Jihočeská univerzita v Českých Budějovicích

Přírodovědecká fakulta

Analýza síťového provozu pomocí metod data miningu s cílem detekce nestandardního chování uživatelů

Bakalářská práce

Daniel Cába

Školitel:Ing. Petr Břehovský

České Budějovice 2012

Bibliografické údaje

Daniel Cába, 2012: Analýza síťového provozu pomocí metod data miningu s cílem detekce nestandardního chování uživatelů.

Anotace

Bakalářská práce se zaměřuje na problematiku detekce útoků na lokální síti. Nejdříve budou vybrány a popsány některé typy útoků, následně pak budou útoky prakticky provedeny, za účelem získání dat. V další části dojde k navržení experimentů a přispůsobení dat. V hlavní části pak dojde k samotným experimentům pomocí programu rapid miner.

Klíčová slova:

dobývání znalostí z dat, shluková analýza, útoky na LAN

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb., v platném znění, souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě, elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb., zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

V Českých Budějovicích, dne 12.12.2012 Podpis……

Poděkování

Zde bych velice rád poděkoval panu Ing. Petru Břehovskému. za odbornou pomoc a ochotu.

Obsah

1. Úvod a cíle práce

1.1 Úvod

Útoky na počítačovou síť je nadčasové téma, vždy zde budou skupiny lidí, jejichž cílem bude z nejrůznějších pohnutek napadnout naší síťovou infrastrukturu nebo odposlechnout naše citlivá data. Firewall v perimetru nedokáže zabránit útokům z vnitřní sítě nebo červům a jinému malwaru, který si uživatelé sami donesou. To ukázalo například hromadné rozšíření červu Conficker a jeho variant v roce 2009 ve velkých organizacích. Zatímco současné IDS založené na signaturách nedokáží včas detekovat úplně nové typy hrozeb, systémy založené na detekci statistických anomalií je často odhalit dokáží.

Tato práce je tedy zaměřena na navržení a otestování metod detekce některých vybraných útoků na vnitřní síť a jinou nestandardní aktivitu na síti pomocí statistických algoritmů a dobývání znalostí z dat.

V první části práce budou vybrány konkrétní útoky, na které se práce zaměří, a provedena analýza primárních a sekundárních zdrojů, které se jimi zabývají. Budou zjištěny charakteristické znaky vybraných útoků a způsob, jakým se podepisují na provozu v síti.

Na jejich základě budou vypracovány metody detekce, způsob zpracování dat a parametry, které budou do algoritmů analýzy vstupovat. Pak přistoupíme k výběru softwarových prostředků, pomocí kterých budeme odchytávat síťovou komunikaci.

Následně v empirické části proběhne získání a potřebná konverze dat a provedení samotných experimentů. Při nich dojde k analýze získaných dat a zhodnocení výsledků.

Práce přinese přehled některých metod detekce útoků na vnitřní sítě pomocí automatizovaných modelů a dalších metod data miningu a zhodnocení jednotlivých algoritmů a navrhnutých řešení při použití v praxi.

1.2 Cíle práce

- Prozkoumat a popsat použitelnost metod data miningu na analýzu provozu na lokální síti
- Navrhnout a otestovat metody detekce vybraných síťových útoků a nestandardního chování uživatelů založené na dobývání znalostí z dat.

2.0 Teorie útoku

V této části budou popsány vybrané útoky, jejich princip a charakteristické vlastnosti.

2.1 DHCP spoof teorie

Klient, který se hlásí na síť poprvé, odešle paket DHCP discover jako broadcast, a čeká na odpověď. DHCP server takový paket obdrží, a pokud má volné adresy k dispozici, odešle DHCP offer paket klientovi, kterým nabízí síťové nastavení. Klient v případě zájmu DHCP serveru odpovídá paketem DHCP request (většinou to bývá ten, kdo se ozve první) a DHCP server mu potvrzuje paketem DHCP ACK.

Teorie útoku

Konkrétní provedení se případ od případu liší, základ ale bývá stejný. Buď se útok provádí v momentě, kdy se oběť poprvé připojuje na síť tzn. ještě před tím, než odešle DHCP discover paket (ideální případ), nebo už od skutečného DHCP serveru adresu obdržela, což je varianta pravděpodobnější, ale pro útočníka poněkud pracnější. V prvním případě stačí být pouze rychlejší než skutečný DHCP server, čehož lze doclít i pouhým rychlejším doručením DHCP offer paketu oběti bez jakéhokoliv dalšího prozrazování se. V druhém případě je ovšem nutné zajistit, aby po uplynutí lease time opravdový DHCP server už nenabídl IP adresu. Toho lze snadno docílit např. vyčerpáním adresového prostoru, ze kterého

má příslušný DHCP server adresy přiřazovat. V takovém případě bude obět pátrat po novém DHCP serveru, na což útočník zareaguje.[1].

Jak se to projeví na síti

Vyčerpání adresového prostoru, popř. nějaký DoS útok na zahlcení DHCP serveru lze zdetekovat poměrně snadno, protože se typycky bude jednat o velký nárůst komunikace na síti. To určitě zdetekujeme snadno a útok tak snadno odhalíme. Ovšem jen tiché zaslání DHCP offer (a ACK) paketu se na objemu komunikace nijak nepodepíše. Systém si musí všimnout faktu, že tento typ paketu přišel z velmi nezvyklé adresy, a tedy se jedná o extrém.

2.2 MAC flood teorie

CAM tabulka je zjednodušeně tabulka o dvou sloupcích: MAC adresy a porty, ke kterým jsou zařízení s příslušnou adresou připojena. Po prvním připojení switche do sítě je tabulka prázdná, a když na nějaký port swtiche přijde paket, dojde k vytvoření záznamu v tabulce. Pokud na switch přijde na paket,jehož cílovou adresu nemá v tabulce, rozešle paket na všechny porty, kromě portu, na který paket přišel. Kapacita tabulky i reakce switche na různé nestandartní události záleží na konkrétním modelu.

Teorie útoku

MAC flooding je útok, při kterém útočník přeplní CAM tabulku switche vymyšlenými adresami, aby přinutil switch k přeposílání paketů na všechny porty a umožnil mu tak odposlouchávat komunikaci. Samotný princip útoku je velmi jednoduchý, jde pouze o generování paketů nějakým automatizovaným nástrojem s náhodou či náhodnou? odchozí adresou. Ve chvíli, kdy útočník tabulku naplní, bude schopen odposlechnout všechny počítače, které se od této chvíle připojí ke switchi. Pokud se chce ovšem odposlouchávat komunikace počítače, jehož adresa

už je v CAM tabulce switche, musí ve svém úsilí chvíli vytrvat. Údaje z CAM tabulky jsou totiž po určité době mazány. Když dojde ke smazání záznamu o oběti, útočník toho využije a zabere jeho místo.[1]

Jak se to projeví na síti

CAM tabulky jsou poměrně rozsáhlé a jejich přeplnění se projeví především masivním nárůstem komunikace na síti. Stejně tak útok prozradí velký počet dosud neznámých adres. Ostatní podezřelé vlastnosti se dost liší podle povahy paketu, např. I adresa příjemce může být smyšlená, ale i reálná, a cílový počítač na takové pakety bude nějakým způsobem reagovat.

2.3 Port sken teorie

Port sken je technika, která slouží k zjišťování, které porty má skenované zařízení otevřené a tedy, jaké služby na zařízení na dané adrese běží. Ačkoliv se nejedná o nic, co by bylo zákonem vysloveně zakázané, skenovat porty druhé osobě bez předchozí domluvy se považuje minimálně za neetické. skenování portů často využívají samotní správci, např. k odhalení trojských koňů nebo k prohlédnutí si služeb, které jimi spravovaný systém nabízí.[2] Na druhé straně, útočník může tyto informace snadno zneužít k útoku a právě port sken je věc, kterou hacker svůj útok začíná. Na sítí se tedy jedná o mimořádně podezřelou aktivitu, tím spíše na lokální síti.

Technik skenování portů je velmi mnoho. Liší se především podle použitého protokolu odesílaných paketů. Dá se říci, že nejběžnějším typem je TCP SYN sken, který jsme zvolili i my. Technika je založena na tom, že útočník předstírá zájem o otevření spojení na vybraném portu odesláním klasického TCP SYN paketu a čeká na odpověd, podle které zjišťuje stav portu. Pokud obdrží TCP SYN/ACK paket, je port otevřený, RST paket indikuje zavřený port. Žádná odpověd značí filtrovaný port. Nmap rovněž považuje port za otevřený, pokud v odpověď obdrží TCP SYN paket bez ACK příznaku díky velmi vzácnému split-

handshake spojení. Jedná se o poměrně rychlou metodu skenu, která není příliš nápadná, protože se TCP handshake vlastně nedokončí, tudíš se spojení nenaváže a tím se snižuje šance, že útočníkovo chování se uloží do logů. Za mínku také stojí i TCP connect skan, který se používá v případě, že operační systém neposkytuje přístup k raw soketu. Pak je nutno použít systémová volání pro navázání spojení na vybraném portu, což je ovšem pomalejší a nápadnější, protože se spojení dokončuje.[3]

Jak se sken projeví na síti

Ať už jde o jakýkoliv typ, kompletní skan se určitě projeví statisticky na vytíženosti sítě. Pravděpodobnost, že TCP SYN skan bude odhalen podle jednotlivých paketů se ale snižuje, neboť se příliž podobá klasickému navazování spojení. Co můžeme ovšem detekovat, je pokus o spojení na velmi neobvyklých portech nebo jiné anomálie.

2.4 DNS spoof teorie

Aby klient mohl využívat službu DNS, musí mít ve svém systému implementovaný resolver, tedy program, který slouží k formování dotazů na DNS (request pakety) servery a zpracování odpovědí (response pakety). Každý request paket obsahuje kromě adresy na překlad rovněž 16 bitové ID číslo, které musí server ve své odpovědi uvést, jinak odpověď nebude akceptována.[1]

Teorie útoku

Z předešlého odstavce je patrné, že přesné provedení útoku se do jisté míry odvíjí od konkrétní implementace resolveru. Vzhledem k tomu, že resolver je téměř vždy naimplementován v operačním systému, dalo by se říci, že záleží na operačním systému oběti. Některé resolvery například nekontrolují, zda obdržely opravdu odpověď na to, na co se ptaly (Win XP service pack 1). V každém

případě je ovšem nutné znát IP adresu DNS serveru, abychom se za něj mohli vydávat, znát číslo portu DNS služby a uhodnout ID číslo dotazu. Adresu DNS serveru mívají všechny počítače na LAN nakonfigurovanou stejně, útočníkovi se tedy obvykle stačí podívat na adresu DNS serveru na svém počítači. Nutné je však odhadnout číslo portu a vždy je nutné uhádnout ID číslo dotazu, což samo o sobě znamená trefit se do 1 z 2^{16} kombinací. Může být tedy nutné spomalit opravdový DNS server např. DoS útokem.[1]

Jak se to projeví na síti

Zde je důležité si všímat především doprovodných jevů poutajících se s tímto typem útoku, tzn. především nezvyklým nárůstem DNS response typu paketů, které útočník musí odesílat, aby uhodnul ID číslo requestu. Dále je třeba si všímat obrovského nárůstu komunikace jednoho počítače s opravdovým DNS serverem v rámci snahy o jeho zpomalení.

3. Provedení útoků

3.1 Útok DHCP spoof

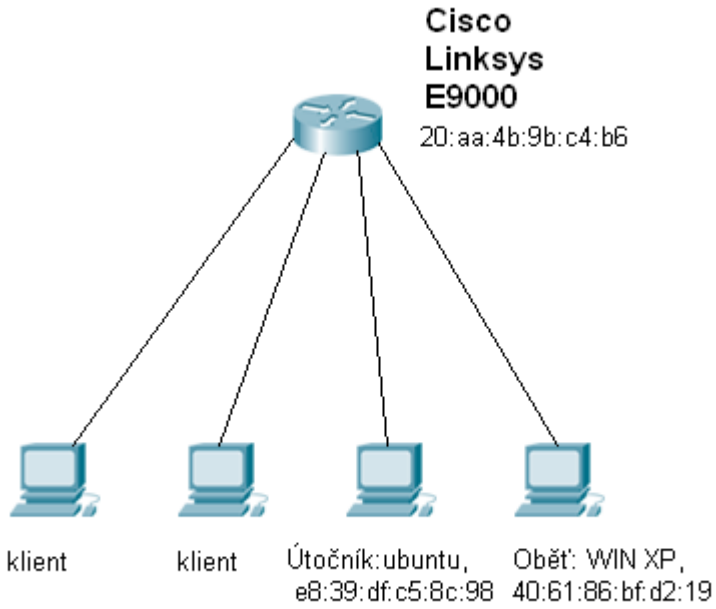

Fig1

V první řadě je třeba odchytat vzorové příklady obyčejného získání adresy z DHCP serveru. Topologie sítě viz Fig1 . Jako prostředek k odchytávání paketů byl použit program wireshark v. 1.6.5. Vě výběru atributů pro vizualizaci byly přidány dva sloupce, pro zdrojový a cílový port, typ adres nastaven na MAC(unresolved). Pro lepší vizualizaci ve wiresharku byly nastaveny filtry komunikace na porty 67 a 68, i když tento krok by mohl být vynechán a příslušná filtrace by se provedla pomocí programu na předzpracování. Následně byla data vyexportována do souboru csv. Pro release adresy z DHCP serveru byly použity konzolové příkazy ipconfig. Odchycená data viz soubor DHCP_vzor. Toto odchycení bylo provedeno několikrát na různých strojích v topologii. Následně byla data roztříděna a předzpracována programem.

Útok

Útočící počítač byl vybaven programem ettercap, který jsme pro útok využili, nainstalovaném na operačním systému ubuntu. Data byla odchytávána na počítači oběti, opět wiresharkem na platformě Windows xp. Bohužel se ukázalo, že GUI ettercapu obsahuje bug, který znemožňuje tento MITM útok provést, takže bylo nutné program ovládat textovými příkazy z konzole. Data viz soubor DHCP_spoof.csv.

Závěr

Útok byl několikrát opakován, pokaždé s úspěchem. To je poměrně překvapivé, protože veškerá komunikace jde přes router, který je zároveň DHCP serverem. Skutečný DHCP server by tedy teoreticky měl být rychlejší v odpovědi než útočník. Fakt, že útočník byl rychlejší, byl pravděpodobně způsoben pomalostí DHCP serveru při zpracovávání požadavku. Jedním z důvodů takového zpomalení může být, že pravý DHCP server musel vybírat adresu z většího rozsahu. Výsledkem tohoto útoku tedy je možnost odposlouchávat a měnit obsah nešifrované nebo nedostatečně šifrované komunikace.

3.2 Útok MAC flood

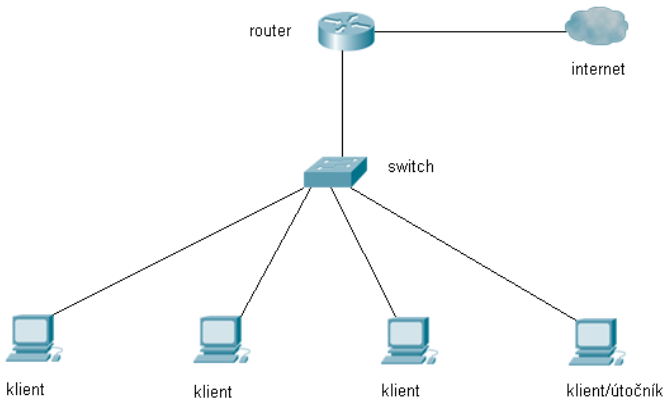

Fig2

Nejprve byly odchyceny vzory pro trénovací množinu, to znamená obyčejná data neobsahující útok. Data viz soubor clean_data.csv. Topologie sítě viz Fig2. Jako prostředek k odchytávání paketů byl použit program wireshark v. 1.6.5 bez filtrů nebo úprav pro zobrazení. Data byla odchytávána v souvislém intervalu zhruba dvou minut. Následně bylo odchytávání ukončeno a data exportována do csv souboru. Jako další fáze proběhlo odchytávání dat během útoku.

Útok

Útočící počítač běží pod systémem ubuntu. Pro samotný útok byl využit program Macof z balíku Dsniff, pomocí kterého jsme generovali velké množství paketů s náhodnými adresami. Data byla zachytávána na počítači přípojeném k

síti pomocí mirror portu na switchi. Data byla zachytávána pomocí nástroje wireshark, pod operačním systémem windows 7. Data byla odchytávána zhruba 50 sekund a po 10 sekundách byl spuštěn útok. (Časové údaje jsou orientační.) Data viz soubor Mac_flood.csv.

Závěr

Podařilo se nám zahltit CAM tabulku switche a získat data pro dobývání znalostí z dat. Výsledkem tohoto útoku je možnost odposlechu dat.

3.3 TCP SYN sken

Útok

Útok probíhal na stejné topologii jako předchozí útok MAC flood (Viz FigA). Samotný útok proběhl pomocí nástroje Nmap nainstalovaném na operačním systému ubuntu. Data byla odchytána pomocí nástroje wireshark, stejně jako v předchozích útocích. Data byla odchytávána po dobu 1,5 minuty, útok započal po 10 sekundách. sken nebyl omezen na rozsah portů.

Závěr

Data se nám podařilo odchytat, útok proběhl úspěšně. Některé porty byly odfiltrovány firewallem na straně oběti (windows firewall).

3.4 Útok DNS spoof

Samotný útok tak, jak je popsán výše, již není dnes použitelný. Operační systémy už neobsahují tolik chyb a protokol je poměrně dobře chráněň. I tak se však jedná o poměrně mocný nástroj například v kombinaci s ARP cache poisoningem, nebo jakýmkoliv dalším útokem, jehož výsledkem je přesměrovnání provozu. V našem pokusu jsme využili právě ARP cache poisoningu, abychom nemuseli hádat číslo dotazu.

Útok

Nejprve došlo k odchytání vzorů jako u předchozích příkladů pomocí programu wireshark. Pak došlo k exportu a zpracování odchycených paketů.Útok byl proveden na stejné topologii jako DHCP spoofing. K útoku byl využit opět program ettercap. Ettercap využívá konfigurační soubor etter.dns k volbě doménových jmen, která chceme podvrhnout, a adres, za které je chceme podvrhnout. Nastavili jsme doménová jména některých často používaných stránek jako google.com a seznam.cz tak, aby se překládala na adresu Jihočeské univerzity. Následně byl nejprve spuštěn ARP cache poisoning a pak DNS spoofing pomocí jednoho z pluginů programu.

Závěr

Podařilo se nám úspěšně přesměrovat provoz a podvrhnout překlady doménových jmen oběti. Efekt tohoto útoku byl pocítěn se spožděním, stejně tak jako jeho odeznění kvůli tomu, že záznamy se ukládají do DNS cache.

4. Návrh postupu

V této části bude řeč o tom, jak celý problém uchopit, a také, jak předzpracovat data.

4.1 Požadavky na metodiku data miningu

Základní věcí, kterou si je třeba uvědomit a která nám omezuje možnosti, je, že se jedná o tzv. One class learning. Jednotlivé řádky datové matice (ať už jednotlivé pakety nebo statistická data) se snažíme klasifikovat do dvou kategorií, podezřelý a nezávadný. K dispozici ovšem máme pouze data jedné z těchto kategorií – data, která jsou v pořádku. Tento přístup nám sice umožňuje reagovat na neučené situace (hacker je vždy o krok před námi), ale zároveň nám v prvotní analýze znemožňuje použít běžné klasifikátory. Budeme se tedy především zabývat separovatelností dat a shlukovou analýzou.

Metody založené na statistice resp. dobývání z dat nebývají bezchybné a ani v tomto případě to není reálný požadavek. Chyby se stávají, je však třeba si uvědomit, jaké chyby jsou zásadní. U takovýchto systémů, zejména v počátku jejich nasazení, často dochází k false positive událostem. Tyto chyby pak musí řešit osoba za to zodpovědná. Přesto, že je to pro takového pracovníka nejspíš vyčerpávající, škody jsou minimální. Navíc jak se postupně plní trénovací množina dalšími a dalšími vzory, počet false positive událostí se snižuje. I jediná false negative událost, tzn neodhalený útok, nám může způsobit velké škody. Proto je důležitější než na celkovou přesnost se v matici záměn soustředit především na true positive, někdy i za cenu vyšší citlivosti.

Další problém, který musíme vyřešit, je počet clusterů při shlukové analýze. Cluster by měl reprezentovat skupinu, do které řádek matice patří. Protože ale nevíme, jestli některý z paketů je opravdu podezřelý, nevíme dopředu kolik tam bude clusterů. Víme pouze, že buď jeden, nebo dva. Nabízí se několik řešení. Zkusíme použít metody s předem daným počtem clusterů (Např. K-MEANS)

a budeme předpokládat, že tam podezřelý paket je. Pokud se tam ve skutečnosti vyskytovat nebude a vrátí nám to nějaký řádek jako podezřelý, podíváme se do trénovací množiny, zda je paket odtud, a pokud to tak bude, můžeme říci, že tento vzor je v pořádku.. Další zajímavou metodou je postupné zkoušení počtu clusterů a měření průměrných vzdáleností vzorů od středu clusteru. Ta se bude samozřejmě postupně snižovat a ve chvíli, kdy se algoritmus dostane ke správnému počtu clusterů, dojde k velmi výraznému a nápadnému zlomu v redukci této vzdálenosti. Tato metoda je ovšem náročná na data, vyžaduje velmi dobře separovatelná data bez outlierů.

4.2 Selekce atributů

Ještě než dojde k vlastnímu odchycení dat, je třeba zvážit, jaké informace jsou nebo by mohly být relevantní pro náš problém, a které naopak by nás mohly pouze mást bez jakékoliv přidané hodnoty. Je potřeba měřit takové vlastnosti provozu, které nám nejen pomohou pro naše konkrétní měření experimentu, ale které by byly užitečné plošně v co největší míře. Tedy i na takové případy, kterými se zde zabývat nebudeme. Je možné měřit různé statistické údaje, je možné přistopit i k analýze jednotlivých paketů. Síťová komunikace je rozmanitá. I hrozeb a útoků je mnoho. Proto bude lépe nežli zvolit jeden přístup, zvolit jejich kombinaci. V první řadě tedy budeme měřit celkový objem komunikace za určitý čas. To může být užitečné pro detekci útoků, které generují velké množství provozu na síti, např. DOS útoky. Stejně tak by mohlo být užitečné měřit zastoupení jednotlivých protokolů v objemu síťové komunikace.

Dále přistoupíme i k analýze paktetů. Je jasné, že statisticky analyzovat payload paketů nemá velký smysl - vzhledem k počtu protokolů a různosti komunikace. Budeme třídit pakety podle typu a zaměříme se především na hlavičky, na zdrojový soket, cílový soket. U výběru podoby adresace zde máme dvě možnosti. Podle vrstvy síťového modelu se rozhodneme, ke kterou zvolit. Ve svých experimentech jsem se rozhodl pro vrstvu linkovou – tedy MAC adresy, protože se zaměřujeme na lokální síť, a hlavně při směrování paketů mimo lokální

síť je zde obrovské množství adres, které bychom nebyli schopni zpracovat. Takto komunikaci z LAN do WAN zjednodušíme na komunikaci mezi uživatelem a bránou.

5. Experimenty

5.1 DHCP spoof experimenty

Množiny dat

Obecně se dá říci, že metody data miningu pracují s velkým objemem dat, se kterým by si člověk neporadil. Na první pohled na naše množiny dat je evidentní, že obsahují poměrně málo příkladů. Je však třeba si uvědomit, že pracujeme s modelem skutečnosti a experimenty probíhají na malé lokální síti. Jestliže tedy máme na síti 1 DHCP server a N klientů, z podstaty útoku je jasné, že v trénovací množině může být pouze N vzorů. Každý další musí být s jiným nutně duplicitní. Naše množina neobsahuje ani N vzorů záměrně, protože nás též zajímá schopnost generalizace, tzn. reagování na neučená data. Při importování dat do RapidMineru použijeme operátor na čtení csv souborů. Všechny atributy jsou (poly)nominální, včetně čísel portů.

Výběr operátorů

RapidMiner nám nabízí mnoho operátorů pro clustering, které reprezentují jednotlivé algoritmy, resp. jejich variace. Zdaleka ne všechny se však hodí pro naše potřeby. Našim potřebám neodpovídá hierarchické shlukování ani fuzzy clustering, ale je třeba mít možnost v parametrech shora omezit počet clusterů. Ideálně, nikoliv však nutně, by operátor měl zvládat nominální atributy. Jedním takovým operátorem je k-medoids, který ovšem omezuje počet clusterů i zdola, kvůlu čemuž bude nutné zpětné porovnání, o kterém jsem se zmiňoval. Další operátor, který tuto vadu nesdílí, je x-means, který je variantou na známý

k- means. Oba tyto operátory však neumožňují pracovat s nominálními hodnotami, proto bude nutná úprava vstupních dat.

K-medoids

Algoritmus k-medoids funguje na jednoduchém principu.

1) Člověk zvolí počet clusterů.
2) Náhodně se zvolí K bodů, které budou reprezentaty clusteru.
3) Každý bod se zařadí k nejbližšímu clusteru.
4) Vybere se nový reprezentant clusteru, tak aby se průměrná vzdálenost bodů od něho minimalizovala.
5) bod 3 a 4 se opakuje, dokud se množina neustálí[5].

X-means

Jak již bylo uvedeno, jedná se o vylepšení algoritmu k-means o odhad počtu clusterů. Nutno uvést, že tento algoritmus není příliš rychlý.

1) Proběhne K-means pro spodní hranici K.
2) Každý centroid se rozdělí na 2 dětské cetroidy.
3) Ty se posunou v náhodném, navzájem opačném vektoru o velikost proporcionální s velikostí oblasti
4) V každé oblasti proběhne K means pro K=2 posunující dětské centroidy.
5) Pro každý dětský pár je otestováno, zda rozdělení přispělo ke zlepšení přesnosti. Na základě tohoto testu jsou děti ponechány, nebo zpět spojeny v rodiče. Jako test se nejčastěji používá BIC scoring.
6) K se zvětšuje a algoritmus je ukončen, když se K dostane na maximální horní hranici.[6]

Experiment s originálními daty

K-medoids

Nejprve byl použit operátor pro sloučení trénovací a testovací matice. Vstupní data viz Fig 3 a Fig4. Následně byl aplikován operátor k-medoids s parametrem k=2., a porběhl clustering. Proces viz Fig5.

SOURCE	DESTINATION	SRC PORT	DST PORT
20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68
20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68
20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68

Fig3

SOURCE	DESTINATION	SRC PORT	DST PORT
20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68
20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68
e8:39:df:c5:8c:98	40:61:86:bf:d2:19	67	68
20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68

Fig4

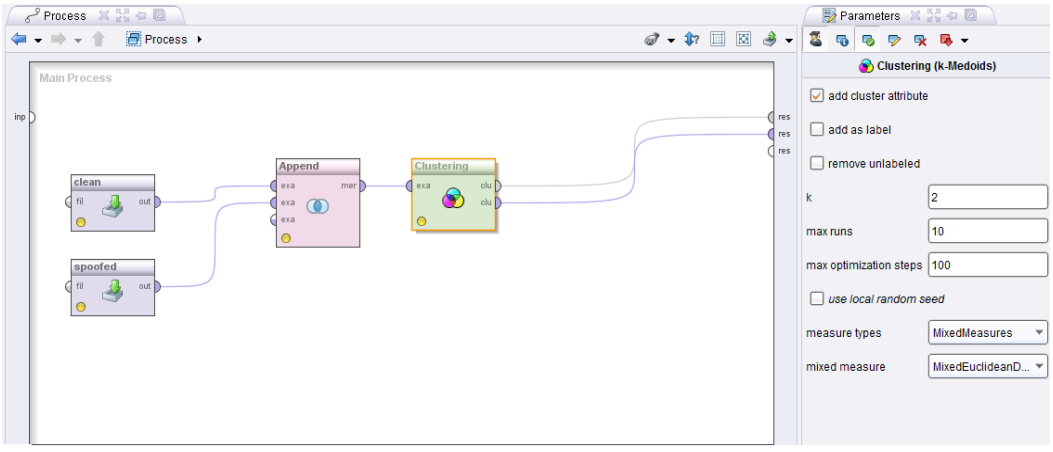

Fig5

Můžeme říci, že úspěšnost byla 100%, pro detailní výsledky viz Fig 6.

Row No.	id	cluster	SOURCE	DESTINATION	SRC PORT	DST PORT
1	1	cluster_0	20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68
2	2	cluster_0	20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68
3	3	cluster_0	20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68
4	4	cluster_0	20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68
5	5	cluster_0	20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68
6	6	cluster_1	e8:39:df:c5:8c:98	40:61:86:bf:d2:19	67	68
7	7	cluster_0	20:aa:4b:9b:c4:b6	ff:ff:ff:ff:ff:ff	67	68

Fig6

X-Means

Kromě operátoru pro sloučení datových matic bylo nutné použít operátor pro konverzi nominálních atributů na číselné. Proces viz. Fig 7.

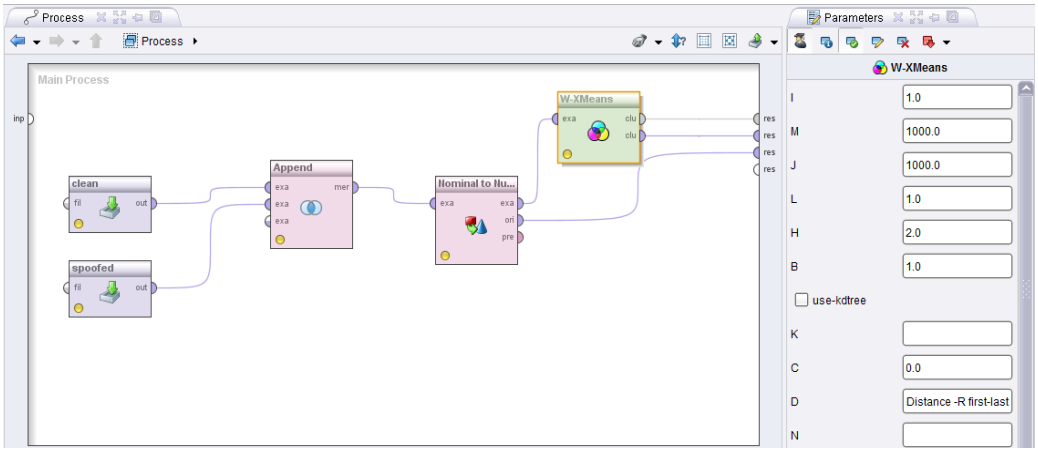

Fig7

Můžeme říci, že úspěšnost byla 100%, pro detailní výsledky viz Fig 8.

Row No.	id	cluster	SOURCE	DESTINATI...	SRC PORT	DST PORT
1	1	cluster1	0	0	0	0
2	2	cluster1	0	0	0	0
3	3	cluster1	0	0	0	0
4	4	cluster1	0	0	0	0
5	5	cluster1	0	0	0	0
6	6	cluster0	1	1	0	0
7	7	cluster1	0	0	0	0

Fig8

Vzhledem k velmi dobré separovatelnosti dat byly výsledky opravdu dobré. Veliký podíl na tom má fakt, že skutečný DHCP server odpovídá na discover pakety broadcastem, zatímco útočník, ve snaze zakrýt svoji činnost, používá v odpovědi unicast. Podívejme se nyní, jak by výsledky vypadaly, pokud by i reálný DHCP server odpovídal na discover pakety unicastem, čímž operátorům práci trochu znesnadníme.

Experiment s pozměněnými daty

Vstupní data byla upravena tak, aby adresy cíle offer paketů neobsahovaly broadcast adresu, ale skutečné fyzické adresy strojů, na které tento paket odpovídal, a data byla záměrně upravena tak, aby vystihla pro analýzu složité případy. Data viz Fig 9, 10.

SOURCE	DESTINATION	SRC PORT	DST PORT
20:aa:4b:9b:c4:b6	00:0f:fe:0f:60:7a	67	68
20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	67	68
20:aa:4b:9b:c4:b6	40:61:86:bf:d2:19	67	68

Fig9

SOURCE	DESTINATION	SRC PORT	DST PORT
20:aa:4b:9b:c4:b6	00:0f:fe:0f:60:7a	67	68
20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	67	68
e8:39:df:c5:8c:98	40:61:86:bf:d2:19	67	68
20:aa:4b:9b:c4:b6	b4:74:9f:e0:34:d1	67	68

Fig10

Proces samozřejmě probíhal stejně, pouze s jinými vstupními daty.

X-means

Je vidět, že tento algoritmus nebyl schopen detekovat útok. Přestože měl vysokou úspěšnost celkově, protože kvantitativně převážila nezávadná data, není pro tento případ vyhovující, protože jeho true positive úspěšnost je 0. Tento neúspěch je pravděpodobně ovlivněn i tím, že bylo nutné data konvertovat z nominálních na číselná. Je třeba mít však na paměti, že se jedná spíše pouze o simulovaná data, nikoliv o skutečná. Výsledky viz Fig11 .

Row No.	id	cluster	SOURCE	DESTINATI...	SRC PORT	DST PORT
1	1	cluster0	0	0	0	0
2	2	cluster0	0	1	0	0
3	3	cluster0	0	2	0	0
4	4	cluster0	0	0	0	0
5	5	cluster0	0	1	0	0
6	6	cluster0	1	2	0	0
7	7	cluster0	0	3	0	0

Fig11

K-medoids

Tento algoritmus byl schopen správně zdetekovat útok i ve složitější množině dat. Úspěšnost 100%. Takovýto experiment by však nebyl kompletní. Je nutné vyzkoušet analyzovat i "čistá"data – data, která útočný paket vůbec neobsahují. Výsledky viz Fig12

Row No.	id	cluster	SOURCE	DESTINATION	SRC PORT	DST PORT
1	1	cluster_0	20:aa:4b:9b:c4:b6	00:0f:fe:0f:60:7a	67	68
2	2	cluster_0	20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	67	68
3	3	cluster_0	20:aa:4b:9b:c4:b6	40:61:86:bf:d2:19	67	68
4	4	cluster_0	20:aa:4b:9b:c4:b6	00:0f:fe:0f:60:7a	67	68
5	5	cluster_0	20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	67	68
6	6	cluster_1	e8:39:df:c5:8c:98	40:61:86:bf:d2:19	67	68
7	7	cluster_0	20:aa:4b:9b:c4:b6	b4:74:9f:e0:34:d1	67	68

Fig12

K-medoids(čistá data)

Protože jsme nastavili počet clusterů na 2, dostali sme opravdu 2 clustery, a máme 2 false positive události. Jak jsem již zmiňoval, je třeba tyto pakety porovnat, zdali nejsou obsaženy v testovací množině, a nejedná se tedy o falešný poplach. Při zpětné kontrole zjišťujeme, že tyto pakety opravdu jsou obsaženy v testovací množině. Jedná se tedy o falešný poplach. Tato zpětná kontrola algoritmus mírně prodlužuje, ale je schopna sama eliminovat většinu false positive událostí.Test byl tedy 100% úspěšný. Výsledek clusteringu viz Fig13.

Row No.	id	cluster	SOURCE	DESTINATI...	SRC PORT	DST PORT
1	1	cluster_0	20:aa:4b:9b:	00:0f:fe:0f:6(	67	68
2	2	cluster_0	20:aa:4b:9b:	00:23:54:4c:	67	68
3	3	cluster_1	20:aa:4b:9b:	40:61:86:bf:(	67	68
4	4	cluster_0	20:aa:4b:9b:	00:0f:fe:0f:6(	67	68
5	5	cluster_0	20:aa:4b:9b:	00:23:54:4c:	67	68
6	6	cluster_1	20:aa:4b:9b:	40:61:86:bf:(	67	68
7	7	cluster_0	20:aa:4b:9b:	b4:74:9f:e0:3	67	68

Fig13

Klasifikátory

Klasifikátory, které rapidminer nabízí, nejsou užitečné pro tento druh experimentů díky již zmíněnému one class learning problému. Přesto však, když už byl útok identifikován a získáme data i druhého typu, můžeme využít i nějaký tradiční klasifikátor. V tomto experimentu jsem použil originální i modifikovaná data a bayesův naivní klasifikátor, abych zjistil, zda je schopen vypozorovat příslušné závislosti. Výsledek viz Fig 14. Stojí ovšem za zamyšlení, zda jsme si jisti, že toto je jediný typ útoku tímto druhem paketu. Pokud si jisti nejsme, a jisti si být nemůžeme, je lepší spolehnout se na shlukovou analýzu.

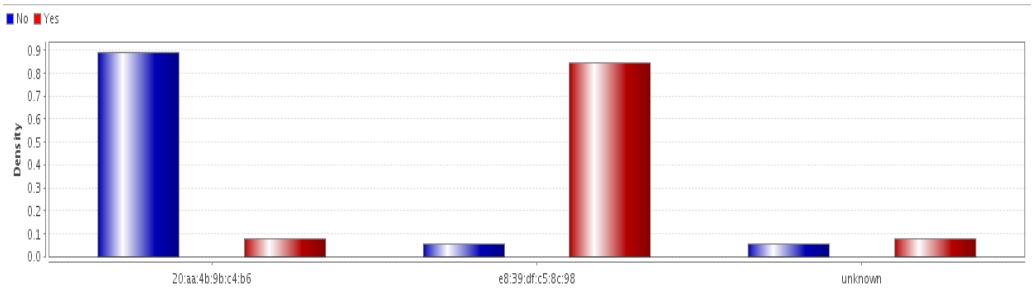

Fig14

5.2 MAC flood experimenty

Množina dat

V každém příkladě z datové matice je kvantitativně zaznamenán počet paketů za dříve zvolenou časovou jednotku, to je zde 10 sekund. Kromě celkového počtu odeslaných paketů je také počítán počet paketů jednotlivých protokolů odeslaných za tento časový úsek. Zde je třeba dodat, že se zdaleka nejedná o všechny protokoly, ani většinu, ale pro laboratorní podmínky a pro naše experimenty je to zcela postačující. Zvolení správného časového intervalu je zde velmi důležitý faktor. Při zvolení většího intervalu budou naše data více potlačovat extrémy, což bude mít za následek méně false positive událostí, ale také více false negative událostí. Zjednodušeně se dá říci, že délka časového intervalu je nepřímo úměrná citlivosti detekce. Tento stav rovnosti časových intervalů je dobře použitelný v našich podmínkách. V reálných podmínkách by musel být přístup trochu pozměněný, protože je jasné, že síť bude různě vytížená v různých hodinách. Například pokud je síť ve tři hodiny ráno stejně vytížená jako v jednu hodinu odpoledne, je zřejmě někde něco špatně.

Výběr operátorů

Obecně se nám budou hodit operátory flat clusteringu, které umí pracovat s numerickými hodnotami a umožňují nám omezit shora počet clusterů. Experiment

provedeme s algoritmy k-means, kde ovšem musíme omezit počet clusterů i shora, a x-means, jehož vhodnější implementace je poskytována prostřednictvím rozšíření rapidmineru WEKA.

K-means

1) Uživatel zadá počet clusterů.
2) Náhodně se zvolí K bodů, které budou reprezentaty clusteru.
3) Každý bod je přiřazen reprezentatu/středu clusteru, kterému je nejblíže.
4) Vypočítá se (nový) střed clusteru.
5) Bod 3 a 4 probíhá do ustálení.[4]

K-means

Data do trénovací matice byla odchytána během souvislých dvou minut, kdy byl odchytáván "běžný" provoz, resp. Jeho simulace. Testovací data byla samozřejmě odchytána na stejné topologii, zhruba během stejného časového intervalu. V průběhu odchytávání do trénovací matice byl spuštěn útok. Data viz Fig 15.

1	period	DNS	ARP	DHCP	ICMP	HTTP	TCP	UDP	total
2	0-10	520	0	0	0	672	12848	0	14215
3	01/10/20	304	4	0	0	757	104921	0	106082
4	20-30	0	0	0	0	191	95378	0	95587
5	30-40	398	11	7	0	610	3689	0	4837
6	40-50	324	0	0	0	60	459	0	922

Fig15

Byly použity operátory pro čtení csv souborů, spojení trénovací a testovací matice a operátor pro příslušný clustering. Proces viz Fig 16.

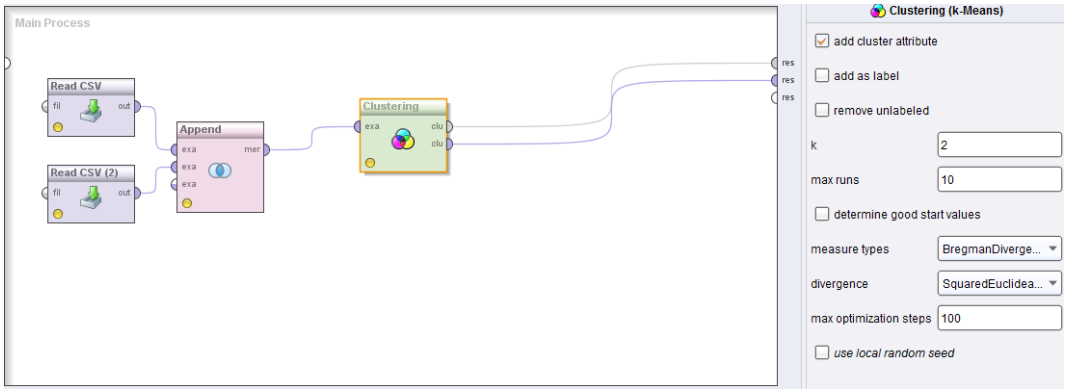

Fig16

Algoritmus byl 100% úspěšný. Výsledky viz Fig17.

Row No.	id	cluster	DNS	ARP	DHCP	ICMP	HTTP	TCP	UDP	total
3	3	cluster_1	148	28	0	2	441	2367	12	3285
4	4	cluster_1	140	8	0	0	339	1203	2	1730
5	5	cluster_1	96	13	7	0	974	3914	0	5113
6	6	cluster_1	169	13	3	0	1050	13405	0	15092
7	7	cluster_1	232	7	0	0	1569	6224	0	8192
8	8	cluster_1	136	8	0	0	776	3858	0	4834
9	9	cluster_1	24	3	0	0	710	3128	0	3936
10	10	cluster_1	40	7	0	0	1143	5595	0	6847
11	11	cluster_1	84	8	0	0	751	10986	0	11860
12	12	cluster_1	120	7	0	0	537	3426	0	4152
13	13	cluster_1	148	6	0	0	762	4559	0	5566
14	14	cluster_1	28	3	0	0	449	2763	0	3274
15	15	cluster_1	44	13	7	0	524	11560	0	12377
16	16	cluster_1	44	6	0	0	250	2147	0	3437
17	17	cluster_1	520	0	0	0	672	12848	0	14215
18	18	cluster_0	304	4	0	0	757	104921	0	106082
19	19	cluster_0	0	0	0	0	191	95378	0	95587
20	20	cluster_1	398	11	7	0	610	3689	0	4837
21	21	cluster_1	324	0	0	0	60	459	0	922

Fig17

X-means

Experiment proběhl se stejnými daty, liší se v použití algoritmu pro clustering. Maximální počet clusterů byl omezen na 2, minimální počet clusterů na 1. Proces viz Fig18.

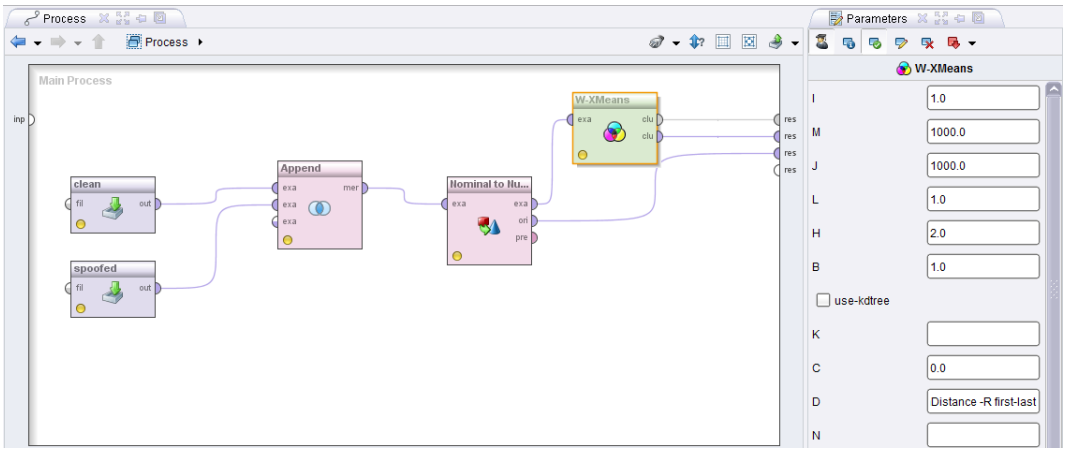

Fig18

Algoritmus nebyl schopen detekovat žádné anomálie. Výsledek viz Fig19.

Row No.	id	cluster	DNS	ARP	DHCP	ICMP	HTTP	TCP	UDP	total
3	3	cluster0	148	28	0	2	441	2367	12	3285
4	4	cluster0	140	8	0	0	339	1203	2	1730
5	5	cluster0	96	13	7	0	974	3914	0	5113
6	6	cluster0	169	13	3	0	1050	13405	0	15092
7	7	cluster0	232	7	0	0	1569	6224	0	8192
8	8	cluster0	136	8	0	0	776	3858	0	4834
9	9	cluster0	24	3	0	0	710	3128	0	3936
10	10	cluster0	40	7	0	0	1143	5595	0	6847
11	11	cluster0	84	8	0	0	751	10986	0	11860
12	12	cluster0	120	7	0	0	537	3426	0	4152
13	13	cluster0	148	6	0	0	762	4559	0	5566
14	14	cluster0	28	3	0	0	449	2763	0	3274
15	15	cluster0	44	13	7	0	524	11560	0	12377
16	16	cluster0	44	6	0	0	250	2147	0	3437
17	17	cluster0	520	0	0	0	672	12848	0	14215
18	18	cluster0	304	4	0	0	757	104921	0	106082
19	19	cluster0	0	0	0	0	191	95378	0	95587
20	20	cluster0	398	11	7	0	610	3689	0	4837
21	21	cluster0	324	0	0	0	60	459	0	922

Fig19

K-means (čistá data)

Tentokrát byla jako testovací množnina použit soubor obsahující pouze další běžná data odchycená stejným způsobem a na stejné topologii jako trénovací data.

Ze 32 vzorů bylo 6 indentifikováno jako extrém. Při kontrole trénovací množiny byly 4 vzory vyloučeny. U dvou vzorů byla tedy nahlášena false positive událost. Jak již bylo řečeno, false positive události jsou při nízkém počtu vzorů v trénovací matici poměrně časté, a zde byla testovací matice zhruba o třetinu delší než trénovací. Úspěšnost 93,75 procent. Výsledek viz Fig20.

Row No.	id	cluster	DNS	ARP	DHCP	ICMP	HTTP	TCP	UDP	total
1	1	cluster_0	146	12	0	0	696	2580	0	6706
2	2	cluster_0	216	2	0	0	1524	5622	0	7551
3	3	cluster_0	148	28	0	2	441	2367	12	3285
4	4	cluster_0	140	8	0	0	339	1203	2	1730
5	5	cluster_0	96	13	7	0	974	3914	0	5113
6	6	cluster_1	169	13	3	0	1050	13405	0	15092
7	7	cluster_1	232	7	0	0	1569	6224	0	8192
8	8	cluster_0	136	8	0	0	776	3858	0	4834
9	9	cluster_0	24	3	0	0	710	3128	0	3936
10	10	cluster_0	40	7	0	0	1143	5595	0	6847
11	11	cluster_1	84	8	0	0	751	10986	0	11860
12	12	cluster_0	120	7	0	0	537	3426	0	4152
13	13	cluster_0	148	6	0	0	762	4559	0	5566
14	14	cluster_0	28	3	0	0	449	2763	0	3274
15	15	cluster_1	44	13	7	0	524	11560	0	12377
16	16	cluster_0	44	6	0	0	250	2147	0	3437
17	17	cluster_0	6	13	0	0	185	543	0	881
18	18	cluster_0	67	14	4	0	1030	4427	0	5676
19	19	cluster_0	92	7	0	0	542	1424	0	2102

Fig20

5.3 Port sken experimenty

Množna dat

Z logiky věci je jasné, že množiny dat musí být zpracovávány stejně pro stejnou metodu detekce. Proto byla množina dat upravena stejně jako v případě experimentů s MAC floodem.

Výběr operátorů

To samé samozřejmě musí platit i o použitých algoritmech, dopředu přece nikdy nevíme, jaká data se k nám dostanou.

K-means

Testovací data byla samozřejmě odchytána na stejné topologii jako při útoku MAC floodem a zhruba během stejného časového intervalu. V průběhu odchytávání do trénovací matice byl spuštěn útok. Data viz Fig 21.

	A	B	C	D	E	F	G	H	I
1	period	DNS	ARP	DHCP	ICMP	HTTP	TCP	UDP	total
2	0-10	0	8	0	0	7	157	0	187
3	01/10/20	16	7	0	0	237	1361	0	1682
4	20-30	1	11	0	0	169	307	0	495
5	30-40	79	11	0	0	468	41705	0	42523
6	40-50	44	8	0	0	301	88491	0	88902
7	50-60	164	7	0	0	289	3042	0	3573
8	60-70	236	8	0	0	837	12239	0	13434
9	70-80	68	6	0	0	409	6983	0	7502
10	80-90	4	5	0	0	31	215	0	285

Fig 21

Byly použity operátory pro čtení csv souborů, spojení trénovací a testovací matice a operátor pro příslušný clustering.

Row No.	id	cluster	DNS	ARP	DHCP	ICMP	HTTP	TCP	UDP	total
7	7	cluster_0	232	7	0	0	1569	6224	0	8192
8	8	cluster_0	136	8	0	0	776	3858	0	4834
9	9	cluster_0	24	3	0	0	710	3128	0	3936
10	10	cluster_0	40	7	0	0	1143	5595	0	6847
11	11	cluster_0	84	8	0	0	751	10986	0	11860
12	12	cluster_0	120	7	0	0	537	3426	0	4152
13	13	cluster_0	148	6	0	0	762	4559	0	5566
14	14	cluster_0	28	3	0	0	449	2763	0	3274
15	15	cluster_0	44	13	7	0	524	11560	0	12377
16	16	cluster_0	44	6	0	0	250	2147	0	3437
17	17	cluster_0	0	8	0	0	7	157	0	187
18	18	cluster_0	16	7	0	0	237	1361	0	1682
19	19	cluster_0	1	11	0	0	169	307	0	495
20	20	cluster_1	79	11	0	0	468	41705	0	42523
21	21	cluster_1	44	8	0	0	301	88491	0	88902
22	22	cluster_0	164	7	0	0	289	3042	0	3573
23	23	cluster_0	236	8	0	0	837	12239	0	13434
24	24	cluster_0	68	6	0	0	409	6983	0	7502
25	25	cluster_0	4	5	0	0	31	215	0	285

Fig22

Algoritmus byl 100% úspěšný. Výsledky viz Fig22.

X-means

Experiment proběhl se stejnými daty, liší se v použití algoritmu pro clustering. Výsledek: Algoritmus ani v tomto případě nebyl schopen detekovat anomálie. Výsledek viz Fig23.

Row No.	id	cluster	DNS	ARP	DHCP	ICMP	HTTP	TCP	UDP	total
7	7	cluster0	232	7	0	0	1569	6224	0	8192
8	8	cluster0	136	8	0	0	776	3858	0	4834
9	9	cluster0	24	3	0	0	710	3128	0	3936
10	10	cluster0	40	7	0	0	1143	5595	0	6847
11	11	cluster0	84	8	0	0	751	10986	0	11860
12	12	cluster0	120	7	0	0	537	3426	0	4152
13	13	cluster0	148	6	0	0	762	4559	0	5566
14	14	cluster0	28	3	0	0	449	2763	0	3274
15	15	cluster0	44	13	7	0	524	11560	0	12377
16	16	cluster0	44	6	0	0	250	2147	0	3437
17	17	cluster0	0	8	0	0	7	157	0	187
18	18	cluster0	16	7	0	0	237	1361	0	1682
19	19	cluster0	1	11	0	0	169	307	0	495
20	20	cluster0	79	11	0	0	468	41705	0	42523
21	21	cluster0	44	8	0	0	301	88491	0	88902
22	22	cluster0	164	7	0	0	289	3042	0	3573
23	23	cluster0	236	8	0	0	837	12239	0	13434
24	24	cluster0	68	6	0	0	409	6983	0	7502
25	25	cluster0	4	5	0	0	31	215	0	285

Fig23

5.4 DNS spoof experimenty

Data byla zpracována stejným způsobem jako v případě detekce DHCP spoofingu a byly použity stejné algoritmy pro detekci.

X-means

Trénovaci množina viz Fig21, testovací množina viz Fig22.

	A	B	C	D
1	SOURCE	DESTINATION	SRC PORT	DST PORT
2	20:aa:4b:9b:c4:b6	b4:74:9f:e0:34:d1	53	1083
3	20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	53	1083
4	20:aa:4b:9b:c4:b6	40:61:86:bf:d2:19	53	1084
5	20:aa:4b:9b:c4:b6	e8:39:df:c5:8c:98	53	1076
6	20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	53	1083
7	20:aa:4b:9b:c4:b6	40:61:86:bf:d2:19	53	1083
8	20:aa:4b:9b:c4:b6	e8:39:df:c5:8c:98	53	1083
9	20:aa:4b:9b:c4:b6	b4:74:9f:e0:34:d1	53	1084
10	20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	53	1076
11	20:aa:4b:9b:c4:b6	e8:39:df:c5:8c:98	53	1083
12	20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	53	1084
13	20:aa:4b:9b:c4:b6	40:61:86:bf:d2:19	53	1076
14	20:aa:4b:9b:c4:b6	b4:74:9f:e0:34:d1	53	1028

Fig21

	A	B	C	D
1	SOURCE	DESTINATION	SRC PORT	DST PORT
2	e8:39:df:c5:8c:98	b4:74:9f:e0:34:d1	53	1150
3	e8:39:df:c5:8c:98	00:23:54:4c:70:03	53	1050
4	20:aa:4b:9b:c4:b6	e8:39:df:c5:8c:98	53	1035

Fig22

Algoritmus nebyl schopen detekovat anomálie. Výsledky clusteringu viz Fig23.

Row No.	id	cluster	SOURCE	DESTINATI...	SRC PORT	DST PORT
1	1	cluster0	0	0	0	0
2	2	cluster0	0	1	0	0
3	3	cluster0	0	2	0	1
4	4	cluster0	0	3	0	2
5	5	cluster0	0	1	0	0
6	6	cluster0	0	2	0	0
7	7	cluster0	0	3	0	0
8	8	cluster0	0	0	0	1
9	9	cluster0	0	1	0	2
10	10	cluster0	0	3	0	0
11	11	cluster0	0	1	0	1
12	12	cluster0	0	2	0	2
13	13	cluster0	0	0	0	3
14	14	cluster0	1	0	0	4
15	15	cluster0	1	1	0	5
16	16	cluster0	0	3	0	6

Fig23

K-means

Ani zde nebyl algoritmus schopen správně detekovat podezřelé pakety. Ukazuje se, že protokol s velkým rozsahem použitých portů je pro navrženou metodu detekce příliš složitý. Řešením by mohl být jiný výběr atributů vstupujících do shlukové analýzy nebo přiřazení službě (protokolu) jednoho neměnného portu. Data viz Fig24.

Row No.	id	cluster	SOURCE	DESTINATION	SRC PORT	DST PORT
1	1	cluster_0	20:aa:4b:9b:c4:b6	b4:74:9f:e0:34:d1	53	1083
2	2	cluster_0	20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	53	1083
3	3	cluster_1	20:aa:4b:9b:c4:b6	40:61:86:bf:d2:19	53	1084
4	4	cluster_0	20:aa:4b:9b:c4:b6	e8:39:df:c5:8c:98	53	1076
5	5	cluster_0	20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	53	1083
6	6	cluster_1	20:aa:4b:9b:c4:b6	40:61:86:bf:d2:19	53	1083
7	7	cluster_0	20:aa:4b:9b:c4:b6	e8:39:df:c5:8c:98	53	1083
8	8	cluster_0	20:aa:4b:9b:c4:b6	b4:74:9f:e0:34:d1	53	1084
9	9	cluster_1	20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	53	1076
10	10	cluster_0	20:aa:4b:9b:c4:b6	e8:39:df:c5:8c:98	53	1083
11	11	cluster_0	20:aa:4b:9b:c4:b6	00:23:54:4c:70:03	53	1084
12	12	cluster_1	20:aa:4b:9b:c4:b6	40:61:86:bf:d2:19	53	1076
13	13	cluster_0	20:aa:4b:9b:c4:b6	b4:74:9f:e0:34:d1	53	1028
14	14	cluster_0	e8:39:df:c5:8c:98	b4:74:9f:e0:34:d1	53	1150
15	15	cluster_0	e8:39:df:c5:8c:98	00:23:54:4c:70:03	53	1050
16	16	cluster_0	20:aa:4b:9b:c4:b6	e8:39:df:c5:8c:98	53	1035

Fig24

6. Závěr

Byly úspěšně provedeny vybrané útoky a získaná data byla upravena tak, aby splňovala požadavky pro navržené experimenty. Ty byly provedeny pomocí programu rapidminer. Navržená řešení byla poměrně úspěšná, ukázalo se však, že mají nedostatky. Nebyli jsme chopni správně detekovat útok na DNS protokol, DNS spoofing. Hlavním problémem se ukázal být příliš velký rozsah možných portů, který je řádově v tisících, což je příliš velké množství nominálních hodnot pro použité metody shlukové analýzy. Řešením by mohlo být přiřazení službě DNS znatelně menšího rozsahu portů k použití. To ale jde částečně proti bezpečnosti služby, neboť útočník, pokud před tím nepoužil jiný MITM útok, musí použité číslo portu uhádnout. Další možností by mohlo být vyřazení čísel portů z množiny zpracovávaných atributů. To by sice nijak neublížilo úspěšnosti našich experimentů, právě naopak, avšak takovéto zjednodušení může být nepraktické vůči jiným typům hrozeb, se kterými jsme neexperimentovali. Množství operátorů (algoritmů) v rapid mineru, které by odpovídaly našim potřebám, se ukázalo jako ne zcela uspokojivé. Zcela chyběly klasifikátory

umožňující one class klasifikaci, s výjimkou SVM operátoru, který, přestože tato funkce nebyla zdokumentována, once class klasifikaci umožňoval. Výsledky však byly velkým zklamáním. Využívali jsme tedy především metod shlukové analýzy se zpětným ověřováním výsledků vůči trénovací množině. Zde se ukázalo jako velmi důležité, kolik neotestovaných vzorů do klasifikace vstupuje. Příliš malé množství bude algoritmus zpomalovat, příliš velké množství bude snižovat přesnost, neboť střed shluku při analýze může být množstvím anomálií vychýlen. Navzdory laboratorním podmínkám jsme byli schopni odhalit zbylé útoky s velkou, v drtivé většině100% přesností (u některých algoritmů), přestože systémům založeným na statistické analýze většinou trvá delší dobu, než získají dostatečně velké množství dat pro analýzu s větší přesností.

7. Použitá literatura

[1] HALLER, Martin. Lupa.cz Odposlouchávame data na přepínaném Ethernetu (1.) Odposlouchávame data na přepínaném Ethernetu. [online]. 2006, s. 1-5 [cit. 2012-12-12]. Dostupné z: http://www.lupa.cz/clanky/odposlouchavame-data-na-prepinanem-ethernetu-1/

[2] MATEJKA, Jan. (Ne)legální port skenning. [online]. 2001 [cit. 2012-12-12]. Dostupné z: http://www.lupa.cz/clanky/nelegalni-port-scanning/

[3] LYON, Gordon. Nmap Network scanning. [online]. [cit. 2012-12-11]. Dostupné z: nmap.org/book/man-port-skenning-techniques.html

[4] Some Methods for classification and Analysis of Multivariate Observations. J.B., MacQueen. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. 1967, 281–297.

[5] ROUSSEEUW, P.J a L. KAUFMAN. Clustering by means of Medoids, in Statistical Data Analysis Based on the L1-Norm and Related Methods. 1987, 405–416.

[6] X-means: Extending K-means with Efficient Estimation of the Number of Clusters. PELLEG, Dan a Andrew MOORE. Proceedings of the Seventeenth International Conference on Machine Learning. 2000, s. 727-734.

8. Přílohy

[A] Dvd s procesy rapid mineru, odchycenými daty, a zdrojovými kódy SW na předzpracování dat.