VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

AUTOMATIZACE TVORBY DETEKTORŮ VYBRANÝCH SÍŤOVÝCH ÚTOKŮ

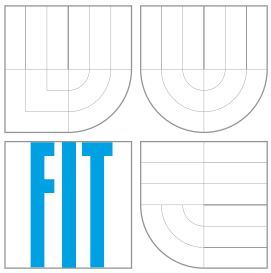
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

LUKÁŠ HUTÁK

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

AUTOMATIZACE TVORBY DETEKTORŮ VYBRANÝCH SÍŤOVÝCH ÚTOKŮ

AUTOMATED DEVELOPMENT OF NETWORK ATTACK DETECTORS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — LUKÁŠ HUTÁK
AUTHOR

VEDOUCÍ PRÁCE — Ing. MARTIN ŽÁDNÍK, Ph.D.
SUPERVISOR

BRNO 2015

Abstrakt

Tato práce se zabývá automatizovanou tvorbou detektorů síťových útoků. Popisuje návrh tvorby vzorů běžného a útočného chování na základě monitorování síťového provozu vybraných služeb. Vzory jsou reprezentovány generovanými statistikami, u kterých se klade důraz na výběr vhodných metrik. Použitím strojového učení se vzory užijí k tvorbě detektorů síťových útoků. Na základě návrhu byl implementován modul do systému Nemea v programovacím jazyce C/C++.

Abstract

The thesis is focused on automated development of network attack detectors. It describes a design of patterns developed for normal and offensive behaviors based on monitoring network traffic of selected services. Patterns are represented by statistics with a focus on suitable metrics. Using machine learning algorithms attack detectors are created from behavioral patterns. Finally, a module was implemented for Nemea system in C/C++ programming language based on the proposal.

Klíčová slova

síťová bezpečnost, NetFlow, honeypot, automatizované zpracování, vzory chování, Nemea

Keywords

network security, NetFlow, honeypot, automated processing, behavioral patterns, Nemea

Citace

Lukáš Huták: Automatizace tvorby detektorů vybraných síťových útoků, bakalářská práce, Brno, FIT VUT v Brně, 2015

Automatizace tvorby detektorů vybraných síťových útoků

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Martina Žádníka, Ph.D.

.....
Lukáš Huták
20.května 2015

Poděkování

Rád bych poděkoval vedoucímu Ing. Martinu Žádníkovi za veškeré odborné informace a pomoc při tvorbě této bakalářské práce. V neposlední řadě děkuji rodině za podporu a trpělivost po celou dobu mého studia.

Obsah

Kapitola 1

Úvod

Naše společnost je v současné době vysoce závislá na množství služeb, které se rozšířily za pomocí sítě Internet a již nyní je většina odvětví postavena na rychlé komunikaci a výměně dat bez omezení hranic a s minimálním časovým zpožděním. Dále je nutné si uvědomit, že to není tak dávno, kdy převládalo ukládání informací jen v papírové podobě. Nyní je celá řada firemních a osobních informací uchovávána v podobě digitální, což usnadňuje a urychluje práci. Na druhou stranu také přináší riziko v podobě jejich relativně snazšího odcizení. Čím více jsou internetové služby rozšířené a uživateli oblíbené, roste o ně i zájem z řad kyberútočníků. Ti v nich hledají bezpečnostní nedostatky a slabiny, které se snaží využít pro svůj prospěch. Takto vznikající incidenty zasahují v lepším případě pouze do soukromí uživatelů. Čím dál častěji se lze ovšem setkat se zneužitím odcizených osobních ale i firemních informací. V důsledku se tak potýkáme se vznikem finančních a jiných škod, které mohou být pro osoby ale i firmy likvidační. Je tedy nezbytné umět odhalit nové ale i stávající hrozby a vytvářet prostředky usnadňující jejich detekci.

Pro detekci hrozeb přímo na zařízení nebo na úrovni lokální sítě existuje řada specializovaných nástrojů. Na druhou stranu lze některé typy incidentů odhalit už na úrovni poskytovatele internetového připojení, který má širší přehled o celkovém provozu sítě, neboť spravuje připojení i pro řadu dalších odběratelů. K tomu, aby bylo možné hrozby vyhledávat, je nutné ideálně v reálném čase sledovat právě probíhající síťové toky. Řada existujících nástrojů se zaměřuje na srovnávání obsahu přenášených dat se známými signaturami útoků. Problém těchto nástrojů ale představuje vysoká výpočetní náročnost, a nejen proto je nelze využít na vysokorychlostních páteřních sítích.

Sledování se tudíž orientuje na získávání síťových statistik přenášených toků. Statistiky se vytváří z pasivního monitorováním uživatelů, kdy nedochází k interakci s uživatelem, a tím pádem ani netuší, že jejich provoz je zaznamenáván a případně vyhodnocován. Nad získanými statistikami lze uplatňovat algoritmy s detekčními pravidly anomálního chování. Pro vznik takovýchto pravidel je nejprve nutné odhalit a sledovat útočníky při probíhajících incidentech a zjistit odlišnosti oproti běžnému legitimnímu chování uživatelů. K tomu mohou pomoci honeypoty, jejichž cílem je přilákat útočníky a zaznamenat jejich činnost. Zjištěné odlišnosti v chování pak je možné za pomoci strojového učení využít pro tvorbu nových detekčních pravidel, které budou následně schopny odhalit útočníka napadajícího i běžné cíle.

Náplní této bakalářské práce je vytvořit nástroj pro automatizovanou tvorbu detekčních pravidel vybraných síťových útoků, který na základě statistik o tocích na síťové služby ze sledovaného provozu vysokorychlostní páteřní sítě vytvoří detekční pravidla. V tomto nástroji budou honeypoty využívány k nalezení útočníků, u nichž bude jejich síťový provoz

cíleně zkoumán a povede k vytvoření vzorů anomálního chování.

Nedílná součást práce se zabývá výběrem vhodných metrik pro síťové statistiky. Ať už je sledován zcela běžný uživatel nebo útočník právě podnikající útok, musí metriky vhodně reprezentovat celkovou charakteristiku chování na síti. Z metrik je nutné vybrat pouze ty, u nichž lze říci, že značnou mírou napomáhají k určení odlišností mezi legitimním a nelegitimním provozem. Pro tvorbu detekčních pravidel se dále využije strojové učení z množiny zachycených statistik, což ovšem vyžaduje, aby statistiky obsahovaly pouze záznamy útočníků a důvěryhodných uživatelů s jejich patřičným odlišením. Z tohoto důvodu se i část práce věnuje možnému způsobu, jak tyto dvě skupiny v nepřeberné záplavě síťových toků určit.

Kapitola 2

Teoretický úvod

Než bude rozbírán návrh principu umožňující zpracovávat síťový provoz za účelem oddělit v něm běžné a nebezpečné uživatele, je potřeba si představit základní technologie, které jsou pro návrh nezbytné. V této kapitole tak mimo jiné budou představeny moderní metody sledování síťového provozu, jejich principy a důvody z jakých je vhodné tyto technologie aktivně využívat. K tomu, aby se ve sledovaném provozu snadněji a s poměrně vysokou mírou důvěry dali útočníci rozpoznat, bude vysvětlena činnost síťových honeypotů. Jakmile známe chování útočníků lze v něm nalézt vzory, a proto se nástrojům pro strojové učení věnuje poslední část kapitoly.

2.1 Metody monitorování sítě a technologie NetFlow

V důležitých sítích, jako jsou páteřní a firemní sítě, ke správě také patří sledování, co se na síti děje a v jakém stavu se nachází síťové prvky. Takovéto informace napomáhají administrátorovi k zajištění méně náchylného provozu sítě a v případech, kdy nastává nečekaná událost, mohou informace o aktuálním stavu pomoci identifikovat příčinu problémů a usnadnit jejich odstranění. Monitorování sítě se obvykle neprovádí nahodile jen ve vybrané období, ale probíhá nepřetržitě, neboť není možné předvídat budoucí problémy a incidenty. Rozlišujeme aktivní a pasivní způsobu monitorování.[10]

Aktivní monitorování vyžaduje vynucenou interakci, kdy dochází k účelné komunikaci s cílem otestování dostupnosti linek, uzlů nebo služeb přítomných na síťových zařízeních. Mezi hojně používané protokoly patří ICMP a SNMP, kde první jmenovaný lze využít (nejen) pro zjištění dostupnosti zařízení (používají jej např. příkazy `ping` a `traceroute`) a druhý vyčítá nejrůznější data o konfiguraci a stavu aktivních zařízení. Obvykle se za pomoci periodického opakování automaticky zjišťuje dostupnost, stav zařízení a služeb. Když se během zvoleného časového období opakovaně nedaří prokázat správnou funkčnost, je informován administrátor. [10]

Pasivní monitorování se oproti aktivnímu odlišuje v tom, že nejsou podnikány přímé akce, ale využívá a sleduje probíhající komunikaci. Zejména sbíráním a rozborem logovacích souborů služeb a zařízení si vytváříme obrázek o jejich stavu. I v tomto případě se může využívat protokol SNMP, kdy jsou asynchronně posílána upozornění na události při jejím vzniku. V neposlední řadě se jedná o zachytávání síťového provozu.

Při analyzování provozu u pasivního monitorování sledujeme nejen linkovou a síťovou vrstvu, ale zkoumání můžeme provádět až do úrovně vrstvy aplikační. Monitorovací sonda se obvykle nasadí na sdílené síťové médium a zachytává procházející provoz nebo provoz, který byl na ní zrcadlen z jiného portu síťového zařízení. V případech, kdy hloubkově analyzujeme provoz s využitím populárních nástrojů `tcpdump` nebo `wireshark`, zachytáváme veškerý obsah přenášených dat. Výpočetní náročnost zpracování takto zachycených dat v reálném čase se u vytížených páteřních sítí stává nereálnou, neboť i na vytížené gigabitové lince za hodinu projdou stovky GB dat. Z hlediska dlouhodobé správy a rozvoje sítě není nezbytně nutné znát povahu momentálně přenášených dat. Proto se spíše monitorují celkové statistiky provozu sítě a uživatelů za delší časové období. Této oblasti se věnují různé technologie, z nichž nejrozšířenější je NetFlow.

Technologie NetFlow

NetFlow bylo navrženo firmou Cisco původně jako proprietární technologie používaná na jejich vybraných síťových prvcích pro monitorování toků na síti, ale v dnešní době ji lze najít i zařízeních jiných výrobců. Problém páteřních sítí představuje masivní množství toků, které není možné z pohledu požadovaného výpočetního výkonu v reálném čase hloubkově analyzovat. Nabízí se řešení, kdy se pakety zkoumají obvykle jen do úrovně transportní vrstvy. Za cenu ztráty detailních informací o provozu se tak příznivě snižují výpočetní nároky, což umožňuje nasazení i na vysokorychlostních sítích.

Při monitorování jsou vytvářeny NetFlow statistiky o jednotlivých síťových tocích (anglicky „flow“). Dle standardu RFC 3954 [7] se za síťový tok považuje množina paketů procházejících na síti bodem pozorování během daného časového intervalu, kde všechny pakety spadající k jednomu toku mají shodnou množinu společných klíčových vlastností odvozených od dat z paketů. Mezi klíčové vlastnosti identifikující tok patří:

- Zdrojová IP adresa
- Cílová IP adresa
- Zdrojový port
- Cílový port
- Protokol
- ToS (typ služeb)
- Vstupní rozhraní

Z uvedených položek lze odvodit, že komunikace mezi dvěma navzájem komunikujícími stroji, i když je komplementární, bude představovat alespoň 2 síťové toky (každý pro jeden směr). Jeden NetFlow záznam tak vždy vyjadřuje jeden směr komunikace. Samotné klíčové informace ovšem neposkytují dostatek informací o pohledu na síť, a proto se ke každému toku ukládají navíc rozšířené informace v podobě položek o počtu přenesených bytů, množství paketů, času prvního a posledního zachyceného paketu a další. Z kombinace klíčových položek a příslušných rozšířených informací se dá například říci, že proběhla komunikace konkrétního zařízení v danou dobu pomocí zjištěného protokolu s jiným zařízením a přeneslo se určité množství dat. Záznam o toku tedy obsahuje pouze statistické informace a nezahrnuje obsah přenosu.

Velmi často se zařízení generující NetFlow záznamy nasazují na hraniční body mezi sítěmi, a to s cílem zjistit, jaký provoz přichází a odchází ze sítě. Správci firemních sítí

ocení možnost snadno zjistit, odkud a kam chodí nejvíce dat. Populární je i detekce anomálií a hrozeb, neboť ze statistických dat lze určit některé síťové útoky nebo i snahu o šíření červů mezi zařízeními.

```
Date first seen         Duration Proto  Src IP Addr:Port     Dst IP Addr:Port  Pkts Bytes
2015-03-15 15:14:52.408    5.409 TCP    AA.BB.71.69:46614 ->  CC.DD.9.21:80       4   172
2015-03-15 15:14:52.547    5.107 TCP     CC.DD.9.21:80    -> AA.BB.71.69:46614    3   132
```

Obrázek 2.1: Zkrácená podoba NetFlow záznamu (anonym. výpis z nástroje `nfdump`)

2.1.1 Komponenty

Pro monitorování sítě je potřeba zařízení snímající provoz v daném bodě sítě a takové zařízení nazýváme exportér (specializovaná hardwarová zařízení označujeme také jako sondy). Získané statistiky z jednoho nebo více exportérů se odesílají do sběrného místa označovaného jako kolektor, jehož činností je data uchovávat. Architektura pro monitorování je tak postavena na dvou typech zařízení, mezi nimiž probíhá komunikace prostřednictvím protokolu NetFlow.

Exportér může být router, L3 switch nebo také specializované hardwarové zařízení označované jako sonda s provozem, jenž byl na ni odbočen. Síťové zařízení sleduje jednotlivé pakety a jejich klíčové vlastnosti určující tok. Pro každý zaznamenaný tok si vytvoří záznam v paměti označované jako „NetFlow cache“. První paket toku způsobí založení záznamu a další ho pouze aktualizují (přidávají počet paketů, bytů, apod.). Toky se po určité době označí jako ukončené, záznamy odešlou na kolektor a z paměti se uvolní.

Kolektor přijímá data z jednoho nebo obvykle více exportéru, dekóduje je a ukládá do datového uložiště. Uložištěm může být databáze navržená pro rychlé vyhledávání podle klíčových položek nebo speciální formáty souborů, ve kterých se za časový interval (např. 5 minut) ukládají statistiky zachycených toků. Z přijímaných dat musí kolektor určit z jaké sondy, a tedy i místa monitorování data pochází. Tok totiž může být pozorován vícero sondami na různých místech - tam kde vzniká, kudy prochází a kde končí. Z toho vyplývá, že pokud jeden a ten samý tok projde přes více sond, budou NetFlow záznamy na kolektoru duplicitní. Záznamy by se daly při budoucím zpracování zdánlivě snadno deduplikovat, aby to ale nebylo tak jednoduché, může nastat situace, kdy např. z důvodu vytížení některé z linek je tok částečně směrován přes jiné body sítě.

Vzhledem k velkému množství toků, které jsou přítomny na vysokorychlostních sítích, se vytváří odpovídající množství statistik, a tak kolektor nezbytně musí umět starší záznamy s toky sám odmazávat. Pro komunikaci kolektoru a uživatele (obvykle administrátora) se často používají webové grafické nadstavby pro pokládaní dotazů nad daty. Obvykle nabízí předpočítané statistiky v podobě rozložení protokolů, komunikace na dané lince, rychlosti apod.

2.1.2 Verze NetFlow a technologie IPFIX

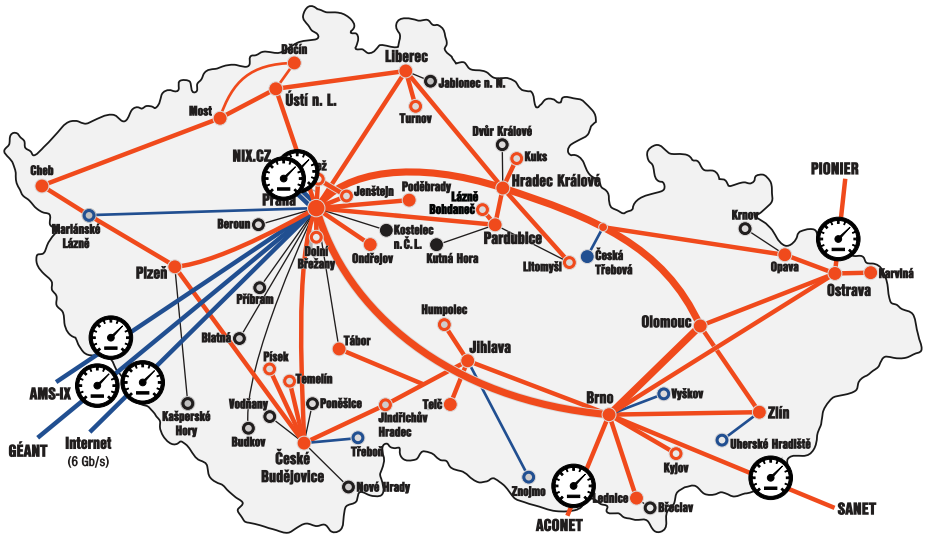
Původní technologie NetFlow měla sloužit jako urychlení pro směrování paketů, kdy se pro první paket vytvořil záznam s informací o směrování, který se aplikoval na následující pakety z daného toku [5]. Protokol NetFlow se v průběhu času upravoval, vyvíjel a své místo si našel v oblasti monitorování sítí. Od verze 5 došlo k jeho masovému šíření a poskytování této technologie i jinými výrobci. Tato verze však měla konstantní podobu zaznamenávaných položek, a proto další významný milník představovala verze 9 s možností flexibilního určení položek sledovaných statistik pomocí šablon.

Od počátku představovalo NetFlow proprietární technologie firmy Cisco. Jak se technologie stávala populárnější, existovala potřeby vytvořit univerzální a společný standard pro přenos informací o tocích. Nově vzniklý protokol IPFIX (Internet Protocol Flow Information Export)[8] silně vychází z NetFlow verze 9, a proto je někdy označován i jako NetFlow verze 10. Tento relativně nový standard dále rozšiřuje funkcionalitu o možnost vytvářet si vlastní položky, což v případě nasazení rozšiřujících modulů do exportéru umožňuje sledovat i aplikační protokoly.

2.1.3 Monitorování síťových toků na síti organizace CESNET

Jelikož se tato bakalářská práce věnuje generování detekčních pravidel z provozu na vysokorychlostní síti, budou veškeré poznatky a výsledky vycházet z provozu sítě sdružení CESNET. Sdružení vzniklo v roce 1996 z iniciativy 26 vysokých škol a Akademie věd České republiky [3]. V České republice plní roli národní výzkumné a vzdělávací sítě (angl. NREN, National Research and Education Network) a činnosti sdružení mimo jiné zahrnují výzkum a vývoj v oblasti informačních a komunikačních technologií. Provozovaná síť je připojena z velké části do sítí cizokrajných výzkumných a vzdělávacích sítí a také do sítě NIX, která sdružuje české i zahraniční poskytovatele internetového připojení.

Na všech hraničních bodech, kde je síť připojena k sítím jiných poskytovatelů, se nachází měřící sondy, které veškeré statistiky odesílají na centrální kolektor. Sondy umožňují sledovat toky přicházející a odcházející ze sítě, ale uvnitř ní se nenachází, a tedy není možné vidět komunikaci mezi jednotlivými připojenými členy ani případné vnitřní incidenty. Za jeden měsíc se vygeneruje necelých 8 TB komprimovaných NetFlow záznamů, které se uchovávají a slouží k případné zpětné analýze. Starší záznamy jsou automaticky odmazávány a nahrazovány novými. Kolektor si pro každou linku zvlášť uchovává data v samostatných souborech po 5 minutových intervalech ve formátu `nfdump`[1].



Obrázek 2.2: Topologie sítě s abstraktně zakreslenými měřícími body[3]

[1] Nfdump je populární balík nástrojů pro sběr a manipulaci s NetFlow daty

2.2 Honeypoty

Na síti Internet se často pohybují útočníci, jejichž cílem je proniknout do nezabezpečených systému. Běžně můžeme vídat, že jsou stále odhalovány nové více či méně kritické zranitelnosti v počítačových systémech a útočníci se je snaží využít, dokud nebyly patřičně opraveny. Klasické bezpečnostní technologie jako antiviry, firewally a IDS/IPS systémy odhalují obvykle známé nebo obdobné pokusy o útok a tím přímo brání zranitelné systémy. U nových zranitelností, které ani nejsou podobné těm již známým, tyto systémy často selhávají.

K odhalování nových zranitelností by bylo vhodné mít nastražené „pasti“, do nichž by se útočník chytil a nevědomky tak ukázal celou podobu svého útoku včetně zranitelností, které k tomuto účelu zneužil. Právě tato oblast je cílem honeypotů, které se snaží přitáhnout útočníky a umožnit jim provést útok na (obvykle zdánlivě) reálné stroje. Záznamy vytvořené z těchto útoků se následně využívají k analýze a případnému zjištění nových slabin. Honeypoty tak nejsou bezpečnostní systémy, které by přímo chránily koncové uživatele nebo lokální sítě. Jejich smysl je odlišný a spočívá ve snaze pomáhat objevit a rozebrat nové zranitelnosti a typy útoků. Tím vylepšují stávající klasické bezpečnostní systémy dříve, než dojde k masovějšímu rozšíření útoku na objevenou zranitelnost.

Obvykle se honeypoty ocitají v pozici serverů očekávajících příchozí požadavek. Snaží se nalákat pouze útočníky s tím, že běžní legitimní uživatelé by s nimi za žádných okolností komunikovat neměli. Potom by se teoreticky dalo říci, že kdokoliv komunikuje s honeypotem, je útočník a jeho aktivita je potenciálně škodlivá a měla by být uchována pro pozdější analýzu. K tomu, aby se odfiltrovali jen útočníci, se v některých případech nasazeným honeypotům nepřiřazují žádné DNS záznamy a tím pádem na ně nevedou z pohledu běžného uživatele žádné odkazy. Útočníci takovéto honeypoty objeví až za pomoci skenování sítě. Skenováním sítě obvykle rozumíme posílání jednoduchých požadavků na spojení s vybranými službami napříč všemi adresami počítačů v dané části sítě, aby si útočníci ověřili, že se na druhé straně někdo nachází, požadavky poslouchá a rozumí jim. Takové skenování sítě je mimochodem běžným postupem útočníků a předchází útokům.

Pro větší efektivitu jsou některé honeypoty schopny vytvořit řadu virtuálních strojů, které obsadí část rozsahu sítě. Jeden honeypot tak obstarává více síťových adres, čímž se efektivně využijí prostředky a zvyšuje pravděpodobnost, že bude kontaktován útočníkem. S touto vlastností se lze setkat zejména u některých nízko interaktivních honeypotů.

2.2.1 Dělení honeypotů

Honepoty může dělit dle několika kategorií. Jednou z nich jsou honeypoty fyzické a virtuální. Jak název napovídá, fyzické jsou provozovány na skutečném hardwaru a virtuální mohou být představovány virtuálním počítačem nebo softwarem simulujícím část služeb. Za nejvýznamnější se ovšem považuje dělení dle míry interaktivity s útočníkem.

Honeypoty s nízkou mírou interakce jsou vhodné, pokud není nutné přímo zaznamenat věrohodné chování útoku. Představují vhodný nástroj, jak odlákat útočníka od primárních systémů a zdržet jej pří jeho činnosti. Průběh útoku nelze rekonstruovat, neboť simulují pouze úzkou část služby. Často jsou implementovány jednoduchým skriptem, který celou komunikaci řídí, a proto by nemělo docházet ke kompromitaci samotného honeypotu. Oproti honeypotům s vysokou mírou interakce jsou snadno nasaditelné, lépe se udržují, ale jejich odhalení nastává už po krátké době interakce.

Honeypoty s vysokou mírou interakce reálně simulují služby a snaží se napodobit skutečnou interakci útočníka se serverem. Může se jednat o plnohodnotné systémy, které však mohou být kompromitovány a útočníkem dále zneužity. Je tedy potřeba zajistit, aby nebyly využity k napadání jiných počítačů nebo aby se např. nepodílely na rozesílání nevyžádané pošty. Veškerý postup útočníka se detailně zaznamená včetně použitých nástrojů a stažených souborů. Analýzou zaznamenaného průběhu se dají identifikovat zneužité zranitelnosti systému použité při útoku.

2.2.2 Honeypoty v síti organizace CESNET

Jelikož samotný nástroj pro tvorbu detekčních pravidel popisovaný v pozdějších kapitolách vyžaduje sledování provozu na honeypoty, bylo nutné je v síti CESNET nalézt. Podařilo se sehnat přibližné informace o počtu adres, které byly v sítích některých připojených organizací vyhrazeny pro honeypoty.

Kvůli aktivnímu nasazení honeypotů na sítích různých provozovatelů, nebudou tyto instituce ani rozsahy adres zveřejněny. Uveřejnění adres by mohlo mít nepříznivý vliv na jejich činnost např. tím, že potenciální útočníci by se jim úmyslně vyhýbali.

Hledání na základě indicií

Jako vhodný počáteční bod pro hledání se zprvu jevilo zanalyzování NetFlow dat útoků, které proběhly v březnu 2013 na významné české servery. Tyto útoky primárně vedly na webové stránky zpravodajských médií, bank a mobilních operátorů. Útok byl mimo jiné veden formou požadavků vytvořených útočníkem s podvrženou zdrojovou adresou tak, aby vypadaly, že odesílatelem je server pod útokem. Požadavek byl odeslán na předem vybrané zneužitelné počítače, které reagovaly na vzniklou komunikaci a útok odrážely na skutečný server, čímž jej zatěžovaly.

Mezi tyto zneužité počítače měly patřit i honeypoty s nízkou mírou interakce, které se údajně projevovaly několikanásobným přeposláním požadavků v případě, že protější strana včas nereagovala. Navíc samotné honeypoty reagovaly na běžně neotevřené porty. Tento způsob vyhledávání je poměrně nepřesný a vyžaduje generování mnoha dotazů nad NetFlow záznamy, což způsobovalo dlouhé doby výpočtu, a proto se od tohoto postupu upustilo.

Hledání v systému Warden

V rámci organizace CESNET je provozován i projekt Warden, což je systém pro sdílení informací o detekovaných bezpečnostních událostech[1]. Zapojení členové do systému odesílají informace o detekovaných síťových incidentech a zároveň si mohou stahovat informace od jiných členů. Nahlášené incidenty pochází z honeypotů a jiných detekčních systémů. Formát záznamu incidentů má ve Wardenu (verze 2.1) konstantní podobu.

```
#id,hostname,service,detected,type,source_type,source,target_proto,target_port,scale,note
80153859,<serverA.cz>,Nemea,2015-03-14 22:49:28,portscan,IP,XX.YY.198.19,TCP,,268,"SYNscan"
80153860,<serverB.cz>,HPScan,2015-03-14 22:47:58,portscan,IP,XX.YYY.54.160,TCP,3389,2,
80153871,<serverB.cz>,HPScan,2015-03-14 22:45:04,probe,IP,XX.YY.252.50,TCP,23,24,
80153872,<serverB.cz>,HPScan,2015-03-14 22:45:06,probe,IP,XX.YY.252.117,TCP,23,32,
80153912,<serverC.cz>,Dionaea,2015-03-14 22:45:39,portscan,IP,XX.YY.141.28,tcp,23,2,
80153913,<serverC.cz>,Dionaea,2015-03-14 22:48:09,portscan,IP,XX.YY.135.131,tcp,2083,1,
```

Obrázek 2.3: Příklad záznamů v systému Warden (zkráceno a anonymizováno)

Výše uvedené položky `hostname` a `service` určují adresu serveru a název služby, která událost nahlásila. Z adresy serveru lze tedy určit síť poskytovatele a z názvu služby se obvykle dá odvodit, jestli událost nahlásil honeypot. Z položek záznamu je patrné, že je pouze nahlášena IP adresa útočníka (položka `source`), ale adresa oběti není uvedena. Ve vybraných případech, kdy jako `service` vystupuje honeypot, tak není známá jeho adresa. Mezi další významné položky ještě patří `detected`, `target_proto` a `target_port`, které představují čas detekce události, protokol komunikace a cílový port útoku.

Z incidentů se vybraly pouze ty záznamy, které byly nahlášeny honeypoty. Jelikož ze zadaného formátu lze vyčíst, že ohlašující služba (honeypot) označila konkrétní IP adresu útočníka a dobu, kdy se incident odehrál, lze trochu netradičně Warden využít v opačném směru k nalezení nahlašující adresy vyhledáním incidentu v NetFlow datech. Pro identifikaci záznamu se pokládaly dotazy, které se skládaly z částí:

`src ip <`*`IP útočníka`*`> and dst net <`*`síť s honeypotem`*`> and dst port <`*`cílový port`*`> and proto <`*`protokol`*`>`

Dotaz hledá NetFlow záznamy, kdy ve specifikovaném časovém okně probíhá komunikace od útočníka (`src ip`) na síť organizace (`dst net`), co útok nahlásila, a obsahuje příslušný cílový port a protokol. Vzhledem k tomu, že se nahlášený čas může lišit od času v NetFlow datech kvůli odlišnému času na statistiky tvořícím exportéru a honeypotu, probíhalo hledání v časových oknech. Časové okno bylo reprezentováno soubory s pěti minutovými NetFlow statistikami uloženými na kolektoru. Ke správné identifikaci příslušných souborů s časovým oknem se musel započítat i vliv letního času a časové zóny kvůli používání odlišného formátu na kolektoru a v systému Warden. Se zadaným filtrem bylo ručně analyzováno pár záznamů incidentů a v případě, že výsledkem dotazu byla pouze jedna unikátní cílová IP adresa, se jednoznačně dalo říct, že se jedná o IP adresu používanou honeypotem. Funkčnost řešení ovšem občas narážela na situace, kdy útočník např. prováděl skenování sítě, což vedlo ve výsledku k celé řadě potenciálních adres honeypotů, a tudíž nešlo jednoznačně identifikovat používanou IP adresu ani rozsah. Tento poměrně častý handicap neměl celkový vliv na odhalování adres, neboť díky značnému množství záznamů o incidentech, byli nalezeni alternativní útočníci, kteří stejné chování nevykazovali. Proces vyhledávání byl zautomatizován napsáním primitivního skriptu v jazyce Python, který pro každý incident položil dotaz, výsledek vyhodnotil a případnou adresu honeypotu poznamenal. Závěrem se některé výsledné adresy ručně sloučily do sítí s příslušným prefixem. Z výsledků bylo patrné, že při situacích, kdy bylo nevhodně zvolené časové okno, se zřídka objevily i chybně určené adresy honeypotů. Kvůli jejich malému zastoupení se ovšem daly snadno ve výsledcích odhalit a chybu eliminovat.

Analýza nalezených honeypotů

Celkově se podařilo nalézt přes 5000 adres, z nichž se většinou jednalo o adresy ve společném prefixu. Analýzou odhalených honeypotů bylo zjištěno, že většina náleží do kategorie s nízkou mírou interakce. Pro strojové učení tak sledování a záznam komunikace útočníka s honeypotem nemá význam kvůli nedostatečné simulaci reálného chování. Při návrhu algoritmu pro tvorbu vzorů chování tak je nutné s touto vlastností počítat. Jelikož však útočníci jsou odhaleni honeypoty, je možné sledovat jejich komunikaci s jinými stroji a využít ji.

2.3 Strojové učení

Lidstvo vyprodukuje každým rokem větší a větší množství dat, které může díky pokroku ve výpočetní technice uchovávat a dohledávat místo toho, aby je ztrácelo. Samotné uchování v dnešní době už nestačí a nadneseně se dá říci, že nevyužití cenných dat představuje plýtváním úložným prostorem. V komplexních datech lze pomocí technik strojového učení odhalit pro člověka těžko spatřitelné vzory, které mohou např. v obchodní sféře představovat markantní konkurenční výhodu. Vhodným příkladem mohou být obchody, jenž si mohou pomocí věrnostních karet nebo internetových účtů k jednotlivým zákazníkům uchovávat jejich nákupní preference, a tak odhadovat, o které další produkty nebo slevy mohou mít zájem podle chování obdobných zákazníků. Uvedený případ demonstruje jeden z neomezeného množství potenciálního využití strojového učení, které se obecně zabývá technikami a algoritmy zpracování dat. Naučený klasifikační systém dokáže na základě získaných znalostí zpracovávat stejné, podobné nebo i úplně nové úkoly inteligentně. Metody strojového učení můžeme podle povahy učení rozdělit do několika níže uvedených skupin. [13]

Učení s učitelem využívá pro každý krok učení vektor trénovacích dat, pro něhož je požadovaný výstup od systému předem známý. Budovaný systém tak přímo ví, jaké ohodnocení připadá pro konkrétní krok učení. Cílem je předpovědět výsledek i pro jiná data než byla použita pro učení. Mezi významné zástupce patří algoritmus *C5.0*, který dokáže vytvářet pro člověka snadno pochopitelné rozhodovací stromy nebo pravidla.

Učení bez učitele nepoužívá při učení žádný předepsaný výstup systému. Vznikající systém sám musí být schopen pomocí technik klastrování či určení závislostí, najít shluky, do kterých vstupní data vhodně zařadí, přičemž nedostává žádnou informaci o správnosti své klasifikace. Minimální pomoc tak někdy představuje jen informace o očekávaném množství shluků. Známým zástupcem této kategorie je např. algoritmus *k-means*.

Posilované učení je podobné k učení s učitelem avšak představuje postup, kdy budovaný systém je až na základě své reakce na testovací vektor ohodnocen kladně nebo záporně a navíc ještě s různou intenzitou.

Implementace klasifikačních algoritmů strojového učení označujeme jako klasifikátory. Po fázi učení použitého klasifikátoru je vhodné vědět, s jakou úspěšností dokáže stejná nebo jiná data než byla použita při učení roztřídit do správných kategorií. Existuje několik postupů, jakými lze otestovat míru znalostí, které získal ze vzorku poskytnutých trénovacích dat. Níže uvedené příklady představují jen část z možných postupů.

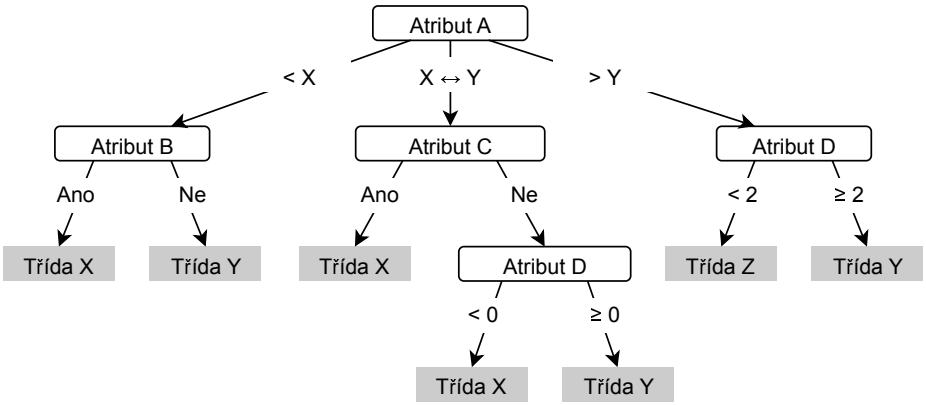
Ověření klasifikátoru na základě dat použitých při učení představuje neobjektivní měření, neboť pro vyhodnocení se používají ta stejná data jako při učení. Lze tedy odvodit, že je také bude umět poměrně dobře roztřídit. Tento způsob testování není vůbec doporučené používat, jelikož může vést k chybným úsudkům a je téměř jisté, že nad jinými daty bude klasifikátor vykazovat nižší přesnost.

Ověření klasifikátoru na základě dat nepoužitých při učení je opakem předchozí testovací metody, neboť používá testovací sadu, kde žádný ze vzorků nebyl použit při učení. Díky tomu je považován i za nejvhodnější. Problém ovšem může nastat, když pro potřeby učení není dostupné dostatečné množství dat a vymezení části dat pro potřeby testování by mohlo mít nepříznivý vliv na učení klasifikátoru.

Křížová validace (Cross-validation) využívá stejná data pro učení a testování. Algoritmus je přitom vhodný i při učení nad omezeným vstupním vzorkem. Princip spočívá v rozdělení dat na obvykle 10 samostatných částí, kdy klasifikátor se bude od začátku učit celkem 10x a to tak, že pro potřeby učení bude vždy využito 9 z 10 částí dat a pro testování se použije zbývající jedna část. Postupně se všechny části prostřídají a celkový výsledek představuje průměr ohodnocení dílčích testování. Je třeba si ovšem uvědomit, že předchozí kroky vedly k určení míry úspěšnosti, avšak samotný výsledný klasifikátor se učí nad kompletními daty [12].

Algoritmus C5.0

Novou generaci algoritmů strojového učení s učitelem založenou na tvorbě rozhodovacích stromů představuje algoritmus *C5.0*[2]. Ze seznamu dostupných atributů a množiny trénovacích případů, kde jsou známy i očekávané výstupy, dovoluje sestavit rozhodovací strom nebo více stromů, které jsou následně využity pro oklasifikování dalších případů, u nichž je třeba výstup teprve určit. Rozhodovací strom se skládá z uzlů, kde každý uzel představuje rozhodování podle jednoho atributu. Z tohoto uzlu vede konečné množství hran do jiných uzlů. Koncové uzly pak představují výslednou třídu klasifikace. Klasifikátor nejprve začíná vytvářet strom od uzlu s atributem, který dokáže ze všech nejlépe odlišit případy v testovacích datech podle jejich výsledné třídy. V nižších uzlech už obdobně pracuje jen s podmnožinou příslušných testovací případů.



Obrázek 2.4: Ukázka rozhodovacího stromu

C5.0 byl vyvinut jako vylepšený nástupce známého a užívaného klasifikátoru *C4.5* a přináší řadu funkcí jako je [4]:

- podpora pro automatický výběr vhodných atributů ještě před začátkem tvorby rozhodovacího stromu. Nevhodné atributy, které se s minimálním významem podílí na klasifikaci, jsou ignorovány, což v důsledku přináší jednodušší výsledné stromy.
- adaptivní „boosting“, což je technika, při níž se sestrojí na základě trénovacích dat více odlišných klasifikátorů. Klasifikátory se vytváří postupně a vždy věnují zvýšenou pozornost případům, kde jejich předchůdce selhával. Při klasifikaci nového případu pak každý z klasifikátorů předpoví výstupní třídu a na základě zastoupení hlasů se určí celková výsledná třída.

- variabilní ohodnocení chybné klasifikace, které v podstatě vyjadřuje cenu za konkrétní chybné označení výstupní třídy místo té správné.
- podpora křížové validace a jiné

Klasifikátor *C5.0* byl pro použití v této práci mimo jiné zvolen kvůli dostupné implementaci s otevřený zdrojovým kódem [2] a široké škále možností konfigurace. Hlavní důvod ovšem představovalo jeho úspěšné nasazení Tomaszem Bujlowem a jeho kolegy v práci [6] zabývající se klasifikací síťového provozu za účelem přiřazení provozu příslušnému typu aplikace (Skype, FTP, SSH, aj.). Ve zmíněné práci, která využívá statistiky získané monitorováním sítě, dosahoval přesnosti přesahující 99 %.

Kapitola 3

Návrh metody tvorby pravidel ze síťového provozu

Jedním z bodů této bakalářské práce je navrhnout způsob, jakým by šlo sledování sítě využít pro tvorbu detekčních pravidel anomálního chování uživatelů na vybraných síťových službách. Je třeba brát na vědomí, že styl komunikace se diametrálně odlišuje v závislosti na cílené službě. Vytvoření univerzální a přitom jednoduchého detekčního pravidla, které by s dostatečnou přesností dokázalo v síťovém provozu identifikovat útočníka a neprojevovalo by se značnou dávkou falešných detekcí, je čirá utopie.

Vytváření pravidel pro jednotlivé služby už lze považovat za uskutečnitelné. Sledováním komunikace mezi důvěryhodnými adresami bychom mohli vytvářet vzory běžné komunikace a sledováním provozu z adres útočníků by naopak šly vytvářet vzory komunikace útočné. Tyto vzory mohou být reprezentovány v podobě statistik odvozených od statistik získaných technologií NetFlow nebo jí podobných. Tento princip je v podstatě univerzální a na různé služby obecně aplikovatelný. Pro člověka je hledání hlubších souvislostí v enormním kvantu statistik takřka nemožné, časově velmi náročné a s nejistým výsledkem. Jedním z cílů této bakalářské práce by tak měl být program, který by celý tento proces zautomatizoval a značně urychlil. Pro návrh předpokládejme nasazení měřicích bodů jen na okrajích sítě.

3.1 Problematika určení důvěryhodných a útočných adres

Nalezení části útočníků nepředstavuje při využití honeypotů zásadní problém. Pokud budeme vycházet z předpokladu, že kdokoliv kdo komunikoval s honeypotem, je považován za útočníka, je možné si jeho adresu zapamatovat. Jeho následné a cílené sledování při komunikaci využívající službu, pro kterou se vytváří detekční pravidlo, poslouží při tvorbě negativních statistik.

Pro strojové učení s učitelem je potřeba mít i vzory správného chování. Získání takových statistik je už obtížné z toho důvodu, že nelze s jistotou určit důvěryhodné adresy. V nekomerčních sítích jako je právě monitorovaná síť organizace CESNET nebo jako jsou sítě ostatních národních výzkumných a vzdělávacích sítí okolních zemí (SANET, PIONEER, aj.), dále označované jen jako důvěryhodné sítě, lze s jistou dávkou optimismu očekávat nižší riziko přítomnosti útočných adres. Proto za komunikaci, která se využije pro tvorbu pozitivních statistik, bude považován provoz mezi sítěmi důvěrnými a sítí monitorovanou. Za další důvěryhodné adresy lze také považovat rozsahy sítí známých technologických organizací jako Google, Apple, Microsoft a jiné.

Byť by mohla nastat situace, že do pozitivních statistik se propašuje chování některých útočníků ukrytých v důvěryhodných sítích, vedlo by to v důsledku pouze k možnému ignorování daného stylu útoku, což je přijatelnější varianta než označování korektní komunikace za útočnou. Další opatření eliminující výskyt takto nevhodných statistik je diskutováno dále.

3.2 Výběr vhodného provozu

Nyní již víme, že budeme sledovat provoz důvěryhodných a útočných adres a vytvářet z něj statistiky. Je dobré brát na vědomí, že ne veškerý provoz těchto adres je přímo pro statistiky vhodný. Pokud totiž část provozu neprochází přes monitorovanou síť, nemohou být statistiky chování úplné a část dat je ztracena. Předpokládejme pro další postup rozdělení IP adres do 5 skupin:

- adresy honeypotů
- adresy monitorované sítě
- adresy důvěryhodných sítí
- adresy útočníků
- adresy ostatní

Pro tyto skupiny platí, že adresy honeypotů mohou být podmnožinou adres monitorované sítě nebo i sítí jiných. Adresy monitorované a adresy důvěryhodných sítí by se naopak neměly překrývat vůbec. Za adresy útočníků budeme předpokládat adresy, které v průběhu monitorování komunikovaly s honeypoty. Následující tabulka 3.1 vyjadřuje všechny kategorie komunikace mezi skupinami adres a níže uvedený popis upřesňuje jejich význam pro tvorbu statistik.

Zdroj → Cíl	Honeypoty	Útočníci	Důvěrné adr.	Monitor. adr.	Ostatní adr.
Honeypoty	E	F	C	C	-[1]
Útočníci	F	G	D	B	G
Důvěrné adr.	C	D	D	A	G
Monitor. adr.	C	B	A	E	G
Ostatní adr.	-[1]	G	G	G	G

Tabulka 3.1: Kategorie komunikací mezi skupinami adres

Kategorie A Provoz mezi sítí monitorovanou a důvěryhodnou je vhodný pro tvorbu pozitivních statistik, neboť veškerý provoz prochází přes hraniční měřící body. Statistiky budou vytvářeny pro monitorovanou i důvěrnou adresu.

Kategorie B Provoz mezi sítí monitorovanou a adresami útočníků je naopak vhodný pro vytváření statistik negativních, protože se opět zachytává veškerý provoz. Zaznamenané chování bude započítáno pouze do statistiky útočníka, aby neovlivnilo případné pozitivní statistiky adresy ze skupiny monitorovaných.

[1]Komunikace mezi ostatními adresami a adresou honeypotu není možná, protože návrh počítá s tím, že adresa ze skupiny ostatních komunikující na honeypot je okamžitě považována za útočníka.

Kategorie C Provoz mezi důvěryhodnými nebo monitorovanými sítěmi a adresami honeypotů vyjadřuje situaci, kdy došlo mezi adresami primárně využívanými pro generování pozitivních statistik k situaci, kdy jsou považovány za útočníky. Takové adresy již nemohou být dále považovány za vhodné a postup korekce jimi vytvořených statistik je popisován dále.

Kategorie D Komunikace mezi těmito adresami představuje potenciálně zajímavý provoz, který je možné sledovat, ale pro tvorbu statistik nemá zásadní význam, jelikož jeho část může procházet mimo monitorovanou síť.

Kategorie E Návrh počítá s monitorováním provozu na rozhraních sledované páteřní sítě, z čehož vyplývá, že provoz mezi monitorovanými adresami by v ideálním případě neměl být vůbec zaznamenán. Jelikož honeypoty obvykle nezahajují komunikaci, lze provoz mezi nimi považovat za vyloučený. Proto budou oba ojedinělé provozy z hlediska tvorby statistik ignorovány.

Kategorie F Obsahově je komunikace mezi honeypotem a útočníkem nezajímavá, neboť nalezené honeypoty byly až na výjimky z kategorie s nízkou mírou interakce.

Kategorie G Ostatní hůře identifikovatelný provoz nespadá do žádné z výše uvedených kategorií. Až na výjimky ani nelze celý sledovat a jeho obsah je diskutabilní, protože může obsahovat korektní komunikaci, ale i komunikaci dosud neodhalených útočníků, kteří se vyhnuli honeypotům. Pro tvorbu statistik je nevhodný.

Z výše uvedených příčin vyplývá, že k tvorbě statistik se využije pouze kategorie A a B. Zvolené řešení má navíc výhodu v tom, že monitorovaná adresa vytvářející pozitivní statistiky může být pod útokem z adresy útočníka, ale negativní chování se neprojeví do její statistiky, neboť se do nich zahrnuje pouze komunikace s důvěrnými adresami.

Statistiky mohou být znečištěny. Za okolností, kdy se normálně vytváří pozitivní statistiky mezi monitorovanou a důvěrnou sítí, může nastat situace, že z důvěrné adresy proběhne spojení s honeypotem. V takové případě je vhodné považovat prefix sítě, do něhož adresa náleží, za potenciálně infikovaný. Jiné IP adresy v rámci stejného prefixu by již neměly být považovány za vhodné pro monitorování. Protože již dříve vytvořené a uchované statistiky potenciálně infikovaných adres mohly být narušeny, budou při dalším zpracování odebrány. Podobná okolnost jako u důvěryhodných adres může hypoteticky nastat i u adres v monitorované síti, ačkoliv řešení je odlišné. Nebude se odebírat celá síť, do které adresa náleží, ale pouze kompromitovaná adresa a to z několika následujících důvodů. Monitorová síť je omezena na počet v ní přítomných adres a odebráním části rozsahu sítě by se tak značně zhoršily schopnosti tvorby statistik. Mnohem významnějším argumentem je skutečnost, že adresy honeypotů jsou převážně z monitorované sítě, a protože uvažujeme měřící body pouze na okrajích, neměl by takový provoz být zachytitelný až na případy, kdy např. útočník využívá podvrhnutou IP adresu.

3.3 Statistiky pro tvorbu pravidel

V tento okamžik je znám typ provozu vybraný pro tvorbu detekčních pravidel. Dále je třeba stanovit, co přesně budou statistiky o provozu obsahovat a jakého budou nabývat rozsahu. Vzhledem k tomu, že tato práce se zaměřuje na monitorování provozu na páteřních sítích,

musí se počítat s nutností včas zpracovat velká kvanta dat. Při návrhu budou zavrhnuty ty metriky, které nejsou vhodné kvůli požadavkům na výpočetní a paměťovou náročnost.

Z pohledu monitorování do zajímavých statistik bezpochyby patří sledování komunikace mezi dvěma adresami. Zpracování takového provozu dokáže zjistit podrobné detaily např. zda při komunikaci měly otevřeno více spojení, kolik mezi sebou přenesly dat apod. Navrhovaná aplikace má ovšem zvládat sledovat a analyzovat komunikaci na páteřní síti, kde jen za dobu 5 minut lze identifikovat více než stovky tisíc unikátních IP adres. Nutno brát na vědomí, že jedna IP adresa může a velmi často komunikuje s více jinými adresami zároveň a pro každou dvojici by bylo nutné samostatně uchovávat a zpracovávat data. Takové analýzy jsou uplatnitelné v rámci lokálních sítí, kde množství přenosů je značně menší než na sítích páteřních. Vhodný kompromis, který byl zvolen pro následnou implementaci, představuje vytvářet pro každou unikátní zaznamenanou IP adresu její obecný statistický profil, který bude pro všechny příchozí a odchozí komunikace společný. Tvorba detekčních pravidel se zaměřuje vždy na jednu vybranou cílenou službu, a tak bude obecný profil rozšířen o specifický statistický profil, který se bude aktualizovat jen v případě, když analyzovaná komunikace náleží cílené službě.

Z hlediska bezpečnosti existuje snaha útočníka detekovat ideálně v průběhu jeho činnosti a to, co nejdříve a s nízkým počtem falešných poplachů. Pro generování statistik byly zvoleny časové intervaly, kdy je komunikace proudově zaznamenávána pro každou adresu po dobu až 5 minut s následným vyhodnocením naměřených statistik. Pokud se daná adresa vyskytne v provozu i po vyhodnocení, je jí vytvořen nový a čistý profil, který bude opět po stanovenou dobu aktualizován a dále zpracováván.

Byť by bylo možné, aby některé metriky byly ovlivněny případně statistikami již dříve vytvořenými, nebude tato možnost uvažována kvůli výpočetním nárokům. Pro každou adresu by to představovalo nutnost ukládat si (alespoň dočasně) historické záznamy, jejich vyhledávání a zpracování.

Pro lepší představu, jaké metriky statistik je třeba sledovat, si pojďme uvést příklad používaného a osvědčeného detekčního pravidla, které se věnuje odhalení adres provádějících na síti horizontální skenování s TCP příznakem SYN. Horizontální skenování je založeno na principu odhalování zařízení v síti se zjištěním dostupnosti vytipované služby. Útočník obvykle zkouší navázat spojení s řadou zařízení a čeká na odpověď. Protože často tato služba neběží nebo na kontaktované adrese žádné zařízení neposlouchá, navázat spojení se mu ve většině případů nepodaří.

```
odchozí toky se SYN příznakem > 200 &&
odchozí toky se SYN příznakem > 20 x odchozích toky s ACK příznakem &&
odchozí toky se SYN příznakem > 5 x příchozích toky s ACK příznakem &&
počet unikátních kontaktovaných adres >= 200 &&
odchozí toky se SYN příznakem > počet všech odchozích toků / 2
```

Obrázek 3.1: Pravidlo detekce horizontálního SYN skenování z modulu *hoststatsnemea*

Pro uvedené pravidlo nebude diskutován význam jednotlivých podmínek, ale spíše to, co vyjadřují z pohledu metrik. Na 1. a 4. řádku si lze všimnout, že se jedná o podmínku prahovou, kde je kontrolováno dosažení určitého detekčního prahu. Na řádcích 2, 3 a 5 se naopak nachází podmínky poměrové, které kontrolují dosažení minimálního poměru mezi metrikami statistiky. Pro účely strojového učení tedy vyplývá, že je vhodné kombinovat statistiky vyjadřující počty výskytů (pro překročení prahů) a poměry mezi sledovanými metrikami.

Při plánování metrik je bezpochyby nutné zohlednit zdroj dat použitý při implementaci návrhu. V tomto případě budou využity hodnoty poskytnuté systémem pro analýzu síťových dat Nemea, který je blíže popsán v kapitole 4.1. Předem je možné říci, že obvyklý formát, kterým jsou předávána data z kolektorů o jednotlivých tocích, označovány jako „collector_flow“ obsahuje položky:

`SRC_IP`	Zdrojová IP adresa	`TIME_FIRST`	Čas prvního paketu
`DST_IP`	Cílová IP adresa	`TIME_LAST`	Čas posledního paketu
`SRC_PORT`	Zdrojový port	`TCP_FLAGS`	Příznaky TCP protokolu
`DST_PORT`	Cílový port	`LINK_BIT_FIELD`	Identifikace linky
`PROTOCOL`	Protokol	`DIR_BIT_FIELD`	Identifikace směru na lince
`PACKETS`	Množství paketů	`TOS`	Typ služeb
`BYTES`	Množství bytů	`TTL`	Zbývající počet skoků

Jaké položky jsou pro monitorování vhodné, mohou napovědět již existující práce, které se zabývaly výběrem vhodných metrik [9, 6]. Zásadní odlišnost těchto prací ale vychází z rozdílného přístupu ke sledování sítě. Na rozdíl od této práce, která využívá agregované monitorovací záznamy, vycházely z detailního snímání provozu na úrovni jednotlivých paketů. Celkově však nabízely řadu zajímavých metrik, které jsou s drobným přizpůsobením aplikovatelné i u agregovaných dat. Pro potřeby strojového učení tak bylo navrhnuto celkem 30 metrik uvedených v příloze B.

Kvůli využití algoritmů pro hledání vzorů je třeba zajistit čistotu dat a dbát, aby samotné metriky jednotlivých statistik byly co nejméně navzájem odvoditelné a na sobě závislé. Takovým příkladem odvoditelnosti může být provázání dvou metrik statistiky o IP adrese, kde jedna vyjadřuje poměr vstupních toků k celkovému počtu a druhá vyjadřuje poměr výstupních toků ke stejnému celkovému počtu. Vzhledem k tomu, že celkový počet toků je součtem vstupních a výstupních toků, bude platit, že součet poměrů je vždy 1, a proto stačí, aby se vyskytovala pouze jedna z metrik.

Kvalitu vytvářených statistik může i významně ovlivnit chování útočníků, kteří se během útoku zároveň snaží provádět dříve zmíněné skenování sítě. V případech, kdy se spojení nenaváže, je obsah přenášených užitečných dat v paketu (označovaných anglicky „payload“) mizivý. Aby se proto význam skenování v rámci experimentů co nejvíce minimalizoval, nebudou se toky, u nichž je průměrný payload na jeden paket menší než 5 bytů (nebo jiná nakonfigurovaná hodnota), do statistik zaznamenávat.

3.4 Charakteristika činnosti programu

Výsledky snažení by měl představovat experimentální program, který bude schopen dlouhodobě analyzovat provoz na vytipovanou síťovou službu a odhalí vzor anomálních chování útočníků z dat dostupných měřením na páteřní síti. Vytipovanou službu v datech identifikuje na základě pravidel o používaných zdrojových/cílových portech a protokolech. Své získané schopnosti nadále musí umět aplikovat na ostrý provoz, v němž bude schopen včas odhalit útočné chování. Jeho činnost se tudíž musí skládat ze dvou samostatných režimů, učení a detekce.

Režim učení bude podle popisu zmíněného v této kapitole filtrovat jen část provozu vhodnou pro učení. Samotný provoz budou představovat již dříve zachycená data a učení tak nebude probíhat z živého provozu. Důvodem je, že proces učení vyžaduje obvykle zpracování záznamů z několikadenního provozu, aby mohl pochytit, co nejvíce situací. Učení z již

zachyceného provozu je rychlejší, není závislé na případných výpadcích dat z jiných zdrojů a dovoluje rychleji experimentovat s konfigurací. Je také třeba, aby se zabránilo přílišnému přeučení na jeden styl komunikace, a proto pro každou IP adresu umožní uchovat jen omezený počet záznamů se vzory chování. Samotné učení se bude skládat ze 3 částí: od sběru statistik sledované služby, odfiltrování znehodnocených nebo nadbytečných statistických záznamů až po učení klasifikátoru, který představuje výsledek této fáze.

Režim detekční částečně provádí podobný postup, jako byl použit při učení s tím rozdílem, že již nebude filtrovat toky podle kategorií určených pro učení, ale bude přijímat veškerý provoz na cílenou službu a generovat statistiky. Statistiky se následně vyhodnotí klasifikátorem, který na základě získaných znalostí rozhodne, jestli se jedná o provoz odpovídající útočníkům. Nasazený program by měl zvládat vyhodnotit ostrý provoz na monitorované páteřní síti v reálném čase. Existují i požadavky na zpětné zpracování již zachyceného provozu, a proto by měl podporovat i vyhodnocování dříve zachycených dat. Příklad takového použití představuje analýza historického provozu u sítí, kde se vyskytovaly problémy a je třeba dohledat zpětně útočníka. Výstup detekčního režimu představuje seznam identifikovaných útočníků s časem jejich útoku.

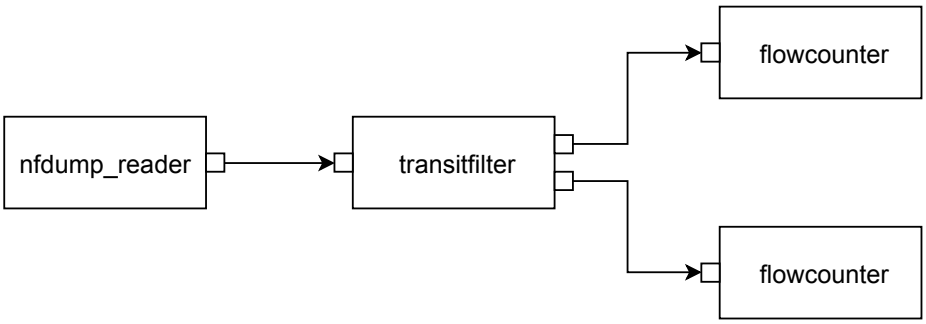
Kapitola 4

Implementace

Návrh činnosti a stěžejních bodů tvorby detekčních pravidel je stanoven. Další fázi logicky představuje implementace. Aby bylo možné se co nejvíce věnovat samotnému algoritmu, bylo pro vzniklý program zvoleno zastřešující řešení v podobě systému pro analýzu síťových dat Nemea[11], který je vyvíjen organizací CESNET.

4.1 Framework Nemea

Snahou frameworku je vytvořit podpůrné prostředky pro snadnou tvorbu a nasazení navzájem komunikujících samostatných programů pro analýzu zachycených dat (NetFlow, IPFIX) z monitorované sítě v reálném čase. Vzniklé programy jsou označovány jako moduly, které mohou být vzájemně propojovány do větších celků předávajících si data. Každý modul disponuje určitým počtem vstupních a/nebo výstupních rozhraní, s jejichž pomocí komunikuje s moduly ostatními. Samostatnost modulům umožňuje být přidán nebo odebrán za běhu, aniž by byly ovlivněny jiné na jejich činnosti nezávisející moduly. Pokud by modul havaroval a byl nahrazen, propojení mezi moduly je schopnost se automaticky obnovit a pokračovat ve své činnosti.



Obrázek 4.1: Ukázka zapojení modulů Nemea

Obrázek 4.1 ukazuje příklad jednoduchého zapojení modulů, kde se modul označený jako `nfdump_reader` věnuje přehrávání zachyceného provozu ze souboru, který předá modulu `transitfilter`, jehož náplní je rozdělit provoz na takový, jenž přes monitorovanou síť pouze prochází a jenž v síti začíná nebo končí. Odfiltrovaná data jsou předána primitivním modulům `flow_couter`, které spočítají celkový počet toků, paketů a bytů v předaných záznamech.

Hlavní komponenty

Nemea, jakožto framework, nabízí prostředky pro tvorbu softwarových modulů. Mezi základní komponenty patří knihovna pro komunikaci mezi moduly, šablonový systém přenášených dat a často používané algoritmy.

Knihovna libtrap představuje implementaci komunikační vrstvy TRAP (TRaffic Analysis Platform) používanou pro výměnu dat mezi moduly. Použití sdílené knihovny odstíní vývojáře modulu od nutnosti zabývat se vytvářením soketů a navazováním připojení, čímž se může více orientovat na samotnou aplikační logiku. Knihovní funkce umožňují nakonfigurovat způsob realizace komunikace. Modul si může určit, jestli požaduje rozhraní blokující (modul bude čekat, dokud se jiný modul nepřipojí) nebo neblokující. Spojení modulů může využívat UNIX soketů, které efektivně zprostředkují výměnu dat mezi moduly v rámci jednoho fyzického zařízení, ale i TCP/IP soketů, kdy propojené moduly mohou běžet na rozdílných výpočetních strojích, což zejména u výpočetně náročných modulů dovoluje rozložit zátěž. Při každém zavolání příjmu dat z TRAPu je vrácen právě jeden záznam odpovídající jednomu toku. Obdobně probíhá odesílání.

Formát UniRec reprezentuje způsob formátování předávaných zpráv zasílaných mezi Nemea moduly přes komunikační vrstvu TRAP. Protože obnos přenášených dat může být enormní a je nutné, aby formát přenášených záznamů byl dostatečně optimální a flexibilní pro potřeby všech druhů modulů, využívá se princip šablon. Šablona reprezentuje význam a umístění dat v rámci přenášeného záznamu, čímž usnadňuje přístup k individuálním položkám. Pro korektní komunikaci modulů je nezbytné, aby používaly na společném rozhraní stejné šablony. Nedílnou součástí je i pro všechny moduly společná reprezentace formátu IP adres a časových značek.

Popis vybraných modulů

Systém pro analýzu síťových dat Nemea disponuje celou řadou již implementovaných modulů, které můžeme obecně rozdělit do kategorií se zaměřením na:

- získávání, reprodukci nebo generování síťových dat
- předzpracování síťových dat (filtrace, deduplikace, apod.)
- analýzu anomálií a detekci útoků
- postprodukci (agregace výsledků, apod.)
- zachytávání, ukládání nebo hlášením výsledků či jiných dat

Propojené moduly mohou být použity i pro zpracování dříve zaznamenaného síťového provozu, neboť existují moduly, které se věnují přehrávání dat ze souborů. Následujících pár uvedených modulů představuje pouze část z dostupné škály a byly vybrány s ohledem na využití v této bakalářské práci.

`nfdump_reader` Modul umožňuje číst soubory ve formátu Nfdump se zachycenými NetFlow záznamy a přeposílat je v UniRec formátu na výstupní rozhraní. Ostatní připojené moduly tak mohou dostávat již dříve zachycená data a nejsou např. při testování závislá na datech živých.

`trapdump` & `trapreplay` Představují dvojici navzájem se doplňujících modulů. Zatímco první modul dokáže uchovávat libovolné UniRec zprávy ze vstupního rozhraní do souboru. Druhý takové soubor umí číst a přehrát je nezměněné na výstupní rozhraní.

`merger` Jeden ze zástupců modulů z kategorie předzpracování síťových dat umí sloučit síťový provoz z více vstupních zdrojů na jeden výstupní. Ostatní moduly nemusí nutně podporovat více vstupních rozhraní při příjmu síťových dat a stačí, když je modul před jejich vstup zapojen.

`hoststatsnemea` Jedná se o detekční modul vytvářející si statistiky o individuálních adresách v síťovém provozu. Ve vypočítaných statistikách hledá útočné chování za pomoci množiny jednoduchý pravidel. Informace o detekovaných adresách poskytuje modulům pro hlášení výsledků.

4.2 Implementace modulu

Návrh aplikace uvedený v kapitole 3.4 představuje základ modulu `automatic_rules` pro systém Nemea. Vzhledem k tomu, že většina modulů v tomto systému pro analýzu síťových dat a další podpůrný software jsou napsány v jazyce C/C++ není ani tento modul výjimkou a převážná část kódu je v jazyce C++.

Navzdory skutečnosti, že návrh se sestává ze dvou samostatných režimů, kde jeden se věnuje učení a druhý detekci, představuje implementovaný modul jeden celek. Hlavním důvodem, proč nedošlo k rozdělení na dva separátní moduly pro jednotlivé režimy, je skutečnost, že činnosti obou se značně překrývají. Hlavní část, kde se přijímají vstupní data a vytváří statistiky, je až na drobné detaily společná.

Veškerý vstup dat obstarávají knihovní funkce libtrap v kombinaci s formátem zpráv UniRec. Na jediném vstupu modulu je očekávána UniRec šablona <*COLLECTOR_FLOW*>. Přenášené záznamy, které této šabloně odpovídají, v sobě zahrnují veškeré standardní položky popisující sítový tok. Část zdrojových kódu a logiky programu vychází i z implementace modulu `hoststatsnemea`, kde převzaté a upravené části jsou dřívější výhradní autorskou prací autora této bakalářské práce.

Komponenty implementovaného modulu

Při studování návrhu aplikace bylo objeveno několik samostatných částí, jež se podařilo oddělit a implementovat do logicky souvisejících tříd. Následující přehled se stručně věnuje vybraným třídám, u kterých je známo, že mají významný vliv na činnost modulu.

Třída `Ip_filter`, jak název napovídá, je základem pro filtrování IP adres. Při konfiguraci jsou ji poskytnuty IP adresy celých sítí nebo jednotlivých zařízení. Její činnost je vskutku prostá, neboť zpracovává pouze dotazy na přítomnost adresy. Protože bylo nezbytné, aby šlo přidávat a odebírat nové adresy i za běhu, třída tyto funkce v rámci implementace podporuje.

Třída `Flow_filter` je nezbytnou součástí pro režim učení, kdy je třeba ze vstupních dat vybrat toky pro pozitivní a negativní statistiky. Při inicializaci jsou jí poskytnuty informace o rozsazích adres honeypotů a také monitorovaných a důvěrných sítích. Za použití několika instancí třídy `Ip_filter` je schopna pro UniRec záznam stanovit jednu z kategorií toku tak, jak bylo určeno v kapitole 3.2. Zároveň při zaznamenání provozu na libovolnou adresu honeypotu identifikuje útočníka a provede nevyhnutelné kroky, aby jej dokázala při dalším výskytu správně zařadit.

Skupina třídy souhrnně označovaných jako `unirec_filter` slouží pro sestavení derivačního stromu pro vyhodnocení filtračních podmínek nad vybranými UniRec položkami. Umožňuje tím za běhu aplikace z uživatelem definovaného výrazu, sestavit strom, který určí, jestli záznam toku odpovídá uživatelem stanoveným podmínkám. Cílem této třídy je ve vstupních UniRec záznamech modulu identifikovat provoz služby, pro kterou se vytváří detekční pravidla nebo se analyzuje. Část kódu je při překladu generována za pomocí dvojice nástrojů *flex* (pro lexikální analýzu) a *bison* (pro syntaktickou analýzu).

Třída `Host_profile` vychází z implementace stejnojmenné třídy z `hoststatsnemea`, reprezentuje ústřední část modulu pro oba režimy a jejím úkolem je pro data o tocích vytvářet statistiky. Každý tok určený na zpracování vede k aktualizaci statistického záznamu zvlášť pro zdrojovou a cílovou IP adresu. Pro záznamy, u kterých platí, že existují příliš dlouhou dobu nebo po stanovený čas nebyly aktualizovány, umožňuje provést jejich vyhodnocení a odebrání. Adresám ze zpracovávaných toků, kterým záznam dosud nebyl vytvořen nebo byl dříve odebrán, vytvoří nový prázdný záznam. Třída navíc pomocí svého napojení na `unirec_filter` částečně vymezuje tvorbu statistik jen na cílenou službu a při učení dovoluje aplikovat i instanci `Flow_filteru` k omezení sledovaných toků.

Třídy odvozené z `Record_handler` se věnují zpracování statistických záznamů, které byly třídou `Host_profile` označeny jako ukončené a určené na vyhodnocení. Odvozené třídy pro režim učení ukládají statistiky do souboru pro pozdější zpracování. Během detekčního režimu se používá odvozená třída `C50_wrapper` reprezentující stěžejní prvek vyhodnocení nasbíraných statistik. Tato třída zaobaluje algoritmus v jazyce C dodávaný k vyhodnocení rozhodovacích stromů vzniklých při učení klasifikátoru *C5.0*. Problém tohoto algoritmu a s ním souvisejících funkcí představuje očekávání, že data pro klasifikaci budou dostupná v souboru a analýza všech statistik proběhne jednorázově, což je v rozporu s návrhem modulu, který počítá s dlouhodobým během a okamžitým vyhodnocením. Vzniklá třída tuto okolnost řeší vytvořením dočasného paměťového bufferu, na který se naváže popisovač souborů. Při každém vyhodnocení statistického záznamu se do bufferu dočasně uloží jeho textová podoba. Na základě následné klasifikace algoritmus klasifikátoru určí, zda se jedná o útočný provoz a určí míru jistoty rozhodnutí.

Zpracování dat

Data na vstupu jsou činností modulu transformována do podoby statistik. Při pohledu na jejich podobu navrhnutu v příloze B je patrné, že zejména poměrové metriky představují hodnoty vypočítané až pro finální zpracování záznamu. Z této příčiny jsou prve vytvářeny jiné statistiky, které např. zvlášť uchovávají počet vstupních a výstupních toků a až při finalizaci záznamu se z nich vypočítá poměr počtu vstupních toků k celkové počtu toků. Pro každou IP adresu, která je zpracována třídou `Host_profile`, jsou vytvořeny dva provázané statistické profily, obecný a specifický. Obecný profil se aktualizuje z veškeré zachycené komunikace, ale k aktualizaci specifického profilu dojde pouze v případech, kdy toky odpovídají službě, pro kterou se vytváří detekční pravidlo. Kombinací obou profilů je možné dopočítat metriky pro klasifikaci.

Záznamy nemohou být aktualizovány věčně a je třeba je včas vyhodnotit. Na základě nastavení aktivního a neaktivního časovače jsou určeny ke zpracování a odebírány. Neaktivní časovač je počítaný od posledního přijatého toku a expiruje v případech, kdy ani po specifikované době nepřišel další tok. S každým novým příslušným tokem se neaktivní časovač nuluje. Na druhou stranu se aktivní časovač počítá od prvního příchozího toku a sleduje, jak dlouho aktualizace záznamu trvá. Pokud trvá déle než stanovenou dobu, tak

bez ohledu na stále přicházející nové toky je záznam označen za ukončený.

Ve statistikách pro klasifikaci se objevuje i informace o tzv. payload bytech. Na vstupu sice informace o počtu bytů, které tok přenesl, je, ale jsou do ní zahrnuty i velikosti IP hlaviček a hlaviček transportního protokolu obsažené v paketech. Požadovanou hodnotu ovšem lze přibližně dopočítat jejich odečtením. Přestože pakety mohou být fragmentované a transportní hlavička se tudíž nemusí vyskytovat v každém paketu, předpokládá výpočet nehorší možnou variantu, kdy se v každém paketu vyskytují hlavičky obě. Fragmentace paketů nastává zejména při přenášení velkého objemu dat a v takovém případě se dá určitá odchylka tolerovat, neboť transportní hlavičky nejsou tak objemné.

4.2.1 Režimy

Hlavní část společnou režimu učení a detekce představuje nekonečný cyklu, který vyčítá data z rozhraní TRAP a předává je na zpracování. Pro oba režimy platí, že dokáží vytvářet nebo analyzovat provoz jen pro jednu službu zároveň. Následující části objasňují, jak jsou jednotlivé režimy implementovány.

Režim učení

Snahou při učení je analyzovat již dříve zachycený provoz, vytvořit z něj statistiky a odhalit v něm vzor chování. Jelikož modul i při učení musí využívat informace o čase pro aktivní a neaktivní časovače, aby věděl, kdy má statistické záznamy vyhodnotit, odvozuje čas z koncových časových značek přijímaných toků. Při tomto režimu se užívá již dříve zachycený provoz, ale stejně jako pro provoz živý i pro něj platí, že časové značky nejdou plynule za sebou, ale jsou proházené tak, jak je vygeneroval exportér. Zkouškami bylo zjištěno, že výpočet aktuálního času z maximální naměřené koncové značky, není úplně vhodný kvůli možným časovým skokům. Jako optimální se jevilo využití průměrných značek posledních 10 toků tak, aby se skoky alespoň částečně vyhladily.

Každých (ve výchozím nastavení) 10 sekund se pozdrží příjem nových dat z TRAPu a na základě časovačů se ověří, zda jsou statistické záznamy stále platné. Když platnost záznamů vypršela, jsou vyhodnoceny a odebrány. Jakmile jsou všechny zkontrolovány, je příjem obnoven.

Po ukončení vstupu modulu, kdy už je ze sledovaného provozu vytvořeno dostatečné množství statistik, se provede očištění dat. Jsou případy respektive záznamy, u nichž je potřeba provést jejich redukci nebo je zcela odebrat. Kandidáty na odstranění jsou všechny adresy, u kterých v průběhu generování statistik bylo zjištěno potenciálně nekorektní chování. Aby se navíc zabránilo přeučení klasifikátoru, kdy jedna adresa může provádět útok po dlouhý časový úsek a celkový počet záznamů by mohl nepříznivě ovlivnit proces učení, pro každou IP adresu se ponechá maximálně 40 záznamů se součtem příchozích a odchozích toků pod množství 50 a maximálně dalších 40 záznamů nad tento limit. Takové rozdělení bylo zvoleno s ohledem na skutečnost, že fáze útoku mohou mít odlišnou intenzitu.

Nad očištěnými záznamy se provede učení klasifikátoru voláním externího programu, který vytvoří rozhodovací strom. Používá se konfigurace *C5.0* s podporou automatického výběru vhodných parametrů, adaptivním boostingem a ohodnocením chybné klasifikace, která falešně pozitivní detekce považuje za závažnější než falešně negativní. Modul s klasifikátorem vygenerují celou řadu souborů, kde obsah je určen dle přípony uvedené v tabulce 4.1. Pro spuštění detekce anomálií jsou už nutné pouze soubory `.names`, `.costs`, `.tree` a `.module_config`.

`.names`	Popis datových typů metrik
`.data`	Statistiky použité pro učení
`.costs`	Ceny chybné klasifikace
`.tree`	Popis rozhodovacího stromu
`.module_config`	Konfigurace modulu v době učení
`.c50_output`	Výpis učení klasifikátoru
...	...

Tabulka 4.1: Význam generovaných souborů dle přípony

Režim detekce

U detekčního režimu je situace při tvorbě statistik velmi podobná. Rozdíl je pouze ve způsobu jejich finálního zpracování, kdy modul záznamy nikde neskladuje, ale používá dříve sestavený rozhodovací strom na určení, zda se jedná o útočný provoz. Pro detekci jsou zvoleny jen statistické záznamy, u který celkový počet zaznamenaných toků překračuje hranici 50. Když si je klasifikátor při určení útočníka na více než 95% jistý správností rozhodnutí, je útočník poznačen do logovacího souboru i s časem začátku a konce pozorování útoku. S tím, že jsou jednotlivé statistiky vždy vyhodnoceny nejpozději po 5 minutách od jejich vzniku (aktivním časovačem), souvisí jev, kdy dlouhotrvající útoky jsou nahlášeny vícekrát. V budoucnu bude možné, aby se záznamy s informací o detekci odesílaly přes TRAP rozhraní jiným modulům, které je např. agregují.

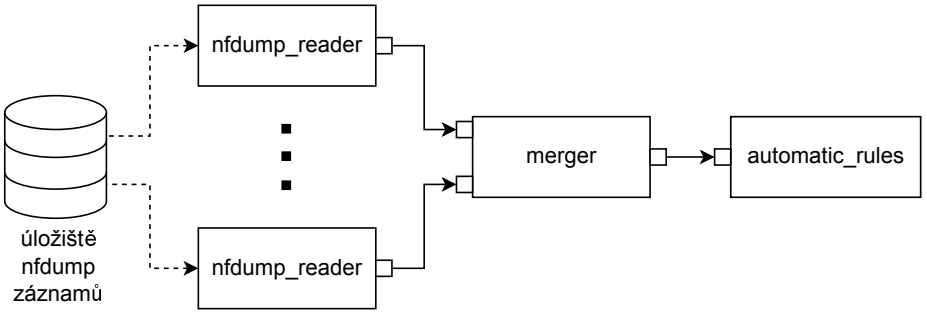
Detekce může pracovat s živým nebo přehrávaným provozem. Při analýze dříve zachyceného provozu se přibližný čas pro vnitřní potřeby určuje z časových značek toků stejně, jak tomu bylo při učení. Pozastavení přijmu nových dat při vyhodnocování záznamů je též totožné.

Při zpracování živého provozu nelze provádět pozastavení přísunu nových dat, aniž by došlo k jejich ztrátě. Z této příčiny se používají dvě spolupracující vlákna, kde jedno má na starost čistě příjem dat a aktualizaci statistických záznamů a druhé záznamy při jejich ukončení vyhodnotí, provede detekci a odebere. Vnitřní čas modulu se navíc určuje z reálného systémového času, čímž nehrozí žádné časové skoky.

Kapitola 5

Testování a vyhodnocení výsledků

Tato kapitola se věnuje testování modulu a jeho schopnosti detekovat útoky na vybrané služby. Jako datová sada pro režim učení byly zvoleny 4 celé dny, které měl implementovaný modul `automatic_rules` zvlášť pro každou službu zpracovat v režimu učení. Pro lepší představu o množství NetFlow záznamů pro zpracování, lze uvést, že za jediný pracovní den se zachytí a zpracuje více jak 7 miliard toků. Zaznamenaná data z kolektoru se uchovávají pro každou monitorovanou linku zvlášť a modul `nfdump_reader`, sloužící pro jejich přehrávání, dokáže zároveň přehrávat pouze data z jedné linky. Pro učení však bylo třeba znát veškerý provoz. Řešením je konfigurace modulů, kdy pro každou linku je vyčleněn jeden `nfdump_reader` a jejich výstup je nasměrován do modulu `merger`, který provoz sjednotí s ohledem na časové značky do jediného výstupního rozhraní připojeného na vstup modulu `automatic_rules`.



Obrázek 5.1: Návrh zapojení pro testování modulu

Při nasazení se ovšem ukázalo, že `merger` neoptimálně pracuje při sjednocování velkého množství linek, kdy pro každé vstupní rozhraní si vytváří vlákno, které s ostatními soupeří o výstupní rozhraní. Pro vyřešení situace byl použit výpočetní stroj s dostatečným počtem výpočetních jader, ale i na něm samotný proces sjednocování záznamů trval nepříjemně dlouhou dobu. Pro rychlé a efektivní experimentování se stejnocenný provoz tudíž zaznamenal prve pomocí modulu `trapdump` a při experimentech pouze přehrával modulem `trapreplay`. Nekomprimovaná sjednocená data za 4 dny zabírala více jak 2TB místa.

Pro účely testování modulu v režimu detekce byl zvolen časový úsek 12 hodin. Zvolená testovací data pocházela z období vzdáleného více jak 2 týdny od dat použitých pro učení. Pro každý sledovaný provoz testované služby bude uvedena stručná charakteristika zneužití, množství záznamů použitých pro učení a výsledky detekcí.

5.1 Útoky na službu SSH

SSH neboli Secure Shell je protokol používaný pro zabezpečenou komunikaci skrze počítačovou síť a využívá se pro správu a řízení vzdálených počítačů. Umožňuje uživatelům spouštět příkazy pomocí příkazové řádky bez fyzické přítomnosti poblíž počítače. Pro spojení je vyžadována autentizace uživatele, která je velmi často prováděna pomocí kombinace uživatelského jména a hesla. Pro útočníky je získání kontroly nad strojem cenné, a proto se za pomocí slovníkových útoků, kdy zkouší nejčastější kombinace uživatelských jmen a hesel, snaží získat přístup. Forma opakovaného monotónního hádání představuje poměrně charakteristické chování z pohledu monitorování sítě.

Režim učení	Počet získaných záznamů	34863
	Počet pozitivních záznamů	793 (2,27%)
Režim detekce	Počet detekcí celkem	54926
	Počet unikátních adres	4801

Tabulka 5.1: Vyhodnocení činnosti modulu pro službu SSH

Při analýze výsledků se ukázalo, že detekce probíhala vcelku úspěšně a podezřelý provoz byl nahlášen. Problém ovšem představovala skutečnost, že mezi detekovanými útočníky bylo i mnoho obětí jejich útoku. Příčinou je skutečnost, že při hádání hesel odpovídají oběti na žádosti útočníků se stejnou monotónností, čímž jsou také detekovány. Při ruční kontrole více než 30 výsledků byl pouze jeden z nich falešně pozitivní. Pro budoucí využití je třeba zapracovat na rozlišování útočníků a obětí.

5.2 Útoky na službu RDP

RDP (Remote Desktop Protocol) umožňuje uživateli připojit se k desktopovému prostředí vzdáleného počítače a ovládat jej. S určitým nadhledem ji lze označit za obdobu služby SSH. Připojení uživatele je opět provázeno autentizací a způsob prolamování hesel je tedy obdobný jako v předchozím případu.

Režim učení	Počet získaných záznamů	111448
	Počet pozitivních záznamů	3490 (2,44%)
Režim detekce	Počet detekcí celkem	11156
	Počet unikátních adres	1374

Tabulka 5.2: Vyhodnocení činnosti modulu pro službu RDP

Jelikož se jedná o podobný styl útoku jako na službu SSH jsou výsledky relativně totožné. U ručně kontrolovaných detekcí nebyly nalezeny žádné chybné záznamy. Z přiložené tabulky 5.2 a její obdoby u služby SSH lze vidět, že při učení počet negativních záznamů značně dominoval. Získávání pozitivních statistik naráží na skutečnost, že ačkoliv bylo důvěryhodných adres dostatek, neprobíhalo mezi nimi a monitorovanou sítí dostatečné množství provozu na tyto službu. Příčinou je pravděpodobně i to, že služby jsou užívány hlavně v rámci sítě samotné a útočníků, kteří je chtějí zneužít je mnohem více.

5.3 Útoky využívající DNS

Není mnoho uživatelů, kteří si pamatují IP adresy jimi navštěvovaných serverů. Proto vznikla služba DNS, jejíž náplní je i mapování doménových jmen na IP adresy a zpět. Tím, že systém je používán i řadou aplikačních protokolů, představuje jeden z pilířů fungování Internetu tak, jak jej známe. Služba DNS používá nejen záznamy pro překlad domén a IP adres, ale i celou škálu jiných specifických záznamů. Útočnicí dokáží DNS servery zneužít např. pro posílení svých DoS/DDoS útoků[1]. Cílem je, aby server na malý požadavek reagoval několikanásobně větší odpovědí. Útočníci podvrhnou svoji IP adresu tak, aby se tvářila jako adresa zařízení, na které chtějí útočit, a pokládáním enormního množství dotazů na DNS server způsobí, že zesílený útok je směrován na oběť.

Režim učení	Počet získaných záznamů	92303
	Počet pozitivních záznamů	27898 (30,22%)
Režim detekce	Počet detekcí celkem	150261
	Počet unikátních adres	32302

Tabulka 5.3: Vyhodnocení činnosti modulu pro službu DNS

Křížovou validací klasifikátoru se zjistilo, že míra chybovosti přesahuje 10,3% a vytvořený rozhodovací strom je natolik košatý, že i jeho použití pro vyhodnocení je nadprůměrně náročné. Při letmém pohledu na data, která byla použita pro učení, bylo vyzkoumáno, že u identifikovaných útočníků sice existovaly specifické vzory chování např. při skenování sítě, ale u řady dalších se chování velmi blížilo chování korektnímu. Jelikož existovaly velmi podobné negativní a pozitivní statistiky, klasifikátorem vytvořený rozhodovací strom je nepoužitelný. Při ruční kontrole se potvrdilo, že většina z ověřovaných záznamů jsou falešně pozitivní detekce.

Z podstaty fungování modulu nelze vzory chování pro DoS/DDoS ani zachytit, neboť útočníci, kteří prochází síť a hledají kompromitovatelné DNS servery, je cíleně zneužijí až s podvrženou adresou. Pro zesílené útoku je podobně zneužitelná služba NTP, která se používá pro synchronizaci času mezi počítači, čímž zajišťuje, aby používaly stejný a přesný čas. Zde klasifikátor dosahoval při validaci chybovosti přesahující 27,4 %. Ani zde tak nebyly výsledky použitelné.

5.4 Zhodnocení testování

Z výsledků testování je patrné, že modul úspěšně dokázal odvodit detekční pravidla pro provoz služeb, kdy útočník hádá hesla tj. SSH a RDP. Zřejmým kamenem úrazu u těchto služeb je ale problémové získávání pozitivních statistik. Byť byla chybovost klasifikátoru při křížové validaci u obou velmi nízká, hrozí, že získané výsledky v režimu detekce mohly být nepříznivě ovlivněny nadměrným množstvím negativních statistik. Pro další vývoj modulu je nezbytné se věnovat vylepšením v této oblasti.

Z poměru počtu detekcí útoků připadajících na počet unikátních IP adres lze usuzovat, že značná část z útoků probíhala delší dobu. Vzhledem k relativně krátké době, po které jsou statistické záznamy vyhodnoceny, je to pochopitelné.

[1] DoS/DDoS je technika útoku, při niž dochází k zahlcení internetové služby velkým množstvím požadavků s cílem službu odstavit z provozu.

Kapitola 6

Závěr

Podstatou práce bylo poskytnout pohled do problematiky monitorování vysokorychlostních sítí, nalezení vzorů chování pro vybrané síťové služby a jejich využití pro identifikaci anomálního chování uživatelů.

V teoretickém úvodu byly představeny základní principy technik pro monitorování provozu na sítích. Jelikož pro identifikaci vzorů chování byl zvolen pasivní přístup, kdy je neinvazivně sledována procházející komunikace, byla podrobně představena technologie monitorování sítí NetFlow. Nalezení útočníků a využití jejich agresivního chování v síťovém provozu bylo pro určení špatných vzorů nezbytné, a proto byly k jejich identifikaci využity bezpečnostní prvky souhrnně označované jako honeypoty. Aby nebylo nutné je v rámci práce nasazovat, byl popsán použitý postup pro jejich odhalení v rámci monitorované sítě u jejich provozovatelů. Automatizované odhalení vzorů chování vyžaduje nasazení metod strojového učení, kterým se věnovala poslední část úvodní sekce.

V dalších částech práce směřovala na návrh metody tvorby detekčních pravidel ze síťového provozu. Pro vznik popisů chování vhodných pro strojové učení byl uveden princip získávání pozitivních a negativních vzorů, kdy negativní vzory se určovaly z provozu adres kontaktujících honeypoty. Vzory pozitivní vznikaly na základě komunikace mezi monitorovanou sítí a sítěmi důvěrnými. Určit podobu chování pro cílenou síťovou službu z veškerého provozu nebylo vhodné, a tudíž se návrh zabýval i výběrem příhodné podmnožiny. Reprezentace vzorů chování používá formu v podobě statistik, kde výběr metrik byl klíčovou součástí návrhu. Bylo poukázáno na metriky, které nebylo možné využít nebo představovaly příliš výpočetně náročné postupy nepoužitelné pro zpracování dat z vysokorychlostních sítí. Použitých 30 metrik bylo pečlivě vybráno, aby výstižně charakterizovaly obecné klíčové znaky provozu síťových služeb.

Na základě vzniklého návrhu byl implementován nástroj v podobě modulu do systému pro analýzu síťových dat Nemea. Ke tvorbě pravidel a testování detekčních schopností modulu byla využita data ze sítě organizace CESNET. Prvotní fáze učení se vždy specializovala na konkrétní vybranou síťovou službu, pro kterou se vytvářely vzory běžného a útočného chování. Na konci fáze učení modul použil nástroj C5.0 pro strojové učení, aby získal rozhodovací strom detekující anomální chování nad danou službou. V rámci experimentů byly následně nad reálnými daty testovány detekční schopnosti modulu. Při zaměření na útoky na služby, které využívaly metodu hádání hesel, se na základě analýzy výsledků potvrdila funkčnost modulu. Ať už byl detekován útočník nebo jeho oběť, většina ručně kontrolovaných záznamů odpovídala charakteristice útoku. U útoku, které naopak cílí na zneužití služeb pro posílení svých útoků, se navržená metoda neosvědčila, neboť nedokáže požadované vzory zachytit.

Další rozvoj projektu by měl směřovat na vyvážení poměru pozitivních a negativních vzorů použitých pro potřeby učení, čímž se zvýší věrohodnost generovaných detekcí. Bez rozvoje v této oblasti nelze modul seriózně využívat. Pro posílení detekčních schopností by šlo také uvažovat o rozšíření metrik a otestování jiných nástrojů pro strojové učení.

Literatura

[1] Architektura [Warden]. [online]. 2014 [cit. 19-05-2015].
URL https://wardenw.cesnet.cz/cs/architecture

[2] Data Mining Tools See5 and C5.0. [online]. March 2013 [cit. 19-05-2015].
URL https://www.rulequest.com/see5-info.html

[3] Historie národní sítě pro vědu, výzkum a vzdělávání. [online]. 2013 [cit. 19-05-2015].
URL http://www.cesnet.cz/sdruzeni/dokumenty/historie-narodni-site-pro-vedu-vyzkum-a-vzdelavani/

[4] Is See5/C5.0 Better Than C4.5? [online]. Únor 2012 [cit. 19-05-2015].
URL http://rulequest.com/see5-comparison.html

[5] NetFlow — Wikipedia, The Free Encyclopedia. [online]. 2015 [cit. 19-05-2015].
URL http://en.wikipedia.org/wiki/NetFlow

[6] Bujlow, T.; Riaz, T.; Pedersen, J.: A method for classification of network traffic based on C5.0 Machine Learning Algorithm. In *Computing, Networking and Communications (ICNC), 2012 International Conference on*, Leden 2012, ISBN 978-1-4673-0008-7, s. 237–241.

[7] Claise, B.: Cisco Systems NetFlow Services Export Version 9. RFC 3954, Říjen 2004.
URL https://tools.ietf.org/html/rfc3954

[8] Claise, B.; Trammell, B.; Aitken, P.: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. RFC 7011, Září 2013.
URL https://tools.ietf.org/html/rfc7011

[9] Homoliak, I.: Metriky pro detekci útoků v síťovém provozu. Diplomová práce, Brno, FIT VUT v Brně, 2012.

[10] Petr Matoušek: *Síťové aplikace a jejich arichtektura*. Brno: VUTIUM, první vydání, 2014, ISBN 9788021437661.

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

[12] Witten, I.; Frank, E.; Hall, M. A.: *Data mining*. Amsterdam: Morgan Kaufmann, třetí vydání, 2011, ISBN 978-0-12-374856-0.

[13] Zbořil, F.: *Základy umělé inteligence*. Brno: FIT VUT, 2012, studijní opora.

Příloha A

Obsah DVD

V kořenovém adresáři DVD se nachází následující adresáře a soubory:

`doc/`	Adresář s obsahem zdrojových kódů textové části bakalářské práce.
`src/`	Adresář s obsahem zdrojových kódů implementovaného modulu `automatic_rules` s návodem na jeho překlad a spuštění.
`examples/`	Adresář s příklady vytvořených detekčních pravidel.
`BP_Hutak.pdf`	Textová část bakalářské práce.
`README.txt`	Popis obsahu DVD disku.

Příloha B

Metriky

in_avg_flow_dur	Průměrné doba trvání příchozího toku [sekundy]
out_avg_flow_dur	Průměrná doba trvání odchozího toku [sekundy]
total_flows	Celkový počet toků
in_to_total_flow_ratio	Poměr mezi počtem příchozích a všech toků
in_flow_per_time	Průměrný počet příchozích toků za sekundu
out_flow_per_time	Průměrný počet odchozích toků za sekundu
in_to_total_pkts_ratio	Poměr mezi počtem příchozích a všech paketů
in_to_total_bytes_ratio	Poměr mezi počtem příchozích a všech payload bytů
avg_pkts_per_in_flow	Průměrný počet paketů na příchozí tok
avg_pkts_per_out_flow	Průměrný počet paketů na odchozí tok
avg_bytes_per_in_flow	Průměrný počet payload bytů na příchozí tok
avg_bytes_per_out_flow	Průměrný počet payload bytů na odchozí tok
in_100B_to_total_inflows	Poměr mezi příchozími toky s průměrným počtem payload bytů na paket $\leq$ 100B a celkovým počtem příchozích toků
out_100B_to_total_outflows	Poměr mezi odchozími toky s průměrným počtem payload bytů na paket $\leq$ 100B a celkovým počtem příchozích toků
in_1000B_to_total_inflows	Poměr mezi příchozími toky s průměrným počtem payload bytů na paket $\geq$ 1000B a celkovým počtem příchozích toků
out_1000B_to_total_outflow	Poměr mezi odchozími toky s průměrným počtem payload bytů na paket $\geq$ 1000B a celkovým počtem odchozích toků
std_dev_in_bytes	Standardní odchylka počtu payload bytů na příchozí tok
std_dev_out_bytes	Standardní odchylka počtu payload bytů na odchozí tok
std_dev_in_pkts	Standardní odchylka počtu paketů na příchozí tok
std_dev_out_pkts	Standardní odchylka počtu paketů na odchozí tok
in_ack_to_inflow_ratio	Poměr mezi příchozími toky s příznakem ACK a všemi příchozími toky
out_ack_to_outflow_ratio	Poměr mezi odchozími toky s příznakem ACK a všemi odchozími toky
in_psh_to_inflow_ratio	Poměr mezi příchozími toky s příznakem PSH a všemi příchozími toky
out_psh_to_outflow_ratio	Poměr mezi odchozími toky s příznakem PSH a všemi odchozími toky
in_fin_to_inflow_ratio	Poměr mezi příchozími toky s příznakem FIN a všemi příchozími toky
out_fin_to_outflow_ratio	Poměr mezi odchozími toky s příznakem FIN a všemi odchozími toky
in_uniq_ips	Počet unikátních IP adres kontaktovaných příchozími toky
out_uniq_ips	Počet unikátních IP adres kontaktovaných odchozími toky
knw_to_total_local_port	Poměr použitých lokálních portů $<$ 10000 a všech portů
knw_to_total_remote_port	Poměr použitých vzdálených portů $<$ 10000 a všech portů