VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

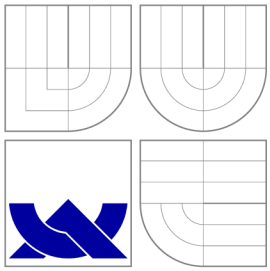
ROZŠÍŘENÍ BEHAVIORÁLNÍ ANALÝZY SÍŤOVÉ KOMUNIKACE URČENÉ PRO DETEKCI ÚTOKŮ

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

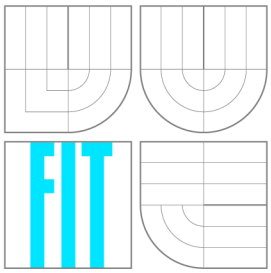
AUTOR PRÁCE
AUTHOR

Bc. MARTIN TEKNŐS

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

ROZŠÍŘENÍ BEHAVIORÁLNÍ ANALÝZY SÍŤOVÉ KOMUNIKACE URČENÉ PRO DETEKCI ÚTOKŮ

EXTENSION OF BEHAVIORAL ANALYSIS OF NETWORK TRAFFIC FOCUSING ON ATTACK DETECTION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MARTIN TEKNŐS

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. IVAN HOMOLIAK

BRNO 2015

Abstrakt

Práce se zaobírá behaviorální analýzou síťové komunikace (NBA) určené pro detekci útoků. Cílem práce je vylepšit NBA zvýšením přesnosti detekce obfuskovaných síťových útoků pomocí ní. Jsou představeny metody a techniky používané pro detekci síťových útoků a klasifikaci síťového provozu. Dále jsou popsány systémy na detekci útoků (IDS) z pohledu jejich funkcionality a možných útoků na ně. Práce popisuje principy vybraných útoků proti IDS a jsou navrhnuty metody obfuskace, které je možné využít pro překonání NBA. Dále byl navržen a implementován nástroj na automatickou exploitaci, který také vykonává navržené obfuskace síťových útoků a sbírá data z této síťové komunikace. Vytvořený nástroj byl použit k vykonání síťových útoků. Pak byli získány data pro experimentování a vykonány různé experimenty, kterých výsledkem bylo zdůraznění trénování klasifikačních modelů NBA s využitím znalosti o obfuskacích.

Abstract

This thesis is focused on network behavior analysis (NBA) designed to detect network attacks. The goal of the thesis is to increase detection accuracy of obfuscated network attacks. Methods and techniques used to detect network attacks and network traffic classification were presented first. Intrusion detection systems (IDS) in terms of their functionality and possible attacks on them are described next. This work also describes principles of selected attacks against IDS. Further, obfuscation methods which can be used to overcome NBA are suggested. The tool for automatic exploitation, attack obfuscation and collection of this network communication was designed and implemented. This tool was used for execution of network attacks. A dataset for experiments was obtained from collected network communications. Finally, achieved results emphasized requirement of training NBA models by obfuscated malicious network traffic.

Klíčová slova

behaviorální analýza síťové komunikace, NBA, detekce síťových útoků, IDS, IPS, klasifikace síťového provozu, útok, útok na síťovou službu, síťový útok, dolovaní z dat, strojové učení, ASNM, exploit, obfuskace, segmentace, fragmentování, modifikace MTU, přehození paketů, duplikace paketů, zahazování paketů, poškození paketů, detekce

Keywords

network behavior analysis, NBA, detection of network attacks, IDS, IPS, network traffic classification, attack, attack on network service, network attacks, data mining, machine learning, ASNM, exploit, obfuscation, segmentation, fragmentation, MTU modification, packet reordering, packet duplication, packet loss, packet corruption, detection

Citace

Martin Teknős: Rozšíření behaviorální analýzy síťové
komunikace určené pro detekci útoků, diplomová práce, Brno, FIT VUT v Brně, 2015

Rozšíření behaviorální analýzy síťové komunikace určené pro detekci útoků

Prohlášení

Prehlasujem, že som tento semestrálny projekt vypracoval samostatne pod vedením pána Ing. Ivana Homoliaka. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Bc. Martin Teknős
27. mája 2015

Poděkování

Rád by som poďakoval vedúcemu mojej práce Ing. Ivanovi Homoliakovi za odbornú pomoc, poskytnuté rady, ochotu a čas strávený pri konzultáciách.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Kapitola 1

Úvod

Bezpečnosť sieťovej prevádzky je naďalej dôležitou oblasťou, ktorá sa neustále vyvíja. Stále je tu veľa oblastí, kde je potrebný ďalší výskum pre zlepšenie obrany proti rôznym typom sieťových útokov. Neustále vznikajú nové hrozby, a staré často zostávajú aktuálne. Klasické spôsoby zabezpečenia už prestávajú postačovať. To bolo jedným z dôvodov prečo postupne vznikla oblasť behaviorálnej analýzy sieťovej komunikácie (ďalej NBA). Tá sa snaží pristupovať k detekcii útokov novým prístupom, ktorý nevyužíva k analýze obsah paketov, ale len ich hlavičiek.

Cieľom tejto práce je prispieť k vylepšeniu NBA systémov a najmä zvýšiť presnosť detekcie obfuskovaných útokov, ktoré sa zvyčajne ťažko odhaľujú. Predpokladom je, že NBA dosiahne nižšiu úspešnosť klasifikácie, ak pri procese trénovania klasifikačného modelu nebude mať informácie o obfuskovaných útokoch.

Po naštudovaní problematiky boli navrhnuté rôzne techniky na obfuskáciu sieťových útokov. Ďalej bol navrhnutý a implementovaný nástroj na automatickú obfuskáciu, vykonanie útoku a zaznamenanie tejto aktivity. Pomocou tohto nástroja boli získané dáta pre neskoršie experimenty. Nakoniec boli nad týmito dátami vykonané rôzne experimenty a zhodnotené ich výsledky.

Práca je rozdelená na niekoľko kapitol. Klasifikácii sieťovej prevádzky sa venuje 2. kapitola. Popisuje rôzne prístupy ku klasifikácii sieťovej prevádzky, praktickým obmedzeniam a nakoniec aj vybrané existujúce techniky. V 3. kapitole sa práca venuje princípom detekcie sieťových útokov a popisuje rôzne metódy detekcie a technológie IDS[1]. Behaviorálnou analýzou sa zaoberá kapitola 4. Popisuje najmä existujúce NBA. V 5. kapitole sú uvedené útoky proti IDS, ich klasifikácia, niektoré existujúce útoky a nakoniec navrhnuté postupy na realizáciu obfuskovaných útokov. Testovaciu infraštruktúru, využité zraniteľné sieťové služby a útoky popisuje 6. kapitola. Návrhom a implementáciou nástroja na automatickú exploitáciu sieťových služieb sa zaoberá 7. kapitola. V 8. kapitole je popísané, ako bola získaná sada dát k experimentom. Experimentom nad touto sadou dát sa potom venuje 9. kapitola. V 10. kapitole sú uvedené možnosti ďalšieho rozvoja. Nakoniec, 11. kapitola zhŕňa dosiahnuté výsledky a možnosti ďalšieho vývoja.

[1] *Intrusion Detection System*

Kapitola 2

Klasifikácia sieťovej prevádzky

Dôležitosť klasifikácie IP sieťovej prevádzky môžno ilustrovať na príklade zákonného odpočúvania . Siete poskytovateľov internetového pripojenia musia umožňovať zákonný odposluch. Techniky klasifikácie sieťovej prevádzky ponúkajú možnosť identifikovať vzory sieťovej prevádzky a ktoré triedy aplikácií využíva určitá osoba v ľubovoľnom čase. [1]

Bežne nasadené techniky klasifikácie IP sieťovej prevádzky sú založené na priamom skúmaní obsahu každého paketu na nejakom bode siete. Jednoduchá klasifikácia odvodzuje identitu aplikácie na základe predpokladu, že väčšina aplikácií dôsledne používa „všeobecne známe“ čísla TCP/UDP portov[1] [1].

Avšak mnoho aplikácií stále vo väčšej miere využíva nepredvídateľné čísla portov [3]. V dôsledku toho, sa začali používať sofistikovanejšie techniky klasifikácie. Tieto techniky odvodzujú typ aplikácie skúmaním/hľadaním údajov špecifických pre danú aplikáciu, alebo podľa známeho správania protokolu v rámci TCP/UDP dátového obsahu (*payload*) [4].

Bohužiaľ účinnosť takejto hĺbkovej inšpekcie paketov sa zmenšuje. Tieto techniky sa spoliehajú na dva súvisiace predpoklady:

1. Tretia strana, ktorá nie je v žiadnom vzťahu so zdrojom ani príjemcom je schopná kontrolovať každý dátový obsah IP paketu.

2. Klasifikátor pozná syntax každého dátového obsahu paketu (tzn. payload je možné interpretovať).

Avšak prvý predpoklad je oslabený, pretože zákazníci môžu používať šifrovanie aby obfuskovali obsah paketu (vrátane čísiel TCP/UPD portov). Ďalším problémom môže byť legislatíva, ktorá stanovuje predpisy o ochrane osobných údajov, ktoré by úplne zamedzovali schopnosť tretích strán zákonne kontrolovať obsah paketov. Druhý predpoklad je ohrozený tým, že vyžaduje opakované aktualizácie, aby boli vždy pred pravidelnými zmenami vo formáte obsahu paketov každej aplikácie. [1]

Výskumníci zareagovali skúmaním klasifikačných systémov, ktoré budú schopné fungovať bez hĺbkovej analýzy obsahu paketov. Novšie prístupy klasifikujú sieťovú prevádzku rozpoznávaním štatistických vzorov v externe pozorovateľných vlastnostiach sieťovej prevádzky (napr. typické dĺžky paketov). Hlavným cieľom je zlúčiť IP sieťovú prevádzku do skupín, ktoré majú podobné vlastnosti, alebo klasifikovať ich do niekoľkých tried podľa oblasti použitia/aplikácie. [1]

[1] Názvy služieb a čísla portov sa používajú na rozlíšenie medzi rôznymi službami, ktoré bežia cez transportné protokoly [2].

Množstvo výskumných pracovníkov skúma zvlášť dôkladne aplikáciu techník strojového určenia (ďalej ML) na klasifikáciu sieťovej prevádzky. Staršie techniky ML sa spoliehali na statickú, off-line analýzu predtým zachytenej sieťovej prevádzky. Novšie práce začínajú riešiť aj praktické požiadavky, ako klasifikácia sieťovej prevádzky v reálnom čase, založená na ML v produkčných sieťach. [1]

Oblasť bezpečnosti tiež využíva techniky ML. Avšak, tu ide o niečo zásadne odlišné od kategorizácie všetkej sieťovej prevádzky. V oblasti bezpečnosti je podstatné nájsť zhodu signatúry na identifikáciu preniknutia, alebo škodlivého programu podľa jeho neštandardného správania medzi inak legitímnou aktivitou.

Sieťová prevádzka sa väčšinou klasifikuje po skupinách paketov, tzv. tokoch:

tok (flow) : Po sebe idúce pakety, ktoré majú rovnakú päticu (typ protokolu, zdrojovú IP adresu i port a cieľovú IP adresu i port). Táto pätica informácií je prítomná v každom pakete. Takéto pakety patria do jedného toku, ktoré chceme klasifikovať. [1]

Táto kapitola začína popisom metrík na vyhodnotenie klasifikácie (sieťovej prevádzky). Pokračuje prehľadom a popisom vlastností rôznych prístupov ku klasifikácii tokov, venuje sa aj jej aplikácii v praxi a problémom. Nakoniec sú popísané niektoré vybrané techniky klasifikácie sieťovej prevádzky (2.6).

2.1 Metriky klasifikácie sieťovej prevádzky

Kľúčovým kritériom je presnosť predikcie (*accuracy*), tzn. ako presný je klasifikátor, keď dostane dáta, ktoré, predtým nevidel. Najskôr je však vhodné zaviesť aj ostatné metriky. Keď máme klasifikátor a inštanciu, môžno dostať jeden zo štyroch výsledkov [5]:

True Positive (ďalej TP) : inštancia je pozitívna a je tak aj klasifikovaná.

False Negative (ďalej FN) : inštancia je pozitívna, ale je klasifikovaná ako negatívna.

True Negative (ďalej TN) : inštancia negatívna a je tak aj klasifikovaná.

False Positive (ďalej FP) : inštancia negatívna, ale je klasifikovaná ako pozitívna.

Keď máme klasifikátor a množinu inštancií, je možné vytvoriť *confusion matrix* (pozri tabuľku 2.1), ktorá reprezentuje dispozície danej množiny inštancií. Tvorí základ pre mnoho bežne používaných metrík. Čísla pozdĺž hlavnej diagonály (TP, TN) reprezentujú správne rozhodnutie klasifikátora. Na druhej diagonále (FP, FN) sú čísla, ktoré reprezentujú chyby medzi rôznymi triedami. [5] Kvalitný klasifikátor má za cieľ minimalizovať FN a FP.

		Skutočná trieda	
		Pozitívna	Negatívna
Trieda predpovedaná klasifikačným modelom	Pozitívna	TP	FP
	Negatívna	FN	TN
	Celkovo v stĺpci	P	N

Tabuľka 2.1: *Confusion matrix* [5].

Medzi hlavné metriky, ktoré sa dajú počítať z *confusion matrix* patria [5]:

***sensitivity* (*recall, true positive rate*) :**

$$sensitivity \approx \frac{TP}{P} \tag{2.1}$$

***false positive rate* :**

$$fp\ rate \approx \frac{FP}{N} \tag{2.2}$$

***specificity* :**

$$specificity = \frac{TN}{FP + TN} = 1 - fp\ rate = \frac{TN}{N} \tag{2.3}$$

***precision* (*positive predictive value*) :**

$$precision = \frac{TP}{TP + FP} \tag{2.4}$$

***accuracy* :**

$$accuracy = \frac{TP + TN}{P + N} \tag{2.5}$$

2.1.1 Presnosť bajtová a toková.

Presnosť jednotlivých metrík môže byť uvádzaná ako percento bajtov/tokov vo vzťahu ku klasifikovanej sieťovej prevádzke. Voľba jedného, alebo druhého môže významne pozmeniť význam uvádzanej dosahovanej presnosti. [1]

Väčšina publikovaných prác zaoberajúcich sa klasifikáciou sieťovej prevádzky sa zameriava na *flow accuracy* (percento tokov z klasifikovanej sieťovej prevádzky). To znamená, že merajú presnosť, s ktorou sú toky korektne klasifikované, vo vzťahu k počtu ostatných tokov v testovacej a/alebo trénovacej sade dát. [1]

Avšak, niektoré práce vyjadrujú ich výpočty presnosti pomocou *byte accuracy* (bajtová presnosť). Zameriavajú sa na počet bajtov prenášaných paketmi, ktoré patria do správne klasifikovaných tokov, v pomere k celkovému počtu bajtov v testovacej a/alebo trénovacej sade dát [6, 7].

Autori článku [8] tvrdia, že bajtová presnosť je rozhodujúca pri vyhodnocovaní presnosti algoritmov na klasifikáciu sieťovej prevádzky. Poznamenávajú, že väčšina tokov na internete je malých a tvoria len malú časť, z celkového počtu bajtov a paketov v sieti (*mice flows*). Na druhej strane, väčšina bajtov sieťovej prevádzky je vygenerovaných malým počtom veľkých tokov (*elephant flows*). Uvádzajú príklad, kde bol klasifikátor optimalizovaný na 99,9% tokovú presnosť. Avšak tých zvyšných 0.1% predstavovalo 46% bajtov z celkového počtu bajtov v sieťovej prevádzke a teda výsledná bajtová presnosť bola len 54%.

Na druhej strane, je pre túto prácu, a celkovo pohľad zo strany detekcie sieťových útokov, je podstatnejšia skôr toková presnosť. Dôležitejšie je detegovať škodlivý tok v maximálnom počte prípadov. Nie je podstatné, ako veľký objem dát daný tok prenáša.

2.2 Limitácie inšpekcie paketov pre klasifikáciu sieťovej prevádzky

Tradične sa klasifikácia sieťovej prevádzky spolieha na skúmanie čísiel TCP/UDP portov paketov (*port-based classification)*, alebo na rekonštrukciu signatúr protokolu v dátovom obsahu paketov (*payload-based classification)*. Každý prístup trpí niekoľkými nedostatkami. [1]

2.2.1 Klasifikácia založená na číslach portov

Historicky mnoho aplikácií využívalo „všeobecne známe“ porty cez ktoré mohli ostatné zariadenia iniciovať komunikáciu. Aplikáciu tak bolo možné odvodiť len podľa cieľového portu paketu a jeho vyhľadaním v zozname registrovaných portov [2], ktorý spravuje organizácia IANA[2]. [1]

Avšak, tento prístup má viaceré obmedzenia. V prvom rade, niektoré aplikácie vôbec nemusia mať porty, ktoré využívajú, registrované v zozname IANA [9]. Aplikácia tiež môže z rôznych dôvodov využívať iné porty ako tie, ktoré sú uvedené v zozname IANA. Ďalšou limitáciou je, že v niektorých prípadoch sú serverové porty dynamicky alokované podľa potreby. Nakoniec, vôbec nemusí byť možné zistiť skutočné čísla portov, ak šifrovanie na IP vrstve obfuskuje TCP/UDP hlavičku. [1]

V dnešnej dobe už známe čísla portov už nemožno použiť pre spoľahlivú identifikáciu sieťových aplikácií. Existuje viacero nových aplikácií, ktoré nepoužívajú dobre známe čísla portov, alebo používajú ďalšie protokoly (napr. HTTP) ako obálky, s cieľom prejsť cez firewall bez blokovania. Jedným z dôsledkov je, že jednoduché skúmanie portov, ktoré používajú pakety (v rámci analyzovaného toku) môže viesť k nepresnej klasifikácii sieťovej prevádzky. Autori článku [10] pozorovali, že táto klasifikačná technika nedosiahla ani 70% bajtovú presnosť s použitím oficiálneho zoznamu IANA. [10]

2.2.2 Klasifikácia založená na obsahu paketov

Tento prístup rekonštruuje reláciu (*session*) a informácie o aplikácii z obsahu každého paketu [1]. Moore a Papagiannaki [10] kombinujú obe techniky a s použitím informácie o porte a prvého kilobajtu z každého toku dosahujú presnosť takmer 79%. Vyššiu presnosť (až takmer 100%) dosiahli len skúmaním celého obsahu paketov.

Značná komplexnosť a zaťaženie procesoru zariadenia, ktoré robí klasifikáciu, sú jednými z veľkých nevýhod tohto prístupu. Zariadenie, ktoré robí klasifikáciu musí byť pravidelne aktualizované, aby stále malo aktuálne informácie o aplikačných protokoloch. Ďalej musí byť toto zariadenie aj dostatočne výkonné, aby stíhalo paralelne analyzovať potenciálne veľké množstvo tokov. Tento prístup je prakticky nepoužiteľný na proprietárne protokoly a šifrovanú sieťovú prevádzku. Nakoniec, problémom môže byť aj narušenie zásad ochrany osobných údajov, alebo porušenie príslušnej legislatívy z dôvodu priamej analýzy relácie a obsahu aplikačnej vrstvy. [1]

2.3 Klasifikácia založená na štatistikách

Novšie prístupy pre identifikáciu aplikácie sa spoliehajú na štatistické charakteristiky sieťovej prevádzky. Predpokladom takejto metódy je, že sieťová prevádzka na sieťovej vrstve má štatistické vlastnosti, ktoré sú jedinečné pre určité triedy aplikácií. Tieto jedinečné štatistické vlastnosti potom umožňujú vzájomné rozlíšenie aplikácií. [1]

Vzťah medzi triedou sieťovej prevádzky a jej odpozorovaným štatistickými vlastnosťami bol uvedený v článkoch [11, 12]. Výsledky týchto prác (a mnohých ďalších) stimulovali výskum a vývoj nových klasifikačných techník (založených na štatistických vlastnostiach sieťovej prevádzky). Potreba pracovať s vzormi sieťovej prevádzky, veľkým objemom dát a mnohorozmernými priestormi tokov a atribútov paketov patria medzi dôvody pre zavedenie techník ML v tejto oblasti. [1]

[2] Internet Assigned Numbers Authority

2.4 Aplikácia strojového učenia v klasifikácii internetovej prevádzky

V roku 1994 bolo ML prvý krát použité na klasifikáciu internetovej prevádzky v kontexte detekcie preniknutia (útoku) [13]. Práca sa stala východiskovým bodom pre mnoho prác využívajúcich techniky ML pre klasifikáciu internetovej prevádzky, ktoré nasledovali.

Vlastnosti (atribúty) tokov sa nazývajú rysy (*features*). Pomocou nich je možné identifikovať a rozlišovať jednotlivé toky. Nie všetky rysy sú rovnako užitočné. Praktické klasifikátory vyberajú najmenšiu množinu rysov, ktoré ešte dávajú efektívne rozlíšenie medzi členmi jednotlivých tried sieťovej prevádzky. [1]

2.4.1 Výber rysov a redukcia ich dimenzie

V ML je výber rysov a redukcia dimenzie veľmi dôležitá. Je to jeden z krokov predspracovania dát, kedy sa odstránia redundantné[3] a irelevantné[4] rysy. Ako je uvedené v [14], presnosť predikcie Naivného Bayesovského algoritmu trpí prítomnosťou irelevantných a redundantných rysov. Schopnosť identifikovať najdôležitejšie rysy internetovej prevádzky je dôležitá pre odhalenie najlepších rysov pre klasifikáciu sieťovej prevádzky, zlepšenie presnosti klasifikácie a zníženie výpočtovej náročnosti (redukciou počtu rysov). [6]

Existujú dva rozdielne prístupy k výberu rysov [6]:

- Metóda ***filter*** využíva charakteristiky trénovacích dát, aby rozhodla dôležitosť a relevantnosť určitých rysov pre klasifikačný problém. Príkladom môže byť úroveň korelácie medzi rysmi a triedou, alebo miera separácie tried na základe uvažovaného rysu.
- Metóda ***wrapper*** používa výsledky určitého klasifikátoru na zostavenie optimálnej množiny. Vyhodnocuje výsledky klasifikátoru na testovacích dátach pre rozdielne kombinácie rysov.

Opakovaným iterovaním jedného z algoritmov môžno identifikovať optimálnu množinu rysov, ktoré sú vhodné pre konkrétnu klasifikačnú metódu. Jednou takouto metódou je dopredný výber (*forward selection*), kde sa začína bez rysov a postupne sa pridáva jeden za druhým a kontroluje sa výsledok po pridaní každého rysu. Druhým prístupom je spätná eliminácia (*backward elimination*), kedy sa začína so všetkými rysmi a postupne sa eliminujú jeden za druhým. Hlavnou nevýhodou tejto druhej metódy je jej veľká výpočtová náročnosť (keďže je nutné vyskúšať všetky rôzne kombinácie rysov). [6]

2.5 Výzvy pre operačné nasadenie

Článok [1] diskutuje viaceré výzvy pre operačné nasadenie ML na klasifikáciu sieťovej prevádzky. Niektoré z nich teraz budú popísané.

2.5.1 Skorá a kontinuálna klasifikácia

Klasifikátor by mal rozhodnúť s použitím čo najmenšieho počtu paketov pre každý tok (namiesto čakania na dokončenie každého toku). Avšak, niekoľko prvých paketov nemusí

[3] Redundantný rys je úzko korelovaný s iným rysom. Dôvodom pre odstránenie takýchto rysov je, že zhoršujú presnosť, alebo zvyšujú preučenie (znižujú zovšeobecňovanie) klasifikačného modelu. [6]

[4] Rys je irelevantný, keď nenesie žiadnu užitočnú informáciu a nemá rozlišovaciu schopnosť [6].

stačiť. Napríklad útoky sa môžu maskovať imitovaním štatistických vlastností dôveryhodných aplikácií na začiatku ich toku. Problémom môže byť aj to, keď je klasifikátor spustený (alebo reštartovaný) zatiaľčo množstvo tokov už je aktívnych (nemôže tak zachytiť začiatky týchto aktívnych tokov). Preto by mal klasifikátor v ideálnom prípade robiť kontinuálnu klasifikáciu – prepočítavať svoje rozhodnutie po celú dobu životnosti každého toku. Klasifikátor musí zohľadňovať aj fakt, že mnoho aplikácií mení ich štatistické vlastnosti v priebehu času. Avšak, tok by mal byť ideálne korektne klasifikovaný, ako patriaci rovnakej aplikácii počas celého života toku danej aplikácie. Výsledky článku [15] ukazujú, že redukcia počtu rysov významne zlepšuje výkonnosť algoritmov v zmysle skrátenia doby vytvárania modelu aj samotnej klasifikácie pre väčšinu algoritmov.

2.5.2 Efektívne využitie pamäte a procesorov

Neefektívny klasifikátor môže byť nevhodný pre prevádzkové využitie bez ohľadu na to, ako rýchlo môže byť natrénovaný a ako presne vie identifikovať toky. Určite je nutné robiť nejaké kompromisy medzi kvalitou klasifikácie klasifikátorom a spotrebou hardvérových zdrojov skutočnej implementácie.

Napríklad články [16, 6] ukazujú vynikajúci potenciál pre presnú klasifikáciu. Avšak, tieto prístupy využívajú veľké množstvo rysov a mnohé z nich sú výpočtovo náročné. Je nutné zvážiť réžiu výpočtovo náročných rysov oproti potenciálnej strate presnosti, ak by sa tieto rysy vynechali.

Článok [15] sa venuje porovnaniu piatich algoritmov ML a tiež varuje pred niektorými kompromismi medzi dobou trénovania a rýchlosťou klasifikácie. Techniky na včasnú a kontinuálnu klasifikáciu inklinovali k návrhu posuvného okna (*sliding window*), v rámci ktorého sú počítané rysy. Zväčšenie dĺžky tohto okna môže zvýšiť presnosť klasifikácie [17, 18]. Na druhej strane, v závislosti od implementácie, to môže spozdiť klasifikačné rozhodnutie a zvýšiť pamäťové nároky na dočasné ukladanie paketov počas výpočtu rysov. Komplexnosť výpočtov by však nemala rásť rýchlejšie, ako dostupný výkon a pamäť.

2.5.3 Prenosnosť a robustnosť

Klasifikačný model možno považovať za prenosný, ak ho možno použiť v rôznych sieťových lokalitách, a stále bude dosahovať dobrú presnosť klasifikácie. Takýto model je robustný, ak poskytuje konzistentne vysokú presnosť aj počas rôznych odchýliek v podobne straty paketov, *traffic shaping*, fragmentácie paketov,... Robustný klasifikátor tiež dokáže efektívne identifikovať sieťovú prevádzku novo vznikajúcich aplikácií. Žiadna z prác, ktoré skúmali v článku [1], vážne neuvažovala ani neriešila problém prenosnosti klasifikačného modelu ani jeho robustnosť.

2.6 Vybrané techniky klasifikácie sieťovej prevádzky založené na strojovom učení

2.6.1 Klasifikácia využívajúca techniky Bayesovskej analýzy

Autori článku [6] navrhli použiť naivnú bayesovskú techniku na kategorizovanie internetovej prevádzky. Použili ručne klasifikovanú dátovú sadu, ktorá im umožňovala presné vyhodnotenie klasifikácie. Vybranú sieťovú prevádzku klasifikovali do týchto kategórií: hromadný prenos dát, databáza, interaktívne, email, služby, WWW, P2P, útok, hry a multimédiá.

Používali 248 rysov na trénovanie klasifikátoru. Po redukcii počtu rysov pomocou metód *Naive Bayes Kernel Estimation* (NBKE) a *Fast Correlation-Based Filter* (FCBF) dosiahli celkovú presnosť lepšiu, ako 95%.

Autori popísali techniku, ktorá využíva schopnosť trénovať klasifikátor so známymi dátami. Ukázali, že klasifikačný model vytvorený s využitím tejto techniky je následne možné aplikovať, aj keď je dostupných oveľa menej informácií o sieťovej prevádzke. Narozdiel od dát z celého obsahu paketu (*full-payload* dáta) použitých k presnej klasifikácii, využívali autori na trénovanie i testovanie diskriminátory[5] odvodené z hlavičiek paketov. Autori zdôrazňujú, že toto je veľká výhoda ich prístupu. Použitie vzoriek vopred známej sieťovej prevádzky na kategorizáciu tokov s využitím len bežne dostupných informácií. [6]

Práca bola ďalej rozšírená o Bayesovsky trénovanú neurónovú sieť [16]. Autori demonštrovali, že sofistikovaná Bayesovsky trénovaná neurónová sieť je schopná klasifikovať toky na základe štatistík odvodených z hlavičiek paketov a bez čísiel portov a IP adries. Dosahovali presnosť 99% na dáta trénované i testované z rovnakého dňa. V takmer realistických podmienkach (využívanie trénovacej množiny dát o osem mesiacov staršej, ako trénovacie dáta) bola dosiahnutá 95% presnosť klasifikácie.

2.6.2 Hybridné prístupy

Erman a ďalší v článku [19] navrhli *semi-supervised* (kombinácia učenia s učiteľom a učenia bez učiteľa) prístup ku klasifikácii sieťovej prevádzky. Motivovali ich k tomu dva dôvody:

1. Označené/dopredu klasifikované (*labeled)* sady dát sú celkom vzácne a nie je jednoduché ich získať. Zároveň platí, že metódy učenia s učiteľom nezovšeobecňujú príliš dobre, ak sú trénované s malým počtom príkladov v dátovej sade.

2. Možnosť príchodu nových aplikácií a nie všetky z nich musia byť vopred známe. Tradičné metódy učenia s učiteľom priraďujú neznámy tok do jednej zo známych tried, ale nie sú schopné detegovať nový typ toku (ktorý by potreboval vlastnú triedu).

Autori to vyriešili klasifikačnou technikou pozostávajúcou z dvoch krokov:

1. Privedenie trénovacích dát pozostávajúcich z označených tokov v kombinácii s neoznačenými (*unlabeled*) tokmi do zhlukovacieho algoritmu.

2. Použitie dostupných označených tokov a ich mapovanie zo zhlukov na rôzne známe triedy. Tento prístup umožní zachovať niektoré zhluky.

Tento nový prístup má sľubné výsledky. Priebežné výsledky boli uvedené v článku [19], kde použili zhlukovací algoritmus K-Means. S dvomi označenými tokmi na zhluk dosiahli 94% tokovú presnosť. Zvyšovanie počtu označených tokov na zhluk nad päť zlepšovalo presnosť len zanedbateľne. Viac detailov možno nájsť v [20].

Autori tvrdia, že navrhnutý prístup je výhodný z pohľadu kratšieho trénovania s malým počtom označených tokov zmiešaného s veľkým počtom neoznačených tokov, je schopný zvládnuť predtým nevidené aplikácie a varianty charakteristík existujúcich aplikácií. Nakoniec uviedli aj možnosť vylepšenia presnosti klasifikácie pridaním neoznačených tokov pre iteračné trénovanie. Avšak, neuviedli/nedemonštrovali to v spomínanom článku. Autori mali aj ďalšie práce, ale bohužiaľ sú na to patenty, takže to už nie je také zaujímavé.

[5] Parametre objektov, ktoré umožňujú rozlišovať medzi jednotlivými triedami sieťovej prevádzky, dostali meno discriminátory (*discriminators*) [6].

Kapitola 3

Princípy detekcie sieťových útokov

Na začiatok je uvedené rozlíšenie niektorých základných pojmov:

preniknutie (*intrusion*) : Úspešné narušenie dôvernosti, integrity, dostupnosti, alebo prekonanie bezpečnostných mechanizmov počítača, alebo siete [21].

detekcia preniknutia (*intrusion detection* (ďalej ID)) : Proces sledovania udalostí, ktoré sa vyskytujú v počítačovom systéme (alebo sieti) a ich analýza na príznaky preniknutia [21].

systém na detekciu preniknutia (*intrusion detection system* (ďalej IDS)) : Softvérový/hardvérový produkt, ktorý automatizuje detekciu preniknutia (ID) [21].

systém k prevencii preniknutia (*intrusion prevention system* (ďalej IPS)) : Softvér, alebo hardvér, ktorý má všetky schopnosti IDS a tiež sa môže pokúsiť zastaviť škodlivé incidenty [22].

Pre zameranie tejto práce nie je dôležité rozlišovať systémy IDS a IPS, preto sa pri zmienke o IDS bude myslieť vždy aj IPS (ak nebude uvedené inak).

Technológie IDS máju spoločnú nevýhodu. Tou je, že nemôžu poskytnúť absolútne presnú detekciu. Presnosť detekcie sa posudzuje pomocou dvoch indikátorov:

- **falošne pozitívne (ďalej FP)** – Ide o prípad falošného poplachu, kedy IDS nesprávne identifikuje legitímnu činnosť, ako škodlivú/nebezpečnú.
- **falošne negatívne (ďalej FN)** – Tu IDS prehliadne/neidentifikuje nebezpečnú aktivitu.

Vzhľadom na to, že je problematické dosahovať čo najnižšie miery pre oba indikátory, je bežne preferované skôr znižovanie FN (a zvýšenie FP). Preferuje sa vyššia bezpečnosť na úkor falošných poplachov. [23]

Autori článku [24] nazbierali prípady FP a FN z reálneho sveta a štatisticky ich analyzovali. Zistili, že väčšina falošných prípadov je práve FN, pretože väčšina aplikácií nie je v úplnej zhode s RFC[1] špecifikáciami. Ďalej zistili, že väčšina FP alarmov sa nevzťahuje na bezpečnostné problémy , ale na politiky riadenia. Nakoniec zistili aj veľmi vysoké percento FN pre dávno známe útoky (napr. pretečenie vyrovnávacej pamäti (*buffer overflow*)).

[1] *Request for Comments*

3.1 Metódy detekcie

Metódy ID možno klasifikovať do troch hlavných kategórií: *signature-based detection* (ďalej SD, viď. časť 3.1.1), *anomaly-based detection* (ďalej AD, pozri podkapitolu 3.1.2) a *stateful protocol analysis* (ďalej SPA, pozri 3.1.3) [23]. V praxi sa reálne používa ich kombinácia v podobe hybridnej metódy (pozri časť 3.1.4). Nasleduje popis princípu, výhod a nevýhod spomenutých metód, ktorý vychádza z [23].

3.1.1 Detekcia na základe signatúr

Signatúra je vzor (alebo reťazec), ktorý korešponduje so známym útokom, alebo hrozbou. Signatúra teda popisuje útok. Varovanie vznikne, ak je aktuálna udalosť identifikovaná ako známe preniknutie (útok) [25]. SD sa snaží odhaliť prípadné preniknutie porovnávaním signatúr so zachytenou udalosťou. Táto metóda využíva nazhromaždené poznatky o špecifických útokoch a zraniteľnostiach systémov. Odtiaľ pochádzajú aj alternatívne názvy *knowledge-based detection* (detekcia na základe znalostí), alebo *misuse detection* (detekcia zneužitia).

Je to najjednoduchšia a efektívna metóda na detekciu známych útokov. Metóda SD robí detailnú analýzu kontextu. Prístup SD poskytuje pre človeka dobre čitateľný popis udalostí preniknutia [25]. Je to práve vďaka tomu, že tieto modely sú zvyčajne vytvorené bezpečnostnými expertmi (s postupným využívaním ML to však prestáva platiť).

Nedostatkom tejto metódy je neefektívna detekcia neznámych, *evasion*[2] a variant známych útokov. SD má slabé znalosti o stavoch a protokoloch. Je náročné udržiavať signatúry aktuálne vzhľadom na stále novo vznikajúce útoky. Časová náročnosť udržiavania znalostí tiež nie je zanedbateľná. IDS založené na SD však nie sú účinné proti novým útokom, pre ktoré ešte nie sú dostupné signatúry [25].

3.1.2 Detekcia na základe anomálií

Anomália je odchýlenie od známeho správania. Je tam definovaný model normálnej (legitímnej) aktivity. Varovanie je vyvolané aktivitou, ktorá sa nejakým spôsobom vychyľuje od normálnej aktivity [25]. Profil reprezentuje normálne/očakávané správanie odvodené zo sledovania bežnej činnosti legitímnych užívateľov, sieťových spojení, počítačov alebo užívateľov v priebehu času. Existujú statické, alebo dynamické profily. AD porovnáva profil normálneho správania s pozorovanou udalosťou, aby rozoznala možný útok. Niektoré zdroje nazývajú tento spôsob detekcie ako *behavior-based detection* (detekcia podľa správania).

Ide o metódu, ktorá je efektívna na detekciu nových a dovtedy nepredvídaných zraniteľností. Je menej závislá od operačného systému. AD zjednodušuje detekciu zneužitia oprávnení.

Vzhľadom na kontinuálnu zmenu sledovaných udalostí nie je presnosť profilov príliš vysoká. Počas zostavovania profilov správania nie je AD dostupná.

3.1.3 Kontrola stavu protokolu

SPA kontroluje obsah paketov a vie/môže sledovať stavy protokolu (napr. párovanie žiadosti a odpovede pri zostavovaní spojenia). Na rozdiel od AD, je SPA závislé od všeobecných profilov, ktoré poskytol dodávateľ IDS pre každý špecifický protokol. Vo všeobecnosti sú modely

[2] Tento druh útokov sa snaží skrývať/vyhýbať odhaleniu (pozri časti 5.2 a 5.4.1).

sieťových protokolov v SPA založené na štandardoch protokolov (napr. od IETF[3]). SPA sa zvykne nazývať aj *specification-based detection* (detekcia založená na špecifikáciách).

Veľkou výhodou SPA je znalosť stavov protokolu. Táto metóda vie dobre odhaliť neočakávané sekvencie príkazov.

Problémom tejto metódy je náročnosť na zdroje, kvôli sledovaniu a analýze stavu protokolu. Nie je schopná odhaliť útoky, ktoré vyzerajú ako bežná sieťová prevádzka (vykazujú bežné správanie protokolu). Slabinou je aj problematická kompatibilita s jednoúčelovými operačnými systémami alebo prístupovými bodmi (*APs*).

3.1.4 Hybridná metóda

V skutočnosti je väčšina reálne nasadených IDS systémov hybridných. Používajú naraz niekoľko metód detekcie pre lepšiu schopnosť detekcie útokov. (Napríklad SD a AD sa navzájom dopĺňajú, pretože jedna sa venuje známym starším hrozbám/útokom a druhá skôr novým/neznámym útokom.) [23]

Vo všeobecnosti môžu byť klasifikačné modely navrhnuté ako kombinácia oboch prístupov (AD a SD) za účelom využitiach výhod každého z nich [26]. Keď sú dostupné reprezentatívne príklady legitímnych i škodlivých vzorov, možno s pomocou techník ML vytvoriť detekčný model, ktorý ich bude rozlišovať. Takýto model môže obsahovať informácie týkajúce sa vzorov preniknutia i normálnej sieťovej prevádzky pre ich odlíšenie. Do istej miery popisuje tento detekčný model rozdiely medzi nimi, nie vzory preniknutia/normálnej prevádzky. Toto umožňuje zvýšiť presnosť SD a dokonca odhaliť úplne nové, nikdy nevidené útoky (*zero-day*[4]). [25]

3.2 Typy technológií IDS

Dnes existuje množstvo typov IDS technológií. Autori článku [23] kategorizujú tieto technológie do štyroch tried, podľa toho, kde sú nasadené a aký typ udalostí môžu rozpoznať. V nasledujúcich podkapitolách sú popísané ich princípy, výhody i nevýhody.

3.2.1 Hosťovské IDS

Host-based IDS (ďalej HIDS) monitoruje a zbiera štatistiky pre: počítače (*hosts*) obsahujúce citlivé informácie, podozrivé aktivity a servery na ktorých bežia verejne dostupné služby.

Softvérový agent HIDS je nasadený *inline*[5]. Ako jediné z IDS technológií dokáže HIDS analyzovať aj šifrovanú komunikáciu medzi koncovými zariadeniami.

Medzi nedostatky patrí nedostatočná znalosť kontextu a z toho vyplývajúca nižšia dosahovaná presnosť detekcie. Nevýhodou sú aj oneskorenia pri generovaní varovaní a centralizované hlásenie správ. HIDS konzumuje zdroje zariadenia, na ktorom beží a problémové sú aj konflikty s existujúcimi bezpečnostnými riešeniami. V neposlednom rade, HIDS je schopné detegovať len prieniky na danom zariadení, kde je nasadený.

[3] *Internet Engineering Task Force*

[4] *Zero-day* útok je útok, ktorý využíva doteraz neznámu chybu zabezpečenia, ktorú vývojári nemali čas vyriešiť a opraviť [27].

[5] Sieťová prevádzka musí prechádzať cez agenta [22].

3.2.2 Sieťové IDS

Network-based IDS (ďalej NIDS) pomocou senzorov zachytáva sieťovú prevádzku v rámci určitej časti siete. Následne analyzuje aktivity aplikácií a protokolov, aby rozpoznal podozrivé incidenty.

Senzor NIDS môžno nasadiť *inline* i pasívne[6]. Vzhľadom na spôsob nasadenia má schopnosť detegovať prieniky v rámci celej podsiete (ktorú monitoruje) a zariadení v nej obsiahnutých. NIDS je schopné analyzovať najširší rozsah aplikačných protokolov.

Nedokáže sledovať bezdrôtové protokoly. Vysoká úroveň FP i FN a neschopnosť detegovať útoky v rámci šifrovanej sieťovej komunikácie patrí medzi ďalšie limitácie tejto technológie. Nakoniec môže byť problematická aj úplná analýza počas vysokého zaťaženia siete. NIDS to jednoducho nemusí stíhať a niektoré pakety musí zahadzovať.

3.2.3 Bezdrôtové IDS

Wireless-based IDS (ďalej WIDS) je prakticky variant NIDS, ale zachytáva bezdrôtovú sieťovú komunikáciu. WIDS má pasívny senzor. Sleduje, a je schopné detegovať, viacero bezdrôtových zariadení. Sú to prakticky všetky zariadenia, s ktorými dokáže komunikovať, resp. pasívne odpočúvať ich komunikáciu. Práve vďaka svojej úzkej špecializácii na bezdrôtovú komunikáciu je WIDS úspešnejšie pri monitorovaní tohto typu komunikácie.

WIDS nemôže monitorovať aktivity protokolov na aplikačnej, transportnej ani sieťovej vrstve. Senzory sú náchylné na útoky fyzickým rušením.

3.2.4 Behaviorálna analýza sieťovej komunikácie

Network Behavior Analysis (ďalej NBA) systém analyzuje správanie siete sledovaním sieťovej prevádzky, aby rozpoznal útoky s neočakávaným tokom paketov. Pri NBA sa väčšinou využíva pasívny senzor. Rovnako ako NIDS, dokáže NBA detegovať prieniky v rámci sledovanej podsiete. NBA má skvelé schopnosti detekcie na prieskumné skenovania siete (najčastejšie aktivita útočníka pri získavaní znalostí o sieti, bežiacich službách, atď.), DoS útoky a rekonštrukciu infekcií škodlivým softvérom. Hlavná nevýhoda NBA je oneskorenie v detekcii útokov. Spôsobuje ho prenos dát o sieťovej prevádzke do NBA v dávkach, a nie v reálnom čase. NBA sa ďalej venuje 4. kapitola.

3.2.5 Zmiešané IDS

Mixed IDS (ďalej MIDS) využíva viacero technológií k dosiahnutiu presnejšej detekcie útokov (oproti samostatným prístupom) [23]. Reálny (resp. aspoň zverejnený v rámci akademického výskumu) MIDS sa *(zatiaľ)* nepodarilo nájsť, ale nie je ťažké si taký systém predstaviť. Napríklad kombináciou HIDS, NIDS a WIDS (ideálne od jedného výrobcu) by bolo možné získavať množstvo informácií o aktivite v podnikovej sieti aj na konkrétnych počítačoch. Samostatné HIDS/NIDS často nemajú dostatok informácií pre kvalifikované rozhodnutie, či je nejaká aktivita škodlivá/legitímna. Takéto MIDS by bolo jedno z možných riešení.

[6] Pasívny senzor monitoruje kópiu sieťovej prevádzky. Cez senzor v skutočnosti žiadna neprechádza [22].

Kapitola 4

Behaviorálna analýza sieťovej komunikácie

Odvetvie behaviorálnej analýzy sieťovej komunikácie (ďalej *NBA*) v podstate dostaneme spojením dvoch oblastí. Prvou je detekcia sieťových útokov, a to najmä detekcia na základe anomálií (ďalej AD). Druhou oblasťou je klasifikácia sieťovej prevádzky, a to hlavne s využitím strojového učenia (ďalej ML).

Je tu značné úsilie vyvinúť metódy detekcie založené na nových metrikách pre popis sieťovej prevádzky, ktoré identifikujú vlastnosti spojenia. Vďaka tomu je možná skorá identifikácia vznikajúcich bezpečnostných incidentov, rýchla detekcia infekcií vrámci vnútornej siete, alebo okamžitá prevencia vznikajúcich útokov. [28]

Ďalší výskum sa zameral na signatúry správania. Vďaka tomu boli vyvinuté viaceré metódy dolovania z dát, ktoré definujú sady metrík popisujúce fázu útoku (charakteristiku jeho správania). Tieto metódy využívajú buď štandard NetFlow, alebo sieťové pakety. NetFlow sa však ukázal ako nedostatočný zdroj informácií o priebehu útoku. Výskumníci si preto začali vytvárať ich vlastné metriky, ktoré umožnili získať viac informácií a kontextu k analyzovaným spojeniam. [28]

Ďalej budú uvedené príklady existujúcich NBA (podkapitoly 4.1 a 4.2). V podkapitole 4.3 bude uvedené ich porovnanie z niekoľkých pohľadov.

4.1 Diskriminátory (na klasifikáciu tokov)

Moore a kol. [29] sa zamerali výhradne na TCP protokol a TCP toky. TCP je stavový protokol, ktorý má presne ohraničený začiatok i koniec. Naopak z princípu protokolu UDP je problematické ohraničiť začiatok/koniec toku. UDP v ich práci autori nepoužívali.

V najjednoduchšom prípade je tok úplne definovaný, keď je zaznamenané úplné zostavenie aj ukončenie TCP spojenia. Pri definovaní toku sa objavujú komplikácie, keď je zostavenie/ukončenie spojenia abnormálne. Autori to vyriešili vďaka množine pravidiel vstavaných do nástroja `netdude`[1]. [29]

Diskriminátory poskytujú širokú škálu rysov na popísanie tokov. Každý rys je dostupný pre oba smery obojsmernej sieťovej prevádzky. Tam, kde to dáva zmysel je dostupná aj štatistika pre celý tok. Nasledujú príklady diskriminátorov [16, 10, 29]:

- metriky toku (trvanie, počet paketov, celkový počet bajtov),

[1] Netdude [30] implementuje TCP stavový protokol v dostatočne robustnej miere aj pri strate paketov.

- doba príchodu medzi paketmi (priemer, rozptyl, 1. a 3. kvartil, medián, minimum, maximum,...),
- veľkosť TCP/IP hlavičiek (priemer, rozptyl, 1. a 3. kvartil, medián, minimum, maximum,...),
- celkový počet paketov (v každom smere aj spolu pre celý tok),
- veľkosť dát (priemer, rozptyl, 1. a 3. kvartil, medián, minimum, maximum,...),
- zoradený zoznam desiatich najdôležitejších komponentov furiérovej transformácie časov medzi príchodmi paketov (pre každý smer) a
- viacero hodnôt špecifických pre TCP (napr. celkový počet prenesených bajtov, celkový počet opakovaných prenosov, celkový počet paketov, celkový počet ACK paketov nesúcich SACK informáciu, minimálna veľkosť segmentu,...).

4.2 Pokročilé bezpečnostné sieťové metriky na popis vektoru útoku

Prvý krát boli tieto metriky uvedene v diplomovej práci [31]. Bola navrhnutá metóda pre extrakciu dát zo sieťovej prevádzky a kontextové oddelenie neúplných spojení s využitím sady metrík, ktoré vytvárajú signatúru definujúcu správanie pripojenia. Aktuálna vylepšená verzia bola uvedená v článku [28].

Homoliak a kol. [28] definujú metódu na generovanie signatúr sieťového správania zo sady sieťových bezpečnostných metrík – Advanced Security Network Metrics (ďalej ASNM). Pozostávajú zo 167 metrík rozdelených do 5 kategórií podľa ich vlastností. Metriky popisujú vlastnosti detekovaného útoku – jeho správanie. Tieto metriky zlepšujú schopnosť detekcie potenciálnych útokov len zo sledovania sieťovej prevádzky. Metóda je založená na extrakcii rôznych rysov z každého analyzovaného TCP spojenia.

4.2.1 Definícia metrík

Všetky metriky boli definované tak, aby popisovali vlastnosti, procesy a správanie sieťových útokov, alebo legitímnych TCP spojení. Spolu používajú 167 metrík na získanie čo najlepšej signatúry TCP spojenia. Kategórie metrík sú pomenované podľa ich princípov. Nasleduje krátky popis týchto kategórií. [28]

Štatistické metriky

Identifikujú štatistické vlastnosti TCP spojenia. Zohľadňujú sa všetky pakety TCP spojenia. Určujú sa štatistiky ako počet, medián, priemer, štandardná odchýlka, pomery niektorých položiek hlavičiek v pakete alebo v paketoch samotných. Tieto metriky čiastočne používajú časovú reprezentáciu výskytov paketov. Zahŕňajú tak dynamické vlastnosti analyzovaného TCP spojenia, ale bez akéhokoľvek kontextu. Väčšina týchto metrík rozlišuje prichádzajúce a odchádzajúce pakety analyzovaného TCP spojenia. Celkovo bolo definovaných 50 štatistických metrík. [28]

Dynamické metriky

Skúmajú dynamické vlastnosti TCP spojenia ako rýchlosť, alebo chybovosť prenosového kanálu. Tieto vlastnosti môžu byť reálne, alebo simulované. Kontext analyzovaného TCP spojenia je zohľadnený v 14 dynamických metrikách. Celkovo bolo definovaných 32 dynamických metrík. Veľa z týchto metrík odlišuje prichádzajúce/odchádzajúce pakety, zohľadňuje štatistické vlastnosti paketov a ich veľkosti. [28]

Lokalizačné metriky

Sledujú statické vlastnosti TCP spojenia. Reprezentujú umiestnenie zúčastnených strán a porty, ktoré použili pri komunikácii. Niektoré metriky vyjadrujú lokalitu nepriamo príznakom, ktorý rozlišuje či zúčastnené strany boli v lokálnej sieti. Lokalizačné metriky nezohľadňujú kontext spojenia, ale rozlišujú jeho smer. Celkovo definovali 8 lokalizačných metrík. [28]

Distribuované metriky

Charakteristikou týchto metrík je, že distribuujú pakety alebo ich dĺžky do pevného počtu intervalov (1s, 4s, 8s, 32s, 64s) za jednotku času. Táto kategória používa vektorovú reprezentáciu. Všetky metriky pracujú v kontexte analyzovaného TCP spojenia. Spolu definovali 34 metrík, ktoré sú výsledkom parametrizácie 2 funkcií. Tieto funkcie používajú parametre ako jednotka času, prah, smer a kontext spojenia. [28]

Behaviorálne metriky

Založené na popise behaviorálnych vlastností TCP spojenia. Ide napríklad o (ne)legálne ukončenie spojenia, počet tokov v určitom časovom intervale, polynomiálna aproximácia dĺžky paketov v časovej úseku, paralelné vytváranie nových služieb, alebo periodická komunikácia. Od predchádzajúceho článku [32] navrhli aj nové behaviorálne metriky. Medzi nimi napríklad:

- počet spoločných TCP spojení zúčastnených strán, pred analyzovaným TCP spojením (ohraničené časovým intervalom). Zohľadňuje to kontext analyzovaného TCP spojenia.
- Počet nových TCP tokov po začiatku analyzovaného TCP spojenia. Pracuje v kontexte analyzovaného TCP spojenia.
- Štandardná odchýlka časových intervalov medzi TCP spojeniami na rovnakých IP adresách a portoch.

Celkovo definovali 43 behaviorálnych metrík. Väčšina z nich používa smer analyzovaného TCP spojenia a 6 z nich zohľadňuje kontext. [28]

4.3 Porovnania

Homoliak a kol. [28] porovnali výkonnosť ASNM s diskriminátormi. Autori diskriminátorov i metrík uvažovali len TCP spojenia. Zistili, že len približne 20% definícií diskriminátorov je principiálne podobných, alebo rovnakých, ako v prípade nimi navrhnutých metrík. Medzi

unikátne vlastnosti definícií diskriminátorov patrí použitie kvartilov pre štatistickú analýzu, analýza selektívneho potvrdzovania TCP, duplikované pakety, a iné.

4.3.1 Experimenty a klasifikácia

Homoliak a kol. sa zameriavali len na presnosť klasifikácie jednotlivých metrík a diskriminátorov. Experimenty prispôsobili pre dosiahnutie maximálnej presnosti klasifikácie vstupných dát. [28]

Najlepšie výsledky (s ASNM) boli dosiahnuté viacerými metrikami polynomiálnej aproximácie. Väčšinou boli lepšie odchádzajúce spojenia. Pri diskriminátoroch boli najlepšie výsledky dosiahnuté diskriminátorom priemernej veľkosti segmentu v smere od klienta k serveru. Dôvodom bude asi fakt, že exploit obsahuje veľké množstvo dát nutných pre útok *buffer overflow* a tieto dáta sú fragmentované. Prezentované výsledky v tomto článku ukazujú podobné detekčné schopnosti oboch skupín rysov (ASNM i diskriminátory). [28]

Rozšírením množiny rysov použitých ku klasifikácii (použili naraz diskriminátory a metriky) dosiahli presnosť 99,9%. Úspešne tak zlepšili úroveň detekcie o 0.9% od predchádzajúcej [16] špičkovej klasifikačnej metódy (bolo spomenuté v podkapitole 2.6.1). [28]

Autori [33] porovnávali výkon Naivného Bayesovského klasifikátora s použitím ASNM a diskriminátorov. Obe množiny rysov reprezentujú akademické experimentálne NBA založené na štatistickej a behaviorálnej analýze tokov sieťovej prevádzky. Pri polynomiálnej klasifikácii dosahovala presnosť ASNM 98,85% a presnosť s diskriminátormi bola 93,74%. Pri ďalšom experimente klasifikovali do troch tried (legitímna sieťová prevádzka, priame útoky a obfuskované útoky). Tu dosiahli ASNM presnosť 99,69% a diskriminátory 98,12%. Oba experimenty demonštrovali lepšiu presnosť pri použití ASNM.

Kapitola 5

Útoky proti IDS

Útočník využíva rôzne techniky, aby sa vyhol detekcii jeho činnosti pomocou bezpečnostných sieťových zariadení (firewall, IDS,...). Má na to zvyčajne aspoň jeden z týchto dôvodov [34]:

- Útočník chce vykonať nejakú akciu a nechce, aby o nej správca zabezpečenia vedel (alebo chce, aby ju ignoroval). Príkladom takejto akcie môže byť kompromitácia lokálneho systéme, odcudzenie dát, atď. Ak bola činnosť útočníka odhalená, administrátor môže vykonať preventívne protiopatrenia, opraviť škody, sledovať útok, alebo podniknúť iné odvetné kroky. A práve tomu sa chce útočník zvyčajne vyhnúť.
- Útočník chce urobiť niečo, čomu by bezpečnostné nástroje/zariadenia (napr. IPS) za normálnych okolností zabránili. Typicky ide o použitie nejakej aplikácie/protokolu, alebo prístup k niektorým zdrojom (počítačom), ktorý nie je povolený.

V tejto kapitole je najskôr popísaná všeobecná architektúra IDS (5.1) a jej vzťah k možným útokom na IDS. V sekcii 5.2 je spomenutá klasifikácia útokov na IDS. Ďalšie časti popisujú niektoré vybrané útoky (viď časti 5.3 a 5.4). Nakoniec je popísaný návrh niektorých postupov na obídenie NBA (a všeobecne IDS), ktoré budú realizované v ďalšej časti práce.

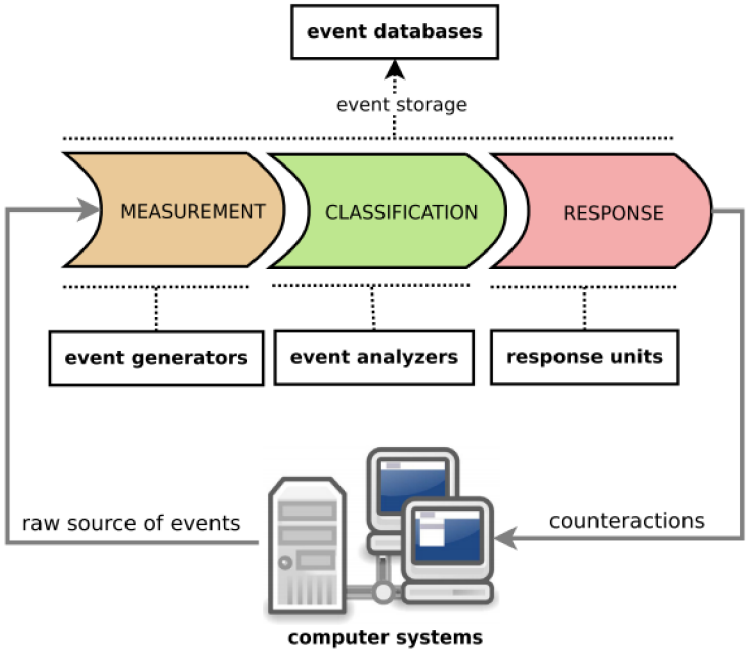
5.1 Všeobecná architektúra IDS

Doteraz bolo navrhnutých mnoho architektúr IDS. Väčšina z nich je založená na štyroch komponentoch. Sú to: generátory udalostí (*event generator*s), analyzátory udalostí (*event analyzer*s), jednotky reagujúce na tieto udalosti (*response unit*s) a databázy udalostí (*event database*s) (viď obrázok 5.1). Pre útoky na IDS sú zaujímavé najmä tri hlavné fázy činnosti IDS [25]:

Meranie – generátory udalostí : Vzor (*pattern*) udalosti je charakterizovaný rysmi o nameranej udalosti. Tieto vzory sú navrhnuté tak, aby presne rozlišovali preniknutie od legitímnej aktivity.

Klasifikácia – analyzátory udalostí : Pre klasifikáciu (typicky v reálnom čase) vzoru udalosti (či ide o preniknutie alebo legitímnu činnosť) sú použité preddefinované modely. Tie popisujú vzory preniknutia a/alebo legitímnej aktivity.

Reakcia – jednotky reagujúce na udalosti : Keď je detekovaný vzor preniknutia, je vyvolaný alarm a (v prípade IPS) je vykonané obranné opatrenie, aby zostali počítačové systémy v bezpečí.



Obr. 5.1: Všeobecné fázy činnosti IDS a príbuzných komponentov [25].

Tieto tri fázy môžno považovať za „vstupné brány“ pre útočníka a jeho útok na IDS.

5.2 Klasifikácia útokov proti IDS

Podľa autorov článku [25] doteraz neexistovala systematická kategorizácia útokov proti IDS. Autori identifikovali šesť hlavných cieľov útoku na IDS:

Evasion : Vzor preniknutia je vhodne upravený tak, aby ho IDS nebolo schopné detegovať (tzn., že útok nespustí alarm).

Overstimulation : Je vytvorených mnoho vzorov tak, aby IDS generovali falošné varovania. To preťaží bezpečnostných operátorov a/alebo analyzátorov týchto falošných alarmov. Potom sa môže ľahko stať, že prehliadnu skutočné útoky.

Poisoning : Do sady dát sú vložené vhodne upravené vzory. Cieľom je oklamať/pomýliť učiaci algoritmus a negatívne ovplyvniť výkon IDS vo fáze klasifikácie (napr. znížením presnosti detekcie). Tieto útoky predstavujú pomerne novú a komplexnú hrozbu, ktorá získava veľký záujem v ML a bezpečnostnej komunite.

Denial of Service **(DoS)** : Odopretie služby v tomto prípade predstavuje zabránenie detekčnej činnosti IDS vygenerovaním vhodne vytvorených vzorov tak, aby došlo k preťaženiu senzoru IDS, alebo spomaleniu jeho „*pattern-matching*“ algoritmu. V prípade IDS pracujúceho v *inline* režime[1] môže tento útok spôsobiť značné zahadzovanie paketov, alebo ich opozdenie. Potom už ide o DoS proti monitorovaným systémom. Tento útok možno tiež využiť na posilnenie *evasion* útoku.

[1] Sieťové pakety nie sú smerované skôr, ako ich preskúmalo IDS.

Response Hijacking : Útočník vytvorí vzor, ktorý spôsobí vygenerovanie nesprávneho popisu udalosti (varovania) a pomýlenie/oklamanie mechanizmu IDS, ktorý reaguje na tieto varovania. Tento mechanizmus môže byť automatický, alebo vykonávaný bezpečnostným operátorom. Cieľom môže byť napríklad neoprávnené blokovanie legitímnych sieťových spojení.

Reverse Engineering : Reverzné inžinierstvo tu predstavuje prípad, kedy si útočník môže nazhromaždiť informácie o vnútornej činnosti IDS (napr. používané rysy, alebo detekčný algoritmus). Tieto informácie možno využiť k vytvoreniu efektívneho útoku, ktorý sa zameria na dosiahnutie jedeného z vyššie uvedených cieľov. Príkladom takéhoto útoku môže byť stimulovanie IDS s dobre známymi útokmi. Útočník si pritom zbiera zodpovedajúce odpovede IDS, aby si neskôr mohol zrekonštruovať detekčnú techniku, ktorú zacielené IDS používa.

Tieto útoky môžu využívať rôzne zraniteľnosti IDS systémov. Avšak, väčšina týchto zraniteľností môže byť jasne asociovaná s jednou z fáz popísaných v časti 5.1. Corona a kol. preto navrhujú kategorizovať zraniteľnosti IDS podľa fázy činnosti IDS, ku ktorej prislúchajú. Cieľ útoku a kategória zneužitej zraniteľnosti môže byť použitá ako rámec na kategorizáciu útokov proti IDS systémom. [25]

5.3 Útoky na fázu merania

V tejto časti sú uvedené príklady na útoky využívajúce nedostatky IDS vo fáze merania. Najskôr je to útok tunelovaním (5.3.1) a potom pomocou segmentácie (5.3.2).

5.3.1 Tunelovanie

Škodlivú sieťovú prevádzku môžno zapuzdriť do „tunela“ (tzn. do druhu sieťovej prevádzky, ktorú senzor IDS neskúma, alebo ani nevidí). Ide o najširšie používanú techniku vyhýbania sa blokovaniu, sledovaniu, atď. V skutočnosti používa tunelovanie aj mnoho bežných používateľov (mnohí z nich si to pritom ani neuvedomujú). Firemný a vládny zamestnanci tiež často používajú tunelovanie (napr. VPN). V týchto prípadoch väčšinou nejde o nekalé úmysly, ale ich cieľ je prakticky rovnaký – vyhnúť sa po ceste (paketov po sieti) detekcii a kontrole, alebo obísť miestne politiky. [34]

Takmer všetky siete umožňujú nejaký typ prichádzajúcej/odchádzajúcej komunikácie. Potenciálny útočník to môže využiť k vytvoreniu tunela. Metódy tunelovania možno rozdeliť do troch hlavných kategórií [34]:

tunelovanie cez neštandardný port : využíva predpoklady IDS, o väzbách medzi aplikáciami, protokolmi a číslami portov. IDS môže (nesprávne) ignorovať sieťovú prevádzku, ktorá prechádza cez známy TCP port (predpokladá tam len legitímne toky). To by už dávno nemalo platiť, ale určite sa ešte nájdu prípady, kde by sa to dalo využiť.

všeobecné zapuzdrenie (tunelovanie) : funguje vložením jedného protokolu do iného.

šifrovaný tunel : využíva šifrovanie k vytvoreniu dátového toku, ktorý nemožno (až na niektoré špeciálne/extrémne prípady) *(až na výnimky ako „NSA špehovanie“)* kontrolovať počas prenosu. Príkladom takéhoto útoku môže byť [35].

NBA sieťových zraniteľností využívajúcich obfuskáciu cez HTTPS tunel

Sofistikovanejšie IDS by sa už dnes nemalo nechať tak ľahko oklamať, alebo dokonca automaticky ignorovať takéto toky. Príkladom takého IDS (NBA) môžu byť články [36, 33]. Toto už síce spadá skôr do oblasti útokov na klasifikačnú časť (5.4), ale tiež sa to hodí aj sem.

Článok [36] uvažuje využitie už napadnutého počítača, ktorý slúži ako prostredník medzi útočníkom a novými zraniteľnými cieľovými počítačmi. Tento počítač tunelovaním obfuskuje celú komunikáciu s útočníkom. Autori používajú obfuskáciu, ktorá bola navrhnutá v diplomovej práci [37].

Tento článok skúma detekčné vlastnosti obfuskovaných sieťových *buffer overflow* útokov vybraných NIDS (Snort) a NBA (AIPS[2]). Obfuskácia bola urobená tunelovaním škodlivej prevádzky v protokoloch HTTP a HTTPS. Cieľom bolo simulovať typické vlastnosti legitímnej prevádzky HTTP. [36]

Ukázali, že SNORT bol schopný detegovať priame (netunelované) útoky. Naopak obfuskované nevedel odhaliť vôbec. [36]

V prvom experimente s AIPS klasifikovali obfuskovanú škodlivú sieťovú prevádzku klasifikátorom, ktorý bol natrénovaný len pomocou priamych (neobfuskovaných) útokov a legitímnej sieťovej prevádzky. Tým chceli otestovať detekciu obfuskovaných útokov. Napriek mnohým snahám dosiahla klasifikácia len 0,00% úspešnosť. To ukazuje, že behaviorálne a štatisticky založené NBA nie je schopné detegovať obfuskované útoky bez akejkoľvek predchádzajúcej informácie o nich. Keď zahrnuli do trénovacích dát aj obfuskované útoky, dosiahli presnosť 97,64% $\pm$ 0,45% v prípade binominálnej klasifikácie, resp. 98,87% $\pm$ 0,99% s polynominálnou klasifikáciou. [36]

Preukázali aj rozdielne štatistické a behaviorálne charakteristiky priamych útokov v porovnaní s obfuskovanými. Autori zdôrazňujú potrebu trénovania štatisticky a behaviorálne založených NBA s rozdielnymi technikami obfuskácie (a jej modifikáciami), aby posilnili schopnosti detekcie takýchto útokov. [36]

5.3.2 Segmentácia

Škodlivý tok je rozdelený do niekoľkých častí, zaslaných v takom poradí, že rekonštrukcia v rámci IDS sa líši od tej, čo sa deje na strane cieľového počítača. Úspech tejto techniky je založený na skutočnosti, že rôzne operačné systémy, môžu spracovávať duplicitné, alebo prekrývajúce sa fragmenty s rôznymi politikami rekonštrukcie, zatiaľčo sieťové senzory (IDS) často využívajú jediný spôsob rekonštrukcie, o ktorej sa predpokladá, že je v súlade so všetkými operačnými systémami. [25]

Väčšie riziko je však v súčasnosti na aplikačnej vrstve. Veľa rozšírených aplikačných protokolov, ako je RPC[3] a SMB[4] umožňujú vlastnú segmentáciu. Skúsenosti však ukázali, že mnoho implementácií (IDS) toto nevie robiť dobre, alebo dokonca vôbec. [34]

Obrana proti takémuto postupu môže byť obtiažna. Ak má senzor niektoré obmedzenia na počet segmentov (a to má prakticky vždy, najmä v režime *inline*), ktoré dokáže uchovávať, alebo na to, ako dobre sa vie prispôsobiť segmentácii, môžu to útočníci využívať. [34]

[2] Tento systém, nazvaný ako *Automated Intrusion Prevention System* (AIPS), bol popísaný v článku [32]. Využíva ASNM a ako zdroj expertných znalostí pre ML používa honeypoty.

[3] *Remote Procedure Call* [38]

[4] *Server Message Block* [39]

5.4 Útoky na klasifikačnú časť

Cieľom tejto časti IDS je klasifikovať každú udalosť (presnejšie vzor udalosti) ako preniknutie, alebo legitímnu aktivitu.[5] Pri výskyte preniknutia by IDS malo poskytnúť i spoľahlivú, a pre ľudí dobre čitateľnú, interpretáciu každého vzoru preniknutia (napr. zacielenú zraniteľnosť, hrozby spôsobené touto zraniteľnosťou, pravdepodobný cieľ útočníka). Klasifikácia sa väčšinou vykonáva v reálnom čase, kde je testovací vzor porovnaný s jedným (alebo i viacerými) modelom popisujúcim vzory preniknutia (*signature detection* (ďalej SD), viď aj sekciu 3.1.1) a/alebo vzory legitímnej sieťovej prevádzky (*anomaly detection* (ďalej AD), viď aj sekciu 3.1.2). [25]

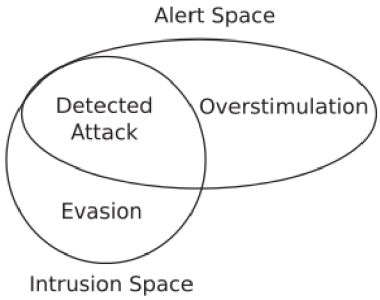
Vzhľadom na stále komplexnejšie hrozby a rýchlo sa rozvíjajúce prostredie je v dnešnej dobe čisto manuálna definícia klasifikačných modelov nevhodná. Preto v posledných rokoch dochádza k nárastu používania algoritmov ML na podporu efektívneho prispôsobenia modelov detekcie. [25]

Algoritmy ML sa môžu automaticky naučiť/vytvoriť presný model na detekciu, ak dostanú štatistický reprezentatívnu množinu vzorov vzťahujúcich sa k preniknutiu a/alebo legitímnym aktivitám. Štatistická reprezentatívnosť sa premieta do troch hlavných praktických problémov [25]:

- **Súkromie** – Vzory udalostí môžu obsahovať citlivé informácie o používateľovi počítačového systému. To môže viesť k obavám o súkromie, keď sa zhromažďujú dáta o legitímnej aktivite používateľa.
- **Reálne prieniky** – Zhromažďovať by sa mali aj čo najaktuálnejšie prieniky z reálneho sveta. Toto je veľmi náročná podmienka, pre veľmi rýchly vývoj prienikov.
- **Presnosť trénovacích dát** – Trieda (označenie) každého vzoru v trénovacích dátach by mala byť dôkladne validovaná, aby sa pokiaľ možno obmedzila prítomnosť šumu v dátach (napr. legitímna prevádzka označená ako prienik). Táto úloha často vyžaduje detailné znalosti ľudského experta (tzn. predchádzajúce znalosti o konkrétnych bezpečnostných problémoch) a to môže byť náročné. Obzvlášť pri veľkom množstve dát. Toto je hlavným dôvodom, prečo sú *poisoning* útoky uskutočniteľné.

Colona a kol. definujú množinu všetkých vzorov preniknutia (*intrusion space* (I)) a množinu vzorov, ktoré vyvolajú alarm (*alert space* (A)), viď. obrázok 5.2. Pre obranu (IDS) je cieľom maximalizovať prienik oboch množín, tzn. znížiť priestor na úspešný útok. Najmä podmnožina $I-A$ (nezachytené alarmy) a $A-I$ (falošné alarmy) môžu byť využité pre útoky *evasion*, resp. *overstimulation*. Tabuľka 5.1 ukazuje generovanie týchto vzorov v závislosti od použitej metódy detekcie. [25]

[5] Teda v prípade binominálnej klasifikácie. Pri polynomiálnej klasifikácii môže ísť napríklad o klasifikáciu útokov a legitímnej sieťovej prevádzky do viacerých tried podľa jednotlivých sieťových služieb.



Obr. 5.2: Všetky vzory preniknutia (I) a všetky vzory spôsobujúce varovania (A) [25].

Cieľ útoku	Model detekcie	
	SD	AD
evasion	Útočník môže zmeniť útok tak, že vzor toho útoku sa nezhoduje so žiadnou signatúrou.	Útočník môže upraviť útok tak, že sa tvári ako legitímna prevádzka (*mimicry* útok). Príkladom je PBA (5.4.1).
overstimulation	Útočník vygeneruje vzory udalostí, ktoré zodpovedajú jednej, alebo viacerým signatúram, ale v skutočnosti nepredstavujú hrozbu pre sledované systémy.	Útočník predloží neštandardné vzory, ktoré v skutočnosti nie sú hrozbou pre sledované systémy. (Tento útok doteraz nikto neoznámil, resp. nie je informácia, že by ho niekto reálne vykonal.)

Tabuľka 5.1: Zhrnutie kľúčových útokov proti IDS založeným na SD a AD [25].

Nasleduje všeobecný prehľad vybraných útokov proti klasifikačnej fáze.

5.4.1 *Evasion* útoky

Autori článku [40] realizovali obfuskáciu sieťových útokov a navrhli novú podtriedu *mimicry* útokov. Nazývajú ju *polymorphic blending attacks* (ďalej PBA). Tieto útoky sa môžu efektívne vyhnúť AD IDS založeným na frekvencii bajtov (*byte frequency-based network anomaly IDS*). Útoky dôkladne upravujú štatisticky svojich paketov, aby kopírovali legitímnu sieťovú prevádzku. Autori demonštrovali efektivitu PBA útokov na PAYL.

5.4.2 *Poisoning* útoky

Perdisci a kol. [41] ukázali, že vloženie jedného „*poisoning*“ červa na každú vzorku červa (tzn. 50% šum v príkladoch prienikov) urobí Polygraph (SD IDS) prakticky zbytočným.

5.5 Navrhnuté metódy na obídenie behaviorálnej analýzy sieťovej komunikácie

Nasledujúce techniky boli navrhnuté na základe: plánovaného použitia ASNM a konzultácií s vedúcim práce. Corona a kol. [25] ukázali množstvo rôznych prístupov ako obísť IDS. Avšak, mnoho z nich nebolo relevantných pre túto prácu. Bolo to najmä z dôvodu uvažovaného spôsobu nasadenia/testovania (viď 6. kapitola). Napríklad DoS útok na NBA by si vyžadoval reálne nasadený systém. ASNM skúmajú len hlavičky paketov. To znamená, že keď vôbec neskúmajú obsah paketov, nemala by veľký zmysel napríklad obfuskácia pomocou PBA[6]. Snahou je vytvoriť nástroj, ktorý bude vedieť vytvoriť/vygenerovať viacero rôznych obfuskovaných útokov. Prioritou je aj počet a pestrosť rôznych metód. Nie len pár vybraných techník. Takto bude možné otestovať NBA na širšej vzorke útokov.

Ako sa ukázalo na príklade tunelovania útokov cez HTTP(S) (5.3.1), NBA bolo schopné detegovať tieto útoky len v prípade, že ich už predtým poznala (mala klasifikačný model natrénovaný s dátami, ktoré obsahovali takéto útoky). Predpokladom tak je, že aj ostatné obfuskované útoky budú pre NBA predstavovať problém, ak pre ňu budú úplne nové, alebo neznáme. Naopak, v prípade, že do trénovacej sady dát budú pridané aj obfuskované útoky, očakáva sa, že ich už NBA dokáže odhaliť (alebo aspoň väčšinu z nich). Cieľom je okrem vytvorenia nástroja na realizáciu týchto útokov aj overenie týchto predpokladov.

Bola navrhnutá sada obfuskácií, z ktorej je možné niekoľko z nich vybrať pre následnú realizáciu. Navrhnuté bolo využiť jednotlivé metódy samostatne a tiež ich v rôznej miere skombinovať, a tak vyskúšať obísť NBA kombináciou viacerých techník naraz. To by ju mohlo dostatočne spliesť a spôsobiť chybnú detekciu. Nasleduje navrhnutá sada obfuskácií.

5.5.1 Roztiahnutie paketov v čase

Realizovať rôzne časové rozdiely medzi príchodmi jednotlivých paketov. Simulovať trpezlivého útočníka. Skúsiť rôzne variácie tohto intervalu:

- od konštantného pozdržania každého paketu,
- cez nejaký kratší (podľa pseudonáhodných čísiel napr. v rovnomernom rozložení),
- až po extrémny prípad. V poslednom prípade skúsiť maximálne roztiahnuť priebeh útoku v čase (prispôsobiť ho tak, aby príjemca len s malou rezervou stíhal prijímať pakety pred vypršaním časového limitu).

Tento prístup by sa dal nazvať aj riadením sieťovej prevádzky (*traffic shaping*). Tu však ide o úplne iný zámer.

5.5.2 Segmentácia a fragmentácia

Toto sa dá realizovať na rôznych vrstvách. Napríklad na sieťovej (IP), transportnej (TCP), alebo aplikačnej. Ide to o snahu rozbiť *exploit* paket, pretože ten je väčšinou posledný a veľký. Nesie *exploit content*, aby spustil napr. *buffer overflow* útok. Jednou z možností, ako to dosiahnuť je modifikácia MTU[7].

[6] Tento útok modifikuje payload paketu a jeho realizácia je pomerne komplikovaná [40].
[7] Maximálna prenosová jednotka (Maximum Transmission Unit).

5.5.3 Zmena poradia paketov

Nasilu poprehadzovať pakety mimo poradia. Využiť napríklad protokoly s posuvným okienkom (*sliding window protocols*).

5.5.4 Simulácia chybového prenosového kanála

Nasilu poškodiť pakety tak, aby nesedel kontrolný súčet (*checksum*). Dôjde k novému prenosu a tým sa zmení tok paketov.

5.5.5 Zahadzovanie paketov

Zahadzovať pakety v rôznej miere za účelom následných opakovaných (i viac krát) prenosov (automaticky by malo dôjsť k premiešaniu paketov a podobne). Tento prístup by sa z časti dal nazvať termínom *traffic policing*, i keď tu ide opäť o iný cieľ.

5.5.6 Legitimná komunikácia súčasne s útokom

Podstatou je komunikovať s počítačom (na ktorý sa útočí) aj legitímnym spôsobom. Cieľom je, aby všetká komunikácia v čase útoku nepredstavovala len samotný útok, ale aby tam bolo pokiaľ možno viac legitímnej komunikácie, ako tej, čo patrí útoku[8]. Tento postup by sa dal zaradiť do kategórie *poisoning* útokov.

Toto pravdepodobne nebude využiteľné pri útokoch na všetky služby vzhľadom na obtiažnu simuláciu legitímnej prevádzky. Avšak, napríklad v prípade zraniteľného FTP serveru, by malo byť možné zautomatizovať legitímnu komunikáciu s FTP serverom.

5.5.7 Modifikácia škodlivého kódu

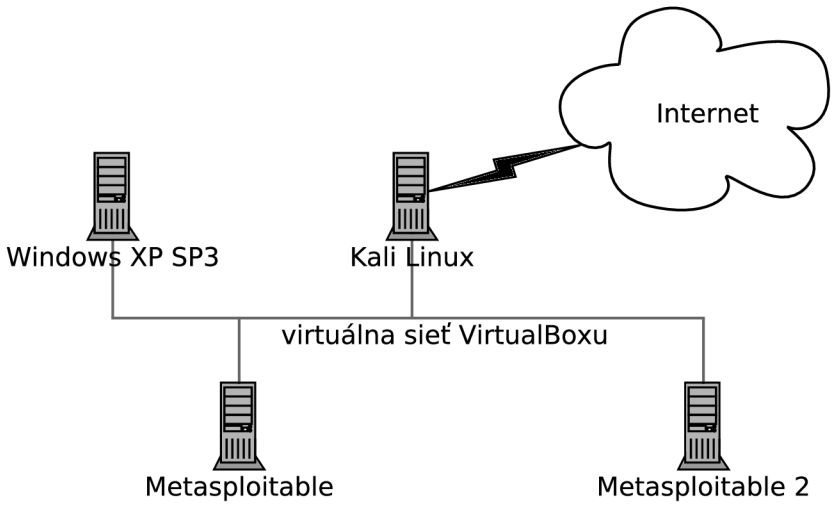
Vyskúšať rôzne techniky na maskovanie škodlivého kódu [42]. Od využitia existujúcich nástrojov (napr. ADMmutate, CLET,...) až po ručnú úpravu škodlivého kódu (*exploit content*-u).

[8] Tak to vyzerá aj v reálnych sieťach, kde je väčšinou legitímna komunikácia.

Kapitola 6

Použitá testovacia infraštruktúra a útoky na sieťové služby

Pre vykonávanie útokov na sieťové služby bolo zvolené prostredie virtuálnej siete (viď obrázok 6.1). Bol pri tom využitý virtualizačný nástroj VirtualBox [43]. Bolo to najmä z dôvodu nezákonnosti[1] a praktických problémov, ktoré sú spojené s vykonávaním sieťových útokov cez Internet. Ako stroj útočníka, bol použitý virtuálny počítač (ďalej VM) s nainštalovaným operačným systémom Kali Linux 1.1.0. Zraniteľné VM, ktoré reprezentovali ciele útokov používali nasledovné operačné systémy: Metasploitable[2], Metasploitable 2 a Windows XP Service Pack 3 (ďalej WinXPSP3). Na všetkých VM boli nastavené statické IP adresy, aby boli zaistené vhodné podmienky pre automatizáciu celého procesu. Na VM s WinXPSP3 bol ešte nainštalovaný Microsoft SQL Server 2005 a vypnutý firewall. Inak boli systémy nainštalované na VM ponechané v pôvodnej konfigurácii. Pre každý zraniteľný systém bol vytvorený snímok (*snapshot*) stavu VM pre prípadnú neskoršiu obnovu po vykonaní útoku.



Obr. 6.1: Použitá sieťová infraštruktúra.

[1] Výnimkou je napr. schválené penetračné testovanie.

[2] Metasploitable je zámerne zraniteľná verzia Ubuntu navrhnutá na testovanie bezpečnostných nástrojov a demonštráciu bežných zraniteľností [44].

Boli vykonané sieťové útoky na rôzne zraniteľné sieťové služby. Práca sa nezamerala na nejaký konkrétny druh útoku. Využité boli všetky dostupné prostriedky na získanie prístupu do systému[3]. Bežný útočník sa tiež nesústredí na spôsob, ako sa dostať do systému. Často využije všetky možnosti, ktoré mu systém/užívateľ umožní a neodmietne ani jednoduchú cestu do systému. Nasleduje stručný popis použitých sieťových služieb a ich zraniteľností. V hranatých zátvorkách sú uvedené hodnoty *Common Vulnerability Scoring System (CVSS)*. V prípadoch, kde je napísaných viac CVSS, je poradie zhodné s poradím v texte, tzn. napr. [slovníkový útok; exploitácia].

- Apache Tomcat 5.5 [CVSS: 7,5; CVSS: 10,0] – Najskôr bol vykonaný slovníkový útok na získanie prístupových údajov [45, 46]. Následne bola zneužitá aplikácia na správu tohto serveru k preneseniu a spusteniu škodlivého kódu [47, 48].

- Microsoft SQL Server 2005 [CVSS: 7,2; CVSS: 2,1] – S pomocou slovníkového útoku boli získané prístupové údaje [49, 50]. Ďalej bola využitá procedúra `xp_cmdshell`, ktorá umožňuje spustenie ľubovoľného kódu na takomto serveri [51, 52].

- Samba 3.0.20-Debian [CVSS: 6,0] – V tejto verzii Samby umožňuje MS-RPC funkcionalita v smdb vzdialené spustenie ľubovoľného kódu, ak je v konfigurácii povolená voľba „username map script“. Pri tomto útoku nie je potrebná autentifikácia. [53, 54]

- Server service (Windows XP SP3) [CVSS: 10,0] – Táto služba umožňuje útočníkovi vzdialene spustiť ľubovoľný kód cez vytvorený *RPC request*, ktorý spôsobí pretečenie zásobníku počas štandardizácie cesty (*path canonicalization*) [55, 56].

- PostgreSQL 8.3.8 [CVSS: 7,5 (pre slovníkový útok)] – Pre získanie prístupových údajov bol opäť použitý slovníkový útok [57, 46]. Štandardné Linuxové inštalácie PostgreSQL môžu zapisovať do adresára `/tmp`. Ďalej môžu vytvoriť užívateľom definované funkcie (ďalej UDF), ktoré využívajú zdieľanú knižnicu z tohto adresára. Útočník to môže využiť a nahrať vlastný kód do adresára `/tmp`, vytvoriť UDF a tak spustiť ľubovoľný kód [58].

- DistCC 2.18.3 [CVSS: 9,3] – Táto zraniteľnosť umožňuje útočníkovi vzdialene spustiť ľubovoľný príkaz cez tzv. *compilation jobs*, ktoré server vykonáva bez kontroly oprávnení [59, 60].

[3]Útoky prebiehali do momentu , kým sa nezískali prístupové údaje, alebo možnosť spúšťať príkazy na napadnutom počítači. V prípade použitia vlastného payloadu do momentu jeho spustenia. Žiadne lokálne útoky (napr. na eskaláciu privilégií) už neboli realizované.

Kapitola 7

Nástroj na automatickú exploitáciu sieťových služieb

Pre účely tejto práce a efektívne získanie sady dát pre experimenty bol navrhnutý a implementovaný nástroj na automatickú obfuskáciu, vykonanie útoku a zaznamenanie tejto aktivity. V nasledujúcich častiach je popísaný návrh (7.1) a implementácia (7.2) tohto nástroja.

7.1 Návrh nástroja

Cieľom bolo navrhnúť nástroj, ktorý bude schopný automaticky vykonávať útoky na zvolené sieťové služby a zaznamenávať sieťovú prevádzku. Okrem priamych útokov musí zvládnuť aj obfuskáciu so snahou o obídenie behaviorálnej analýzy. Ďalej by mal pred ukončením obnoviť všetky modifikované systémové nastavenia. V neposlednom rade by bolo vhodné, aby umožňoval obnovu cieľového počítača (na ktorý sa útočí) po každom útoku. Pre každý útok, alebo útok a obfuskáciu, bol zavedený pojem scenár útoku (skrátene len scenár). Program by mal v predvolenej konfigurácii postupne vykonať všetky dostupné scenáre útoku na zvolený počítač a zraniteľnú sieťovú službu.

7.1.1 Použité techniky obfuskácie

Z navrhovaných metód na obídenie behaviorálnej analýzy sieťovej komunikácie (viď časť 5.5) bola nakoniec časť z nich vybraná pre použitie v navrhovanom nástroji. V tabuľke 7.1 sú zhrnuté zvolené techniky obfuskácie. Uvedené hodnoty boli získané empiricky, tzn. skúšaním rôznych hodnôt dovtedy, kým útoky neboli úspešné z dôvodu rôznych sieťových problémov. Účelom týchto techník je rôznymi spôsobmi modifikovať toky paketov. NBA skúma rôzne štatistické a behaviorálne vlastnosti v tokoch paketov a tieto metódy by mohli priniesť určité komplikácie pre detekčný model NBA.

7.1.2 Využitie existujúcich nástrojov

Nástroj by mal zvládať automatické útoky na zraniteľné počítače (resp. ich služby). K tomu sa ukázalo ako najvhodnejšie použiť Metasploit [61] a konkrétne jeho `msfconsole`.

Na zaznamenávanie sieťovej prevádzky sa použije použitý overený program `tcpdump`. Ten bude zachytávať sieťovú prevádzku medzi útočníkom a cieľovým(i) počítačom. Nástroj

Skupina	**Metóda**	**Id**
Roztiahnutie paketov v čase (*delay*)	konštantný delay: 1s	2a
	konštantný delay: 8s	2b
	delay 5 sekúnd s ± 2,5s variáciou podľa normálneho rozloženia a 25% koreláciou	2c
Zahadzovanie (strata) paketov (*loss*)	strata 25% paketov	2d
Simulácia chybového prenosového kanála (*corrupt*)	poškodenie 25% paketov	2e
	poškodenie 35% paketov	2f
	poškodenie 35% paketov s 25% koreláciou	2g
Duplikácia paketov (*duplication*)	duplikácia 5% paketov	2h
Zmena poradia paketov (*reorder*)	preusporiadanie 25% paketov (prehodené pakety sú posielané s 10ms delay) s 50% koreláciou	2i
	preusporiadanie 50% paketov (prehodené pakety sú posielané s 10ms delay) s 50% koreláciou	2j
Modifikácia MTU	MTU 1000	2k
	MTU 750	2l
	MTU 500	2m
	MTU 250	2n
Kombinácia viacerých techník	delay: 10ms, 20ms normálne rozlož., 25% korelácia; loss: 23%; corrupt: 23%; reorder: 23%	2o
	delay, 7750ms, 150ms norm. rozl., 25% korelácia; loss: 0,1%; corrupt: 0,1%; dupl.: 0,1%; reorder: 0,1%	2p
	delay, 6800ms, 150ms norm. rozl., 25% korelácia; loss: 1%; corrupt: 1%; dupl.: 1%; reorder 1%	2q

Tabuľka 7.1: Použité obfuskačné techniky.

`tcpdump` umožňuje pomocou argumentu `-F` definovať súbor, ktorý obsahuje, tzv. *filter expression*. V prípade tejto práce to bude využité na odfiltrovanie prevádzky, ktorá nepatrí danému útoku.

K realizácii väčšiny obfuskácií je najlepšie použiť program `tc` [62], resp. jeho nadstavbu `NetEm` [63]. Tá umožňuje pridávať latenciu, stratu paketov, duplikáciu paketov, preusporiadanie paketov a ďalšie charakteristiky odchádzajúcim paketom z vybraného sieťového rozhrania. Pre zmenu MTU sa použije program `ifconfig`.

Po každom útoku je vhodné automaticky obnoviť počítač, ktorý bol cieľom útoku. To si užívateľ musí nakonfigurovať, pretože to je závislé od konkrétneho systému a testovacej infraštruktúry. V tejto práci bol použitý program `ssh`, verejný kľúč bez potreby hesla a skript na obnovenie[1] snímku (*snapshot*-u) virtuálneho stroja pomocou programu `vboxmanage` [64]. Užívateľ si môže nastaviť pre každý útok rôznu „obnovovaciu procedúru“. Samozrejme v predvolenej konfigurácii to nie je aktivované.

7.1.3 Návrh tried

Diagram tried, ktoré boli navrhnuté pre tento nástroj je na obrázku 7.1. Hlavnou triedou je trieda `Exploitator`. Vstupné a implicitné argumenty spracováva trieda trieda `Arguments-Parser`. Trieda `DirectoriesWorker` pracuje so vstupným a výstupným adresárom a všeobecne so všetkými súbormi čo k tomu patria (napr. spracuje vstupný adresár, kde sú zdrojové súbory pre daný útok a prípadne súbor pre tcpdump filter a CVE identifikátor útoku, ktorý sa kopíruje do výstupného adresára.). Scenáre útoku sú nastavované v triede `AttackScenarioSetter`. Pre obfuskácie (presnejšie obfuskátory) bola navrhnutá abstraktná trieda `ObfuscatorBase`, ktorú ďalej dedia triedy `NetemObfuscator` (stará sa o obfuskáciu pomocou NetEm) a `MTUObfuscator` (nastavuje MTU). Ďalšia trieda `Tcpdump` slúži na zachytávanie paketov. Nakoniec, trieda `Attacker` je navrhnutá na účely vykonávania útokov.

7.1.4 Navrhované režimy behu

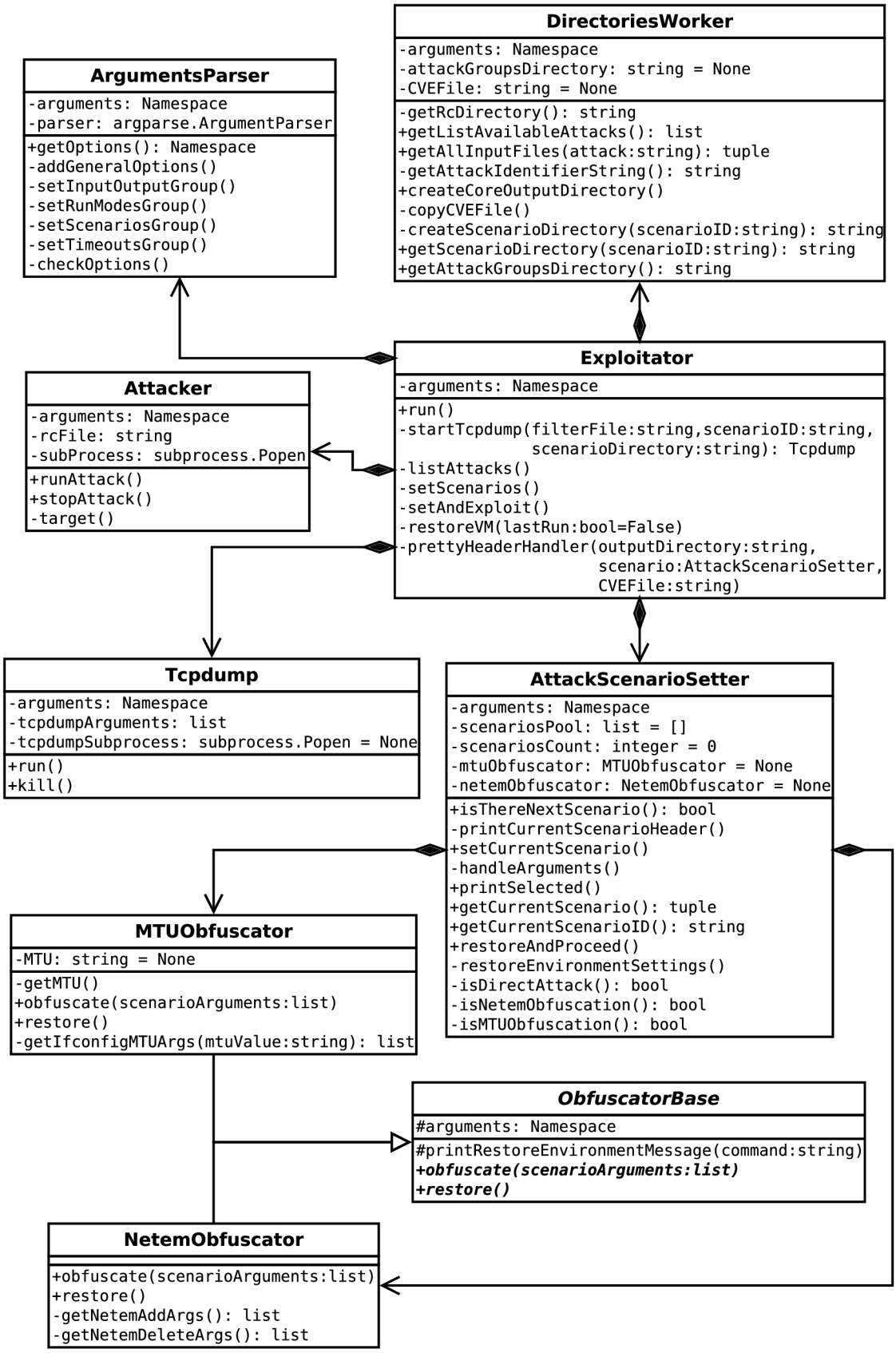
Nastroj by mal umožniť okrem hlavnej činnosti (priamy/obfuskovaný útok a zaznamenanie útoku) aj samostatné nastavenie scenára útoku (bez vykonania útoku). Ďalej by mal nástroj umožniť vypísanie zoznamu dostupných útokov a scenárov. Nasleduje popis niektorých režimov behu.

Navrhovaná hlavná činnosť programu

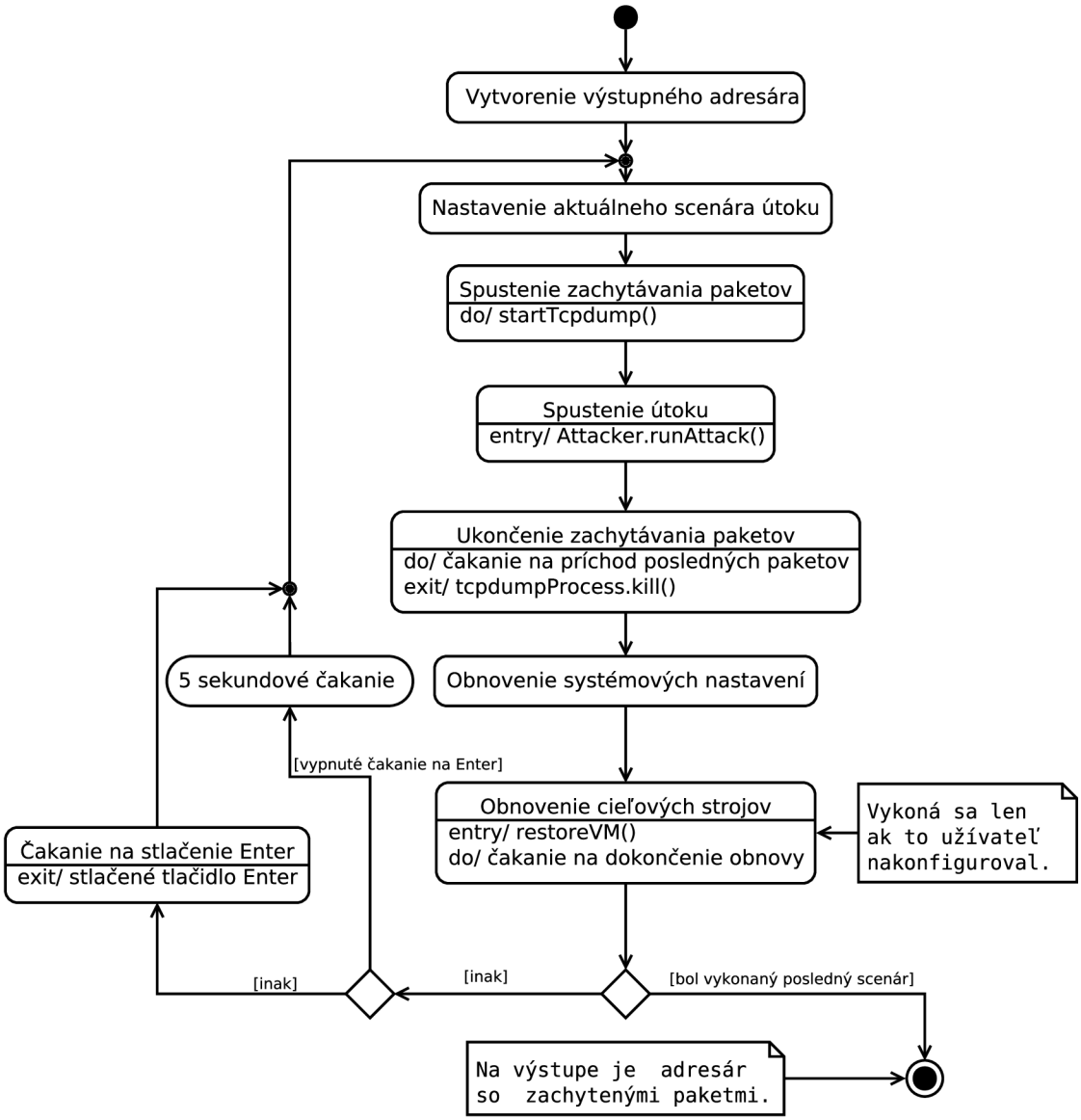
Na obrázku 7.2 je stavový diagram [65], ktorý zachytáva hlavnú činnosť programu. Stav *Obnovenie systémových nastavení* (ako názov napovedá) obnoví stav systému útočníka. Preto je potrebné si v prípade obfuskácie pomocou modifikácie MTU zaznamenať aktuálnu hodnotu, aby program vedel, čo má obnoviť. V rámci stavu *Nastavenie aktuálneho scenára* (ešte pred samotným nastavením) je tak nutné zistiť túto hodnotu.

Stav *Spustenie útoku* skutočnosti ešte zahŕňa vo vnútri možnosť prerušenia útoku v prípade vypršania časového limitu (užívateľom nastaviteľné). V stave *Ukončenie zachytávania paketov* je znázornené čakanie, ktoré je vhodné (záleží od konkrétnej obfuskácie) pred ukončením samotného zachytávania paketov. Takto sa zaistí príchod všetkých paketov. Nástroje na extrakciu metrík (viď. 8. kapitolu) pracujú hlavne s kompletnými (korektne ukončenými) tokmi. Tieto nástroje síce pracujú aj s neukončenými spojeniami, ale v predvolenom nastavení uvažujú len prvých 5 minút komunikácie. Niektoré útoky však môžu trvať aj viac

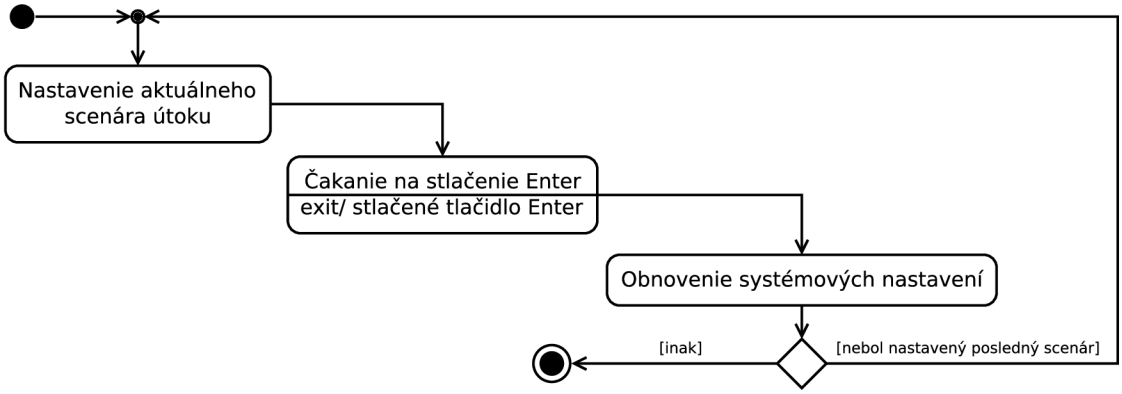
[1] Reálne ten skript pozostával z vypnutia VM, obnovy snímku (*snapshot*) VM a spustenia VM.



Obr. 7.1: Diagram tried pre navrhnutý nástroj.



Obr. 7.2: Stavový diagram pre základnú zjednodušenú funkcionalitu.



Obr. 7.3: Stavový diagram pre režim nastavovania scenára útoku.

ako 15 minút a tak by sa mohli vyskytnúť zbytočné problémy, ktorým sa takto dá vyhnúť. Inak by sa niektoré posledné spojenia nemuseli dostať do extrahovanej sady dát.

Medzi každým útokom je vhodné obnoviť stav počítača, na ktorý sa útočí, alebo aspoň počkať pár sekúnd. V praxi sa totiž ukázalo, že niektoré útoky dostali zraniteľný cieľový počítač do takého stavu, že nasledujúce útoky už neboli úspešné. Z toho dôvodu bola navrhnutá okrem automatického obnovenia aj alternatíva pre manuálne obnovenie. V predvolenej konfigurácii totiž program z princípu nemôže byť dopredu nastavený na automatickú obnovu. Program by preto mal po každom útoku čakať na užívateľa, kým neurobí potrebné kroky (stav *Čakanie na stlačenie Enter*). Avšak, ak užívateľ vie čo robí, je potrebné mu umožniť automatizáciu, tzn. vypnutie tohto čakania po útoku.

V stave *Obnovenie cieľových strojov* prebieha obnova počítača (alebo viacerých počítačov) na ktoré sa útočilo. Avšak, užívateľ musí mať možnosť si to dostatočne prispôsobiť a nastaviť, pretože nie je prakticky možné, aby to program dopredu mal nastavené/realizované univerzálne pre všetky možné konfigurácie infraštruktúry nasadenia. Program by mal umožniť spustenie užívateľom definovaného príkazu pre daný útok a službu. Ako príklad môže byť spustenie skriptu, ktorý obnoví snímok virtuálneho počítača (na ktorý sa útočilo) po dokončení útoku.

Nastavenie scenára bez vykonávania útoku

Nástroj by mal umožniť aj samostatné nastavenia scenára bez vykonávania útoku (viď obrázok 7.3). Toto je vhodné implementovať hlavne pre používateľa, ktorý by chcel pridať nový útok do množiny útokov, s ktorými pracuje nástroj. S touto voľbou nástroj pre užívateľa len aktivuje rôzne scenáre útoku (prednastavené na všetky dostupné) a užívateľ vždy ručne odskúša správanie útoku pri danom scenári.

7.2 Implementácia nástroja

Podľa návrhu bol implementovaný nástroj na automatickú exploitáciu a obfuskáciu útokov v programovacom jazyku Python.

Programy `tcpdump a msfconsole` sú v rámci behu programu spúšťané ako podprocesy. V prvom prípade to je pomocou `subprocess.Popen`. V druhom prípade bolo potrebné použiť aj `threading.Thread`, ktoré umožňuje nastaviť procesu časový limit. Rovnako aj

pre spustenie všetkých príkazov, ktoré sú spúšťané v rámci metód `obfuscate()`/`restore()` je využitá trieda `subprocess.Popen`.

Vstupný adresár obsahuje zložky pre jednotlivé služby a v nich sú adresáre pre jednotlivé útoky. Každý adresár musí obsahovať minimálne zdrojový súbor pre program `msfconsole` s príponou `rc`. Voliteľne môže obsahovať ešte súbor `filter.txt`, ktorý je potom nastavený ako filter pre `tcpdump`. Nakoniec tam môže byť aj súbor s názvom v tvare `CVE-YYYY-NNNNN`, ktorý sa kopíruje do výstupného adresára. Program prehľadáva vstupný adresár a podľa identifikátora aktuálneho útoku a regulárnych výrazov postupne nájde danú zložku a súbory.

Výstupom nástroja je adresár, ktorý zodpovedá adresárovej štruktúre, ktorá je kompatibilná s nástrojmi na extrakciu metrík. Prakticky tak je možné pri vhodnom nastavení plne automatizovať všetko od vykonania útokov až po extrakciu metrík a získanie `csv` súborov. Nasleduje zoznam zdrojových súborov a ich obsah:

- `run_exploitator.py` – Implementácia triedy `Exploitator`. Hlavný súbor, ktorým sa nástroj spúšťa.
- `argparser.py` – Implementácia triedy `ArgumentsParser`.
- `directories_worker.py` – Implementácia triedy `DirectoriesWorker`.
- `tcpdump.py` – Implementácia triedy `Tcpdump`.
- `attacker.py` – Implementácia triedy `Attacker`.
- `scn/attack_scenario_setter.py` – Implementácia triedy `AttackScenarioSetter`.
- `scn/netem_obfuscator.py` – Implementácia triedy `NetemObfuscator`.
- `scn/mtu_obfuscator.py` – Implementácia triedy `MTUObfuscator`.
- `scn/scenarios.py` – Definícia jednotlivých scenárov útoku.
- `scn/obfuscator_base.py` – Implementácia abstraktnej triedy `ObfuscatorBase`.
- `utils.py` – Rôzne pomocné funkcie využívané v rámci celého programu.
- `config.py` – Konfiguračný súbor a zároveň preddefinované nastavenia programu.
- `err.py` – Reťazce pre chybové hlásenia.

7.2.1 Použitie a ovládanie nástroja

V hlavnom režime musí užívateľ programu poskytnúť (a ešte predtým sám otestovať) vstupný adresár, ktorý obsahuje najmä zdrojové súbory s príkazmi pre `msfconsole`. Nástroj potrebuje minimálne súbor s príponou `rc`, ktorý potrebuje pre automatické vykonanie útoku pomocou `msfconsole`. Príklad, ako by mal tento vstupný adresár vyzerať je v preddefinovanom vstupnom adresári `src/exp-set`.

- Program má štyri režimy behu a podľa toho i nasledovné parametre:
  - `-lA, --list-attacks` – vypíše zoznam dostupných útokov.
  - `-lS, --list-scenarios` – vypíše zoznam dostupných scenárov útoku.

  - `--set-and-exploit` – hlavný režim programu. Toto je predvolený režim.
  - `--set-scenario` – režim nastavenia scenára bez vykonania útoku.

- Rôzne voliteľné parametre:
  - `-h, --help` – výpis nápovedy.
  - `-a ATTACK, --attack ATTACK` – výber útoku.
  - `-i INTERFACE, --interface INTERFACE` – sieťové rozhranie pre zachytávanie paketov.
  - `-c, --continue` – vypnutie čakania po útoku (nečaká na Enter)
  - `-v, --verbosity` – zvýšenie detailnosti výstupu programu. Je možné použiť viac krát.

- Voľba scenárov útoku (ak nie je použitý žiadny z nasledujúcich parametrov, tak program postupne aplikuje všetky scenáre):
  - `--scenario SCENARIO` – vykonanie jediného scenára.
  - `--from-to-scenario FROM TO` – vykonanie scenárov definovaných prvým (`FROM`) a posledným (`TO`).
  - `--from-scenario FROM` – pokračovanie od (`FROM`) definovaného scenára až po posledný dostupný.
  - `--to-scenario TO` – postupne vykonáva scenáre až do definovaného (`TO`).

- Vstupný a výstupný adresár:
  - `-r INPUT_DIRECTORY, --input-directory INPUT_DIRECTORY` – vstupný adr.
  - `-w OUTPUT_DIRECTORY, --output-directory OUTPUT_DIRECTORY` – výst. adr.

- Časové limity:
  - `-tT SECONDS, --tcpdump-timeout SECONDS` – časový limit pred ukončením programu tcpdump (po ukončení útoku).
  - `-tA SECONDS, --attack-timeout SECONDS` – časový limit pre vykonanie útoku.

7.2.2 Konfigurácia nástroja

Bol vytvorený konfiguračný súbor `config.py`, ktorý zároveň obsahuje aj predvolené hodnoty. Je možné tu nastaviť nasledovné vlastnosti a funkcionalitu programu:

- *vstupný adresár* (`INPUT_FILES_DIRECTORY`), kde sú umiestnené zdrojové súbory pre program `msfconsole` a prípadne aj príslušné súbory s filtrom pre tcpdump a CVE-YYYY-NNNNN súbor s popisom útoku, ktorý sa nakopíruje do výstupného adresára. Prednastavený vstupný adresár je `exp-set` v adresári so zdrojovým súbormi.
- *výstupný adresár* (`OUTPUT_DIRECTORY`),
- *Sieťové rozhranie* (`INTERFACE`) na ktorom má `tcpdump` zachytávať pakety.

- *časový limit pred ukončením zachytávania paketov* (`TCPDUMP_TIMEOUT`),
- *časový limit pre dokončenie útoku* (`ATTACK_TIMEOUT`),
- zakázať/povoliť *čakanie po každom útoku* (`DO_NOT_WAIT`),
- nakonfigurovať obnovenie počítačov po vykonávaných útokoch (`RESTORE_VMs` a ďalšie).

Tento prístup s konfiguračným súborom bol zvolený najmä pre pohodlie užívateľa. Vďaka tomu nemusí vždy nastavovať často používané a opakujúce sa parametre. Stačí, aby si ich nastavil pri prvom použití. Následne mu stačí len zvoliť útok a prípadne len niektoré scenáre útoku.

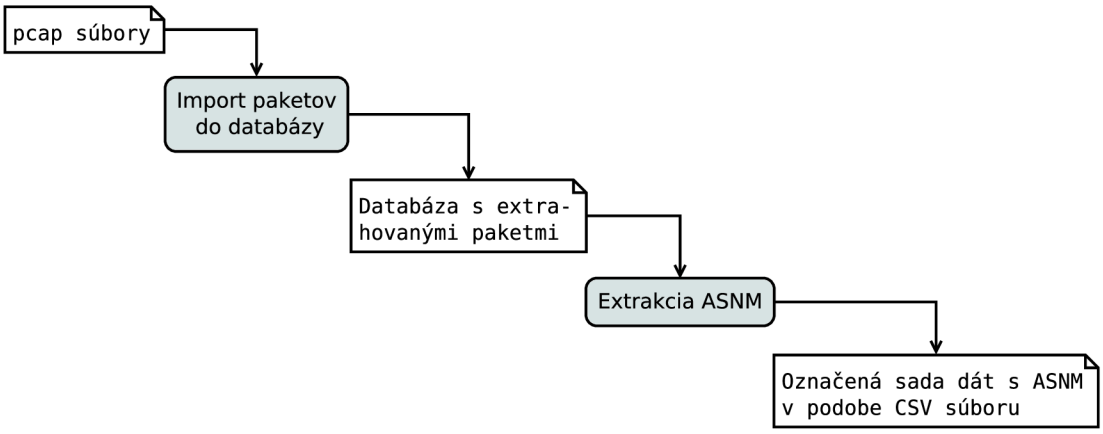
Kapitola 8

Získanie sady dát k experimentom

Vytvorený nástroj bol použitý na automatickú exploitáciu zvolených sieťových služieb a získanie záznamov sieťovej prevádzky z týchto aktivít. Celkovo vykonávanie útokov zabralo mnoho dní a bolo získaných 358 `pcap` súborov, ktoré obsahovali zaznamenanú nelegitímnu sieťovú komunikáciu. Legitímna sieťová komunikácia bola získaná z rôznych zdrojov. Prvú časť tvorí legitímna komunikácia medzi virtuálnymi počítačmi (ďalej VM) podľa navrhnutého modelu nasadenia (viď. obrázok 6.1). Ďalej bolo zaznamenané bežné používanie Internetu (Web, email,...). Tu bola aj simulovaná nespoľahlivá sieť s rôznymi (a postupne sa meniacimi) nastaveniami pomocou nástroja NetEm. V neposlednom rade boli využité aj niektoré verejne dostupné záznamy legitímnej komunikácie medzi použitými sieťovými službami. Získané `pcap` súbory sú uvedené na DVD prílohe v adresári `data`.

Zjednodušený postup, ako boli získané záznamy sieťovej prevádzky použité je na obrázku 8.1. Sada nástrojov na extrakciu ASNM z pcap súborov (adresár `tools` na DVD prílohe) bola poskytnutá vedúcim tejto práce a ich hlavným autorom [31]. Na začiatku bolo potrebné vytvoriť databázu (PostgreSQL), kde sa ukladajú informácie o paketoch. Na to slúži skript `tools/db/scheme.sql`. Následne bola upravená konfigurácia pre pripojenie do databázy[1], aby zodpovedala lokálnym podmienkam. Nakoniec bolo ešte upravený očakávaný tvar vstupného adresára v `TCPDumpWalker.process_file()`[2].

[1] súbory `tools/db-tcpdump-fill/tcpdump_fill.py` a `tools/metricsExtractor/Config.py`
[2] súbor `tools/db-tcpdump-fill/tcpdump_fill.py`



Obr. 8.1: Spracovanie `pcap` súborov a generovanie ASNM.

Vďaka návrhu a implementáci vytvoreného nástroja, ktorý dopredu počítal s využitím týchto nástrojov bolo využitie výstupov (`pcap` súborov) priamočiare. Pri pokuse naraz spracovať všetky `pcap` súbory mal použitý počítač s takým veľkým množstvom paketov problémy (hlavne pre veľké nároky na RAM). Extrakcia prebiehala oddelene pre legitímnu a nelegitímnu sieťovú prevádzku. Získanie ASNM pre nelegitímne toky trvalo skoro dve hodiny. V prípade legitímnych tokov to boli takmer štyri hodiny.

Pre experimentovanie so získanými dátami bol zvolený program RapidMiner[3] (ďalej RM). Získané `csv` súbory s (ne)legitímnymi tokmi a ASNM boli preto ešte pred samotným experimentovaním naimportované do databázy nástroja RM. Na tento účel bol vytvorený v RM proces, ktorý spracoval vstupné `csv` súbory do podoby rôznych dátových tabuliek (s ohľadom na nasledujúce experimenty) v databáze RM. Zo sady dát bolo odstránených 7,75% spojení s chýbajúcimi atribútmi. Väčšinu z toho predstavovala bežná legitímna prevádzka. Výsledná sada dát používaná pri experimentoch je zhrnutá v tabuľke 8.1. Ide o prepojenie štatistík z dvoch tabuliek pre binominálnu klasifikáciu a pre klasifikáciu do troch tried (viď kapitolu 9 k experimentom). Pre službu DistCC nie je dostupná žiadna legitímna prevádzka, pretože pri pokuse o jej použitie nefungovala správne a odmietala komunikovať so vzdialenými počítačmi, takže nebola žiadna komunikácia, ktorý by bolo možné zachytiť. Nepodarilo sa pre ňu získať ani dáta z iných verejných zdrojov.

Sieťová služba / Druh komunikácie	**Počet TCP spojení**			
	Legitímne	**Priame útoky**	**Obfuskované útoky**	**Útoky spolu**
Apache Tomcat	48	48	849	897
MSSQL	46	37	635	672
Samba	2	17	348	365
DistCC	n/a	6	66	72
PostgreSQL	3	44	728	772
Bežná legitímna komunikácia	2299	n/a	n/a	n/a
Spolu	2398	152	2626	2778

Tabuľka 8.1: Distribúcia TCP spojení do klasifikačných tried.

[3]Nástroj na dolovanie z dát, strojové učenie a prediktívnu analytiku [66].

Kapitola 9

Experimenty

Táto kapitola obsahuje výsledky experimentov nad získanou sadou dát (viď kapitolu 8). Dáta obsahovali extrahované ASNM a špeciálne atribúty, ktoré označovali: klasifikačné triedy, sieťové služby, obfuskácie a jednotlivé druhy útokov. Na dolovanie z dát bol zvolený program RapidMiner [66] (ďalej RM). Väčšina experimentov využívala krížovú validáciu s 5 validáciami (5-fold).

Pre klasifikáciu bol zvolený Naivný Bayesovský klasifikátor, keďže bol v minulosti používaný aj vo väčšine relevantných prácach [6, 16, 36, 33]. Nastavenie parametrov klasifikátora bolo nasledovné:

- *laplace correction*: povolená,
- *estimation mode*: full,
- *bandwidth selection*: fix a
- *bandwidth*: premenlivý (uvedené pri každom experimente).

Trénovanie modelu klasifikátora sa sústredilo na maximálnu úspešnosť klasifikácie. Pre každý experiment boli hľadané optimálne parametre pre nastavenie klasifikátora. Váhovanie, diskretizácia ani normalizácia nepriniesla žiadne zlepšenie. Skôr naopak. Preto neboli použité.

V tabuľke 9.1 je uvedených 15 metrík z ASNM (viď časť 4.2), ktoré sú použité pri všetkých experimentoch. Bola vybratá množina metrík, ktoré sú založené na čase, indexe a ďalších vlastnostiach, ktoré sú uvedené v tabuľke 9.1. Tieto metriky boli zvolené pre lepšiu porovnateľnosť s výsledkami v článku [36], kde boli využívané rovnaké metriky. Boli vyskúšané i metódy na výber najlepších rysov (najmä s operátorom *Optimize Selection* a algoritmom *forward selection*), avšak RM nebol schopný v konečnom čase nájsť tieto rysy.

V nasledujúcich častiach sú výsledky experimentov. V tabuľkách sa pod úspešnosťou klasifikácie vždy myslí *accuracy*, tzn. presnosť klasifikácie do klasifikovaných tried (viď vzťah 2.5). Termín presnosť predikcie (*class precision*) hovorí o presnosti predikovanej triedy (viď vzťah 2.4) . Správnosť predikcie (*class recall*) je percento správne predikovaných objektov do danej triedy (viď vzťah 2.1).

Metrika	Popis	Kategória	Vlastnosti			
			Čas	Index	Veľkosti pak.	Počet pak.
PktPerSesIn	Počet paketov počas TCP spojenia v smere dnu.	Dynamické				X
CntAckOut	Celkový počet odoslaných ACK paketov TCP.	Dynamické				X
sumSessPerPort	Počet TCP spojení v intervale 5 minút od aktuálneho spojenia, ktoré prebiehajú na rovnakom cieľ. porte.	Dynamické	X			X
InPktLen32s10i[4]	DÍžky paketov za 32 sekúnd distrib. do 10 intervalov v smere dnu.	Distrib.	X		X	X
SigPktLenSrc	Smerodajná odchýlka veľkosti zdrojového paketu.	Štatistické			X	X
ratInoutB	Pomer odoslaných a prijatých B.	Štatistické			X	X
modTTLIn	Modus hodnoty TTL pre zdrojové pakety.	Štatistické				X
polynomIndexes-3ordOut[1]	Aproximácia odchádzajúcej komunik. polynómom 3. rádu. Osou x je index paketu v rámci spojenia.	Behavior.		X	X	X
polynomIndexes-13ordIn[0]	Aprox. prich. komunikácie polyn. 13. rádu. Osou x je index paketu v rámci spojenia.	Behavior.		X	X	X
polynomIndexes-13ordIn[12]		Behavior.		X	X	X
fourCoefs-GonAngleIn[3]	Koeficienty Fourierovej rady v goniometrickej reprezentácii (uhol). Smer prichádz. komunikácia.	Behavior.		X	X	X
intervalsIPsSig	Smerodajná odchýlka časových intervalov medzi spojeniami prebiehajúcich na rovnakých adresách IP. Uvažuje začiatky spojení na určenie intervalov.	Behavior.	X			
intervalsIPsSig2	Smerodajná odchýlka časových intervalov medzi spojeniami prebiehajúcich na rovnakých adresách IP. Uvažuje začiatky aj konce spojení na určenie intervalov.	Behavior.	X			
gaussProds4Out[1]	Normalizované produkty výstupnej komunikácie so 4 gaussovými krivkami.	Behavior.		X	X	X
gaussProds8Out[2]	Normalizované produkty výstupnej komunikácie s 8 gaussovými krivkami.	Behavior.		X	X	X

Tabuľka 9.1: ASNM, ktoré boli využité pri experimentoch.

9.1 Binominálna klasifikácia

V tejto časti boli vykonané experimenty s klasifikáciou do dvoch tried. Toky v sade dát boli označené ako legitímna, alebo nelegitímna sieťová prevádzka. Tento spôsob klasifikácie je najbežnejší aj pre IDS, ktoré sa snaží odhaliť akýkoľvek útok. Nie je tak dôležité či sa útok snaží rôznymi technikami zamaskovať (obfuskovaný útok), alebo nie (priamy útok).

9.1.1 Trénovanie bez znalosti obfuskovaných útokov

V tabuľkách 9.2 sú výsledky experimentov s klasifikátorom, ktorý bol natrénovaný bez obfuskovaných útokov (tab. 9.2a). Vytvorený klasifikačný model nebol schopný odhaliť všetky útoky (bolo pridaných 2626 obfuskovaných útokov) a dosiahol menšiu presnosť pri otestovaní na všetkých dátach (tab. 9.2b). Správne predikoval len 96,95% obfuskovaných útokov. Parameter bandwidth bol nastavený na hodnotu 5,6.

Úspešnosť klasifikácie: 99,96% ± 0,08%		**Skutočná trieda**		**Presnosť predikcie**
		Legitímne toky	**Priame útoky**	
Predp. trieda	**Legitímne toky**	2398	1	99,96%
	Priame útoky	0	151	100,00%
Správnosť predikcie		100,00%	99,34%	

(a) Krížová validácia (5-fold) na priamych útokoch a legitímnej sieťovej prevádzke.

Úspešnosť klasifikácie: 98,45%		**Skutočná trieda**		**Presnosť predikcie**
		Legitímne toky	**Všetky útoky**	
Predp. trieda	**Legitímne toky**	2398	80	96,77%
	Všetky útoky	0	2698	100,00%
Správnosť predikcie		100,00%	97,12%	

(b) Overenie vytvoreného modelu na všetkých dátach.

Tabuľka 9.2: Trénovanie bez obfuskovaných dát a test na všetkých dátach.

9.1.2 Porovnanie detekcie obfuskovaných a priamych útokov oproti všetkým útokom.

Pri nasledujúcich experimentoch (tab. 9.3) boli vždy nastavené rovnaké podmienky. Klasifikátor mal nastavený parameter bandwidth na 5,5. V prípade tabuľky 9.3a bola použitá sada všetkých dát. Ďalej boli priame (tab. 9.3b), resp. obfuskované (tab. 9.3c) útoky odstránené zo sady dát.

Cieľom obfuskácií bolo upraviť behaviorálne a štatistické vlastnosti útokov nasledovným spôsobom: dosiahnuť maximálnu podobnosť s legitímnymi tokmi a minimálnu podobnosť s priamymi útokmi. Aktuálny experiment sa snaží analyzovať tento cieľ. Pred vykonaním experimentu bol uvažovaný predpoklad, že detekcia len priamych/obfuskovaných útokov dosiahne vyššiu presnosť, ako v prípade, keď sú obe triedy útokov spojené do jednej. Tento predpoklad bol založený na očakávaní, že priame a obfuskované útoky budú mať rozličné správanie a teda aj rozdielne behaviorálne a štatistické vlastnosti, ktoré sú spracovávané pomocou ASNM.

Výsledky však ukázali, že predpoklad nebol naplnený. Jedným dôvodom mohli byť vhodné vlastnosti zvolených ASNM, ktoré dobre charakterizovali dané toky dát. Ďalšou príčinou mohli byť neúčinné obfuskácie.

Úspešnosť klasifikácie: 100,00% ± 0,00%		**Skutočná trieda**		**Presnosť predikcie**
		Legitímne toky	**Všetky útoky**	
Predp. trieda	**Legitímne toky**	2398	0	100,00%
	Všetky útoky	0	2778	100,00%
Správnosť predikcie		100,00%	100,00%	

(a) Celá sada dát.

Úspešnosť klasifikácie: 99,98% ± 0,04%		**Skutočná trieda**		**Presnosť predikcie**
		Legitímne toky	**Obfus. útoky**	
Predp. trieda	**Legitímne toky**	2397	0	100,00%
	Obfus. útoky	1	2626	99,96%
Správnosť predikcie		99,96%	100,00%	

(b) Obfuskované útoky a legitímna sieťová prevádzka.

Úspešnosť klasifikácie: 99,96% ± 0,08%		**Skutočná trieda**		**Presnosť predikcie**
		Legitímne toky	**Priame útoky**	
Predp. trieda	**Legitímne toky**	2398	1	99,96%
	Priame útoky	0	151	100,00%
Správnosť predikcie		100,00%	99,34%	

(c) Priame útoky a legitímna sieťová prevádzka.

Tabuľka 9.3: Krížová validácia (5-fold) na rôznych sadách dát.

9.2 Klasifikácia do troch tried

Pri nasledujúcom experimente bola trieda útokov rozdelená na 2 triedy (obfuskované/priame útoky). Celkovo tak (spolu s legitímnymi tokmi) boli 3 triedy, do ktorých sa klasifikovalo. Parameter bandwidth klasifikátora bol nastavený na hodnotu 0,3.

Bol uvažovaný predpoklad, že priame a obfuskované útoky budú mať odlišné štatistické a behaviorálne vlastnosti. Klasifikátor by tak pomocou zvolených ASNM mal dokázať oddeliť tieto útoky do samostatných tried.

V tabuľke 9.4 vidieť, že klasifikátor nebol schopný správne rozdeliť obfuskované a priame útoky vo všetkých prípadoch. Avšak, bola zachovaná nadtrieda útokov, keďže len 2 útoky boli klasifikované ako legitímne toky. Nesprávna predikcia obfuskovaných útokov (podľa jednotlivých druhov obfuskácií) pre tento experiment je ďalej uvedená v tabuľke 9.8a a na obrázku 9.2 je porovnaná aj s polynomiálnou klasifikáciou.

Úspešnosť klasifikácie: 94,57% ± 0,39%		**Skutočná trieda**			**Presnosť predikcie**
		Legit. toky	**Priame útoky**	**Obfus. útoky**	
Predp. trieda	**Legitímne toky**	2398	0	2	99,92%
	Priame útoky	0	43	170	20,19%
	Obfus. útoky	0	109	2454	95,75%
Správnosť predikcie		100,00%	28,29%	93,45%	

Tabuľka 9.4: Krížová validácia (5-fold) na všetkých dátach.

Úspešnosť klasifikácie: 92,67% +/- 0,66%			Skutočná trieda													Presnosť predikcie
			Priame útoky				Obfuskované útoky				Legitímna prevádzka					
			Apache Tomcat	MSSQL	PostgreSQL	Samba	Apache Tomcat	MSSQL	PostgreSQL	Samba	Apache Tomcat	MSSQL	Bežná legitímna komunikácia	PostgreSQL	Samba	
Predpovedaná trieda	Priame útoky	Apache Tomcat	28	0	0	0	86	0	0	0	0	0	0	0	0	24,56%
		MSSQL	0	15	0	0	0	43	0	1	0	0	0	0	0	25,42%
		PostgreSQL	0	0	0	0	0	0	22	0	0	0	0	0	0	0,00%
		Samba	0	0	0	1	0	1	0	30	0	0	0	0	0	3,12%
	Obfuskované útoky	Apache Tomcat	20	0	0	0	753	0	2	0	1	0	1	0	0	96,91%
		MSSQL	0	19	0	2	0	550	0	24	0	0	0	0	0	92,44%
		PostgreSQL	0	0	44	0	10	0	702	1	0	0	0	0	0	92,73%
		Samba	0	3	0	14	0	41	2	292	0	0	0	0	0	82,95%
	Legitímna sieťová prevádzka	Apache Tomcat	0	0	0	0	0	0	0	0	44	0	0	0	0	100,00%
		MSSQL	0	0	0	0	0	0	0	0	0	45	0	0	0	100,00%
		Bežná legitímna komunikácia	0	0	0	0	0	0	0	0	3	1	2298	3	0	99,70%
		PostgreSQL	0	0	0	0	0	0	0	0	0	0	0	0	0	0,00%
		Samba	0	0	0	0	0	0	0	0	0	0	0	0	2	100,00%
Správnosť predikcie			58,33%	40,54%	0,00%	5,88%	88,69%	86,61%	96,43%	83,91%	91,67%	97,83%	99,96%	0,00%	100,00%	

Tabuľka 9.5: Krížová validácia (10-fold) pre vybrané toky.

9.3 Polynomiálna klasifikácia

Pre tento experiment boli dáta rozdelené do 13 tried. Zo sady dát boli najskôr odstránené toky pre službu DistCC, kde chýbali legitímne toky. Zvyšok bol rozdelený podľa služieb a druhu (priame/obfuskované útoky a legitímne toky) sieťovej prevádzky na 12 tried a 13. triedu tvorila ostatná legitímna prevádzka. Klasifikátor mal parameter bandwidth nastavený na hodnotu $5 \cdot 10^{-5}$.

Výsledky experimentu sú v tabuľke 9.5. Červenou farbou sú vyznačené nesprávne klasifikované toky v prípade útokov. Je vidieť , že tvoria rovnobežnú diagonálu s hlavnou diagonálou. V týchto prípadoch neboli ASNM schopné rozlíšiť obfuskované útoky od priamych útokov, resp. ukázali odolnosť voči použitým obfuskáciám, keďže mali o nich informáciu v procese trénovania modelu. V konečnom dôsledku sa ich charakteristiky javili ako rovnaké, resp. podobné. Pre zvýraznené červené diagonály platí aj to, že bola zachovaná nadtrieda útokov i nadtrieda služieb.

V tabuľkách 9.6 sú ešte uvedené pomery (ne)správne detekovaných priamych a obfuskovaných útokov. Druhý stĺpec oboch tabuliek zodpovedá príslušným *class recall* z tabuľky 9.5. Tretí stĺpec oboch tabuliek znamená pomer nesprávne klasifikovaných útokov, ktoré sú zvýraznené červenou farbou v tabuľke 9.5 (rovnobežne s hlavnou diagonálou)[1].

[1]Nesprávne klasifikované útoky, ktoré sa nevyskytujú vo zvýraznených diagonálach boli zámerne vynechané. Preto súčet hodnôt z 2. a 3. stĺpca netvorí po sčítaní 100%.

Služba	Predik. ako priame	Predik. ako obfus.
Apache Tomcat	58,33%	41,67%
MSSQL	40,54%	51,35%
Samba	0,00%	100,00%
PostgreSQL	5,88%	82,35%

(a) Priame útoky.

Služba	Predik. ako obfus.	Pred. ako priame
Apache Tomcat	88,69%	10,13%
MSSQL	86,61%	6,77%
Samba	96,43%	3,02%
PostgreSQL	83,91%	8,62%

(b) Obfuskované útoky.

Tabuľka 9.6: Pomery (ne)správne detekovaných priamych a obfuskovaných útokov.

9.4 Zhodnotenie použitých obfuskačných techník

V tejto časti sú uvedené rôzne porovnania použitých techník obfuskácie. Označenie obfuskácií je zhodné s označením v tabuľke 7.1.

V tabuľke 9.7 sú uvedené pomery nesprávne klasifikovaných obfuskovaných útokov (podľa jednotlivých druhov obfuskácií) pre experiment s binominálnou klasifikáciou z časti 9.1.1 (tab. 9.2b), kde bol model trénovaný bez informácie o obfuskovaných útokoch. Je tu vidieť, že najúspešnejšie boli obfuskácie, ktoré kombinovali viacero techník (2q, 2o a 2p). Ďalšia v poradí bola obfuskácie s poškodením 35% paketov s 25% koreláciou (2g). Úspešnosť nad 4% ešte dosiahla obfuskácia, ktorá nastavovala MTU na hodnotu 500 (2m). Uvedené pomery sú zobrazené aj na obrázku 9.1.

V ďalších experimentoch bola k dispozícii informácia o obfuskovaných útokoch v procese trénovania. Tabuľky 9.8 obsahujú pomery obfuskovaných útokov (podľa jednotlivých druhov obfuskácií), ktoré neboli klasifikované ako obfuskované. Avšak, takmer vo všetkých prípadoch bola správne klasifikovaná nadtrieda útokov (všetky priame a obfuskované útoky spolu)[2]. To svedčí o dobrej detekčnej schopnosti ASNM, keďže v praxi nás zaujíma najmä detekcia útokov. Detaily, ako druh útoku (napr. či bol obfuskovaný) nás už nezaujímajú.

Ďalej sa dá pri tabuľkách 9.8 hovoriť o odlišnosti obfuskovaných a priamych útokov, ktoré boli v mnohých prípadoch celkom vysoké. Na prvom a druhom mieste boli zhodne obfuskácie, ktoré modifikovali MTU (2k, 2m). V prvej päťke boli v oboch tabuľkách stále rovnaké obfuskácie, len v trochu inom poradí. Pre lepšiu názornosť sú uvedené pomery zobrazené aj na obrázku 9.2.

Nedá sa jednoznačne povedať, že by niektorá obfuskácia bola jednoznačne lepšia, alebo horšia. Často boli obfuskácie, ktoré úspešne odlišovali obfuskované útoky od priamych útokov rozdielne pri binominálnej klasifikácii (obr. 9.1) a klasifikácii do viacerých tried (obr. 9.2).

[2]Výnimkou sú len 2 toky v tabuľke 9.4, ktoré boli klasifikované ako legitímne.

Identifikátor obfuskácie	Chybne klasifikované ako legitímne toky
2q	8,11%
2o	5,92%
2p	5,03%
2g	4,11%
2m	4,08%
2f	3,87%
2d	3,29%
2c	3,07%
2l	2,72%
2e	2,67%
2h	2,65%
2b	2,47%
2i	0,68%
2j	0,68%
2k	0,67%
2a	0,00%
2n	0,00%

Tabuľka 9.7: Pomery nesprávne klasifikovaných obfuskovaných útokov.

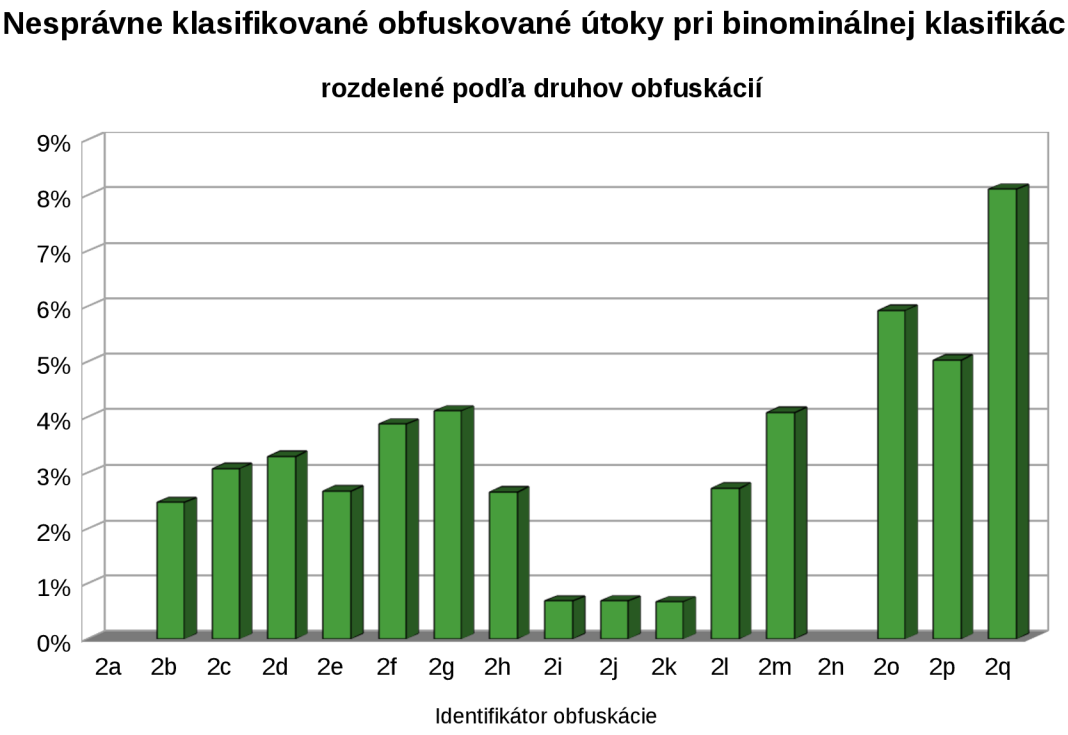
Identifikátor obfuskácie	Klasifik. ako priame útoky
2k	20,67%
2m	18,37%
2l	14,29%
2j	11,64%
2e	11,33%
2i	10,88%
2g	7,53%
2f	4,52%
2a	3,23%
2n	2,88%
2q	2,16%
2h	1,99%
2d	1,97%
2b	1,23%
2p	1,12%
2o	0,66%
2c	0,61%

(a) Pomery pre experiment v časti 9.2.

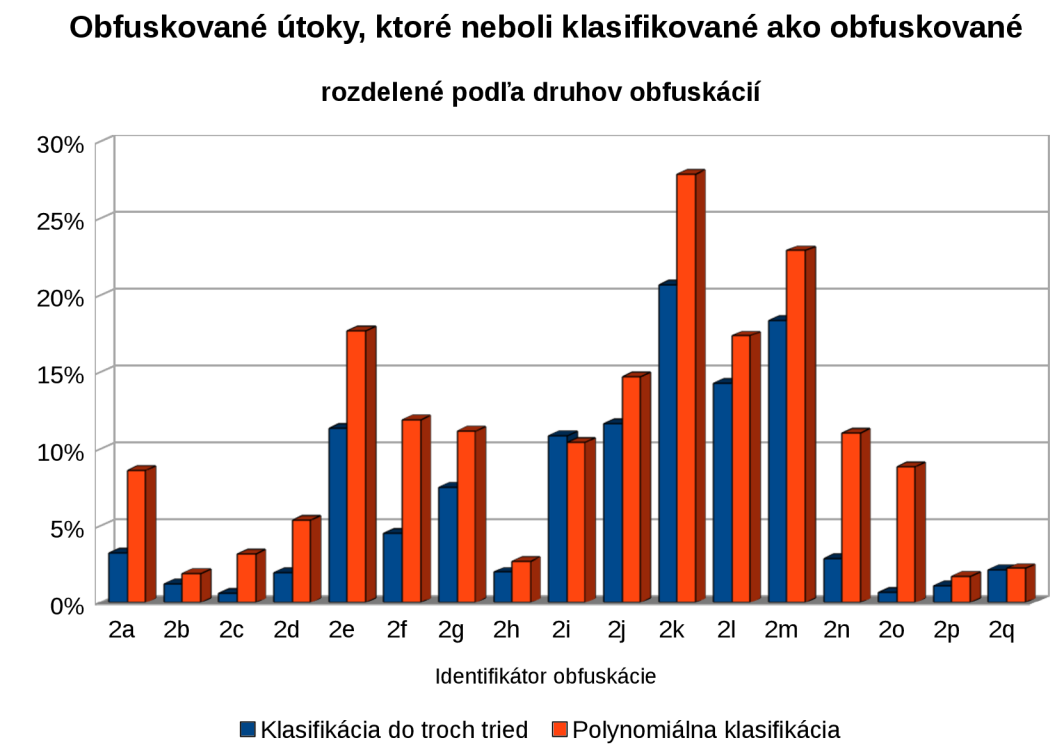
Identifikátor obfuskácie	Klasifik. ako priame útoky
2k	27,89%
2m	22,92%
2e	17,69%
2l	17,36%
2j	14,69%
2f	11,92%
2g	11,19%
2n	11,03%
2i	10,42%
2o	8,84%
2a	8,61%
2d	5,37%
2c	3,16%
2h	2,70%
2q	2,23%
2b	1,92%
2p	1,73%

(b) Pomery pre experiment v časti 9.3.

Tabuľka 9.8: Pomery obfuskovaných útokov, ktoré neboli klasifikované ako obfuskované (pre jednotlivé druhy obfuskácií).



Obr. 9.1: Obfuskované útoky chybne klasifikované ako legitímne toky.



Obr. 9.2: Obfuskované útoky, ktoré neboli klasifikované ako obfuskované.

Kapitola 10

Možnosti ďalšieho rozvoja

Prvou oblasťou, kde by sa dala táto práca rozšíriť sú obfuskačné techniky. Je tu ešte priestor pre vyskúšanie ďalších obfuskácií. Určite by bolo zaujímavé overiť predpoklad, že obfuskácie ktoré fungujú len na úrovni *payload*-u nebudú predstavovať pre NBA väčší problém. Práca [25] uviedla mnoho spôsobov na prekonanie IDS. V prípade reálne nasadeného IDS by sa dalo overiť množstvo obfuskácií, ktoré v tejto práci nebolo možné zrealizovať.

S predchádzajúcim odsekom súvisí aj prípadné rozšírenie vytvoreného nástroja. Ďalšie obfuskácie, ktoré by boli testované, by mohli byť pridané do tohto nástroja. Za istých okolností by sa dalo po väčšom rozšírení nástroja uvažovať o jeho zverejnení a poskytnutí širokej bezpečnostnej komunite. Tá by určite ocenila možnosti, ktorými útočníci už dávno disponujú. Bezpečnostný výskumníci samozrejme majú možnosti využívať rôzne obfuskácie, ale určite by im pomohlo, ak by mali nástroj tohto druhu podobne ako majú napríklad `nmap`, `metasploit` a mnohé ďalšie. Ďalšia možnosť, ktorá sa ponúka na rozšírenie je implementácia využitých (i nových) obfuskácií v rámci nástroja `metasploit`.

Širokou oblasťou pre výskum je ešte aj dolovanie z dát, kde by bolo možné analyzovať vykonávané útoky ešte z iných pohľadov. Jednou z možností je pozrieť sa na (ne)úspešnosť klasifikácie tokov a hlavne úspešnosť jednotlivých obfuskácií v spojení s jednotlivými druhmi útokov. Zistiť, vzťah medzi jednotlivými technikami obfuskácií, druhmi útokov a úspešnosťou týchto obfuskácií. Nemusí pritom ísť o polynomiálnu klasifikáciu do mnohých tried. Stačí to analyzovať následne po predikcii, kde sa urobí analýza nesprávnych predikcií pre jednotlivé skupiny.

Kapitola 11

Záver

V tejto práci boli naštudované princípy detekcie sieťových útokov a klasifikácie sieťovej prevádzky. Špeciálne bola skúmaná hlavne časť behaviorálnej analýzy sieťovej komunikácie (ďalej NBA). V ďalšej časti sa práca venovala existujúcim útokom na systémy k detekcii preniknutia (IDS). Vo vzťahu k cieľom práce, boli navrhnuté metódy, ktoré je možné využiť na obfuskáciu sieťových útokov (časť 5.5), a tým obísť NBA.

Pre efektívne získanie sady dát pre následné experimenty bol navrhnutý a implementovaný nástroj na automatickú obfuskáciu (podľa navrhnutých metód obfuskácie), uskutočnenie útoku a zaznamenanie sieťovej komunikácie (7. kapitola). Tento nástroj bol použitý k vykonaniu útokov na zvolené zraniteľné sieťové služby. Vytvorený nástroj bol veľmi užitočný a umožnil spoľahlivým a systematickým spôsobom získať záznamy zo sieťových útokov pre všetky vykonané sieťové útoky (a ich obfuskácie). Ďalej boli získané aj záznamy legitímnej sieťovej komunikácie. Spôsob, akým bola z týchto komunikácií získaná sada dát k experimentom bol uvedený v 8. kapitole. Nakoniec boli vykonané rôzne experimenty nad získanou sadou dát (9. kapitola). V 10. kapitole boli ešte uvedené možnosti ďalšieho rozvoja.

Ako prvý bol v časti 9.1.1 vykonaný experiment, kde bol klasifikátor trénovaný bez znalosti obfuskovaných útokov. Klasifikátor dosiahol úspešnosť (99,84% $\pm$ 0,08%) pri klasifikácii legitímnej prevádzky a priamych útokov. Avšak, ten istý model dosiahol pri klasifikácii celej sady dát (zahrnuté obfuskované útoky) len 96,95% správnosť predikcie obfuskovaných útokov. Pri tomto experimente sa tak ukázala potreba trénovania klasifikátora aj s obfuskovanými útokmi pre dosiahnutie vyššej presnosti klasifikácie. Bol tak potvrdený predpoklad, že bez tejto informácie neboli ASNM schopné odhaliť všetky útoky.

Experiment v časti 9.1.2 ukázal najmä slabú podobnosť obfuskovaných útokov a legitímnej komunikácie. Ďalšie experimenty ukázali, že ASNM nevedia dobre oddeliť všetky obfuskované útoky od priamych (časti 9.2 a 9.3), resp. ukázali odolnosť voči použitým obfuskáciám. Na druhej strane, nadtrieda útokov bola zachovaná takmer vo všetkých prípadoch, a to je pre nás najdôležitejšie. Podstatná je detekcia útokov bez ohľadu na ich prípadnú obfuskáciu. Z toho dôvodu je detekcia nadtriedy útokov prijateľná.

Zhodnotenie použitých obfuskačných techník (časť 9.4) ukázalo schopnosť niektorých obfuskácií zmeniť charakteristiku priameho útoku. V tabuľke 9.7 boli uvedené pomery nesprávne klasifikovaných obfuskovaných útokov. Celkovo bola najúspešnejšia obfuskácia (8,11% chybovosť), ktorá kombinovala viacero navrhnutých obfuskácií (2q). Ďalšie v poradí boli tiež kombinované obfuskácie (2o a 2p). Ďalej sa v tabuľke 9.8 (a na obrázku 9.2) ukázalo, že obfuskácie vedia pozmeniť charakteristiku útoku takým spôsobom, že klasifikátor často nevedel či ide o priamy útok, alebo obfuskovaný útok. Dá sa tu hovoriť o odlišnosti obfuskovaných a priamych útokov, ktoré boli v mnohých prípadoch celkom vysoké. Na prvom

a druhom mieste boli zhodne obfuskácie, ktoré modifikovali MTU (2k, 2m).

Celkovo sa tak (z časti) potvrdila potreba poskytnúť v procese trénovania klasifikačného modelu čo najviac informácií o útokoch a teda aj obfuskované útoky. Avšak použité obfuskácie neboli až tak účinné, ako tie v článkoch [36, 33] (pozri aj časť 5.3.1). Hlavným dôvodom je pravdepodobne samotná podstata obfuskačných techník, kde v prípade tejto práce boli zachované pôvodné sieťové protokoly pri nelegitímnej komunikácii a menili sa len ich štatistické a behaviorálne vlastnosti. Naopak, v uvedených prácach bolo využité tunelovanie nelegitímnej komunikácie v HTTP(S) protokoloch a tým pádom sa oveľa viac podobala legitímnej komunikácii. Ďalším rozdielom je aj množina vykonaných útokov, kde v prípade tejto práce boli vykonané rôzne druhy útokov na sieťové služby. Nie len útoky typu pretečenie zásobníku, ktoré boli uskutočnené v uvedených prácach.

V prípade NBA (ASNM) sa v tejto práci ukázalo, že si s vykonanými útokmi vedela poradiť celkom dobre. Najmä v prípade, že boli v procese trénovania dostupné informácie o obfuskovaných útokoch. Potom už vedela tieto útoky odhaľovať s vysokou úspešnosťou.

Na túto prácu by sa v budúcnosti dalo naviazať najmä rozšírením použitých obfuskácií. S tým by súviselo aj prípadné rozšírenie realizovaného nástroja. V prípade dostupného reálne nasadeného NBA systému by bolo možné vyskúšať množstvo ďalších techník obfuskácie, ktoré v tejto práci nebolo možné zrealizovať. Od tejto práce by bolo možné pokračovať a zrealizovať verejne dostupný nástroj na obfuskáciu útokov, ktorý by bol dostupný bezpečnostným výskumníkom. Mnoho možností ešte ponúka aj ďalšia detailná analýza úspešnosti klasifikácie a obfuskácie, vo vzťahu s jednotlivými obfuskáciami a sieťovými útokmi.

Literatúra

[1] Thuy TT Nguyen a Grenville Armitage. A survey of techniques for internet traffic classification using machine learning. *Communications Surveys & Tutorials, IEEE*, 10(4):56–76, 2008.

[2] Joe Touch, Eliot Lear, Allison Mankin, Markku Kojo, Kumiko Ono, Martin Stiemerling, Lars Eggert, Alexey Melnikov, Wes Eddy, Eddie Kohler, a Allison Mankin. Service Name and Transport Protocol Port Number Registry [online]. `http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml`, 2014 [cit. 2014-12-24].

[3] T. Karagiannis, A. Broido, N. Brownlee, K. Claffy, a Michalis Faloutsos. Is P2P dying or just hiding? [P2P traffic measurement]. In *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, volume 3, strany 1532–1538 Vol.3, Nov 2004.

[4] Subhabrata Sen, Oliver Spatscheck, a Dongmei Wang. Accurate, Scalable In-network Identification of P2P Traffic Using Application Signatures. In *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, strany 512–521, New York, NY, USA, 2004. ACM.

[5] Tom Fawcett. An introduction to {ROC} analysis . *Pattern Recognition Letters*, 27(8):861–874, 2006. {ROC} Analysis in Pattern Recognition.

[6] Andrew W Moore a Denis Zuev. Internet traffic classification using bayesian analysis techniques. 33(1):50–60, 2005.

[7] Thomas Karagiannis, Konstantina Papagiannaki, a Michalis Faloutsos. BLINC: multilevel traffic classification in the dark. In *ACM SIGCOMM Computer Communication Review*, volume 35, strany 229–240. ACM, 2005.

[8] Jeffrey Erman, Anirban Mahanti, a Martin Arlitt. Byte Me: A Case for Byte Accuracy in Traffic Classification. In *Proceedings of the 3rd Annual ACM Workshop on Mining Network Data*, MineNet '07, strany 35–38, New York, NY, USA, 2007. ACM.

[9] Matthew Roughan, Subhabrata Sen, Oliver Spatscheck, a Nick Duffield. Class-of-service mapping for QoS: a statistical signature-based approach to IP traffic classification. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, strany 135–148. ACM, 2004.

[10] Andrew W Moore a Konstantina Papagiannaki. Toward the accurate identification of network applications. In *Passive and Active Network Measurement*, strany 41–54. Springer, 2005.

[11] Vern Paxson. Empirically derived analytic models of wide-area TCP connections. *IEEE/ACM Transactions on Networking (TON)*, 2(4):316–336, 1994.

[12] Christian Dewes, Arne Wichmann, a Anja Feldmann. An analysis of Internet chat systems. In *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, strany 51–64. ACM, 2003.

[13] J. Frank. Artificial Intelligence and Intrusion Detection: Current and Future Directions. In *Proceedings of the National 17th Computer Security Conference*, 1994.

[14] Ian H Witten a Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[15] Nigel Williams, Sebastian Zander, a Grenville Armitage. A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification. *ACM SIGCOMM Computer Communication Review*, 36(5):5–16, 2006.

[16] Tom Auld, Andrew W Moore, a Stephen F Gull. Bayesian neural networks for internet traffic classification. *Neural Networks, IEEE Transactions on*, 18(1):223–239, 2007.

[17] Thuy TT Nguyen a Grenville J Armitage. Training on multiple sub-flows to optimise the use of Machine Learning classifiers in real-world IP networks. In *LCN*, strany 369–376, 2006.

[18] Thuy Nguyen a Grenville Armitage. Synthetic sub-flow pairs for timely and stable IP traffic identification. In *Proc. Australian Telecommunication Networks and Application Conference*, 2006.

[19] Jeffrey Erman, Anirban Mahanti, Martin Arlitt, Ira Cohen, a Carey Williamson. Semi-supervised network traffic classification. In *ACM SIGMETRICS Performance Evaluation Review*, volume 35, strany 369–370. ACM, 2007.

[20] Jeffrey Erman, Anirban Mahanti, Martin Arlitt, Ira Cohen, a Carey Williamson. Offline/realtime traffic classification using semi-supervised learning. *Performance Evaluation*, 64(9):1194–1213, 2007.

[21] Rebecca Bace a Peter Mell. NIST special publication on intrusion detection systems. Technical report, DTIC Document, 2001.

[22] Karen Scarfone a Peter Mell. Guide to intrusion detection and prevention systems (idps). *NIST special publication*, 800(2007):94, 2007.

[23] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, a Kuang-Yuan Tung. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24, 2013.

[24] Cheng-Yuan Ho, Yuan-Cheng Lai, I-Wei Chen, Fu-Yu Wang, a Wei-Hsuan Tai. Statistical analysis of false positives and false negatives from real traffic with intrusion detection/prevention systems. *Communications Magazine, IEEE*, 50(3):146–154, 2012.

[25] Igino Corona, Giorgio Giacinto, a Fabio Roli. Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues . *Information Sciences*, 239:201–225, 2013.

[26] Taeshik Shon a Jongsub Moon. A Hybrid Machine Learning Approach to Network Anomaly Detection. *Inf. Sci.*, 177(18):3799–3821, September 2007.

[27] Tony Bradley. Zero Day Exploits: Holy Grail Of The Malicious Hacker [online]. `http://netsecurity.about.com/od/newsandeditorial1/a/aazeroday.htm`, [cit. 2015-01-12].

[28] I Homoliak, M Barabas, P Chmelař, M Drozd, a P Hanáček. ASNM: Advanced Security Network Metrics for Attack Vector.

[29] Andrew Moore, Denis Zuev, a Michael Crogan. *Discriminators for use in flow-based classification*. Queen Mary and Westfield College, Department of Computer Science, 2005.

[30] Christian Kreibich. Netdude [online]. `http://netdude.sourceforge.net`, 2007 [cit. 2015-01-07].

[31] HOMOLIAK Ivan. Metriky pro detekci útoků v síťovém provozu. Master's thesis, 2012.

[32] Maroš Barabas, Michal Drozd, a Petr Hanáček. Behavioral signature generation using shadow honeypot. *World Academy Science Engineering Technology*, 2012:829–833, 2012.

[33] Ivan Homoliak, Daniel Ovšonka, Karel Koranda, a Petr Hanáček. Characteristics of Buffer Overflow Attacks Tunneled in HTTP Traffic. In *International Carnahan Conference on Security Technology*, 48th Annual International Carnahan Conference on Security Technology, strany 188–193. IEEE Computer Society, 2014.

[34] Brian Hernacki, Jeremy Bennett, a James Hoagland. An overview of network evasion methods. *Information Security Tech. Report*, 10(3):140–149, 2005.

[35] Matt Keil. Encrypted tunnels enable users to circumvent security controls [online]. `http://threatpost.com/en_us/blogs/encrypted-tunnels-enable-users-circumvent-security-controls-060109`, 2009 [cit. 2015-01-09].

[36] Ivan Homoliak, Daniel Ovšonka, Matěj Grégr, a Petr Hanáček. NBA of Obfuscated Network Vulnerabilities' Exploitation Hidden into HTTPS Traffic. In *Proceedings of International Conference for Internet Technology and Secured Transactions (ICITST-2014)*, strany 311–318. IEEE Computer Society, 2014.

[37] Daniel Ovšonka. Obfuskace síťového provozu pro zabránění jeho detekce pomocí IDS. Master's thesis, 2013.

[38] Sun Microsystems. RPC: Remote Procedure Call Protocol specification: Version 2. RFC 1057 (Informational), June 1988.

[39] SMB: The Server Message Block Protocol [online]. `http://ubiqx.org/cifs/SMB.html`, [cit. 2015-01-12].

[40] Prahlad Fogla, Monirul Sharif, Roberto Perdisci, Oleg Kolesnikov, a Wenke Lee. Polymorphic blending attacks. In *Proceedings of the 15th conference on USENIX Security Symposium-Volume 15*, strany 241–256. USENIX Association, 2006.

[41] Roberto Perdisci, David Dagon, Wenke Lee, Prahlad Fogla, a Monirul Sharif. Misleading worm signature generators using deliberate noise injection. In *Security and Privacy, 2006 IEEE Symposium on*, strany 15–pp. IEEE, 2006.

[42] Ilsun You a Kangbin Yim. Malware Obfuscation Techniques: A Brief Survey. In *Broadband, Wireless Computing, Communication and Applications (BWCCA), 2010 International Conference on*, strany 297–300. IEEE, 2010.

[43] Oracle VM VirtualBox [online]. `https://www.virtualbox.org/`, [cit. 2015-05-22].

[44] Virtual Hacking Lab - Browse /os/metasploitable at SourceForge.net [online]. `http://sourceforge.net/projects/virtualhacking/files/os/metasploitable/`, [cit. 2015-05-22].

[45] Tomcat Application Manager Login Utility [online]. `http://www.rapid7.com/db/modules/auxiliary/scanner/http/tomcat_mgr_login`, [cit. 2015-04-12].

[46] CVE-1999-0502 [online]. `https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-1999-0502`, 2008 [cit. 2015-04-12].

[47] Apache Tomcat Manager Application Deployer Authenticated Code Execution [online]. `http://www.rapid7.com/db/modules/exploit/multi/http/tomcat_mgr_deploy`, [cit. 2015-04-12].

[48] CVE-2009-3843 [online]. `https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2009-3843`, 2012 [cit. 2015-04-12].

[49] MSSQL Ping Utility [online]. `http://www.rapid7.com/db/modules/auxiliary/scanner/mssql/mssql_ping`, [cit. 2015-04-12].

[50] CVE-1999-0506 [online]. `https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-1999-0506`, 2008 [cit. 2015-04-12].

[51] Microsoft SQL Server Payload Execution [online]. `http://www.rapid7.com/db/modules/exploit/windows/mssql/mssql_payload`, [cit. 2015-04-12].

[52] CVE-2000-0402 [online]. `https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2000-0402`, 2008 [cit. 2015-04-12].

[53] Samba username map script Command Execution [online]. `http://www.rapid7.com/db/modules/exploit/multi/samba/usermap_script`, [cit. 2015-04-12].

[54] CVE-2007-2447 [online]. `https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2007-2447`, 2011 [cit. 2015-04-12].

[55] MS08-067 Microsoft Server Service Relative Path Stack Corruption [online]. `http://www.rapid7.com/db/modules/exploit/windows/smb/ms08_067_netapi`, [cit. 2015-04-12].

[56] CVE-2008-4250 [online]. `https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2008-4250`, 2012 [cit. 2015-04-12].

[57] PostgreSQL Login Utility [online]. `http://www.rapid7.com/db/modules/auxiliary/scanner/postgres/postgres_login`, [cit. 2015-04-12].

[58] PostgreSQL for Linux Payload Execution [online]. http://www.rapid7.com/db/modules/exploit/linux/postgres/postgres_payload, [cit. 2015-04-12].

[59] DistCC Daemon Command Execution [online]. http://www.rapid7.com/db/modules/exploit/unix/misc/distcc_exec, [cit. 2015-04-12].

[60] CVE-2004-2687 [online]. https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2004-2687, 2008 [cit. 2015-04-12].

[61] Penetration Testing For Risk Validation with Metasploit — Rapid7 [online]. http://www.rapid7.com/products/metasploit/index.jsp, [cit. 2015-05-20].

[62] tc(8) - Linux manual page [online]. http://man7.org/linux/man-pages/man8/tc.8.html, [cit. 2015-05-20].

[63] netem — The Linux Foundation [online]. http://www.linuxfoundation.org/collaborate/workgroups/networking/netem, [cit. 2015-05-20].

[64] Chapter 8. VBoxManage [online]. http://www.virtualbox.org/manual/ch08.html, [cit. 2015-05-20].

[65] UML State Machine Diagrams [online]. http://www.uml-diagrams.org/state-machine-diagrams.html, [cit. 2015-05-20].

[66] RapidMiner Studio - RapidMiner [online]. https://rapidminer.com/products/studio/, [cit. 2015-05-22].

Zoznam použitých skratiek a symbolov

skratka	celý názov	vysvetlenie
AD	Anomaly Detection	Detekcia anomálií.
ASNM	Advanced Security Network Metrics	Pokročilé bezpečnostné sieťové metriky na popis vektoru útoku
CVSS	Common Vulnerability Scoring System	Systém skórovania zraniteľností.
DoS	Denial of Service	Odmietnutie služby.
FCBF	Fast Correlation-Based Filter	Druh filtra na výber rysov.
HIDS	Host-based IDS	Hosťovské IDS
HTTP	Hypertext Transfer Protocol	Internetový protokol určený pre výmenu hypertextových dokumentov.
HTTPS	Hypertext Transfer Protocol Secure	Nadstavba sieťového protokolu HTTP.
IANA	Internet Assigned Numbers Authority	IANA je zodpovedná za koordináciu niektorých kľúčových prvkov, ktoré udržujú Internet v hladkej prevádzke.
IDS	Intrusion Detection System	systém na detekciu preniknutia
IPS	Intrusion Prevention System	systém k prevencii preniknutia
IP	Internet Protocol	Protokol pre komunikáciu v Internete.
MIDS	Mixed IDS	Zmiešané IDS
ML	Machine Learning	Strojové učenie.
MTU	Maximum Transmission Unit	Maximálna prenosová jednotka.
NBA	Network Behavior Analysis	Behaviorálna analýza sieťovej komunikácie
NBKE	Naive Bayes Kernel Estimation	Zovšeobecnenie naivnej bayesovskej metódy.
NIDS	Network-based IDS	Sieťové IDS
RFC	Request for Comments	Dokument popisujúci Internetový štandard.
RM	RapidMiner	Nástroj na dolovanie z dát.
SD	Signature Detection	Detekcia signatúr.
SPA	Stateful Protocol Analysis	Kontrola obsahu paketov a sledovanie stavu (sieťového) protokolu.
TCP	Transmission Control Protocol	Protokol pre komunikáciu v Internete.
VM	Virtual Machine	Virtuálny stroj.
WIDS	Wireless-based IDS	Bezdrôtové IDS

Dodatok A

Obsah DVD

- `data` – zaznamenaná sieťová komunikácia,
- `datamining` – použitý repozitár programu RapidMiner, kde sú uložené dátové tabuľky, procesy a výsledky,
- `doc` – zdrojové kódy textovej časti diplomovej práce v LaTeX-u,
- `src` – zdrojové kódy vytvoreného nástroja,
- `tools` – použité nástroje na extrakciu ASNM zo zaznamenanej sieťovej komunikácie,
- `README.en` – popis obsahu DVD v anglickom jazyku,
- `README.sk` – popis obsahu DVD v slovenskom jazyku.