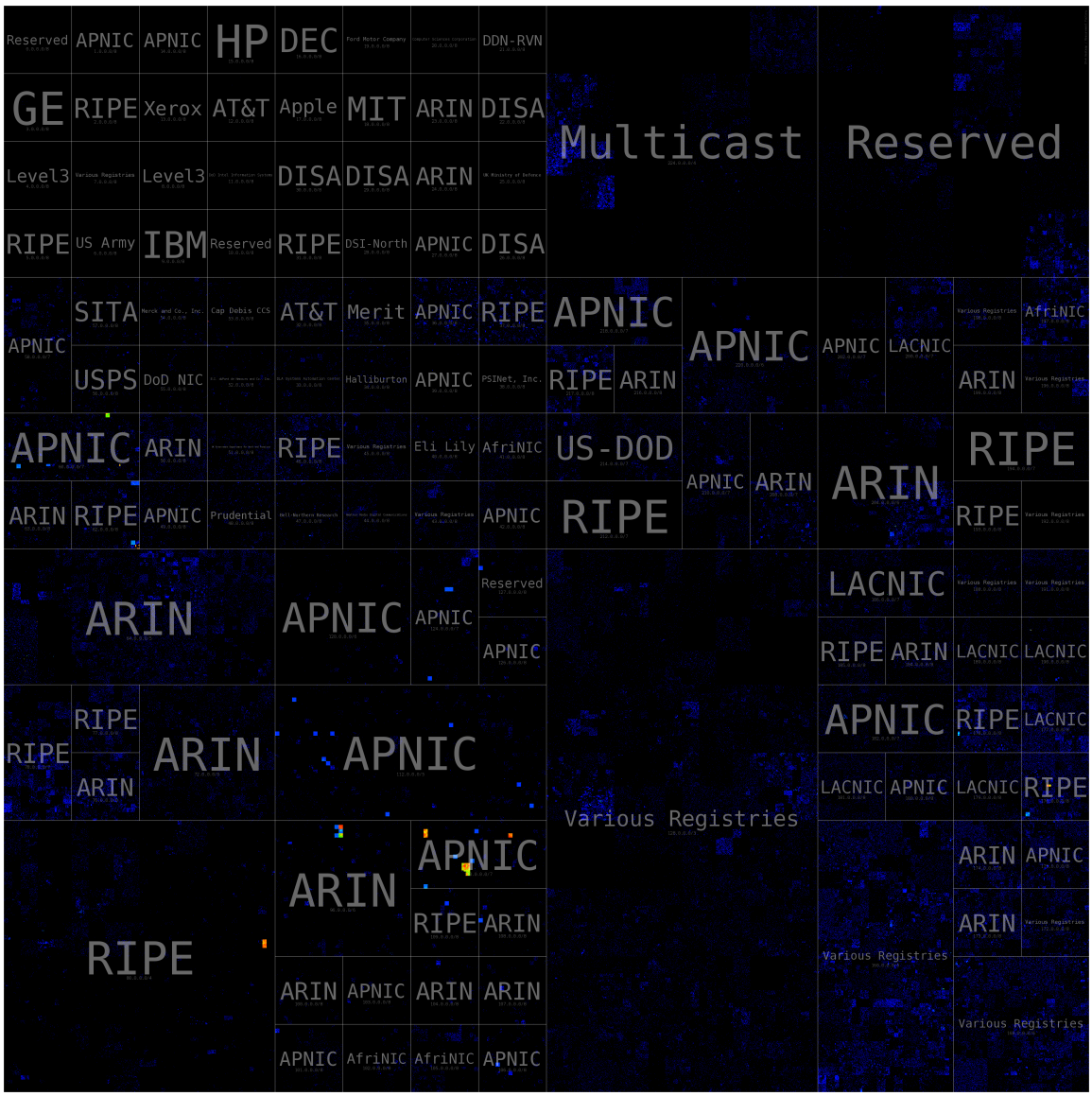
Seznam příloh

A Náhled heat-mapy **36**

B Obsah CD **38**

Příloha A

Náhled heat-mapy



Obrázek A.1: Ukázka heat-mapy

Příloha B

Obsah CD

- Text bakalářské práce ve formátu PDF, včetně zdrojových souborů této zprávy pro sázecí systém LaTeX
- Zdrojové soubory systému Nemea a vyvíjeného modulu
- Makefile pro překlad zdrojových kódů
- soubor README s návodem na instalaci
- ukázky výstupů statistik