Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

Detekce bezpečnostních hrozeb v síti

Tomáš Vecko

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Tomáš Vecko

Systémové inženýrství a informatika
Informatika

Název práce

Detekce bezpečnostních hrozeb v síti

Název anglicky

Detection of Network Security Threats

Cíle práce

Hlavním cílem práce je implementace systému pro detekci bezpečnostních hrozeb v síti (NIDS) pro zvolené firemní prostředí.
Dílčí cíle:
- Charakterizovat systém pro detekci bezpečnostních hrozeb v síti
- Zpracovat porovnání dostupných systémů
- Návrh vybraného systému a jeho implementace
- Realizovat monitoring pro vybraný systém
- Optimalizace systému
- Hodnocení a závěry

Metodika

Teoretická část je založena na studiu zdrojů a analýze řešené problematiky.

V praktické části bude nejprve provedeno porovnání dostupných systémů (technické řešení, ekonomika). Dále bude realizován výběr konkrétního řešení, pro které bude proveden návrh systému a jeho implementace. Pomocí dat z monitoringu bude provedena optimalizace.Závěry práce budou vyvozeny z teoretických i praktických poznatků vlastního řešení.

Doporučený rozsah práce

40 – 50 stran

Klíčová slova

bezpečnost, síť, monitoring, NIDS, Snort, Suricata, Nagios, optimalizace, detekce hrozeb

---

Doporučené zdroje informací

BEJTLICH, Richard. The practice of network security monitoring: understanding incident detection and response. San Francisco: No Starch Press, [2013]. ISBN 978-159-3275-099

HUNT, Cameron. Mastering network security: understanding incident detection and response. 2nd ed. San Francisco: Sybex, c2003. ISBN 978-078-2141-429.

Scaling in the Linux Networking Stack. Kernel.org [online]. Dostupné z: https://www.kernel.org/doc/Documentation/networking/scaling.txt

Suricata, to 10Gbps and beyond. Regit [online]. 30.7.2012. Dostupné z: https://home.regit.org/2012/07/suricata-to-10gbps-and-beyond/

Suricata User Guide [online]. Dostupné také z: https://suricata.readthedocs.io/en/latest/

VANNEY, Ivan. Configure Snort IDS and Create Rules. Linuxhint [online]. Dostupné z: https://linuxhint.com/configure-snort-ids-create-rules/

Předběžný termín obhajoby

2019/20 LS – PEF

Vedoucí práce

Ing. Jiří Vaněk, Ph.D.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 8. 5. 2019

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 14. 10. 2019

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 27. 02. 2020

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Detekce bezpečnostních hrozeb v síti" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 12. 3. 2020 _____

Poděkování

Rád bych touto cestou poděkoval panu Ing. Jiřímu Vaňkovi, Ph.D. za odborné vedení práce a konzultace. Dále chci poděkovat společnosti Gem System a.s. za spolupráci při tvorbě této práce.

Detekce bezpečnostních hrozeb v síti

Abstrakt

Tato práce se se zabývá analýzou, návrhem a implementací vybraného IDS do firemní sítě. První část práce je zaměřena na teoretické uvedení do problematiky sítí. Pokračuje s uvedením příkladů systémů pro sledování sítě, které jsou současné době dostupné. Je představena jejich funkcionalita a licenční použití. Z teoretické části byl vybrán IDS vhodný pro danou společnost. Další část práce se věnuje implementaci vybraného systému. Ukazuje se zde postup instalace, konfigurace, a i práce s konkrétním IDS. Pro zvolený systém je vypracován monitoring a optimalizace. V závěrečné části dochází ke zhodnocení zvolených nástrojů v síti organizace.

Klíčová slova: bezpečnost, síť, monitoring, NIDS, Suricata, Snort, optimalizace, detekce hrozeb, Nagios

Detection of Network Security Threats

Abstract

This bachelor's thesis is focused on the analysis, design, and implementation of IDS into the corporate network. The first part is concentrated on the theoretical introduction to the problematics of computer networks. It continues with a preface of examples for network monitoring, which are currently available. There is presented their functionality and license usage. From the theoretical part of the work is the selected optimal system for the chosen company. The next part of the thesis is devoted to the implementation of the chosen IDS. There is shown procedure of installation, configuration and work with selected IDS. Monitoring and optimization are made for an elected system. In the final part of the work, there is a recognition of the benefits of selected tools for the network.

Keywords: security, network, monitoring, NIDS, Suricata, Snort, optimization, detection of threats, Nagios

Obsah

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

IDS Intrusion Detection System
IPS Intrusion Prevention System
OSI Open Systems Interconnection
LLC Logical Link Control
MAC Media Access Control
UDP User Datagram Protocol
TCP Transmission Control Protocol
TTL Time to live
MTU Maximum transmission unit
NIC Network interface controller
VLAN Virtuální LAN
SPAN Catalyst Switched Port Analyzer
RSPAN Remote SPAN

1 Úvod

Síťová infrastruktura tvoří nedílnou součást každé organizace. Bez komunikace uvnitř organizace a se světem kolem se dnes žádná firma neobejde. Počítačová síť umožňuje uživatelům komunikovat či sdílet informace po celém světě.

S růstem těchto technologií také rostou útoky v počítačových sítích. Útočníci napadají systémy a snaží se získávat citlivá data organizací a uživatelů. Kybernetické útoky jsou čím dál tím více sofistikovanější a obyčejný firewall na ně již nestačí. Je nutné neustále rozpoznávat nové hrozby a přizpůsobit jím úrovně zabezpečení. Existuje mnoho způsobů, jak těmto útokům předcházet, detekovat je a dále řešit. Pro prevenci a detekci existuje řada specializovaných softwarových i hardwarových řešení. Jedním z nich jsou systémy pro detekci a prevenci hrozeb (dále jen IDS a IPS). Tyto systémy umožňují velmi detailní analýzu síťového provozu skenováním hlavičky a obsahu paketů. Tím, na rozdíl od jiných řešení, dokáží detekovat i sofistikovanější útoky, nebo různé zranitelnosti systémů v síti, či podezřelé chování uživatelů.

Nielsenův zákon [1] tvrdí, že kapacita linky roste každý rok o padesát procent. Je důležité, aby IDS systémy dokázaly stíhat analyzovat tento rostoucí provoz. Jakmile systém nebude stíhat pakety analyzovat, tak je začne zahazovat a systém začne být neefektivní.

Tato práce se zabývá porovnáním systémů pro detekci hrozeb a následným nasazením v organizaci GEM System a.s. (dále jen Organizace), která poskytla pro tuto práci server HPE ProLiant DL320e Gen8 v2. Systém bude nasazen na tento server a bude na něj nastaveno zrcadlení veškeré komunikace v síti na úrovni přepínače. Dále bude vytvořen monitoring a optimalizace nasazeného systému.

2 Cíl práce a metodika

2.1 Cíl práce

Hlavním cílem bakalářské práce je implementace systému pro detekci bezpečnostních hrozeb v síti pro zvolené firemní prostředí. Mezi dílčí cíle patří charakterizace IDS a jejich porovnání. Dalším cílem je návrh vybraného systému a jeho optimalizace. Na základě všech poznatků je vypracován závěr.

2.2 Metodika

Teoretická část je založena na studiu zdrojů a analýze řešené problematiky. V první kapitole Počítačová síť je nastíněna problematika počítačových sítí, který je základem pro pochopení funkcionality IDS. V následující kapitole IDS bude pojednávat o těchto systémech. Dále budou charakterizovány jednotlivé systémy a jejich komparace.

Po dokončení teoretické části bude vybrán IDS systém. Implementace vybraného IDS bude následně zpracována v druhé, praktické části práce. Ta je vypracována v kapitole Implementace. V následující kapitole Monitoring IDS bude realizován monitoring vybraného systému pomocí nástrojů Nagios a Grafana. Ze získaných dat bude provedena optimalizace implementovaného systému v kapitole Optimalizace. Na základě získaných poznatků bude provedeno hodnocení a závěr.

3 Teoretická východiska

3.1 Počítačová síť

Počítačová síť je propojení výpočetní techniky za účelem sdílení prostředků a poskytování služeb [2]. Nejzákladnější definicí je propojení uzlů pomocí linek. Tyto uzly komunikují dle předem definovaných protokolů [3]. Počítačové sítě jsou nejčastěji založeny na rodině protokolů TCP/IP a jsou hierarchicky děleny na části zvané subnety.

3.1.1 Model ISO/OSI

Referenční model ISO/OSI představuje síťový model počítačové sítě. Rozděluje síť do sedmi vrstev. Tyto vrstvy a služby na nich dále popisuje. Usnadňuje tak pochopení komunikace a specializace konkrétních vrstev. Vrstvy byly vytvořeny tak, aby počet interakcí přes rozhraní mezi vrstvami byl co nejmenší. Komunikace probíhá od aplikační vrstvy k fyzické. Každá vrstva přidává údaje navíc a zapouzdřuje stávající. Spodní tři vrstvy jsou zaměřeny na přenos dat. Horní tři vrstvy se orientují na aplikace a prostřední vyrovnává rozdíly mezi nimi. [2]

Fyzická vrstva specifikuje fyzickou komunikaci a fyzické propojení komunikujících stran. Zabývá se vlastnostmi přenosových médií a signálů. Definuje napěťové úrovně, převádí data na signál a zajišťuje sériový přenos na úrovni bitů.

Linková vrstva plní funkci přenosu bloku dat mezi sousedními uzly. Řídí přístup k přenosovému médiu. Zajišťuje také zahajování, průběh a také ukončení spojení na lince. Na této vrstvě jsou switche, které pracují s rámci. Na této vrstvě se také používá MAC adresa. Linková vrstva obsahuje dvě podvrstvy – Logical Link Control (dále jen LLC) a Media Access Control (dále jen MAC). LLC pracuje se síťovou vrstvou. Do rámce vkládá údaj o tom, který protokol sítové vrstvy je zapouzdřen v rámci. MAC sousedí s fyzickou vrstvou. Pomocí hardwarově definovaných operací řídí přístup k médiu. Zajišťuje adresování a úpravu dat dle požadavků na přenos signálu po médiu. [3]

Síťová vrstva zajišťuje komunikaci mezi zdrojových a cílovým hostitelem. Využívá fragmentace pro rozdělení zprávy. Routery umožňují směrování, které vyhledává nejlepší cestu pro paket. Tato vrstva zavádí IP adresu, která slouží k jednoznačné identifikaci síťového rozhraní v síti.

Transportní vrstva zajišťuje přenos dat mezi procesy operačního systému vzdálených uzlů. Pracuje se segmenty. Vyšší vrstvě poskytuje kvalitnější přenosové služby. Používá dva transportní protokoly – TCP a UDP. Protokol UDP umožňuje nespolehlivý a nespojovaný přenos dat v síti. Negarantuje tak přenos dat, ale je velmi rychlý. Naproti tomu protokol TCP je spojovaný a spolehlivý. Garantuje doručení zprávy, tím že po každé zprávě vyžaduje potvrzení o přijetí. Tím je také pomalejší. Aplikaci si tak mohou vybrat mezi těmito protokoly. Dalším důležitým aspektem transportní vrstvy jsou porty. Porty se používají pro rozlišení jednotlivých služeb v rámci jednoho uzlu [2] [4].

Mezi aplikační vrstvy modelu TCP/IP se řadí relační prezentační a aplikační. Model TCP/IP je spojuje do jedné. Relační vrstva vytváří relace, což jsou časové intervaly, kdy probíhá komunikace mezi aplikacemi. Rozhoduje, kdy se v rámci relace smí přenášet data. Prezentační vrstva zajišťuje reprezentaci, šifrování a kompresi dat. Aplikační vrstva poskytuje prostředky pro propojení aplikací. Předepisuje způsob komunikace mezi nimi. [4]

3.1.2 **Model TCP/IP**

Model TCP/IP obsahuje pouze čtyři vrstvy, zatímco referenční model ISO/OSI jich vymezuje sedm. Některé vrstvy modelu ISO/OSI se téměř nepoužívaly a model TCP/IP je sloučil. Nižší počet vrstev a menší složitost jsou jedny z důvodů, proč se v dnešní době preferuje tento model. Vrstva síťového rozhraní obsahuje fyzickou a linkovou vrstvu ISO/OSI modelu. Síťová a transportní vrstva zůstala zachována. Aplikační vrstva modelu TCP/IP sloučila relační, prezentační a aplikační vrstvy. [2]

3.1.3 **IP adresace**

IP adresa je logická adresa uzlů v počítačové síti. Nachází se na třetí vrstvě OSI modelu. V IPv4 je adresa velká 32 bitů. Dělí se do čtyř dekadických hodnot, každá o velikosti jednoho bytu. Tyto části jsou označovány jako oktety a oddělují se tečkou [3].

Mezi další část IP adresy patří prefix neboli identifikátor sítě. Prefix sítě jde určit pomocí masky sítě. Maska je číslo, které udává velikost sítě. Lze ji zapsat jako číslo z rozsahu 0–32, nebo jako specifickou IP adresu. Číselná reprezentace udává, kolik bitů z levé strany má hodnotu 1. Tyto bity jsou určeny pro podsíť a zbylé bity, které mají hodnotu 0 jsou použity pro adresaci uzlů. Síťová část IP adresy je použita pro směrování. Maska se v IPv4 zapisuje stejně jako IP adresa pomocí oktetů. Za validní adresy se považují pouze ty, které mají v binárním zápisu z levé strany jedničky následované nulami. Příkladem síťové masky může být 255.255.255.0 [5].

Výchozí brána je označení ve směrovací tabulce pro cestu paketu do jiného subnetu. V případě, kdy cílová IP adresa neodpovídá žádnému záznamu ve směrovací tabulce je paket poslán do výchozí brány. Ta je umístěna na posledním místě v tabulce.

3.1.4 **Paket**

Paket je základní přenosovou jednotkou v sítích. Vyskytuje se na třetí vrstvě modulu ISO/OSI. Skládá se z hlavičky a dat. Struktura hlavičky IPv4 paketu je vidět v tabulce 1. Obsahuje údaje potřebné pro doručení paketu k příslušnému cíli. Doručení paketu zajišťuje router, který pracuje na základě IP adres. Hlavička mimo jiné také obsahuje limit TTL, který omezuje dobu platnosti dat. Tato hodnota se snižuje při každém průchodu routerem. Za hlavičkou je obsah paketu, který obsahuje data. [2] [4]

<table>
<tr><td>Verze</td><td>IHL</td><td>Typ služby</td><td colspan="2">Délka</td></tr>
<tr><td colspan="3">Identifikátor</td><td>Příznaky</td><td>Offset fragmentu</td></tr>
<tr><td colspan="2">TTL</td><td>Protokol</td><td colspan="2">Kontrolní součet hlavičky</td></tr>
<tr><td colspan="5">IP adresa odesílatele</td></tr>
<tr><td colspan="5">IP adresa příjemce</td></tr>
</table>

Tabulka 1 - Hlavička paketu. Zpracováno dle [3]

3.1.5 **Rámec**

Ethernetový rámec označuje přenášenou jednotku na linkové vrstvě (L2) referenčního module ISO/OSI. Maximální velikost je dána pomocí maximální délky rámce (dále jen MTU). Standartní velikost rámce se pohybuje mezi 46–1500 byty. Každý rámec má hlavičku, která obsahuje zdrojovou a cílovou MAC adresu a typ či délku. Typ určuje použitý protokol vyšší vrstvy. Rámci předchází preambule a oddělovač začátku rámce, které oddělují začátek rámce. Na konci je poté kontrolní součet rámce, který umožňuje detekovat poškozené rámce. [2] [4]

3.2 **Systém pro detekci hrozeb**

Nejzákladnější obranou v každé síti je firewall, který slouží jako první linie obrany. Firewall je aplikace, která slouží k řízení komunikace mezi systémy za účelem ochrany před nežádoucími přístupy prostřednictvím TCP/IP protokolů na základě informací z hlaviček protokolů vyšších vrstev. Filtruje datové jednotky jednotlivých datových proudů podle předem definovaných pravidel. Firewally se mohou vyskytovat na koncových i mezilehlých uzlech jako ACL. Samotný firewall však k obraně sítě nestačí. V dnešní době stále více útoků využívá nedostatků a chyb v konkrétních programech. Tyto útoky jsou pak z pohledu firewallu skryté v normální

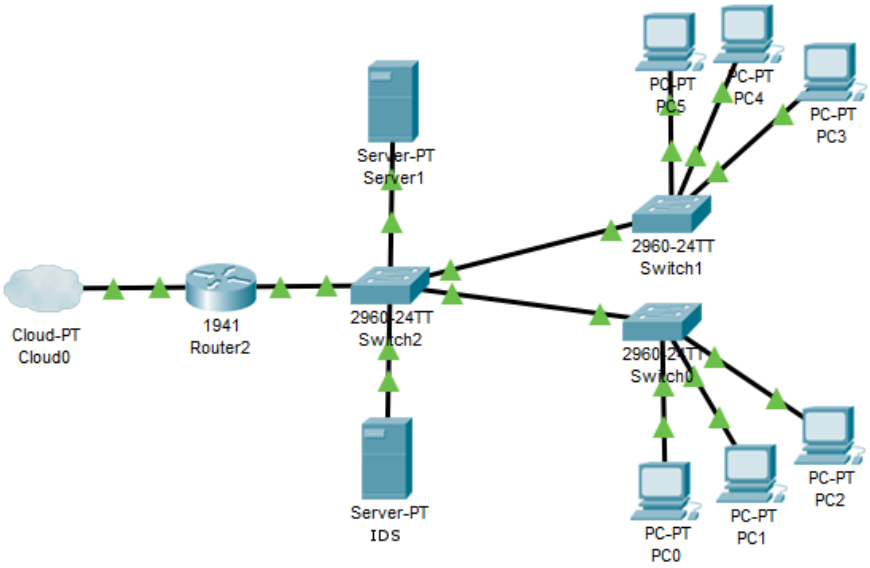
komunikaci. Právě proto ho doplňují IDS, či IPS systémy. V sítích mohou být také nastraženy Honeypoty, dle kterých se dá zjistit, co útočník použil za metody útoku.

IDS je specializovaný nástroj, který monitoruje podezřelou aktivitu v síti. Dokáží číst logy uzlů v síti, analyzovat jejich komunikaci a dále je vyhodnocovat. Tyto systémy většinou obsahují databázi pravidel či signatur známých útoků a jsou schopny je porovnávat s aktivitou v síti, nebo s logy připojených zařízení. Pokud najdou shodu, tak vypíší upozornění. Musí tak být aktualizována pravidelně databáze pravidel, aby systém mohl detekovat nejnovější typy útoků. Sofistikovanější IPS systémy mohou provést další akce od vypnutí linky, či vypnutí postiženého zařízení po zahájení různých protiopatření a dalšího sbírání důkazů. IDS systémy tyto aktivity pouze zaznamenávají. Vyžadují, aby se správce sítě podíval do záznamů a určil další postup. Podle definovaných pravidel pak vypisuje upozornění o podezřelé aktivitě v síti. Tyto pravidla poskytují různé organizace a mohou se jednoduše doplnit či upravit. Také dle jejich definice může vznikat mnoho falešně pozitivních upozornění neboli planých poplachů. Je pak na správci sítě, aby je prověřil a případně upravil pravidla. Počet planých upozornění závisí na mnoho proměnných. Mezi ně patří velikost sítě, počet serverů, uživatelů, pravidel a definice sítě.

Existuje velké množství IDS systému. Mezi nejzákladnější dělení patří síťové a koncové. Koncové IDS monitorují záznamy systémů a zjišťují, zda nedošlo k neoprávněnému vniknutí. Síťové IDS zkoumají komunikaci v síti, dle předem připravených pravidel. Pokud komunikace odpovídá pravidlu, tak systém vypíše upozornění [6]. Standartně nastavená síťová karta zahazuje vše, co není určeno přímo pro ni. Toto se rozlišuje dle MAC adresy. Aby NIDS mohl analyzovat komunikaci mezi ostatními uzly, je třeba, aby jeho síťová karta operovala v tzv. promiskuitním módu. Tím dokáže přijmout komunikaci, která jde i na jinou MAC adresu. V tomto nastavení dokáží NIDS odposlouchávat veškerou komunikaci, která na něj přijde. NIC pak nezahazuje tyto pakety, ale předává je dále systému [6] [7].

Umístění NIDS systému v sítí záleží dle toho, co je třeba monitorovat. Na obrázku 1 je vidět možné základní umístění NIDS v nákresu malé sítě. NIDS je zde umístěn hned za firewallem, který může fungovat také jako router. Systém pro analýzu komunikace je zapojen do switche, který replikuje komunikaci na port kde je zapojen. Systém je pak schopný monitorovat komunikaci z a do internetu ze serverů a také z ostatních koncových zařízení. Pokud je v síti použita technologie VLAN, tak je také analyzována, jelikož je třeba pro její směrování router. Tím je možno zajistit velké pokrytí v síti, kdy je pokryto připojení do internetu a přístup na důležité servery. V případě, že uživatelé a servery jsou v rozdílném subnetu. Pak je komunikace

mezi těmito subnety směrována přes router. Tím je pak docíleno kopírování interní komunikace v síti. Toto umístění však nemonitoruje komunikaci mezi koncovými uzly ve stejném přepínači a ve stejné VLAN. Tyto komunikace lze pokrýt umístěním jednotlivých sond na každém přepínači a poté komunikaci analyzovat na těchto uzlech, či je dále posílat do jednoho systému s větší výpočetní kapacitou pro analýzu. Dále je možné použít technologii RSPAN, která dovoluje poslání komunikace na předem definovaný port na jiném uzlu. Tato technologie není dostupná na všech přepínačích.

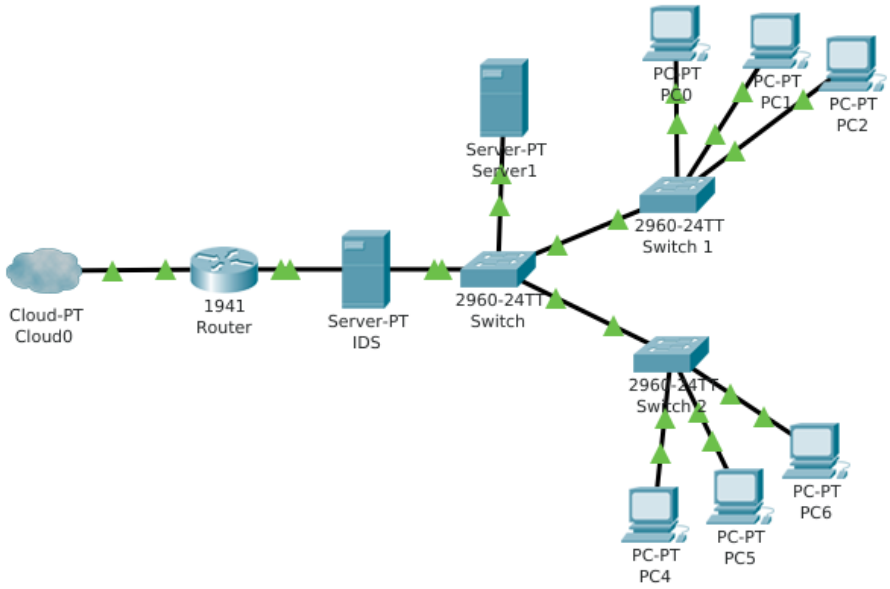


Obrázek 1 - Příklad umístění IDS v síti. Zdroj: vlastní zpracování

3.3 Systém pro prevenci hrozeb

Systém pro detekci a prevenci průniku je systém, který monitoruje komunikaci v reálném čase v počítačové síti s cílem prevence škodlivé činnosti. Mezi hlavní činnosti IPS systémy patří identifikace škodlivé komunikace a následné blokování této komunikace. Touto činností rozšiřují IDS systémy. Jsou zařazeny rovnou do síťové cesty, obdobně jako firewall. Pokd je detekována škodlivá komunikace, tak je ihned zahozena. Tím je komunikace zablokována a není puštěna dále do sítě. IPS systémy jsou aktivním prvkem v síti. Na rozdíl od IDS systémů, které pouze detekují hrozby a jsou pouze pasivním prvkem v síti. Na obrázku 2 je vidět možné umístění systému v jednoduchém návrhu sítě. Systém je umístěn jako aktivní prvek mezi hraničním routerem a přepínačem, kde filtruje komunikaci. Všechna komunikace do a z internetu tak musí projít přes tento systém, než se dostane do defaultní brány. Zároveň je před systémem umístěn firewall, který blokuje většinu útoků z internetu. IPS pak analyzuje

prošlou komunikaci, kterou firewall nechá projít. I v tomto případě, je komunikace mezi VLAN analyzována.



Obrázek 2 - Příklad umístění IPS v síti Zdroj: vlastní zpracování

IPS systémy využívají stejně jako IDS systémy databázi pravidel neboli signatury. Dále je mohou doplňovat o další sofistikovanější způsoby detekce. Tyto systémy mohou dle průměrné tendence v síti vypracovat obraz a poté porovnávat provoz proti tomuto obrazu. Například pokud se uživatel najednou přihlásí z jiné země, nebo jestliže se u něho podezřele zvýšil datový tok. Pokud by tyto komunikace byly mimo stanovené hranice, tak IPS systémy provedou předem definovanou akci. Toto může být užitečné při útocích typu odmítnutí služeb [8]. Tyto systémy pak mohou využít mnoho způsobů pro odvrácení útoků, či podezřelé komunikace. Velkou nevýhodou těchto systémů jsou falešně pozitivní upozornění, stejně jako IDS. Na rozdíl od nich je komunikace dále rovnou zahozena a může způsobovat potíže ve společnosti, kdy pak komunikace nemusí dorazit v pořádku nebo některá služeb v síti není dostupná. Vyžadují tak mnohem větší správu v rámci pravidel. Systémy pracující na úrovni signatur mají obvykle méně falešně pozitivních upozornění než systémy pracující na anomáliích. [7] Pravidla se dají předem přesně nadefinovat a správce sítě nad nimi má mnohem větší kontrolu.

3.4 Typy NIDS

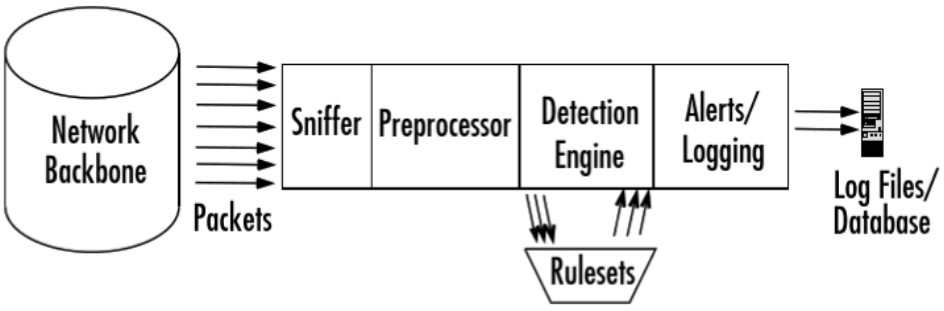
Existuje mnoho specializovaných zařízení a softwarových řešení pro monitoring sítě. Jedním z nejdůležitějších požadavků těchto systému je výkon. Tyto systémy musí zpracovávat komunikaci v síti v reálném čase. Pokud tuto komunikaci nebudou stíhat, tak ji začnou

zahazovat a vzniká riziko, že nedetekují podezřelou aktivitu, nebo že bude část komunikace chybět. Dalším požadavkem je cena. Pod jakou licencí systém vznikl a pokud není volně dostupný, tak zda nabízí atributy navíc, aby se vyplatil. Posledním důležitým aspektem je podpora. Zda má systém aktivní vývojářskou podporu a je stále vyvíjen. To samé platí pro jeho pravidla. Aktivní správa pravidel je klíčová pro správnou detekci podezřelých aktivit.

3.4.1 **Snort**

Snort patří mezi nejznámější a nejpoužívanější NIDS/IPS systémy. Je schopen pracovat, jak v IDS režimu, tak i jako IPS. Jeho vysoká stabilita a otevřenost tomuto softwaru zajistily vysoký podíl na trhu. Dá se používat téměř na každém operačním systému. Může být nainstalován na různé distribuce Linuxu, nebo i na Windows. Je volně dostupný pod všeobecnou veřejnou licencí GNU [9].

Tento systém se skládá z několika částí, které jsou vidět na obrázku 3. První částí je paketový analyzátor, který odposlouchává komunikaci a připravuje pakety pro preprocesor. Analyzátor může být hardwarový, nebo softwarový. Tato část získává pakety z linky. Další částí je preprocesor, který kontroluje pakety. Používá se k úpravě a přeuspořádání paketů. Snaží se detekovat jednoduché chyby v hlavičce paketů. Je důležitou součástí každého IDS systému pro přípravu paketů k budoucí analýze. Třetí částí je detekční blok porovnávající pakety s uloženými pravidly. Tato část je největší částí systému. Její účel je analýze všech procházejících paketů dle uložených pravidel. Pokud najde shodu, tak použije definovanou akci v pravidle. Pokud shodu nenajde, tak paket zahodí. Tato část je nejvíce náročná a trvá nejdelší dobu. Délka záleží na počtu a složitosti pravidel. Poslední část obstarává zaznamenání do databáze.



Obrázek 3 - Architektura systému Snort. Zdroj [7]

Snort vznikl v roce 1998 jako síťový analyzér. Později byl upraven na IDS systém. Společnost Sourcefire, který Snort vyvíjela, byla v roce 2013 odkoupena společností Cisco [10]. Tento systém se stal tak jedním z průkopníků IDS systémů, díky jeho velké komunitě.

Jeho hlavní nedostatek vychází z toho, jak je tento systém starý. Snort je pouze jedno vláknový. To znamená, že pro analýzu paketů využívá pouze jedno jádro. Toto před dvaceti lety nebyl problém, ale v dnešní době má téměř každý procesor minimálně 4 vlákna a serverové procesory dosahují 64 vláken. Toto se mění ve verzi 3.0, kdy je přidána podpora více vláken. Verze je momentálně v beta verzi a datum vydání stabilní verze není znám.

Pro analýzu paketů využívá signatury neboli pravidla. Pravidlo se skládá ze dvou částí, hlavičky a obsahu. V tabulce 2 je dále vidět rozdělení hlavičky pravidla. První je akce, co se má pro dané pravidlo stát. Dále je použitý protokol, zdrojová a cílová adresa a port. V obsahu pravidla se dále definuje vzor chování či obsah paketu.

Akce	Protokol	Zdrojová IP	Zdrojový port	Směr	Cílová IP	Cílový port

Tabulka 2 - Složení pravidla pro systém Snort. Zpracováno dle [7]

Snort umožňuje použití proměnných v pravidlech. V konfiguračním souboru se může nastavit proměnná dle IP adresy a masky sítě. Jednoduché pravidlo pro vypsání každého navštívení stránky z interní sítě by vypadalo následovně:

```
alert tcp $HOME_NET any -> any 443 (msg: “uis”; content: is.czu.cz; sid:10001)
```

Systém Snort pro každý paket projde všechna pravidla. Paket je porovnávám postupně s pravidlem z levé strany na pravou. Po kontrole IP adres, protokolů a portů přejde Snort na porovnání obsahu paketu. Pokud je v pravidle definováno, mimo jiné, pole content, tak Snort použije Boyer – Moore algoritmus. Tento algoritmus je velice rychlý pro hledání určitého textu. Obsah může být uložen jako http nebo také v binární podobě [11]. Pokud najde shodu, tak paket předá do další části pro zaznamenání. [7]

Syntaxe pravidel je relativně jednoduchá pro pochopení a další vytvoření pravidel. Dají se vytvářet mnohem složitější pravidla pro odchycení každého paketu [12]. Oficiální pravidla pro Snort jsou dostupná z webových stránek snort.org. Existuje mnoho dalších serverů, které nabízí pravidelně aktualizovaná pravidla. Pro pravidelnou aktualizaci pravidel je dostupný nástroj Pulled Pork, který byl vytvořen v programovacím jazyku Perl. Tento nástroj umožňuje automatickou aktualizaci pravidel z předem definovaných serverů. Uvedený systém nemá žádné uživatelské rozhraní. Spoléhá na software třetích stran, jako například Squil nebo Snorby.

3.4.2 **Suricata**

Suricata je dalším NIDS/IPS systémem. Je architekturou podobná Snortu, ale nejzásadnější rozdíl je, že je více vláknová. Tím dokáže využít všechny vlákna procesoru pro vyšší výkon. Zvyšuje tak počet přijatých paketů a zmenšení šance ignorování paketů kvůli omezené kapacitě. Systém Suricata byl navržena tak, aby doplnil nedostatky systému Snort ve více jádrových systémech. Využívá stejnou syntaxi pravidel. Je tedy možné používat stejná pravidla jako systém Snort. Má již zabudovaný nástroj pro automatickou aktualizaci pravidel. Spoléhá také na podporu třetích stran pro uživatelské rozhraní. Používá standardizovaný výstup, takže lze použít více nástrojů pro zobrazení zachycených upozornění. Mezi nejpopulárnější rozhraní patří SELK. Toto rozhraní má velké nároky na operační paměť, ale nabízí mnoho možností a je velmi rychlé. Mezi další podporované rozhraní patří také Snorby. Systém umožňuje akceleraci pomocí GPU pro monitoring vysokorychlostních linek. Umožňuje také uložení zachycené komunikace k pozdější inspekci. [13] Dále nabízí mnohem více možností optimalizace a škálovatelnosti pro analýzu vysokorychlostních linek.

První stabilní verze vyšla v roce 2010 a je stejně jako Snort dostupná pod všeobecnou veřejnou licencí GNU [14].

3.4.3 **Zeek**

Zeek je otevřený software pro monitoring počítačové sítě. Původně známý pod názvem Bro odvozeného z románu George Orwella 1984 kde se vyskytuje pojem Big Brother. Momentálně je vydán pod BSD licencí [15].

Systém může pracovat jako klasický systém pro detekci bezpečnostních hrozeb v síti dle signatur, ale může být dále doplněn o další funkcionality. Mezi ně patří detekce dle anomálií v síti či detekce změn v chování uzlů. Systém obsahuje nástroje pro prohlížení detekcí, ale také podporuje další i nástroje. Hlavní rozdíl oproti systémům Suricata a Snort je použití statistik. Systém Zeek dokáže monitorovat komunikaci pro jednotlivé hosty, či porty a pokud tato komunikace přesáhne určitou hranici, tak je vyhodnocena jako nález. Tento systém je speciálně tvořen pro monitoring vysokorychlostních linek, které mohou dosahovat až 100 GE. [15] V dokumentaci nejsou uvedeny minimální hardwarové požadavky pro tento systém. Jsou zde pouze vypsány softwarové požadavky. Z dokumentace tak není jasné, zda na systém v této práci je možné nainstalovat tento systém.

3.4.4 Cisco Stealthwatch

Stealthwatch je systém pro detekci bezpečnostních hrozeb v sítí, od společnosti Cisco. Pro monitoring komunikace používá strojové myšlení. Monitoruje činnosti uživatelů a jejich zařízení v síti. Dokáže pracovat s ostatními uzly v sítí a v případě průniku izolovat i určité segmenty. Systém dokáže analyzovat zašifrovanou komunikaci. Obsahuje také plné uživatelské rozhraní pro kontrolu upozornění. Stealthwatch je dostupný pod licencí od společnosti Cisco. Jeho cena se odvíjí od velikosti sítě a počtu uživatelů. [16]

3.5 Aktuální stav

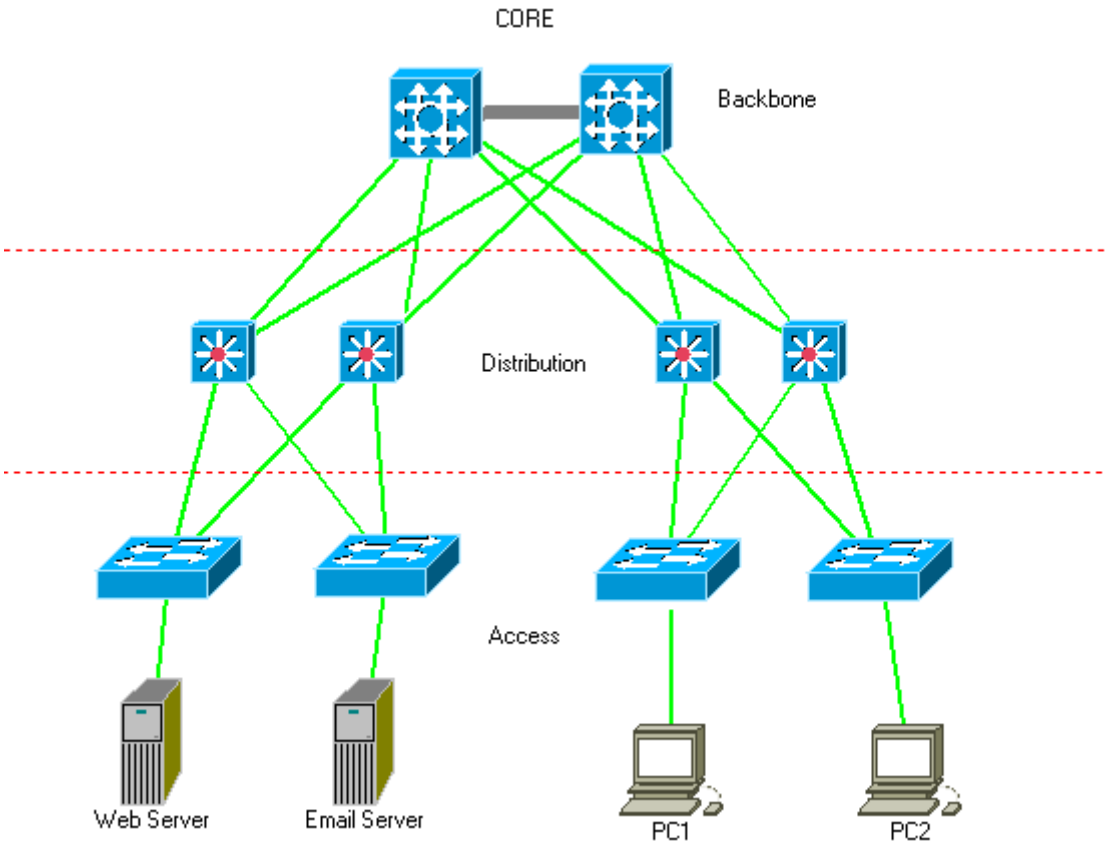
V organizaci byl dříve implementován IDS Snort, ale server, na kterém systém běžel, přestal fungovat. Systém Snort fungoval s podpůrným programem Barnyard2 a webovým rozhraním Snorby. Organizace do tohoto rozhraní přidala několik funkcionalit a vyžaduje jeho další použití v implementaci nového IDS. Společnost poskytuje internetové připojení pro více než sedmdesát pracovníků a nutně potřebuje řešení pro monitoring počítačové sítě. Dostala už od poskytovatele internetu upozornění, že bylo zjištěno stahování obsahu přes nástroj BitTorrent. Společnost momentálně nemá, jak zjistit, který uživatel stahuje, jaký obsah a navštěvuje jaké stránky. Společnost také začala pracovat na certifikaci ISO/IEC 27001 a 27002, kterou vyžadují klienti při výběrových řízení.

Implementace IDS v této společnosti dokáže detekovat stahování obsahu přes P2P a také stahování dalšího obsahu pro které se mohou vytvořit pravidla. Pro certifikaci ISO, zvláště v bodu A.12.2.1, je dále klíčová detekce hrozeb v síti a detekce navštívení podezřelých škodlivých stránek.

4 Vlastní práce

Praktická část této práce se podrobně zabývá implementací a optimalizací systému pro detekci bezpečnostních hrozeb v síti ve středně velké společnosti. Společnost pro tuto práci poskytla server HP Proliant dl320e gen8 v2. Server je zapojen do sítě pomocí ethernetového kabelu rychlostí 1 Gbps. Řešení je realizováno na linuxové distribuci CentOS 7.

Síť společnosti se skládá ze tří vrstev. Server je zapojen do přepínačů, v core vrstvě, pomocí ethernetové kabeláže. Do těchto přepínačů je také zapojen router. Na switchích byla nastavena metoda SPAN pro zrcadlení portu, který vede do routeru neboli výchozí brány. Touto metodou je docíleno zrcadlení veškeré komunikace, která vede do nebo z routeru na server kde bude nainstalován vybraný systém IDS. To obsahuje komunikaci na internet a také mezi VLAN, softwarového rozdělení sítě. Server tak bude analyzovat většinu komunikace v síti společnosti. Tímto způsobem není pokryta komunikace mezi uzly ve stejném síťovém subnetu. Tento přenos dat neprochází přes router a nedostane se tak do serveru k analýze. Tato komunikace byla označena jako nepodstatná, jelikož nejdůležitější část spojení je na produkční servery v jiném subnetu nebo je z internetu. Na obrázku 4 je vidět schéma zapojení dle třívrstvé architektury Cisco.



Obrázek 4 - Třívrstvý model sítě. Zdroj [17]

4.1 Výběr systému

Pro praktickou část této práce byl vybrán systém Suricata. Mezi kritéria pro výběr patřilo:

- Volné užití
- Optimalizace pro analýzu vysokorychlostních linek
- Pravidelné aktualizace pravidel
- Softwarová podpora

Mezi kritéria organizace patřilo, aby systému fungoval na operačním systému CentOS 7 a také aby měl podporu webového rozhraní Snorby.

Konečný výběr byl mezi systémy Suricata a Snort. Obě řešení odpovídají požadavkům organizace. Systémy jsou si velice podobné. Mají otevřený kód a velkou komunitní a vývojářskou podporu. Vývojářská podpora znamená, že systém je pravidelně aktualizován. Komunitní podpora je také velmi důležitá. V případě problému je vyšší šance, že se dá najít řešení online a že se s touto chybou již někdo setkal a vyřešil. Není nutné případné chyby řešit

s podporou autorů systému. Řešení chyb je tak mnohem jednodušší pro systém, který používá více uživatelů. Zásadní rozdíl mezi těmito systémy je podpora více vláken neboli multithreading. Systém Suricata má tuto podporu a dokáže tak analyzovat více komunikace. Pro systém Snort má tato funkcionalita přibýt ve verzi 3.0, která je již několik let v beta testování [18]. Pro produkční nasazení není žádoucí nasadit software v nestabilní verzi. Server kde se bude systém implementovat, má procesor Intel Xeon CPU E3-1231 v3 3.40GHz, který má osm vláken, proto je podpora více vláken pro tuto práci klíčová.

Systém Suricata podporuje mnoho možností výstupů, ale pro tuto práci byl vybrán výstup binární unified2, který používá systém Snort. Organizace tento systém v dřívější době používala a pro tento systém bylo vytvořeno webové rozhraní Snorby. Toto rozhraní získává data z databáze, ale tento systém neumí zapisovat do databáze, ale pouze využívá binární výstup. Pro uložení těchto dat do databáze vznikl program Barnyard2, který vezme binární data a zapíše je do databáze pro webové rozhraní. Organizace takto používala tyto programy.

Jelikož Suricata vychází ze Snortu, tak nabízí více možností ukládání výstupů. Alternativou je například systém SELK. Tento systém používá programy Elasticsearch, Logstash, Kibana a Scirius pro práci s výstupy IDS a zobrazení těchto dat. Systém je také dostupný jako ISO obraz na operačním systému Debian. Tyto programy jsou napsány v programovacím jazyku Java, který je velice náročný na operační paměť. Tvůrci Elasticsearch doporučují pro produkční nasazení, server s 64 GB operační paměti [19]. Server, který zapůjčila společnost má pouze 8 GB operační paměti. Je důležité nastavit co nejvíce paměti IDS pro efektivní analýzu komunikace. V tomto ohledu je program Barnyard2 a rozhraní Snorby lepší volbou, jelikož nemají tak velké nároky na operační paměť serveru.

V případě, kdyby měl server více operační paměti, tak by byl systém SELK zajímavější volbou. Tento systém je modernější, nabízí více možností a má větší podporu. Nicméně organizace požaduje, použijí Snorby a pro účely organizace a této práce je toto rozhraní dostačující.

4.2 Implementace

Instalace je prováděna na linuxové distribuci CentOS 7. Instalace systému je možná z repositáře, ze zdrojového balíku nebo ze systému Git. Nejjednodušší instalace je pomocí repositáře, kde nástroj yum obstará veškeré závislosti, které jsou třeba k instalaci. Tento nástroj také zjednodušuje aktualizaci systému a další správu.

Byla provedena kontrola aktuální verze na webových stránkách OISF, organizace spravující systém Suricata, a repozitáře a bylo zjištěno, že na stránkách je verze systému 4.1.4 a v repositáři je dostupná verze 4.0.1. Z tohoto důvodu byla zvolena metoda instalace ze zdroje. Touto metodou je možné nainstalovat nejnovější verzi systému a není třeba čekat, než bude přidána do repositáře. Tato metoda také umožňuje větší kontrolu nad instalací než ostatní metody. Tomto metodou lze určit, kam bude systém nainstalován a s jakými parametry.

Před samotnou instalací systému bylo třeba nainstalovat a do konfigurovat všechny potřebné nástroje a závislosti. Všechny požadované nástroje a závislosti byly nainstalovány pomocí příkazu yum, pro snadnější aktualizaci a správu. Dále bylo nutné upravit nastavení síťové karty. Přes příkaz ethtool bylo zjištěno aktuální nastavení síťové karty. Bylo zjištěno, že na síťové kartě je zapnuta technologie offload. Tato technologie musí být vypnuta, aby IDS dokázalo správně vidět komunikaci tak jak standartně probíhá [20]. Pomocí příkazu ethtool byla tato technologie na síťové kartě zakázána. Z této síťové karty bude dále IDS analyzovat komunikaci.

Systém byl zkompilován a nainstalován pomocí následujících příkazů:

```
tar xzvf suricata-4.1.4.tar.gz
cd suricata-4.1.4
./configure --sysconfdir=/etc --localstatedir=/var
make
make install
```

V příkazu configure bylo nastaveno, kde budou uloženy konfigurační soubory systému. Cesta v tomto případě byla /etc/suricata. Dále byla také nastavena cesta k log souborům - /var/log/suricata. V těchto souborech budou uloženy záznamy systému, statistiky a také uložená komunikace pro hlubší analýzu.

4.2.1 **Konfigurace IDS**

Před spuštěním systému bylo třeba upravit konfiguraci, aby správně fungoval, načetl požadovaná pravidla a analyzoval správné IP rozsahy. Změny konfigurace byly prováděny v konfiguračním souboru suricata.yaml, který se nachází v adresáři /etc/suricata.

Jedna ze základních věcí, co bylo třeba nastavit, jsou proměnné pro pravidla. Pravidla systému Suricata a Snortu používají ve své syntaxi proměnné, kde se definují IP adresy a protokoly. Čím je definice přesnější, tím je lepší přesnost pravidel a také lepší výkon systému. Do proměnné HOME_NET byly definovány rozsahy, které se ve vybrané organizaci používají. Proměnná

EXTERNAL_NET byla pak definována jako negace vnitřní sítě, tedy: !$HOME_NET. Dále byla definována proměnná DNS_SERVERS, kde byly zadány adresy DC serverů v organizaci. Další proměnné byly definovány jako domácí síť.

V další části konfiguračního souboru bylo zapnuto vypisování statistik pro zjištění efektivnosti systému. Ta bude měřena jako podíl mezi analyzovanými pakety a zahozenými. Tyto údaje budou dále rozepsány v kapitolách Monitoring systému a Optimalizace systému.

Pro výpis nálezů byl vybrán způsob unified2, který je potřebný pro požadované webové rozhraní Snorby. V konfiguračním souboru bylo dále nastaveno síťové rozhraní, ze kterého se má analyzovat komunikace. Server má dvě síťové rozhraní, jedno kde je nastaveno zrcadlení komunikace a druhé pro vzdálený přístup. Bylo vybráno pouze rozhraní, kde je zrcadlena komunikace. Rozhraní pro správu nebylo vybráno, jelikož pro přístup na toto rozhraní prochází komunikace z jiného subnetu a je tak analyzována na druhém rozhraní a útok ze stejného subnetu kde jsou produkční servery, se nepředpokládá.

V konfiguračním souboru byly definovány rozsahy IP adres specifické pro danou organizaci a nastaven výstupní formát pro nálezy IDS systémem. Tento výstup bude dále zpracován, aby byl dostupný ve webovém rozhraní. Konfigurace byla otestována pomocí argumentu -T. Tím byla provedena kontrola syntaxe a výpis případných chyb.

4.2.2 Instalace a konfigurace podpůrných systémů

Systém Suricata slouží pouze pro analýzu komunikace. Systém vyhodnocené nálezy uloží dle definovaných parametrů v konfiguračním souboru na disk. Aby mohl tyto data pracovník zpracovávat a vyhodnocovat, bylo třeba pro tento systém doinstalovat webové rozhraní.

Organizace požaduje použití rozhraní Snorby. Toto rozhraní používá MySQL databázi, ze které získává data a zobrazuje je uživatelům. Systém Suricata a Snort dokáží data ukládat přímo do databáze, ale je doporučeno používat externí program, aby se systém pro detekci hrozeb nezpomaloval připojováním a zapisováním do databáze. Tuto činnost je doporučeno nechat externímu programu. Proto je doporučeno nainstalovat program Barnyard2, který převede uložená data na disku do databáze rozhraní Snorby [21].

Rozhraní Snorby je vytvořeno v programovacím jazyku Ruby. Pro instalaci rozhraní je třeba nainstalovat Ruby, Rails a lokální databázi. Databáze byla nainstalována přes příkaz yum. Byla použita poslední verze MariaDB z repositáře. Dále byl z repositáře nainstalován Ruby. Po jeho

instalaci, byla spuštěna instalace Rails. Při instalaci se vyskytlo několik problémů se závislostmi, kdy bylo třeba několik závislostí ručně nainstalovat v jiné verzi. Také bylo třeba doinstalovat modul pro server Apache a vytvořit virtuální host kde bude rozhraní poslouchat. Po instalaci všech závislostí bylo rozhraní Snorby staženo ze serveru Git a nahráno na server. Byly upraveny dva konfigurační soubory pro nastavení, na jaké IP adrese bude rozhraní fungovat a také úprava připojení do databáze. Dále byla vytvořena lokální MySQL databáze a spuštěna instalace rozhraní. Instalace vytvoří databázové schéma a spustí rozhraní. Výsledné rozhraní Snorby je vidět na obrázku 5.

U služeb pro webový server a lokální databázi bylo nastaveno, aby byly spuštěny po zapnutí operačního systému v případě vypnutí či restartu serveru. V případě, že by byl server restartován a služby by nebyly spuštěny, tak by rozhraní nefungovalo.

Obrázek 5 - Úvodní stránka programu Snorby. Zdroj: vlastní zpracování

Byla prozkoumána možnost napojení rozhraní na Active Directory (dále jen AD) společnosti. Bylo zjištěno, že toto rozhraní neposkytuje možnost napojení do AD. V případě napojení by byla jednodušší správa uživatelů. V tomto případě se musí uživatelé vytvářet lokálně přímo pro rozhraní. Při instalaci byl automaticky vygenerován výchozí účet pro správu rozhraní. Po přihlášení na tento účet byl vytvořen jmenný administrátorský účet a výchozí účet byl zakázán z bezpečnostních důvodů.

Program Barnyard2 je dostupný na serveru Git. Pro jeho instalaci bylo třeba program zkompilovat pomocí příkazů configure a make. U příkazu configure byl přidán parametr databáze, do které se bude program připojovat. V případě této práce je to databáze MySQL. Po instalaci systému Barnyard2, byl upraven konfigurační soubor barnyard2.conf, kde byly upraveny parametry pro připojení k lokální MySQL databázi a adresář kam se ukládají nálezy z IDS. Dále byl upraven parametr pro síťové rozhraní. Program se spouští přes binární program a po spuštění běží na popředí, kde čeká na vstup uživatele. Tímto je blokována příkazová řádka a po odhlášení uživatele se program vypne. Toto chování je nežádoucí a je vyžadováno, aby tento program běžel neustále v pozadí. Pro tento program byla vytvořena služba, aby program běžel na pozadí jako démon a byl spuštěn při startu operačního systému.

Jeden z problémů byl odhalen u funkcionality rozhraní Snorby. U výpisu příkazu top bylo zjištěno, že existuje několik zombie procesů. Po další analýze se došlo k závěru, že tyto procesy vlastní rozhraní Snorby, a to konktrétně program wkhtmltopdf. Rozhraní umožňuje ve výchozím nastavení vytvářet pravidelné hlášení o upozornění a posílat je na email. Po objevení této chyby, byla tato funkcionalita vypnuta, aby se předešlo dalším zombie procesům. Stávající zombie procesy byly odstraněny pomocí restartu serveru Apache a Snorby. Tím, že byl zabit Parent proces, tak jsou automaticky zabiti také všechny Child procesy. V tomto případě se jedná o zombie procesy. Na tuto chybu je otevřen požadavek na serveru Git, ale zatím bez opravy.

4.2.3 **Nastavení a aktualizace pravidel**

Tato kapitola se bude zabývat nastavením pravidel a jejich automatické aktualizace. Pravidelná aktualizace bezpečnostních pravidel je velice důležitá, jelikož se každý den objevují nové bezpečnostní zranitelnosti. Automatická aktualizace pravidel bude nastavena na večerní hodinu, kdy ve společnosti nikdo již není a systém nebude vytížen.

Pravidla je možné stahovat a instalovat manuálně, ale je doporučováno používat na tuto činnost specializované nástroje. Systém byl nainstalován s nástrojem pro aktualizaci pravidel. Tento

nástroj se jmenuje suricata-update. Nástroj provede stažení pravidel z předem definovaných zdrojů do jednoho souboru, který bude definován v konfiguračním souboru suricata.yaml. Pomocí následujícího příkazu bylo provedeno stažení nejnovějších dostupných pravidel:

```
suricata-update --disable-conf /etc/suricata/disablesid.conf --local /etc/suricata/rules/
```

Parametrem local se definují lokální pravidla, která budou přidána do výsledného souboru. V souboru disableid.conf jsou definovány identifikátory pravidel, které budou zakázána. Ve výsledném souboru budou tak za komentována a systém je bude ignorovat.

Zdroje byly použity z výchozího nastavení, jak je vidět na obrázku 6. Zdroje s komerční licencí jsou v příkazu ignorovány a nejsou pro stažení použity.

```
root@cerberus:~# suricata-update list-sources
13/11/2019 -- 14:08:42 - <Info> -- Using data-directory /var/lib/suricata.
13/11/2019 -- 14:08:42 - <Info> -- Using Suricata configuration /etc/suricata/suricata.yaml
13/11/2019 -- 14:08:42 - <Info> -- Using /etc/suricata/rules for Suricata provided rules.
13/11/2019 -- 14:08:42 - <Info> -- Found Suricata version 4.1.4 at /usr/bin/suricata.
Name: oisf/trafficid
  Vendor: OISF
  Summary: Suricata Traffic ID ruleset
  License: MIT
Name: et/open
  Vendor: Proofpoint
  Summary: Emerging Threats Open Ruleset
  License: MIT
Name: scwx/security
  Vendor: Secureworks
  Summary: Secureworks suricata-security ruleset.
  License: Commercial
  Parameters: secret-code
  Subscription: https://www.secureworks.com/contact/ (Please reference CTU Countermeasures)
Name: scwx/malware
  Vendor: Secureworks
  Summary: Secureworks suricata-malware ruleset.
  License: Commercial
  Parameters: secret-code
  Subscription: https://www.secureworks.com/contact/ (Please reference CTU Countermeasures)
Name: et/pro
  Vendor: Proofpoint
  Summary: Emerging Threats Pro Ruleset
  License: Commercial
  Replaces: et/open
  Parameters: secret-code
  Subscription: https://www.proofpoint.com/us/threat-insight/et-pro-ruleset
Name: ptresearch/attackdetection
  Vendor: Positive Technologies
  Summary: Positive Technologies Attack Detection Team ruleset
  License: Custom
Name: sslbl/ssl-fp-blacklist
  Vendor: Abuse.ch
  Summary: Abuse.ch SSL Blacklist
  License: Non-Commercial
Name: tgreen/hunting
  Vendor: tgreen
  Summary: Heuristic ruleset for hunting. Focus on anomaly detection and showcasing latest engine features, not performance.
  License: GPLv3
Name: etnetera/aggressive
  Vendor: Etnetera a.s.
  Summary: Etnetera aggressive IP blacklist
  License: MIT
```

Obrázek 6 - Ukázka zdrojů pravidel IDS. Zdroj: vlastní zpracování

Tímto příkazem byl vytvořen nový soubor, kam byla vypsána všechna stažená pravidla. Systém Suricata ve výchozí instalaci obsahuje několik základních druhů pravidel. Tato pravidla analyzují základní typy útoků, ale také i chyby v komunikaci. Tyto pravidla byla v konfiguračním souboru zakázána a byla nastavena cesta k novému souboru s pravidly, který

byl vytvořen. Výchozí pravidla byla zakázána z důvodu, že nejsou aktualizována a pro systém byl vytvořen nový aktualizovaný zdroj. Není tak nutné mít v systému původní, dále neaktualizovaná pravidla. Každé pravidlo vytěžuje systém, jelikož IDS musí paket otestovat všemi pravidly. Čím více je tak pravidel, tím je systém pomalejší.

Suricata nabízí pro aktualizaci pravidel vlastní nástroj. Tento nástroj je však nutné pustit manuálně a není k dispozici žádná automatizace. Pro automatizaci a pravidelnou aktualizaci byl vytvořen skript.

Na začátku skriptu je zapsáno do logu datum startu. Poté je provedena aktualizace pravidel a výstup příkazu je také uložen do logu. Návratový kód je uložen pro vyhodnocení. Pokud se návratový kód rovná nule, tak je aktualizace vyhodnocena jako úspěšná a zapíše do dalšího souboru aktuální datum, které se použije při monitorování, zda aktualizace skončila správně. Poté jsou obě služby restartovány. U služeb je použito explicitně zastavení a zahájení. S použitím restartu u služeb se naskytl problém, kdy se špatně ukončily a běžely poté dvakrát. Na konci skriptu je zapsán čas, kdy skript skončil. Tento záznam slouží pro informaci administrátorovi, v případě, kdy by se vyskytl problém s aktualizací pravidel.

```
#!/bin/bash
log="/var/log/suricata/rules-update.log"
echo "Suricata rules update starting at `date`.." >> $log
suricata-update --sid-msg-map /etc/suricata/sid-msg.map --force --disable-conf
/etc/suricata/disablesid.conf --local /etc/suricata/rules/ >> $log 2>&1
updatestate=$?
echo "suricata-update exit code: $updatestate" >> $log
if [ $updatestate -eq 0 ]; then
  date +"%Y-%m-%d %T" > /var/log/suricata/lastokupdate;
fi
systemctl stop suricata  >> $log
systemctl stop barnyard2 >> $log
systemctl start suricata >> $log
systemctl start barnyard2 >> $log
echo "PulledPork update finished at `date`.." >> $log
```

Přínosem tohoto skriptu je pravidelná aktualizace pravidel o nejnovější bezpečnostní hrozby. Znovu načtení pravidel je provedeno přes cron v mimo pracovní dobu, kdy je systém nejméně zatížen a neanalyzuje komunikaci pracovníků organizace. Odstávka systému trvá řádově jednu minutu.

4.3 Spuštění systému

V operačním systému byla vytvořena služba suricata, ve které je spouštěn následující příkaz:

```
/usr/bin/suricata -c /etc/suricata/suricata.yaml -i eth1 --user suri -F /etc/suricata/bpf.filter
```

V příkazu je definována cesta ke konfiguračnímu souboru, který byl upraven pro potřeby této práce. Je také definováno, pod jakým uživatel se má IDS spustit a jaké síťové rozhraní má analyzovat. Je také definována cesta k BPF filtru, pro možnost ignorování komunikace.

Po spuštění služby, byl systém úspěšně spuštěn a začal analyzovat příchozí komunikaci na vybraném rozhraní. Spuštění systému bylo ověřeno přes status služby, jak je vidět na obrázku 7.

Obrázek 7 - Start systému Suricata. Zdroj: vlastní zpracování

Pro otestování, zda systém vidí pakety v pořádku, bylo vytvořeno následující pravidlo:

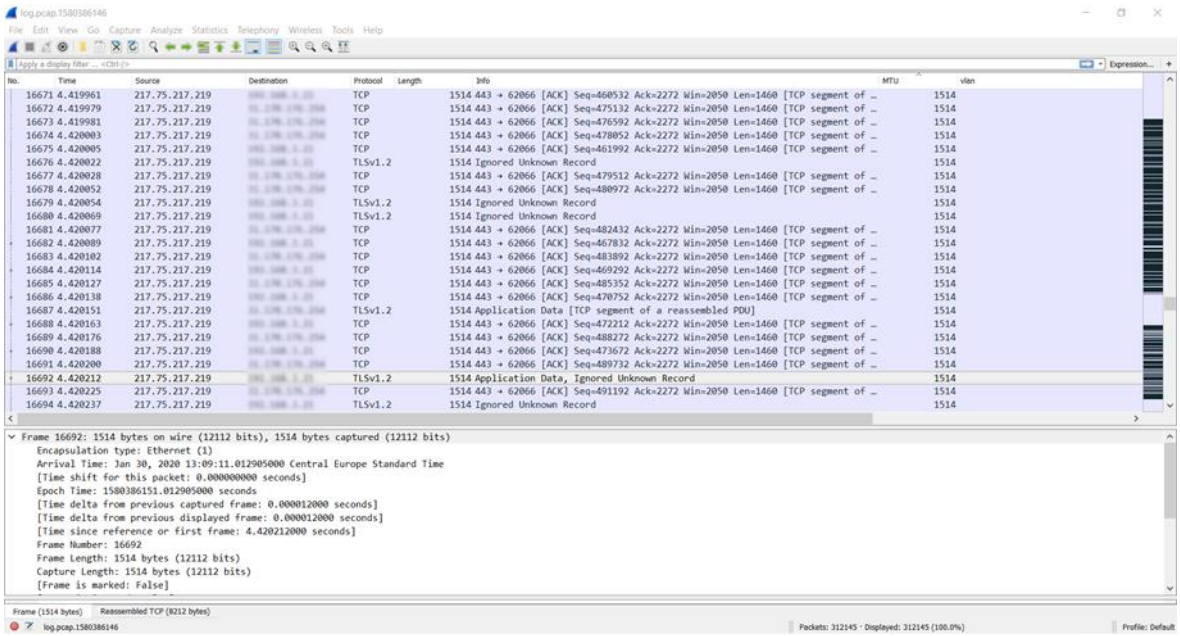
```
alert icmp $HOME_NET any -> 1.1.1.1 any (msg: "Testovaci ICMP alert"; sid:2100001; rev: 1;)
```

Pravidlo zachycuje komunikaci na server s adresou 1.1.1.1, což je veřejný DNS server od společnosti Cloudflare. Pravidlo bylo napsáno do souboru custom.rules v adresáři /etc/suricata/rules. Po úpravě pravidel byl spuštěn skript, pro aktualizaci pravidel. Skript provedl aktualizaci pravidel z internetu a program pro aktualizaci zahrnul i nově napsané pravidlo. Skript také provedl restart systémů pro znovu načtení nových pravidel. Po spuštění systému bylo provedeno ověření konektivity na adresu 1.1.1.1 pomocí programu ping. V systému tato komunikace nebyla zachycena. Objevovali se nálezy na ostatní pravidla, ale pro nově napsané pravidlo, nález nebyl nalezen. Pro další ověření byla použita stránka http://testmyids.com/. Tato stránka vznikla za účelem testování funkčnosti IDS systémů. Stránka byla navštívena z firemní sítě, aby komunikace byla zachycena pomocí IDS. Systém obsahuje pravidlo s identifikačním číslem 2015524, které kontroluje, zda se nepodařilo útočníkovi využít zranitelnosti v jazyce JavaScript.

```
alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET WEB_CLIENT c3284d Malware Network Compromised Redirect (comments 3)"; flow:established,from_server; file_data; content:"/*c3284d*/"; reference:url,blog.unmaskparasites.com/2012/06/22/runforestrun-and-pseudo-random-domains/; classtype:trojan-activity; sid:2015524; rev:2; metadata:created_at 2012_07_25, updated_at 2012_07_25;)
```

Po navštívení stránky nebylo vytvořeno upozornění. Jako první byly ověřeny všechny služby, zda správně běží a byly prověřeny jejich souborové záznamy. V záznamech nebyly nalezeny žádné chyby, ale ve statistikách systému Suricata byl podezřele vysoký počet špatně přijatých paketů. Byla provedena analýza pomocí programu Wireshark. Bylo ověřeno, zda se správně přijímá komunikace na síťovém rozhraní. Rozhraní přijímalo pakety v pořádku a v programu Wireshark byla vidět odchycená komunikace také v pořádku. Byla zde také vidět komunikace na adresu 1.1.1.1. Nebyl tak zjištěn žádný očividný problém.

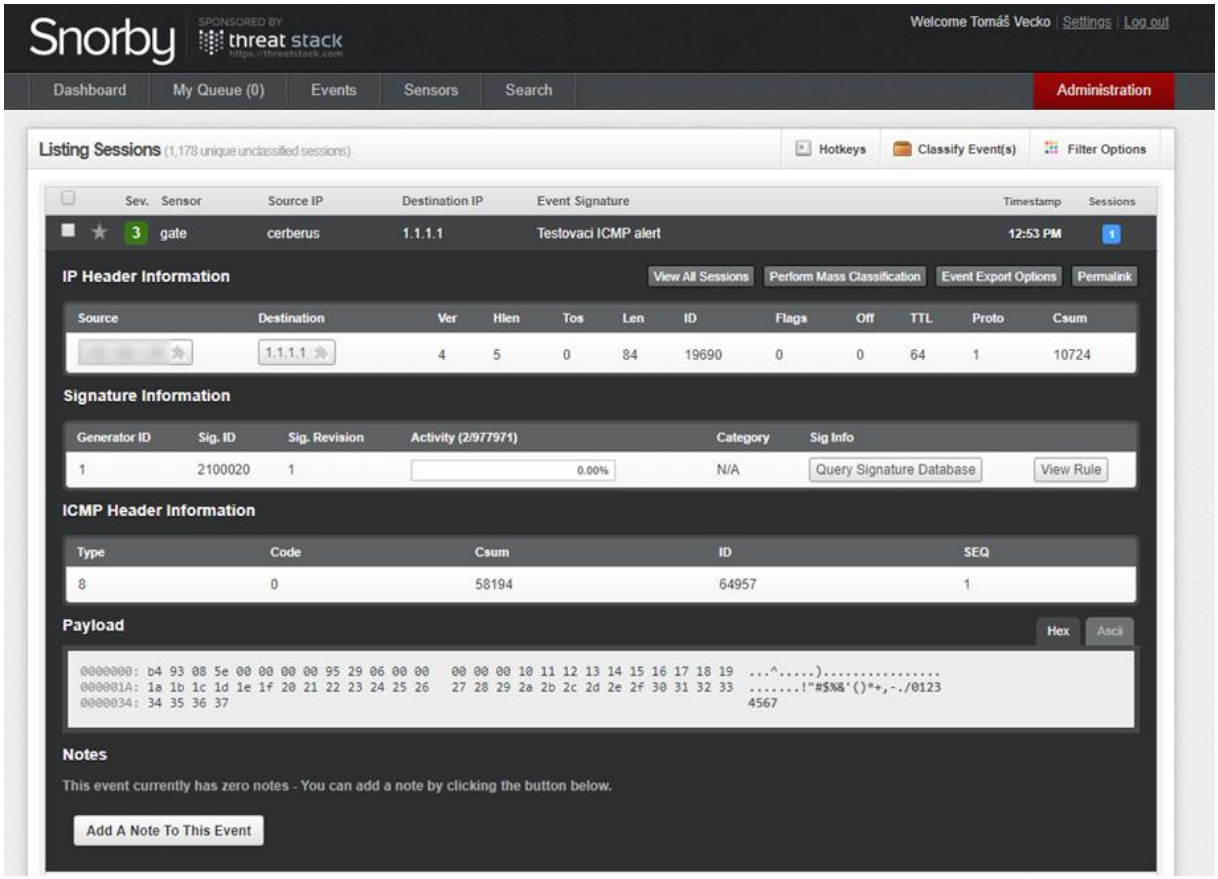
Dle statistik systému Suricata byly zahazovány pakety. Po prověření dokumentace, co může být příčinnou, byla zkontrolována velikost MTU. V konfiguračním souboru je ve výchozím nastavení velikost rámce nastavena na 1500 bajtů. Byla odchycena komunikace pomocí programu tcpdump. Bylo uloženo 200 MB komunikace ve formě pcap souboru. Tento soubor byl otevřen v programu Wireshark, kde byly vidět odchycené pakety. Některé měly velikost rámce nad 1500, konkrétně 1514 bajtů. Této komunikace bylo cca 33 %. Zachycená komunikace je vidět na obrázku 8.



Obrázek 8 - Zachycená komunikace v programu Wireshark. Zdroj: vlastní zpracování

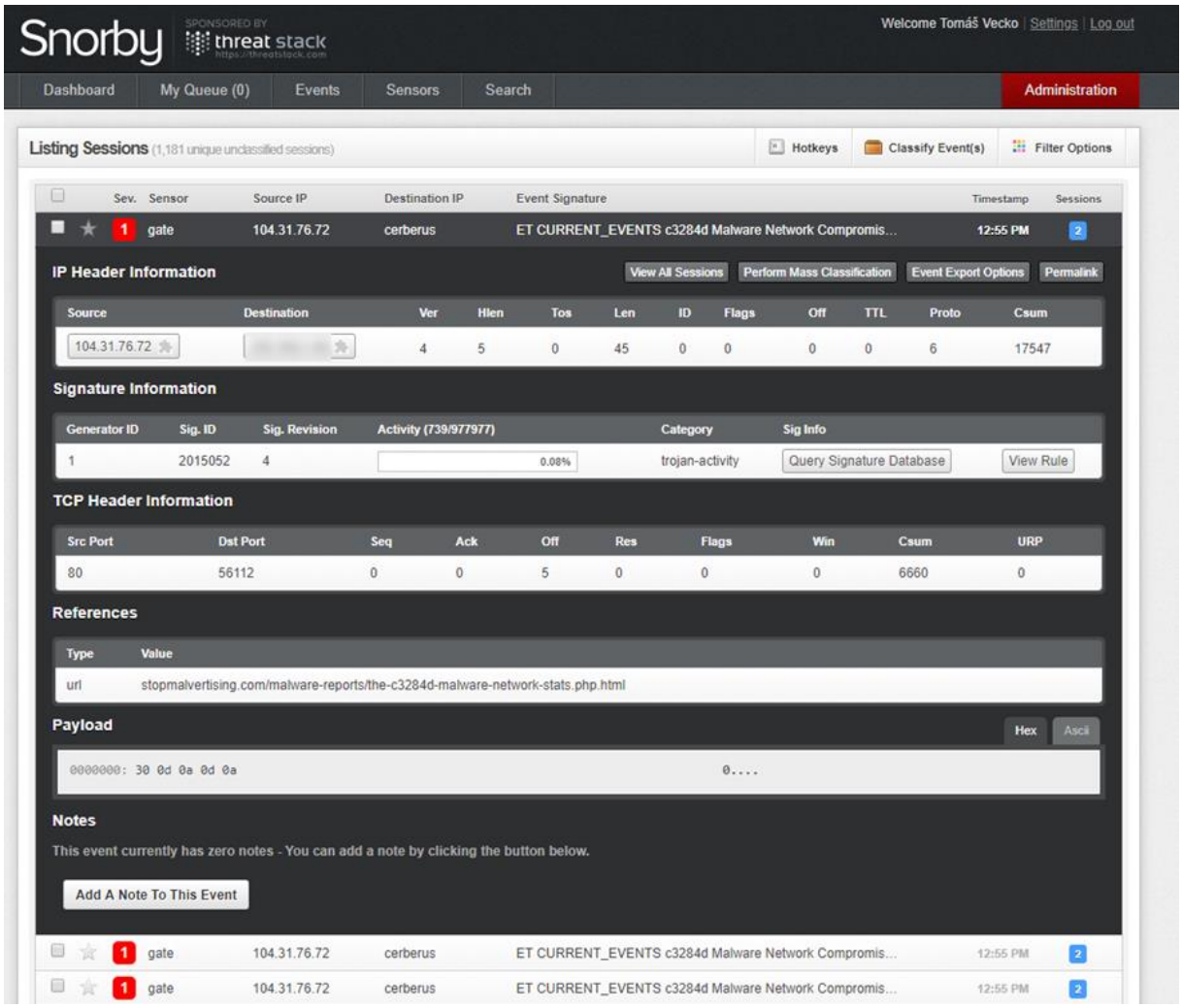
Po dalším studiu literatury bylo zjištěno, že standard IEEE 802.3ac zvyšuje maximální délku rámce až na 1522 bajtů. Toto zvýšení bylo z důvodu přidání čtyř bajtového VLAN označení [2]. Společnost, ve které tento systém běží, používá standard IEEE 802.1Q. Tím bylo vysvětleno zahazování paketů systémem. Systém má momentálně nastavenou maximální velikost rámce 1500 bajtů, dle výchozí nastavení. Pokud přijde do systému rámec, který má větší velikost než maximálně definovanou, tak jej systém zahodí. Proto není vidět testovaná komunikace. Tento parametr v systému definuje, kolik si má systém připravit paměti pro jednotlivé rámce. Pokud by tato velikost byla nastavena na moc velkou, systém by tak plýtval pamětí. Pokud je tato hodnota nastavena na moc nízkou, tak systém rámce s větší hodnotou zahodí. Dle standardu IEEE 802.3ac byla nastavena v systému velikost rámce na 1522 bajtů. Systém si tak pro každý rámce bude alokovat tuto velikost. Větší pakety budou zahozeny, ale v sítí společnosti by se tyto rámce neměly vyskytovat.

Po dalším otestování komunikace na adresu 1.1.1.1 je zachycená komunikace již vidět ve webovém rozhraní po úspěšném zachycení systémem. Zachycená komunikace je vidět na obrázku 9.



Obrázek 9 - Zachycená komunikace ping. Zdroj: vlastní zpracování

Dále byla navštívena stránka http://testmyids.ca/. IDS tuto komunikaci již zachytilo a vyhodnotilo ji jako porušení pravidla 2015052. Na obrázku 10 je vidět zachycená komunikace. Dle testovacích scénářů vypadá, že systém vidí pakety v pořádku a dokáže je již správně vyhodnocovat.



Obrázek 10 - Zachycená zranitelnost jazyka JavaScript. Zdroj: vlastní zpracování

4.4 Úprava pravidel

Tato kapitola obsahuje úpravu sady pravidel a řadu také zakazuje.

Systém momentálně obsahuje 20300 pravidel. Některé pravidla generují příliš mnoho upozornění, nebo falešně pozitivní. IDS za 5 minut provozu vygenerovalo více než 2000 upozornění. S takovým množstvím dat administrátor nemůže pracovat. Z tohoto důvodu byla zakázána další pravidla, což vedlo ke zlepšení práce s webovým rozhraním.

Jedno z výchozích pravidel detekuje, zda se správně ukončilo TCP spojení. Toto pravidlo vygenerovalo více než 50% upozornění z 2000. Existují aplikace, které správně neukončují

spojení, a není v rámci možností administrátora tyto aplikace upravovat. Pravidla tohoto typu byla vybrána k zakázání, jelikož pouze zahlcovala webové rozhraní a správce s nimi nemohl nic dělat. Bylo zjištěno také hned zjištěno několik desítek falešně pozitivních nálezů. U všech těchto nálezů bylo zjištěno ID pravidla a bylo vyhodnoceno zakázání těchto pravidel jednou z dostupných možností pro zakázání pravidel. Systém Suricata nabízí čtyři možnosti, jak ignorovat komunikaci:

- BPF filtr
- Disablesid
- Threshold
- Pass pravidla

Berkeley paketový filtr se používá pro filtraci komunikace, která postupuje do systému pro analýzu. BPF se používá také u programu Wireshark, kde se používá pro filtraci dle IP adresy nebo portu [22]. Pomocí negace lze vytvořit filtr, která komunikace se má ignorovat a nebude IDS zatěžovat. Ignorovaná komunikace se vůbec nedostane do systému k další analýze a bude zahozena.

Soubor disablesid.conf slouží pro zapisování ID pravidel, které se mají ignorovat. Při aktualizaci pravidel se automaticky vyřadí a IDS je neaplikuje. Zakázané pravidla pak nezatěžují systém. V souboru threshold.config se definují také ID pravidel, ale je zde více možností pro filtrování, jako například zakázání pravidla v určitý čas nebo ignorování stejného upozornění pro definovaný čas. Tento soubor používá program barnyard2, ale IDS ho ignoruje. Pravidla jsou tak v IDS zpracována a až další systém je bude ignorovat a v grafickém rozhraní se nezobrazí.

Pravidla pro schválení se mohou použít také pro potlačení upozornění. Systém nejdříve vyhodnocuje schvalovací pravidla a až poté pravidla pro upozornění. Teoreticky tak lze napsat schvalovací pravidlo pro pod část pravidla pro upozornění. Tato možnost by se měla využívat velmi zřídka, jelikož neodlehčuje sadu pravidel IDS, ale naopak sadu zatěžuje o další pravidla.

Každá možnost je vhodná pro specifické použití a je tak na správci, aby vybral vhodné řešení pro zakázání upozornění. Pro účely této práce byla využita možnost přes soubor disablesid.conf, kdy bylo třeba zakázat určitá pravidla. Tím je dosaženo ignorování určitých pravidel a zakázaná pravidla nezatěžují IDS.

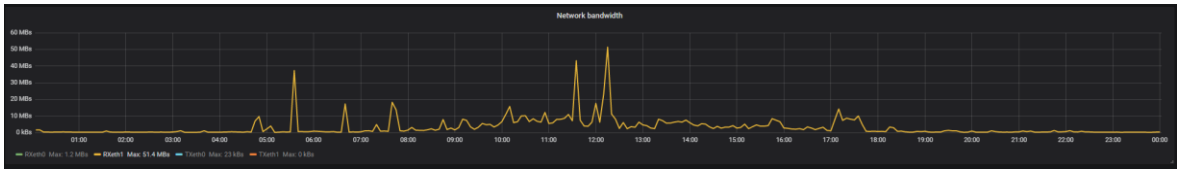
4.5 Monitoring systému

Tato kapitola se bude zabývat nastavení sledování IDS pomocí systému Nagios. Sledování bude použito pro zjištění efektivnosti systému a budoucí optimalizace. Sledování bude také použito pro sledování systémových prostředků, a zda systém funguje správně.

Společnost poskytla šest skriptů pro sledování systémových prostředků. Tím bylo pokryto sledování procesoru, paměti, času, disku, procesů, zátěže, hardwaru a zdroje nepřerušovaného napájení. Bylo nastaveno sledování, zda běží procesy barnyard2, suricata a mysql. Pro účely monitoringu byl vytvořen uživatel nagcheck, přes kterého se skripty spouští. Skripty jsou volány každých pět minut systémem Nagios a vrací návratové kódy a hodnoty, které jsou uloženy do grafů v systému Grafana.

Pro účely této práce byly vytvořeny další skripty pro sledování komunikace na síťové kartě, zatížení disku a sledování statistik IDS. Skript pro sledování propustnosti síťové karty získává hodnoty, které operační systém ukládá do souborů v adresáři /sys/class/net. V těchto souborech je vidět kolik bytů prošlo určitou síťovou kartou v obou směrech od posledního zapnutí operačního systému [23]. Hodnoty se postupně zvyšují. Pokud server odešle jeden byt, tak se přičte jedna k této hodnotě. Skript byl navržen tak, aby získal aktuální hodnoty ze souborů a zapsal je do pomocného souboru s aktuálním časem. Systém Nagios umožňuje zobrazení základních grafů ze získaných hodnot. Kdyby nebyl použit pomocný soubor a vracely se data, jak jsou v souboru, tak by se graf pouze zvětšoval. Systém Grafana dokáže vypočítat rozdíl mezi dvěma hodnotami. Organizaci se rozhodla skript použít i u dalších serverů a požadovala možnost zobrazení grafu v systému Nagios. Při dalším zavolání skriptu se znovu získají aktuální data a provede se rozdíl mezi hodnotami. Hodnoty jsou pak vyděleny uplynulým časem. Tím je získán průměrný počet bytů za sekundu, zkráceně Bps. Funkčnost a správnost skriptu byla ověřena pomocí programu wget, který ukazuje průměrnou rychlost stahování. Skript byl spuštěn a ve stejný čas byl spuštěn program wget pro stažení souboru o velikosti 2 GB z lokální sítě. Příkazy byly odděleny středníkem. Tím bylo docíleno spuštění obou příkazů ve stejný čas. Po skončení stahování program vypsal průměrnou rychlost stahování souboru. Skript byl spuštěn znovu pro kalkulaci průměrné rychlosti. Hodnoty se lišili v rámci 5 %. Tato hodnota odpovídá chybě měření a je přijatelná. Testování bylo opakováno desetkrát se stejným výsledkem.

Skript je spouštěn každých pět minut a výsledky jsou ukládány do grafu v systému Grafana. Graf za 24 hodin je vidět na obrázku 11. Je zde vidět, že ráno v 5:30 je velký vrchol. V tu dobu nejspíše probíhá záloha serverů. Poté se cca od 8 hodin postupně zvyšuje propustnost na síťové kartě, kdy přicházejí zaměstnanci do společnosti a začínají pracovat a po 18 hodině zase klesá, kdy jdou zaměstnanci společnosti domů.



Obrázek 11 - Graf sítě. Zdroj: vlastní zpracování

Skript pro sledování zatížení disku byl vytvořen na podobném principu jako sledování síťového provozu. Skript získává hodnoty ze složky /sys/block. V této složce jsou data pro jednotlivé disky [24]. Skript získává aktuální data o přístupové době k disku a pracuje s nimi stejně jako skript pro sledování síťové komunikace. Na výstupu skriptu je průměrná čekací doba disku. Tím je možné určit, jak je vytížen disk a zda server nepotřebuje spíše disk typu SSD. Graf sledování disku je vidět na obrázku 12. Dle grafu bylo zjištěno, že server není tak vytížen na io operacích, aby vyžadoval rychlejší disky.

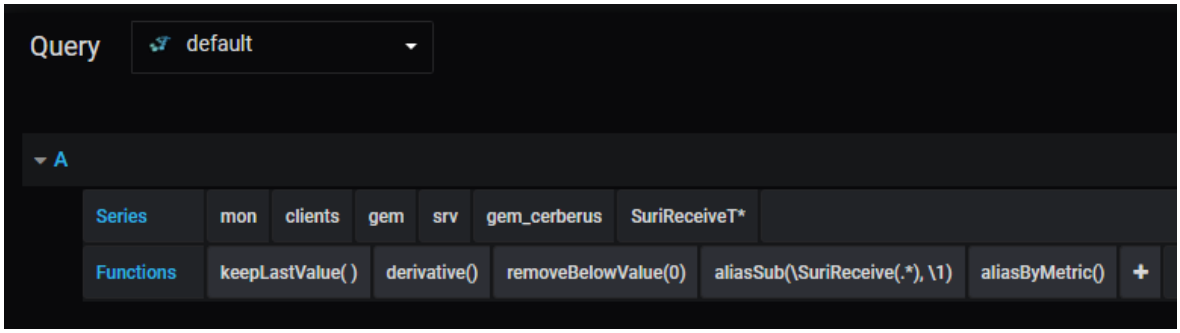
Obrázek 12 - Graf vytížení disku. Zdroj: vlastní zpracování

4.5.1 Statistiky systému

Systém Suricata nabízí možnost vypisování statistik o své činnosti do souboru. Ve výchozím nastavení je interval zapsání statistik 8 sekund. Pro ukládání těchto statistik do nástrojů Nagios a Grafana byl vytvořen další skript, který získává data ze souboru /var/log/suricata/stats.log. Skriptem je měřena efektivita IDS.

Volání skriptu je prováděno každých 5 minut. Hodnoty v souboru se postupně zvětšují jako u statistik operačního systému. Hodnoty jsou ze souboru získány a předávány do systému Nagios. Jelikož se jedná o sledování statistik jednoho systému v rámci této práce, tak grafy v systému Nagios nebudou použity, ale budou použity hlavně grafy v systému Grafana. Hodnoty jsou předávány přímo do systému Grafana, kde je použito několik funkcí, aby systém ukázal pouze

rozdíl mezi dvěma hodnotami volání skriptu. Použité funkce jsou vidět na obrázku 13. Těmito funkcemi je zajištěno zobrazení rozdílu mezi hodnotami.



Obrázek 13 - Funkce pro zobrazení v systému Grafana. Zdroj: vlastní zpracování

V systému Nagios nelze zaručit, kdy bude skript spouštěn, nastavený interval se může lišit v rámci sekund. Pokud by bylo nastaveno v IDS zapisování každých 5 minut, tak by mohla vzniknout situace, kdy bude proveden skript pro získání hodnot a o pár sekund později se uloží nové aktuální hodnoty. Mohly by se tak získávat hodnoty, které by byly skoro až 5 minut staré. Interval zapisování statistik byl tak upraven na 60 sekund, aby příliš nevytěžoval systém vypisováním statistik a také, aby hodnoty byly dostatečně aktuální.

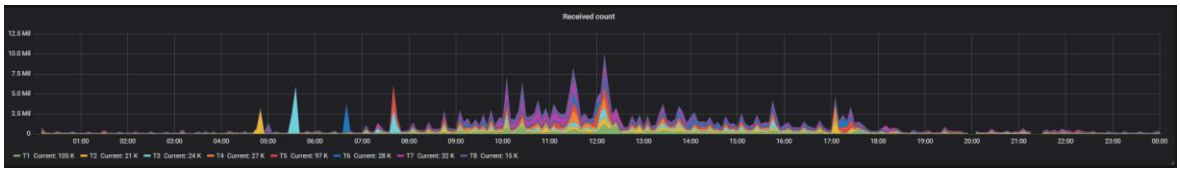
Vytvořeným skriptem jsou získávány data, dle nichž je vyhodnocována efektivita systému. Mezi získávaná data patří průměrná rychlost komunikace, nebo také propustnost, kterou systém musí zpracovávat. Graf je vidět na obrázku 14. Graf je totožný s grafem na obrázku 11, jelikož není použito BPF. V případě jeho použití by se grafy mohly lišit, jelikož by se do IDS nedostala komunikace. Pakety by byly zahozeny ještě předtím, než by se dostaly k analýze.

Obrázek 14 - Graf přijatých paketů. Zdroj: vlastní zpracování

Dále jsou ze souboru získávány hodnoty o celkových přijatých a zahozených paketech. Graf je vidět na obrázku 15. Graf příchozích paketů kopíruje graf na obrázku 14, ale jsou zde také vidět zahozené pakety, které systém nedokázal analyzovat. Je vidět že systém při velké zátěži při analýze nočních záloh nedokázal všechny pakety prověřit a také během dne zahazoval pakety. Celkově 5,6 milionu z 360 milionu jich zahodil. Poměrem přijatých paketů k zahozených lze zjistit aktuální procento zahozených paketů. Za měřený den v grafu je to 1,56 %. Dle této hodnoty bude dále měřena efektivita systému.

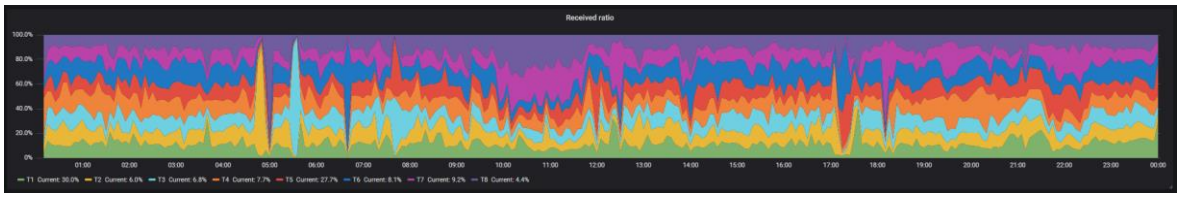
Obrázek 15 - Graf zahozených paketů. Zdroj: vlastní zpracování

V konfiguračním souboru bylo nastaveno, aby systém vypisoval statistiky pro jednotlivá vlákna. Skriptem jsou tak sledovány, kolik paketů zpracovávají jednotlivá vlákna a které vlákna zahazují pakety. Zde je ukázáno, jak systém používá více vláken oproti systému Snort. Ve výchozím nastavení Suricata používá algoritmus cluster_flow, který distribuuje pakety ze stejného zdroje do stejného vlákna. Tím je urychlena analýza, kdy pak IDS neztrácí systémové prostředky pro znovu sestavení komunikace dohromady z jednotlivých vláken. Dalším typem distribuce je round_robin kdy jsou pakety distribuovány rovnoměrně mezi vlákna. Na obrázku 16 je tak vidět rozdělení zátěže mezi jednotlivé vlákna.



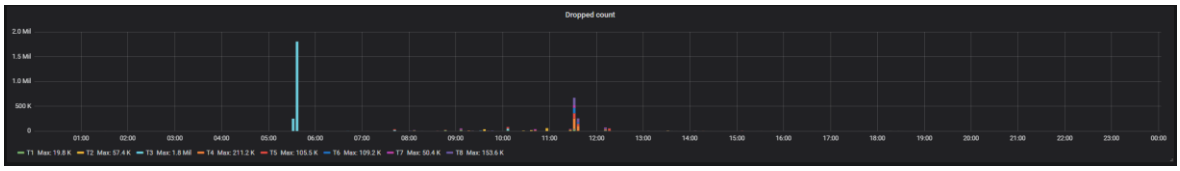
Obrázek 16 - Rozdělení paketů mezi vlákny. Zdroj: vlastní zpracování

Ze stejných dat je vytvořen graf pro zobrazení poměru mezi jednotlivými vlákny. Poměr je vypočítán z paketů pro jednotlivé vlákna k celkovému počtu paketů. Graf je vidět na obrázku 17. Na grafech je patrné, že systém okolo 13 hodiny analyzoval 2 velké toky, které byly zpracovávány v T7 a T8. Dále je vidět, že v 5:30, kdy IDS zahodilo pakety, vlákno T7 analyzovalo 93% celkové komunikace. Vysvětlením je, že v tu dobu probíhala komunikace z jednoho zdroje a systém jí nestíhal zpracovávat jedním vláknem. Ostatní vlákna v tu dobu nebyla využita.



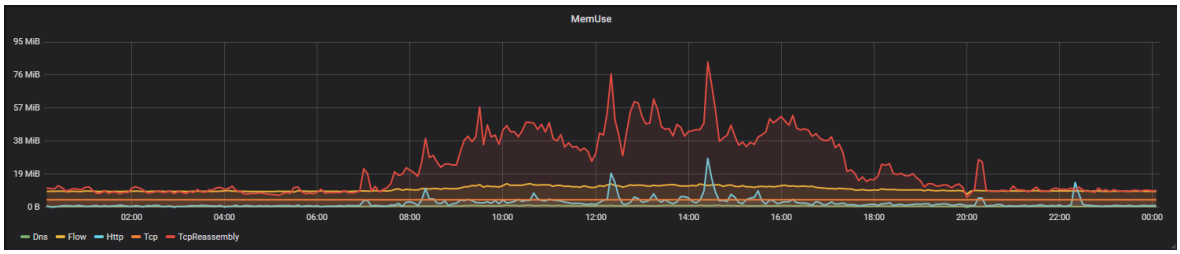
Obrázek 17 - Poměr rozdělených paketů. Zdroj: vlastní zpracování

IDS také poskytuje informace o tom, které vlákno zahodilo pakety. Graf je vidět na obrázku 18. V tomto případě je vidět, že většinu paketů zahodilo vlákno T7, které v tu dobu bylo přetíženo. Dále je vidět i zahazování ostatních vláken během dne, kdy bylo IDS přetíženo.



Obrázek 18 - Graf zahozených paketů dle vláken. Zdroj: vlastní zpracování

Mezi poslední získávané hodnoty patří, kolik IDS využívá paměti pro jednotlivé operace. Na grafu na obrázku 19 je vidět, že systém nejvíce využívá paměti pro znovu sestavení komunikace. Během dne, kdy jsou ve firemní síti uživatelé, je využití paměti vyšší, jelikož je více dat k analýze a k znovu sestavení.



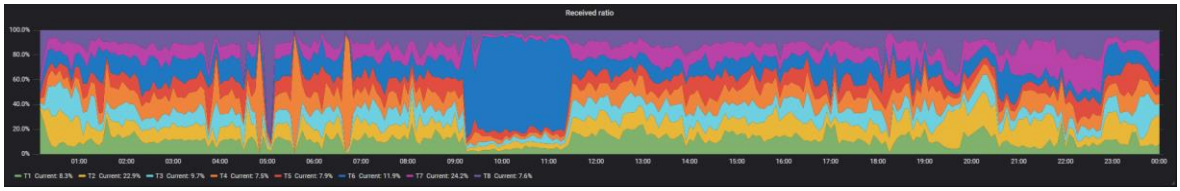
Obrázek 19 - Graf využití paměti. Zdroj: vlastní zpracování

Po několika dnech běhu systému bylo zjištěno, že systém stále zapisuje statistiky do stejného souboru i po restartu. Tento soubor se s časem stával větší a po několika dnech byla velikost přes 100 MB. Systém nenabízí žádné možnosti pro řešení tohoto problému. Pro zálohování log souborů se v linuxových distribucích používá program logrotate. V případě použití tohoto programu, byl soubor zálohován, ale systém Suricata se snažil stále psát do stejného souboru, který již neexistoval. Pro znovu vytvoření souboru byl třeba restart. Použití tohoto programu v případě této práce nebylo optimální. Byl využit skript pro aktualizaci pravidel. Tento skript každý den restartuje systém IDS. Mezi příkazy stop a start služby byl přidán příkaz pro zálohu souboru. Výše navržené řešení přineslo pravidelné zálohování statistik systému a odvrátilo riziko přeplnění disku.

4.6 Optimalizace systému

Tato kapitola se bude zabývat úpravou parametrů IDS pro optimalizaci a zvýšení efektivnosti systému. IDS momentálně za jeden pracovní den zahodí průměrně 1,56 % paketů a pakety zahazuje v průběhu pracovního dne. Cílem této kapitoly je vylepšení této statistiky a omezení zahazování paketů během dne. Server HPE ProLiant DL320e Gen8 v2 má 8 GB operační paměti a procesor Intel Xeon E3-1231 v3, 3.40GHz s 8 vlákny. Tento server je velmi poddimenzován, ale pro účely této práce a pro potřebu organizace stačí. Při postupu optimalizace bude zaměřeno využití na více vláken a co nejrychlejšího zpracování komunikace.

Pro účinné měření změn v konfiguraci je nutné vytvoření kontrolovaných podmínek testování. Program iPerf umožňuje změřit síťovou propustnost mezi dvěma zařízeními. Byl vytvořen virtuální server, na kterém byl tento program spuštěn v režimu server. To umožňuje, aby program čekal na otevření spojení ze strany klienta. Program začal poslouchat na všech dostupných adresách a na portu TCP 5001. Virtuální server byl umístěn do sítě s IDS. Jako klient byl vybrán notebook v jiném subnetu. Na straně notebooku, byl program spuštěn s adresou virtuálního serveru v klientském režimu. Po navázání spojení bylo přenesena data a byla změřena propustnost mezi zařízeními. Propustnost byla průměrně 844 Mbps. Testování bylo nastaveno, aby probíhalo po dobu dvou hodin. Tím na virtuální server byla posílána komunikace. Virtuální server byl v jiném subnetu sítě, a tak komunikace procházela přes router. Tím bylo docíleno přeposílání na IDS server, který tuto komunikaci analyzoval. Na obrázku 20 je vidět, jak tato komunikace byla analyzována. Na obrázku je vidět poměr kolik procent paketů jednotlivá vlákna zpracovala. Jelikož je v systému nastaveno, že stejné spojení zpracovává jedno vlákno, tak je na obrázku vidět, že v dobu testování většinu paketů zpracovalo vlákno 6 s modrou barvou a ostatní vlákna téměř nic neanalyzovala mezi 9 až 11 hodinou. Bylo tak zjištěno, že tímto typem testování není využit systém Suricata na maximum, jelikož ostatní vlákna nejsou využita. Tento typ testování je neefektivní pro účel této práce.



Obrázek 20 - Graf pro testování zátěže. Zdroj: vlastní zpracování

Pro další testování byly vytvořeny další virtuální servery, které nasimulují spojení pro další vlákna. Testování bylo spuštěno po dobu dvou hodin od 21 do 23 hodin, jak je vidět na obrázku 20. Po skončení testování bylo zjištěno, že všechna vlákna byla více zaneprázdněna a IDS zahodilo 19 % paketů. Tento typ testování dokázal více nasimulovat reálné podmínky provozu a vytížil všechna vlákna systému. Tímto testováním bylo docíleno maximální využití linky k serveru a byl tak docílen nejhorší možný scénář testování kdy systému musí analyzovat 1 sGbps komunikace.

Jako další typ testování byl vybrán běžný pracovní den. Systém je tak otestován reálnou komunikací, kterou generují zaměstnanci a servery a je tak vidět, jak bude systém v reálném provozu zatížen. IDS za pracovní den zahodilo průměrně 1,56 % paketů.

Dle dat z monitoringu systém využíval pouze 1,1 GB operační paměti z dostupných 8 GB. Server měl tak nevyužitou operační paměť, kterou mohl využít pro analýzu komunikace. Prvním krokem optimalizace bylo zvýšení dostupných parametrů, aby systém využil všechny dostupné prostředky. Všechny prostředky může využít samotný IDS, jelikož celý server je přímo dedikovaný pro tento účel. Tím je možno naplno využít všechny serverové prostředky.

4.6.1 **Úprava OS**

Linuxový kernel má vyrovnávací paměť pro síťovou kartu, která slouží pro uchování paketů, než si je aplikace ze síťové karty vyzvedne. V případě zatížení serveru může tato doba být vyšší. Z tohoto důvodu byla tato paměť navýšena z výchozích 256 KB na 16 MB pomocí těchto příkazů:

```
echo 'net.core.rmem_max=16777216' >> /etc/sysctl.conf
echo 'net.core.rmem_default=16777216' >> /etc/sysctl.conf
echo 'net.core.optmem_max=16777216' >> /etc/sysctl.conf
sysctl –p
```

Do souboru sysctl.conf byly přidány parametry pro navýšení paměti. Pomocí argumentu p, bylo provedeno znovu načtení konfigurace. Z důvodu, že byla konfigurace zapsána do tohoto souboru, tak bude znovu použita i po restartu serveru [25].

Dále byla optimalizována síťová karta serveru. Pomocí příkazu ethtool byla zjištěna aktuální a maximální hodnota paměti pro příjem paketů. Aktuální hodnota byla 511 a maximální možná byla 2047. Pomocí stejného příkazu bylo nastaveno, aby se použila maximální možná hodnota. Tím je docíleno, že síťová karta, bude mít dostatek vyrovnávací paměti pro situace, kdy náhle přijde více paketů.

```
root@cerberus:~# ethtool -g eth1
Ring parameters for eth1:
Pre-set maximums:
RX:             2047
RX Mini:        0
RX Jumbo:       1023
TX:             511
Current hardware settings:
RX:             2047
RX Mini:        0
RX Jumbo:       100
TX:             511
```

4.6.2 **Konfigurace paměti pro ring_size**

Jako první sekce pro optimalizaci IDS byla vybrána sekce af_packet a ring_size v konfiguračním souboru. S tímto parametrem je možno definovat kolik paketů bude systém držet v zásobníku pro zpracování pro jednotlivé vlákna [26]. Například:

```
ring-size: 100000
```

Toto nastavení by znamenalo, že systém bude držet 100000 paketů pro každé vlákno. V této sekci je také definováno, že systém Suricata má použít všechna dostupná vlákna procesoru. Server má dostupných osm vláken. To znamená 8 × 100000 = 800000 paketů dohromady. Pro výpočet, kolik toto nastavení spotřebuje operační paměti, je třeba znát hodnotu MTU.

```
Velikost MTU × ring-size × počet vláken
```

Neboli:

```
1522 × 100000 × 8 = 1217600000 bajtů = 1,1 GB
```

To znamená, že při tomto nastavení, IDS spotřebuje 1,1 GB operační paměti ihned po startu. Využití operační paměti po startu IDS bylo v operačním systému vidět jako 1,2 GB. To bylo bez optimalizace konfigurace systému. Pro otestování metodiky byl parametr ring-size navýšen z výchozí hodnoty 2048 na 100000. Po restartu systému bylo vidět, že využití operační paměti je na 2,3 GB. Tím byla potvrzena správnost počítání. Pomocí výše zmíněné metodiky bylo vypočítáno, že zvýšení ring-size na hodnotu 450000 by znamenalo zvýšení paměti o 5,1 GB. Finální využití paměti serveru by pak bylo 6,3 GB. Po změně ring-size na tuto hodnotu byl proveden restart systému a využití operační paměti po restartu bylo 6,5 GB. Dle příkazu free má server využito 7,5 GB neboli 87 % operační paměti.

Efektivita optimalizace bylo otestována reálným provozem v pracovní den. Na konci pracovního dne byl poměr zahozených paketů 0,5 %. Na obrázku 21 jsou vidět grafy k tomuto dni. Je zde vidět, že zatížení serveru odpovídá přijaté komunikaci k analýze. Při větší přijaté komunikaci je systém více zatížen. Je zde také vidět, že systém již během dne nezahazuje pakety, ale zahazuje je pouze v 5:35, kdy nejspíše probíhá ranní záloha některého serveru. IDS v tu dobu zahodilo 2 milionu paketů. Před optimalizací v tuto dobu zahodilo až 3 miliony paketů. Toto nastavení tak výrazně zlepšilo efektivnost systému.

Obrázek 21- Grafy systému po optimalizaci. Zdroj: vlastní zpracování

U testování pomocí virtuálních serverů bylo zjištěno také zlepšení efektivity a systém již zahodil pouze 6 % paketů. Toto řešení výrazně zlepšilo efektivitu systému a systém již během pracovního dne nezahazuje pakety a dokáže komunikaci plně analyzovat.

4.6.3 Limit spojení

Pro optimalizaci byl dále vybrán časový limit u spojení v systému. IDS si nechává spojení v paměti po určitý čas. Toto je užitečné, pokud je další komunikace z jednoho zdroje nebo do jednoho cíle. Systém má tak uchované spojení v paměti a dokáže ho dále sestavit a analyzovat více do hloubky. Toto nastavení je určeno v konfigurační souboru parametrem time-out. Pro každý typ komunikace má systém Suricata různé množství stavů. Pro TCP spojení má stavy: nový, zavedený a uzavřený. Pro UDP spojení má pouze nový a zavedený. Pro každý z těchto stavů dokáže systém určit jiné časové limity. Stav nový u TCP spojení znamená dobu během trojcestného handshaking (anglicky three-way handshake). Stav zavedený je od zavedení trojcestného handshaking. Uzavřený stav je, když se komunikace uzavře s jedním ze způsobů pro uzavření. Stav nový u UDP je, pokud je komunikace vedena pouze z jednoho zdroje a zavedený, když je komunikace obousměrná. IDS zachycené komunikace uchovává v paměti

pro další analýzu. V případě většího množství zdrojů komunikace může docházet k problémům s operační pamětí a systém začne zahazovat pakety.

Pro otestování efektivnosti parametrů byly časové spojení změněny na polovinu původní hodnoty. IDS si údaje o komunikaci po této změně nechávalo pouze polovinu času a mělo teoreticky menší nároky na paměť. Konfigurace byla otestována běžným provozem v pracovní den. Testování pomocí virtuálních serverů bylo označeno v tomto případě jako neužitečné, jelikož případě tohoto testování je jen omezené množství spojení. Z tohoto důvodu byla zvolena metoda testování podle pracovního dne. Na konci pracovního dne byl poměr zahozených paketů řádově stejný. IDS během dne nezahazovalo pakety a pouze je zahodilo zase ráno v 5:30. Toto chování lze vysvětlit tím, že ráno je komunikace z jednoho zdroje a není v tu dobu využito tolik paměti pro uložení dalších spojení. V tu dobu je pouze jedno spojení. Toto nastavení má vliv, v případě této práce na tom, jak IDS se spojením během dne pracuje. V tuto dobu má systém více spojení najednou. Pro otestování samotné této změny, bylo z konfiguračního souboru odebrána hodnota ring-size. V konfiguračním souboru tak byly změněny pouze hodnoty pro časové spojení. Tím bylo zajištěno otestování těchto parametrů. Systém byl po změně konfiguračního souboru restartován.

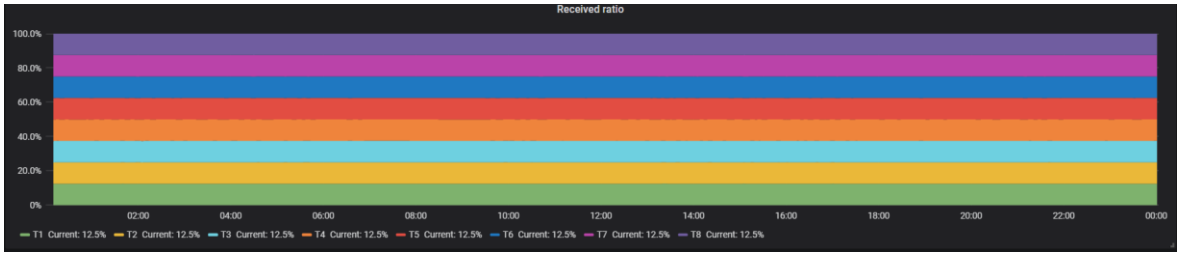
Konfigurace byla otestována běžnou komunikací v pracovní den. Na konci dne byl poměr zahozen paketů řádově stejný jako před optimalizací a to 1,51 %. Systém byl ponechán s touto konfigurací celý pracovní týden, aby se předešlo možným chybám. Na konci měření byl průměrně poměr mezi zahozenými a přijatými pakety 1,54 %. To je oproti hodnotě před optimalizací zlepšení o 0,02 %. Tato hodnota byla vyhodnocena jako zanedbatelná, s výsledkem, že toto nastavení nemá na efektivitu, v případě této práce, vliv. Je možné, že v případě většího provozu ve vysokorychlostních sítích by tyto parametry měly vliv, ale zde nebyl ukázán zásadní rozdíl. IDS v tomto případě nemá problémy s uchováváním spojení do paměti a tyto hodnoty byly nechány ve výchozích hodnotách.

4.6.4 **Vyvážení zátěže**

V kapitole 4.6.2 byla navýšena paměť IDS systému. Systém od té doby zahazuje pakety v běžném provozu pouze v 5:30, kdy je prováděna záloha přes síť a systém nestíhá analyzovat komunikaci, jelikož je přetíženo jedno vlákno systému a ostatní vlákna jsou nečinná. Toto nastavení je dáno přes parametr cluster-type. V konfiguračním souboru je nastavena hodnota cluster_flow, kdy všechny pakety ze stejného spojení jsou posílány do stejného vlákna. Toto

nastavení je dle dokumentace doporučováno, jelikož IDS neplýtvá prostředky pro spojení celé komunikace, které jsou zpracovány v různých vláknech do jednoho.

Na obrázku 21 je vidět, že systém má každý den v 5:30 přetížené jedno vlákno. Z tohoto důvodu byla změněna hodnota na cluster_round_robin, kdy jsou jednotlivé pakety rovnoměrně posílány do jednotlivých vláken. Server po restartu IDS měl využito pouze 2 GB operační paměti. Dle logů bylo zjištěno, že systém Suricata se spustil pouze s jedním vláknem pro analýzu. V konfiguračním souboru bylo u parametru threads hodnota auto. Tím měl systém použít všechna dostupná vlákna. V tomto případě je nepoužil. U tohoto parametru bylo na pevno nastaveno číslo osm, podle počtu vláken. Po restartu IDS použilo již správně všech osm vláken a využití operační paměti bylo již dle očekávání. Konfigurace byla ověřena dle dat z monitoringu. Na obrázku 22 je vidět poměr rozdělení paketů mezi jednotlivé vlákna. Z tohoto obrázku je patrné, že každé vlákno zpracovává stejný počet paketů a zátěž vláken je rovnoměrná.



Obrázek 22 - Graf vláken při rovnoměrném rozdělení. Zdroj: vlastní zpracování

Konfigurace byla otestována reálnou komunikací v běžný pracovní den. Na konci tohoto dne systém nezahodil žádný paket. V 5:30 kdy byla velká zátěž pro jedno vlákno se nyní rozdělila mezi všechna vlákna. IDS bylo následně otestováno pomocí zátěže z virtuálních serverů. Po třech hodinách testování systém Suricata zahodil stejně jako dříve 6 % paketů. Tímto nastavením bylo ošetřena situace, kdy bylo vytíženo jedno vlákno jednou komunikací. Nyní je zátěž rozdělena mezi více vláken a systém během normálního pracovního dne již nezahazuje pakety.

S touto konfigurací nyní dle dokumentace spotřebovává více zdrojů pro sestavení komunikace z jednotlivých vláken dohromady, ale dle monitoringu již nezahazuje pakety, z tohoto důvodu byla konfigurace ponechána, i když není doporučována.

4.6.5 **Modul pf_ring**

Tato kapitola se bude zabývat instalací modulu pf_ring a následně konfigurací systému Suricata pro jeho využití. Modul pf_ring je volně dostupný modul pro síťový socket, který zvyšuje rychlost odchycení komunikace [27].

Pro instalaci byla zvolena metoda přes repositář. Tato metoda je vhodnější pro budoucí správu a pozdější aktualizace. Verze z repositáře a ze serveru Git je navíc shodná, tudíž byla vybrána tato metoda. Repositář byl přidán ze stránek společnosti Ntop, která spravuje pf_ring. Při instalaci pf_ring pomocí příkazu yum, bylo zjištěno, že v závislostech modulu je i aktualizace kernel a kernel-devel. To v tomto případě nebylo možné, jelikož je server od společnosti HP a ta v době psaní této práce nevydala ovladače pro RAID na tento server pro nový kernel. Po instalaci nového kernelu by server již nebylo možné spustit s novým kernelem. Byla tak nainstalována starší verze kernel-devel, která odpovídá nainstalovanému kernelu a instalace pf_ring byla spuštěna znovu. Instalace nyní proběhla v pořádku. V konfiguraci modulu bylo nastaveno, z jakého síťového rozhraní má být prováděno odchytávání a jaké rozhraní slouží pro správu. Dále byl nastaven explicitně promiskuitní režim a zapnuta služba pf_ring.

Pro využití tohoto modulu byl systém Suricata znovu zkompilován s podporou pf_ring. U příkazu configure byla doplněna podpora tohoto modulu a cesta k nově nainstalovaným knihovnám. Podpora modulu byla otestována přes argument build-info u příkazu suricata. Příkaz vypsal podporu jednotlivých modulů systémem IDS. U nově nainstalovaného modulu byla podpora již dostupná. V konfiguračním souboru Suricata bylo v sekci pf_ring doplněno rozhraní k analýze a počet vláken k použití.

Pro testování byly použity virtuální servery s programem iPerf a po třech hodinách testování, IDS nezahodilo žádný paket. Dále bylo provedeno testování pomocí reálného provozu v běžný pracovní den. Po týdnu testování, systém žádný paket nezahodil. Pro snížení použití dalších zdrojů serveru bylo v konfiguračním souboru znovu nastaveno přidělování paketů dle spojení, místo rovnoměrné zátěže. Tato konfigurace byla dále otestována. Při testování pomocí virtuálních serverů, systém nezahodil žádný paket. Pro další otestování byl systém otestován reálnou komunikací v síti organizace. Po týdnu provozu bylo zjištěno, že systém zahazuje pakety opět v ranních hodinách, kdy probíhá komunikace z jednoho zdroje. Poměr zahozených paketů byl zlepšen na hodnotu 0,56 %. V této situaci systém nedokáže analyzovat jednu komunikaci jedním vláknem. Bylo vyhodnoceno, že procesor serveru na tuto operaci není dostatečně rychlý. Po dalším testování bylo zjištěno, že pokud je komunikace z více zdrojů, tak

ji systém v pořádku analyzuje, ale pokud jedna komunikace využije naplno dostupnou linku, tak systém ji již nestíhá analyzovat. Z tohoto důvodu byla ponechána konfigurace rovnoměrné zátěže vláken, kdy systém s touto konfigurací nezahazuje pakety.

5 Výsledky a diskuse

V analýze stávající síťové infrastruktury společnosti bylo zjištěno optimální umístění pro IDS. Na přepínači byla nastavena technologie SPAN pro zrcadlení síťového provozu do serveru IDS. Zrcadlení je nastaveno z portu, který vede na přepínači do routeru, který je výchozí branou sítě. Tím je docíleno zrcadlení internetové komunikace a také komunikace mezi jednotlivými VLAN. Tato komunikace je zrcadlena do zapůjčeného serveru, kde bylo nainstalováno IDS, které tuto komunikaci analyzuje.

Na základě analýzy a výsledků teoretické části a požadavků společnosti, byl vybrán pro účely této práce systém Suricata jako IDS. K tomuto systému byly dále vybrány programy Barnyard2 a Snorby. Tyto systémy byly implementovány na zapůjčený server a byly nakonfigurovány pro specifické účely společnosti. U systému Suricata byla nastavena automatická aktualizace pravidel bezpečnostních hrozeb. Dále byly upraveny pravidla, aby operátor nebyl zahlcován falešnými nálezy při práci ve webovém rozhraní. V rozhraní Snorby byly vytvořeny jmenné účty pro uživatele a v rozhraní je možný přehled o nálezech IDS systému. Pro hlubší analýzu je v systému nastaveno uložení odchycené komunikace v rozhraní pcap. Administrátor může procházet uloženou komunikaci například programem Wireshark a dále ji vyhodnotit.

Pro vybraný systém byl vytvořen monitoring pomocí nástrojů Nagios a Grafana. Pro účely této práce byly vytvořeny specifické skripty pro sledování IDS, jako například sledování propustnosti na síťové kartě a statistik IDS. Pomocí nástroje Nagios je administrátor notifikován o případných problémech s IDS a může na ně okamžitě zareagovat.

Na základě získaných dat z monitoringu byla provedena optimalizace systému Suricata. Systém před optimalizací zahazoval průměrně 1,57 % paketů v pracovní den. V konfiguračním souboru byly upraveny parametry systému a IDS byl znovu nainstalován s přídavným modulem pf_ring. Po těchto optimalizačních krocích, systém již přestal zahazovat pakety.

IDS Suricata nabízí další možnosti optimalizace a škálovatelnosti, ale případě této práce je již nebylo možné použít z důvodu malé velikosti operační paměti. Organizace plánuje připojení IDS serveru optickými vlákny. Teoretická propustnost na server by tak byla 20 Gbps. V případě navýšení propustnosti k serveru je nutné na server doinstalovat další operační paměť. Pro účely této práce a momentálního zapojení serveru do sítě bylo 8 GB operační paměti dostatečné, ale v případě vylepšení infrastruktury organizace tato konfigurace již nebude dostatečná. Po zvýšení operační paměti by bylo možné systém dále škálovat dalšími parametry.

Společnost nyní dokáže analyzovat komunikaci uvnitř sítě a detekovat bezpečnostní hrozby a útoky v síti. Za tři měsíce provozu systém detekoval dva malwary a několik uživatelů stahující obsah přes P2P síť. Všechny tyto nálezy byly zaevidovány do knihy bezpečnostních incidentů. V případě odstávky systému, nemá společnost přehled o síťové komunikaci a nemůže detekovat případné hrozby a útoky v síti. V případě rostoucích potřeb organizace je možné přidat další pravidla do definovaného souboru a budou v příští aktualizaci pravidel přidána do rozsahu pravidel pro analýzu.

6 Závěr

Tato práce se zabývala přehledem a představením IDS systémů pro monitoring firemní počítačové sítě. Ukázala důležitost těchto systémů u středně velkých až velkých společností jako Gem System a.s., ve které byl vybraný systém implementován do produkčního prostředí. IDS bylo konfigurováno specificky pro vybranou společnost. Dále byla vytvořena automatizace pravidelné aktualizace pravidel o nejnovější hrozby. K tomuto systému byly dále nainstalovány další podpůrné programy, včetně webového rozhraní. Ve webovém rozhraní jsou vidět všechny výstrahy, které IDS nalezne. Pravidla byla upravena, aby operátor nebyl zahlcován při práci s webovým rozhraním.

Pro implementovaný systém byl vytvořen monitoring pro sledování stavu a statistik IDS pomocí nástrojů Nagios a Grafana. Pomocí těchto nástrojů je administrátor informován o případných problémech s IDS a může tak okamžitě zareagovat. Na základě sledovaných statistik byl systém optimalizován, aby nezahazoval při analýze pakety.

Výše navržené řešení přineslo společnosti nástroj pro analýzu síťové komunikace. Společnost má přehled o síťové komunikaci uživatelů a také dokáže detekovat bezpečnostní hrozby v síti a síťové útoky. V případě dalších potřeb je možné vytvoření přídavných pravidel pro detekci požadované komunikace.

7 Seznam použitých zdrojů

[1] NIELSEN, jakob. Nielsen's Law of Internet Bandwidth. *Nielsen Norman Group* [online]. [cit. 2019-06-19]. Dostupné z: https://www.nngroup.com/articles/law-of-bandwidth/

[2] COMER, Douglas a David STEVENS. *Internetworking with TCP/IP*. 3rd ed. Upper Saddle River, N.J.: Prentice Hall, 1999. ISBN 978-013-2169-875.

[3] KABELOVÁ, Alena a Libor DOSTÁLEK. *Velký průvodce protokoly TCP/IP a systémem DNS*. 3. aktualiz. a rozš. vyd. Praha: Computer Press, 2002. Všechny cesty k informacím. ISBN 80-722-6675-6.

[4] FALL, Kevin a W. STEVENS. *TCP/IP illustrated*. 2nd ed. Upper Saddle River, NJ: Addison-Wesley, 2012. Addison-Wesley professional computing series. ISBN 03-213-3631-3.

[5] BOUŠKA, Petr. Adresování v IP sítích. *Samuraj-cz* [online]. [cit. 2019-05-28]. Dostupné z: https://www.samuraj-cz.com/clanek/adresovani-v-ip-sitich/

[6] YU, Zhenwei a Jeffrey TSAI. *Intrusion detection: a machine learning approach*. Hackensack, NJ: Distributed by World Scientific, 2011. Series in electrical and computer engineering, v. 3. ISBN 978-184-8164-475.

[7] FOSTER, James a Jeffrey POSLUNS. *Snort 2.0 Intrusion Detection*. Elsevier Science & Technology Books, 2003. ISBN 9781931836746.

[8] BRENTON, Chris a Cameron HUNT. *Mastering network security*. 2nd ed. San Francisco: Sybex, 2003. ISBN 978-078-2141-429.

[9] Snort Licence. *Snort* [online]. [cit. 2019-06-22]. Dostupné z: https://www.snort.org/license

[10] What is the relationship between Snort and Cisco?. *Snort* [online]. [cit. 2019-06-24]. Dostupné z: https://www.snort.org/faq/what-is-the-relationship-between-snort-and-cisco

[11] FISK, Mike a George VARGHESE. *Applying Fast String Matching to Intrusion Detection* [online]. In: . Los Alamos National Laboratory, University of California San Diego, s. 22 [cit. 2019-06-21].

[12] VANNEY, Ivan. Configure Snort IDS and Create Rules. *Linuxhint* [online]. [cit. 2019-05-25]. Dostupné z: https://linuxhint.com/configure-snort-ids-create-rules/

[13] *Suricata* [online]. b.r. [cit. 2019-06-22]. Dostupné z: https://suricata-ids.org/

[14] Open Source. *Suricata* [online]. [cit. 2019-06-23]. Dostupné z: https://suricata-ids.org/about/open-source/

[15] Introduction. *Zeek* [online]. [cit. 2019-06-24]. Dostupné z: https://docs.zeek.org/en/stable/intro/index.html

[16] Cisco Stealthwatch: At a glance. In: *Cisco Public* [online]. s. 2 [cit. 2019-06-21].

[17] CATEGORY: 3 LAYER SWITCHING MODEL. *Loopedback* [online]. [cit. 2020-02-20]. Dostupné z: https://loopedback.com/category/ccnp-switch/3-layer-switching-model/

[18] *Snort 3.0* [online]. b.r. [cit. 2019-09-19]. Dostupné z: https://www.snort.org/snort3

[19] *Hardware* [online]. b.r. [cit. 2019-09-20]. Dostupné z: https://www.elastic.co/guide/en/elasticsearch/guide/current/hardware.html

[20] *Suricata dokumentace* [online]. b.r. [cit. 2020-01-01]. Dostupné z: https://suricata.readthedocs.io/en/latest/

[21] *Barnyard2* [online]. [cit. 2020-02-19]. Dostupné z: https://github.com/firnsy/barnyard2

[22] CORBET, Jonathan. The BPF system call API, version 14. *Lwn.net* [online]. [cit. 2020-01-04]. Dostupné z: https://lwn.net/Articles/612878/

[23] Dokumentace sysfs-class-net-statistics. *Kernel* [online]. [cit. 2020-02-19]. Dostupné z: https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-class-net-statistics

[24] Block layer statistics in /sys/block/<dev>/stat. *Kernel* [online]. [cit. 2019-12-21]. Dostupné z: https://www.kernel.org/doc/Documentation/block/stat.txt

[25] SUCHÁNEK, Marek, Milan NAVRÁTIL, Laura BAILEY a Charlie BOYLE. Red Hat Enterprise Linux 7: Performance Tuning Guide. *RedHat* [online]. [cit. 2020-02-18]. Dostupné z: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/pdf/performance_tuning_guide/Red_Hat_Enterprise_Linux-7-Performance_Tuning_Guide-en-US.pdf

[26] MANEV, Peter. Playing with memory consumption, algorithms and af_packet ring-size in Suricata IDPS. *IT Security through Open Source: IT infrastructure and network security, Suricata and such...* [online]. [cit. 2020-02-19]. Dostupné z: https://pevma.blogspot.com/2014/05/playing-with-memory-consumption.html

[27] MANEV, Peter. Suricata (and the grand slam of) Open Source IDPS - Chapter II - PF_RING / DNA , Part One - PF_RING. *IT Security through Open Source: IT infrastructure and network security, Suricata and such...* [online]. [cit. 2020-01-11]. Dostupné z: https://pevma.blogspot.com/2013/12/suricata-and-grand-slam-of-open-source_4.html