Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Návrh, analýza a implementace IDS/IPS pravidel

2024

Ing. Lukáš Lichnovský

Vedoucí práce:
Mgr. Radek Janoštík, Ph.D.

Studijní program: Informační technologie,
kombinovaná forma

Bibliografické údaje

Autor:	Ing. Lukáš Lichnovský
Název práce:	Návrh, analýza a implementace IDS/IPS pravidel
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2024
Studijní program:	Informační technologie, kombinovaná forma
Vedoucí práce:	Mgr. Radek Janoštík, Ph.D.
Počet stran:	40
Přílohy:	elektronická data v úložišti katedry informatiky
Jazyk práce:	český

Bibliographic info

Author:	Ing. Lukáš Lichnovský
Title:	Design, analysis and implementation of IDS/IPS rules
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2024
Study program:	Information Technologies, combined form
Supervisor:	Mgr. Radek Janoštík, Ph.D.
Page count:	40
Supplements:	electronic data in the storage of department of computer science
Thesis language:	Czech

Anotace

Tato bakalářská práce představuje IDP/IPS systémy a jejich pravidla, kde je zaměřena především na jejich tvorbu a analýzu. Hlavním výsledkem této práce je nástroj k usnadnění a urychlení analýzy podezřelé komunikace. Součástí je také virtuální a reálná implementace těchto systémů v rámci malé sítě.

Synopsis

This bachelor thesis introduces IDP/IPS systems and their rules, focusing mainly on their design and analysis. The main result of this work is a tool to facilitate and accelerate the analysis of detected suspicious communications. It also includes a virtual and real implementation of these systems within a small network.

Klíčová slova: IDS; IPS; Suricata; Snort; ruleset; Raspberry Pi

Keywords: IDS; IPS; Suricata; Snort; ruleset; Raspberry Pi

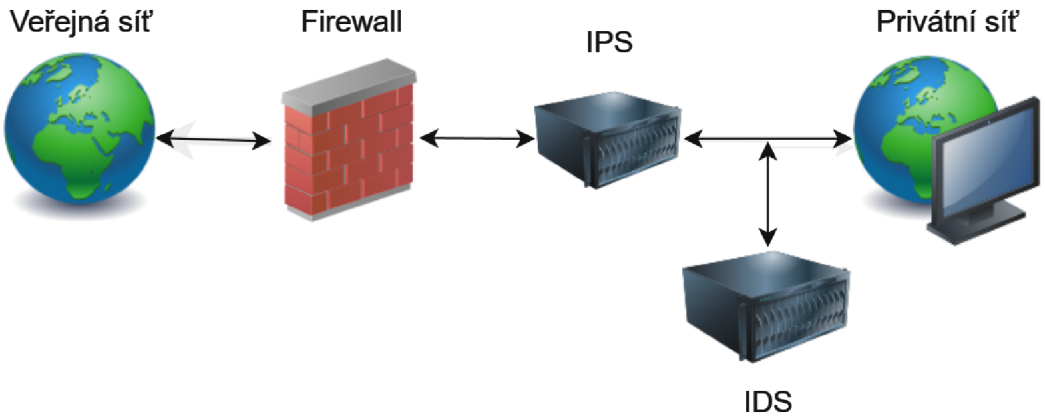
Chtěl bych poděkovat svému vedoucímu bakalářské práce Mgr. Radku Janoštíkovi, Ph.D. za odborné vedení, za pomoc a rady při zpracování této práce.

Odevzdáním tohoto textu jeho autor/ka místopřísežně prohlašuje, že celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

Obsah

1 IDS/IPS systémy

Systémy detekce a prevence úniků (IDS/IPS, Intrusion Detection Prevention Systems) jsou síťová bezpečnostní zařízení, která zkoumají příchozí a odchozí síťový provoz a porovnávají jej s podezřelým, předem známým vzorem – vytvořeným pravidlem (signaturou), které může indikovat síťovou, systémovou nebo i jinou škodlivou činnost. Oba tyto systémy spolu souvisejí, kdy IPS je rozšířená varianta IDS [1]. Jejich obecné schématické zapojení lze vidět na obrázku 1, kde IPS systém musí být zapojen přímo v toku komunikace. Ta proudí skrze tento systém, aby mohl případnou škodlivou komunikaci IPS systém ihned blokovat. Oproti tomu IDS systém je zapojen jako další větev komunikace, do které je veškerý provoz zrcadlen a až nad tímto provozem se provádí analýza a případná detekce podezřelé činnosti. Hlavním rozdílem je to, že IPS vysílá výstrahu a zároveň blokuje podezřelý provoz, zatímco IDS vydává pouze výstrahu.



Obrázek 1: Základní zapojení IDS/IPS systémů

Obecně IDS/IPS systémy mohou vydávat čtyři druhy výstrah. Jedná se konkrétně o *True-positive (útok-výstraha)*, kdy v tomto případě se jedná o reálný útok a systém vydal správnou výstrahu, na základě správného pravidla, popřípadě blokoval správný provoz. Jedná se tedy o správnou detekci. Druhým typem je *False-positive (žádný útok-výstraha)*, kdy se jedná o případ, ve kterém nedošlo k žádnému útoku, avšak systém vydal výstrahu o útoku. Třetím typem výstrahy je *False-negative (útok-žádná výstraha)*, kdy při reálném útoku není vydána výstraha. Posledním typem je *True-negative (žádný útok-žádná výstraha)*, kdy se jedná o druhé správné chování IDS/IPS systému [1]. Příčin nechtěných výstrah může být mnoho a většinou je důležité najít důvod, proč je tomu tak v dané konkrétní síti. Je nutné analyzovat jednotlivá pravidla a upravit jejich strukturu tak, aby byla navržena přímo pro danou síť a popřípadě některá i zakázat.

1.1 Intrusion Detection System (IDS)

Jedná se o bezpečnostní síťové zařízení, které může monitorovat jak příchozí, tak i odchozí provoz v privátní síti. Zapojení IDS v rámci sítě má svá specifika a jak již bylo popsáno výše, tak přes IDS sondu prochází pouze zrcadlený provoz. IDS tento provoz zpracovává a analyzuje a v případě detekce problému vydává výstrahu (alert) ohledně potenciální hrozby. Tento samotný alert může mít různou podobu. Může se jednat o email, zvukovou výstrahu, SMS zprávu, avšak ve většině případů se tato výstraha zobrazí v nějakém jiném centrálním systému např. SIEM (Security Information and Event Management). Na základě toho, jakým způsobem je detekce prováděna, lze tyto IDS rozdělit na tzv. [1]:

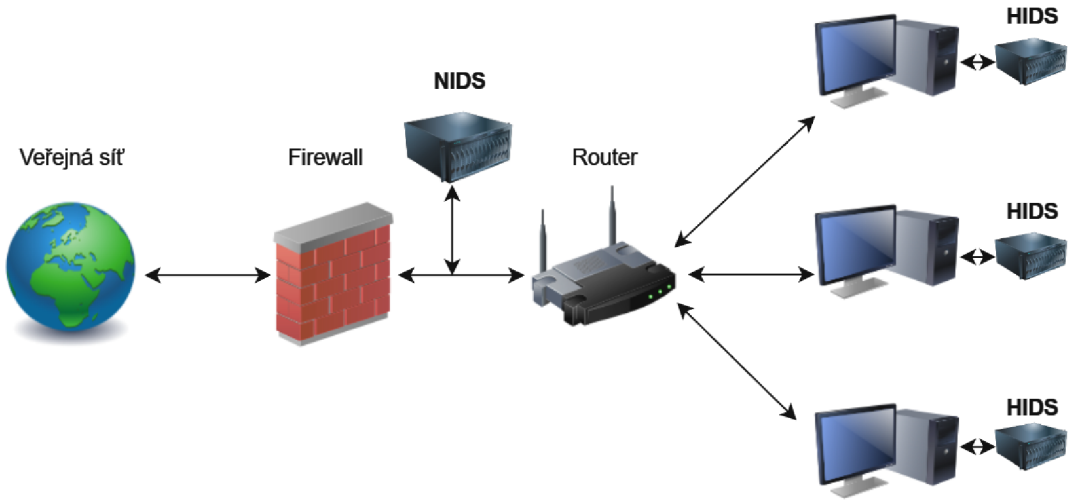
- **Signature-based IDS** – detekce je založena na základě sady předem definovaných pravidel – signatur. Tyto signatury se skládají ze vzorů, které označují jednoznačnou škodlivou činnost a ty se dále skládají z indikátorů kompromitace (IoC, Indicator of Compromise). Vzory jsou porovnávány s probíhající komunikací, datovými pakety a v případě shody, je vydána výstraha. Tvorba těchto vzorů je založena například na základě proběhlých útoků, známého chování malwaru nebo zranitelnostech hardwaru i softwaru. Každá z těchto činností se tedy vyznačuje jinými indikátory kompromitace.
- **Anomaly-based IDS** – tyto IDS jsou založeny na základě pozorování normálního provozu a sledují a vydávají výstrahy v případě odchylky od tohoto provozu. Pojem normální chování je velmi individuální a záleží na mnoha faktorech, například uživatelích, aplikacích, využívání jednotlivých služeb, apod. Jako příklad lze uvést nadměrný počet odeslaných emailů nebo velký počet špatných autentizací jednoho uživatele. Na rozdíl od signature-based IDS, mohou tyto IDS detekovat také dosud neznámé útoky [1].

Tyto systémy lze rozdělit také na základě toho, kde jsou umístěny a jaký provoz analyzují. Jedná se o tři kategorie [1]:

- **NIDS (Network IDS)** – tyto senzory jsou umístěny na klíčových místech, ze kterých je možné monitorovat celou nebo část privátní sítě. Jedná se například o umístění před nebo za firewallem, v blízkosti routerů a další místa, tak jako to lze vidět na následujícím obrázku 2. Tento systém musí pracovat v promiskuitním režimu, ve kterém zařízení zachytává síťovou komunikaci, která není přímo pro něj určena, jinými slovy síťová karta tohoto zařízení přijímá všechny rámce v síti a ne jen ty, které jsou určeny pro dané zařízení (konkrétní MAC adresu) [2].
- **HIDS (Host IDS)** – tyto senzory fungují především na samotných zařízeních, tak jako je znázorněno na obrázku 2, na kterých jsou umístěna. Dochází k monitorování pouze daného zařízení. Systém většinou

pracuje v nepromiskuitním módu, neboli přijímá jen rámce určené pro dané zařízení (MAC adresu) [2].

- **Hybrid IDS** – Jedná se o kombinaci obou předchozích řešení, kdy sondy jsou umístěny na koncových zařízeních, avšak mají možnost monitorovat část, popř. celou síť [1].



Obrázek 2: Zapojení NIDS a HIDS [3]

IDS systémy lze dělit podle mnoha dalších kritérií například podle struktury, kde se může jednat o distribuované nebo centralizované systémy, podle zdroje dat, která mohou být ve formě výsledku nějakého auditu, síťového paketu nebo systémových stavů. Posledním kritériem může být pomocí průběhu monitoringu z hlediska času, kdy se může jednat o monitoring intervalový nebo o monitoring v reálném čase [1].

1.2 Intrusion Prevention System (IPS)

Systém IPS lze považovat za rozšíření IDS systému. Ve většině případů také jednotlivá dostupná řešení umožňují přepínání mezi těmito systémy a správce si tak může vybrat, ve kterém módu bude daná sonda pracovat. V mnoha případech je však prospěšné na různých místech v síti implementovat obě řešení. Hlavním rozdílem oproti IDS je v tom, že umožňuje nejen vydávat výstrahu, ale rovnou také komunikaci ukončovat/blokovat. Vše se opět děje v závislosti na stejných pravidlech – signaturách. Druhým zásadním rozdílem je, že IPS je zapojeno přímo v cestě komunikace, tudíž veškerý provoz jde přes tuto sondu. Na základě toho lze také tvrdit, že IPS systém je dalším možným Single Point of Failure (SPOF) a je třeba tento fakt brát v úvahu již při samotném návrhu sítě a implementovat takové mechanismy, které umožní síťovou komunikaci i v případě poruchy tohoto IPS zařízení. Rozdílné jsou také požadavky na hardwarové zařízení, kdy systém

IDS vyžaduje především větší úložný prostor pro ukládání logů, zatímco IPS musí disponovat výkonějším procesorem a větší pamětí RAM pro zpracování komunikace v reálném čas [1].

1.3 Komponenty IDS/IPS systému

Celkový IDS/IPS systém se skládá z komponent, které zabezpečují různé funkce. Ty zahrnují sběr informací a dat z mnoha různých systémů a síťových zdrojů, až po analýzu těchto dat. Mezi hlavní komponenty tedy patří [1]:

- Síťový senzor
- Analytický nástroj
- Výstražný systém
- Ovládací konzole
- Systém odezvy
- Databáze pravidel – signatur nebo chování

Síťový senzor je hardwarové nebo softwarové zařízení, které je připojeno v síti a předává data k analýze IDS, tzn. pracuje jako sběrné místo dat. Tento senzor je zodpovědný také za formátování předávaných dat pro jednodušší zpracování. Nejčastěji se tedy data dále předávají ve formě jednotlivých událostí [1].

Ovládací konzole je software, který poskytuje grafické rozhraní pro ovládání, analýzu a sledování událostí IDS/IPS systému, zatímco výstražný systém umožňuje sledovat vydané výstrahy a sledovat tak potenciální hrozby v síti. Systém odezvy tedy souvisí s výstražným systémem a na základě vydaných výstrah dělá proti opatření jako například může být blokování dotčených účtů, blokování zdrojových adres útočníků, restartování zařízení nebo služeb a mnoho dalších [1].

Poslední komponenta, databáze pravidel, slouží jako úložiště pro všechna pravidla, která jsou následně použita systémem pro analýzu a detekci hrozeb a anomálií v provozu. Samotná pravidla jsou tématem následujících kapitol této práce.

1.4 Dostupná řešení

V této kapitole jsou popsána některá dostupná open-source řešení IDS/IPS systémů a jejich popis. Nejsou zde zahrnuta komerční řešení těchto systémů.

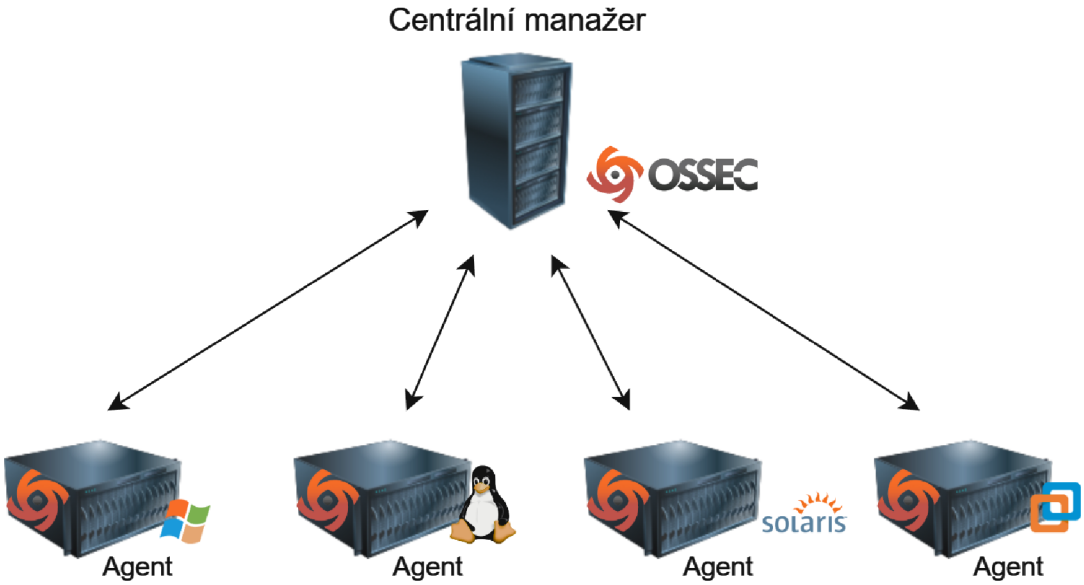
1.4.1 OSSEC

Jedná se o open-source platformu určenou k monitoringu a kontrole systémů. Zahrnuje nástroje pro IDS systémy typu HIDS, které umožňují analýzu logů, kontrolu integrity, monitorování registrů operačního systému Windows, detekci rootkitů, vydávaní výstrah a odezev v reálném čase. V této platformě je zahrnut také systém SIEM, pro lepší práci s výstupy. Tento systém dokáže běžet na mnoha operačních systémech, například Linux, OpenBSD, FreeBSD, macOS, Solarix nebo Windows. Tato platforma může také být implementována ve virtuálním prostředí [4].

OSSEC se skládá z několika částí, kdy hlavní část tvoří centrální manager pro monitoring a přijímání informací od agentů, syslogů, databází a od tzv. *agentless* zařízení, tak jako je to možné vidět na následujícím obrázku 3. Jsou zde také uložena veškerá pravidla, dekodéry pro extrahování informací z logů, hlavní nastavení a také nástroj pro hromadnou správu agentů [4].

Agent je malý program, popřípadě více programů, které jsou nainstalovány v systému, který má být monitorován. Tato část sbírá informace a předává je dále centrálnímu manažeru k provádění analýz. Některé informace mohou být sbírány v reálném čase, ostatní v časových intervalech. Tento agent komunikuje s centrálním manažerem na portu 1514/udp [4].

Agentless zařízení je takové zařízení, na kterém nemůže být agent nainstalován například z důvodu kompatibility, omezených systémových zdrojů nebo bezpečnosti. Toto zařízení poskytuje pouze kontrolu integrity dat, která je prováděna pomocí kontrolních součtů nebo sledování rozdílů dat. Používá se pro monitoring firewallů, routerů nebo i Linux/BSD systémů [4].



Obrázek 3: Architektura OSSEC [4]

K analýze nežádoucího provozu je potřeba mít definována nějaká pravidla.

Ta se skládají z různých částí a podmínek a jsou porovnávána s logy, kdy v případě shody dojde k vydání výstrahy. Pravidla musí mít přesně danou strukturu a pro platformu OSSEC jsou definována pomocí značkovacího jazyka XML, což lze vidět ve zdrojovém kódu 1.

Syntaxe k těmto pravidlům je přesně definována, kdy v uvedeném příkladu lze vidět v prvním řádku definování skupiny pravidel a jejich názvu, ve druhém řádku následuje identifikační číslo samotného pravidla a úroveň závažnosti, kdy těchto úrovní je definováno 16 (0 až 15), v další části jsou podmínky, kdy pomocí přepínače *if_sid* je dána podmínka, že pravidlo bude provedeno pouze pokud pravidlo mezi těmito přepínači úspěšně detekovalo pravidlo jiné, zde s číslem 31100. Značí to vázanost na jiné pravidlo. Mezi dalším přepínačem, *url_pcre2* se již nachází samotný vzor, který se porovnává s daty a v případě jeho nalezení je pravidlo splněno a vyslána další akce – porovnání jiného pravidla, výstraha a další. V tomto případě se jedná o vzor ve formátu PCRE – Perl Compatible Regular Expressions. Následuje už pouze název samotného pravidla a název podskupiny. Další syntaxe je v oficiální dokumentaci pro psaní pravidel [5].

```
1  <group name="web,accesslog,">
2    ...
3    <rule id="31164" level="6">
4      <if_sid>31100</if_sid>
5        <url_pcre2>=%27|select%2B|insert%2B|%2Bfrom%2B|
6             %2Bwhere%2B|%2Bunion%2B</url_pcre2>
7        <description>SQL injection attempt.</description>
8      <group>attack,sqlinjection,</group>
9    </rule>
10   ...
11 </group> <!-- Web access log -->
```

Zdrojový kód 1: Příklad pravidla pro OSSEC platformu

1.4.2 ZEEK (Bro)

ZEEK je open-source pasivní síťový analyzér, který slouží k podpoře vyšetřování podezřelých aktivit. Tento systém obsahuje i spoustu ostatních funkcí, například k řešení systémových problémů. ZEEK oproti jiným systémům využívá logů, které obsahují mimo základní informace mnoho rozšířených informací popisujících síťový provoz, jako například HTTP relace, DNS dotazy a odpovědi, MIME typy, odpovědi serverů, SSL certifikáty, klíčový obsah SMTP relací
a mnoho dalších. Veškeré informace jsou zapisovány ve struktuře JSON a lze je velmi dobře využít pro další zpracování. Tento systém obsahuje velké množství vestavěných funkcí, které dokáží například extrahovat soubory z HTTP komunikace, detekovat malware za pomocí externích zdrojů, hlásit zranitelnosti

softwaru detekovaného v síti, identifikovat populární aplikace a mnoho dalších funkcí [6].

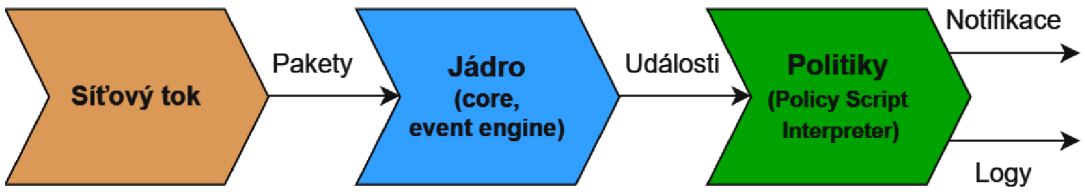
Tento systém je plně modifikovatelný a rozšiřitelný, a to z důvodu, že je založen na Turingovsky úplném programovacím jazyku pro vyjádření analytických úloh. Obsahuje již předpřipravené knihovny, avšak je zde možnost vytvářet si analytické úlohy pomocí vlastního kódu. Veškeré základní analýzy jsou definovány pomocí skriptů. Na oficiálních stránkách je označeno, že se nejedná o klasický systém IDS založený na pravidlech, i když tuto funkci také umožňuje, ale především z důvodu využití skriptovacího jazyka zahrnuje širší spektrum přístupů k nalezení podezřelých aktivit [6].

ZEEK se zaměřuje především na monitorování vysokorychlostních sítí, jako jsou například sítě superpočítačových center, univerzitní, laboratorní nebo vládní sítě. Pro zpracování těchto velkých objemů dat ZEEK také obsahuje nastavitelný load-balancer, který dokáže zátěž rozdělit na několik výkonných prvků a pomocí jednoho centrálního prvku poskytuje koordinaci, synchronizaci, konfigurací a centrální správu všech těchto prvků [6].

Architektura

Celkovou architekturu lze vidět na následujícím obrázku 4. Základem celého systému je jádro (core, event engine), které převádí síťové pakety na sérii událostí. Jádro také obsahuje části, které zabezpečují paketovou analýzu, která provádí analýzu již od protokolů na linkové vrstvě, analýzu navázání spojení (session), která zahrnuje analýzu protokolů na aplikační vrstvě, a také analýzu souborů [6].

Druhou důležitou součástí je interpret politik (Policy Script Interpreter). Ten na události aplikuje skripty, které jakýmsi způsobem definují politiky v dané síti a v případě problému vydává notifikaci a výsledné logy, tak jako je znázorněno na následujícím obrázku 4 [6].



Obrázek 4: Architektura ZEEK [6]

Samotné skripty jsou napsány pomocí vlastního skriptovacího jazyka, což umožňuje dokonalé přizpůsobení politik vůči dané síti. Umožňuje to také rozšíření funkcionality samotného celého systému a kvůli tomuto také často tento systém není označen jako klasický IDS založený na pravidlech [6].

1.4.3 Suricata

Suricata je vysoce výkonný IDS/IPS systém a může pracovat také jako nástroj k bezpečnostnímu sledování sítě (NSM, Network Security Monitoring), kdy již jedna instance dokáže zpracovat obrovské množství provozu, v řádech GB/s, a to především z důvodu, že dokáže využívat více vláknovou detekci. Je tedy mnohem výkonnější než velmi podobný systém Snort. Jedná se o open-source řešení, které je spravováno nadací OISF (Open Information Security Foundation). Vytváří vysokou úroveň detailních informací z aplikační vrstvy v rámci síťového provozu. Umožňuje také provádět pokročilou analýzu za pomocí skriptovacího jazyka Lua [7].

Obecně lze systém Suricata nasadit jako IDS, IPS, NSM (zabezpečení logů, toků, popř. souborů), FPC (Full Packet Capture) nebo CPC (Conditional Packet Capture), tak jako je znázorněno na následujícím obrázku 5. Je možné využít také různých kombinací těchto schopností [8].

Obrázek 5: Schopnosti systému Suricata [8]

V případě provozování systému IDS/IPS se jedná o systém založený na pravidlech, kdy existují dvě verze těchto pravidel, konkrétně skupina neplacených pravidel, které jsou zaměřeny na detekci známých a obecných principech škodlivé činnosti a skupina placených, profesionálních pravidel, které jsou určeny k detekci například již konkrétního typu a projevu malwaru. V případě také této implementace musí být dodrženo jednotlivé rozlišné zapojení obou systémů tzn. v případě IPS přímo v cestě síťového toku a v případě IDS jako pasivní sonda analyzující zrcadlený provoz.

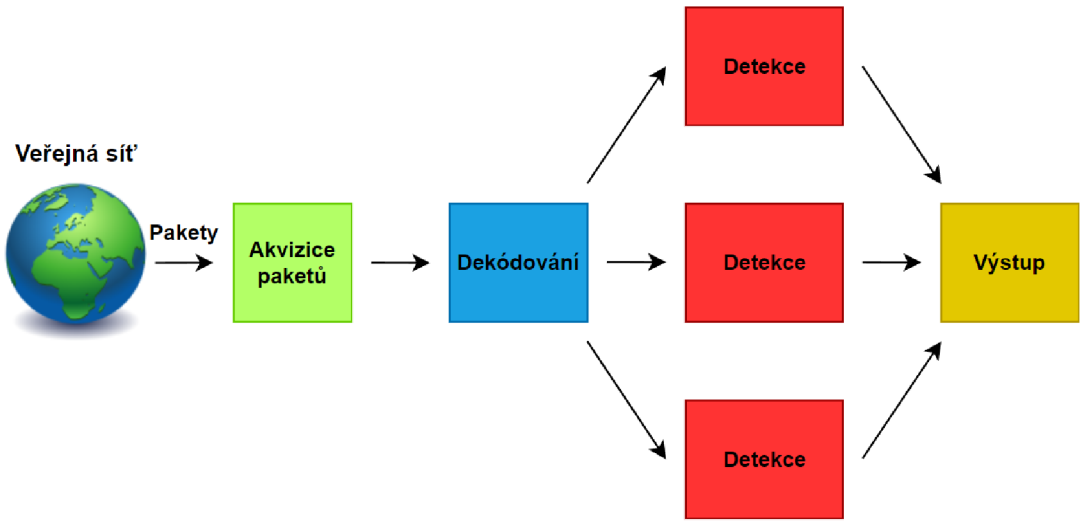
V režimu NSM nejsou využívána pravidla, ale jedná se o zpracování dat z provozu sítě jako je zpracování logů, ukládání TLS certifikátů, extrahování souborů z provozu a další. Suricata také disponuje automatickým detekováním protokolů a portů. Veškeré výsledky lze získat ve formátu JSON a tudíž je možné data dále jednoduše využít např. v nástrojích Splunk, Scirius (Evebox) nebo Kibana [8].

Suricatu lze nainstalovat na různé operační systémy, konkrétně Windows,

Linux, FreeBSD, Unix a macOS. Lze na ni narazit také jako součást operačního systému OPNsense firewall, což je představeno v kapitole 3. Tento systém využívá pravidla, která mají stejnou syntaxi jako následující systém Snort. Jejich popis bude obsahem následujících kapitol.

Architektura

Schématická architektura sytému Suricata je znázorněna na obrázku 6. Jak lze vidět, tak v první části jsou pakety získávány a prvotně zpracovávány z provozu na síti. Následně jsou řazeny a dekódovány na základě IP adresy, portů, protokolů a dalších atributů a jsou poskytovány detekčním blokům. Jak lze vidět, tak je zde znázorněna více vláknová detekce, tudíž je možné zpracovat velké množství dat. V těchto blocích dochází k porovnávání paketů se skupinou pravidel a v případě shody je uveden výstup, ať už v podobě výstrahy, logu nebo blokace komunikace [9].



Obrázek 6: Suricata architektura [9]

1.4.4 Snort

Jedná se o open-source IDS/IPS systém, který je založen na pravidlech, to znamená, že pomocí předem definovaných pravidel dokáže detekovat/blokovat škodlivou činnost v síti a informovat o provedené akci. Celý tento sytém byl navržen jako jednoduchý software, který by mohl běžet i na méně výkonných zařízení a využívá tedy pouze jedno vláknovou detekci, proto nedosahuje takového výkonu jako předchozí systém Suricata. Snort primárně dokáže pracovat ve třech módech, konkrétně jako [10]:

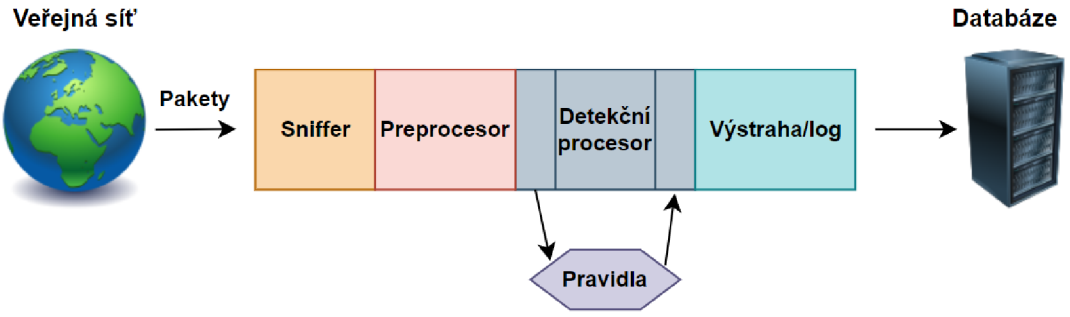
- **Packet sniffer** – zachytává v reálném čase všechny pakety a poskytuje je k dalšímu zpracování

- **Packet logger** – vytváří z provozu logy, záznamy, které poté poskytuje k dalšímu zpracování
- **IDS/IPS systém**

Snort podporuje mnoho operačních systémů, například Ubuntu, CentOS, FreeBSD, Debian nebo Fedora. Protože je tento systém založen na pravidlech, tak i v tomto případě zde je několik verzí pravidel, kdy první z nich, a také nejvíce obecná, jsou pravidla dostupná zdarma, druhou skupinou jsou pravidla dostupná po registraci a poslední jsou konkrétněji zaměřena pravidla na konkrétní útoky nebo projevy malwaru, která již je nutno zaplatit. Tato pravidla mají stejnou syntaxi jako pravidla u předešlého systému Suricata a jejich popis je součástí následujících kapitol [10] [11].

Architektura

Jak lze vidět na následujícím obrázku 7, tak základní architektura tohoto systému se skládá ze snifferu, který slouží k samotnému připojení k síti, zachytávání paketů provozu a poskytuje také prvotní zpracování těchto paketů. Tyto pakety jsou dále předány do preprocesoru, kde dochází k dalšímu zpracování, třídění a předzpracování i za pomocí rozšířených funkcí tohoto preprocesoru. Může se jednat například o rozšíření pro skenování portů nebo HTTP komunikace. Následně jsou roztříděné pakety pomocí detekčního procesoru (detection engine) testovány na shodu se skupinou pravidel (ruleset). Pokud dojde ke shodě v paketu s některým z pravidel, tak se následně z toho paketu udělá výstraha nebo log v závislosti, v jakém módu je Snort nastaven. Tyto výstrahy nebo logy jsou dále ukládány do různých databázových systémů a jsou dále poskytovány dalším nástrojům ke zpracování. V režimu IPS by zde byla tato komunikace blokována. V případě, že ke shodě nedojde, paket v detekčním procesoru končí a výstraha ani log se nevytvoří [2].



Obrázek 7: Snort architektura [2]

2 Tvorba a analýza pravidel

Celá tato práce se zabývá především pravidly pro IDS/IPS systémy, jejich tvorbou a implementací, a proto v této kapitole jsou samotná pravidla a jejich syntaxe popsána podrobněji. Jedná se o pravidla pro IDS/IPS systémy Suricata a Snort, kdy oba systémy, až na pár výjimek, využívají téměř totožnou syntaxi. Větší důraz je zde kladen na pravidla systému Suricata, kdy i samotný software pro tvorbu a analýzu pravidel je vytvořen primárně pro tento systém.

Těmto pravidlům se často říká signatury. Oba tyto pojmy jsou zaměnitelné a mají v této práci stejný význam. Velmi obecně a abstraktně lze říct, že signatura je způsob, jak lze popsat stav sítě. Existují jak základní jednoduchá pravidla, popisující především základní stavy sítě, tak i pravidla velmi složitá, která se již soustředí pouze na specifické stavy sítě. Proto, aby bylo možné porozumět stavům sítě (upozorněním), tak je nutné znát syntaxi a tvorbu těchto pravidel. Jako příklad lze uvést slovní popis stavu sítě (upozornění) [2]:

Pokud paket obsahuje řetězec \cmd.exe a směřuje k IIS webovému serveru v DMZ, vytvoř upozornění, informuj bezpečnostní tým a zachyť paket pro další analýzu [2].

Lze vidět, že se jedná o specifické pravidlo, které zaznamenává jasný stav sítě. Aby však tomu rozuměly i IDS/IPS systémy a bylo možné vytvořit mnoho jiných pravidel je nutné jej převést do jednotného stylu. Snort a Suricata využívají velmi podobnou syntaxi a předchozí pravidlo převedené do této podoby vypadá následovně [2]:

```
alert tcp $EXTERNAL_NET any -> $IIS_WEB_SERVERS
$HTTP_PORTS ( msg:"/cmd.exe going to the IIS Webserver";
flow:established, to_server; content:"/cmd.exe";
depth:30; metadata:created_at 2023_09_07, updated_at
2023_09_07; )
```

Pravidlo má přesně danou strukturu, kdy obsahuje jak prováděnou akci v případě shody (alert), tak komunikující strany a za ní mnoho dalších argumentů. Popis jednotlivých částí a argumentů pravidel je součástí následujících kapitol.

Pro IDS/IPS systém Suricata existuje několik skupin pravidel takzvaných rulesetů. Jedná se o soubor s příponou *.rules*, který obsahuje několik samostatných pravidel (od jednotek až po tisíce). Často jsou tyto rulesety rozděleny podle různých kritérií, jako například pro různé druhy útoků, různé síťové komunikace a další. Mezi nejpoužívanější a nejrozšířenější patří rulesety od společnosti Proofpoint. Ty jsou označeny jako ET OPEN (Emergency Threat OPEN) a obsahují signatury začínající stejným označením (často jen pouze ET). Tyto rulesety jsou volně ke stažení a umožňují detekovat obecně známé hrozby, síťové anomálie a obecnější i konkrétnější stavy sítě. Druhou skupinou jsou

rulesety označeny jako ET PRO. Jedná se již o placená pravidla. Ta jsou tvořena profesionálními bezpečnostními týmy a jsou zaměřena na konkrétní specifické hrozby a anomálie. Mimo tyto dva rulesety existuje také velké množství jiných volně dostupných i placených řešení. Jsou označeny ve tvaru *tvůrce/oblast* například *oisf/trafficid* nebo *scwx/malware.*

2.1 Základní části pravidel

Tato kapitola se zabývá detailnějším popisem pravidel v tzv. formátu Snort. Jsou zde také vysvětleny jednotlivé argumenty pravidel a jejich parametry. Tento základní formát pravidel využívají IDS/IPS systémy Snort a Suricata, kdy však lze u specifických pravidel jednotlivých systémů najít malé rozdíly, což je dáno různými funkcemi těchto systémů. Práce se tedy více zaměřuje na systém Suricata a vychází také z jeho oficiální dokumentace [8].

Pro samotnou tvorbu signatur je nejvhodnější využít editor zdrojového kódu Visual Studio Code spolu s rozšířením Suricata Language Server, a to z důvodu, že se jedná o hotové řešení a jeho implementace je velmi jednoduchá. Suricata Language Server poskytuje kontrolu syntaxe, autokompletaci pravidel a nápovědu ihned při psaní signatur. Tvorba signatur tímto způsobem je nejen rychlejší, ale lze tím také předejít nechtěným chybám. Samotné zprovoznění je velmi jednoduché a lze jej přidat přímo v editoru.

Obecně se tato pravidla skládají ze 4 hlavních částí. Konkrétně z akce (vyznačeno červenou barvou), hlavičky (vyznačeno modrou barvou) a parametrů (vyznačeno zelenou barvou) oddělených středníkem. Jako další součást se uvádí metadata (vyznačenou fialovou barvou), která jsou součástí parametrů a společně jsou umístěny v závorce, tak jako je znázorněno na následujícím pravidle. Metadata mají pouze informační charakter, který vypovídá o daném pravidle a nemají vliv na samotnou analýzu paketu [2] [8]. Barevné označení lze vidět také na příkladu výše.

```
alert hlavička (parametry; metadata)
```

2.1.1 Akce

Jak lze vidět na předchozím příkladu, tak první část akce se skládá pouze z jednoho slova, které může být zvoleno v závislosti na požadované akci v případě zachycení tímto pravidlem z argumentů [8]:

- *alert* – je vytvořeno upozornění
- *pass* – je zastavena další analýza daného paketu (pro IPS systém)
- *drop* – paket je zahozen a je vytvořeno upozornění (pro IPS systém)
- *reject* nebo *rejectsrc* – odesílateli je poslána chyba (RST/ICMP)
- *rejectdst* – příjemci je poslána chyba (RST/ICMP)

- *rejectboth* – chyba je poslána na obě strany (RST/ICMP)

2.1.2 Hlavička

V druhé části, hlavičce, se jako první element vyskytuje použitý protokol komunikace (označeno na následujícím příkladu oranžovou barvou), což znamená, pro jaký provoz je pravidlo určeno a na kterém druhu provozu bude testováno, což lze odvodit ze samotné zkratky protokolu. Mohou se zde vyskytovat hodnoty jako například *tcp, icmp, udp, http* (pro verze 1 i 2 nebo lze i samostatně http1 nebo http2), *dns, smb, ssh, dhcp, imap, smtp* a mnoho dalších. Speciální hodnotou je *ip*, která zde zastupuje všechny protokoly a tudíž, tohle pravidlo je testováno na každou komunikaci. Mohou se zde objevit také další dvě speciální hodnoty, a to *tcp-pkt* a *tcp-stream*, kdy první z nich je testována pouze v obsahu jednotlivých paketů, zatímco druhá hodnota je testována na celém jednom spojení. Dostupnost všech těchto paketů je závislá na jejich nastavení v konfiguračním souboru *suricata.yaml* [8].

Dalšími hodnotami, co obsahuje hlavička jsou hodnoty zdrojových a cílových adres včetně jejich portů. První hodnotou v pořadí je hodnota zdrojové adresy a druhá hodnota je cílová adresa (označeny na následujícím příkladu zelenou barvou). Hodnoty adres mohou být v různých formách, ať už v klasické formě IP adresy (např. 10.1.1.2), CIDR formátu (např. 10.0.0.0/16), skupiny konkrétních IP adres (např. [10.1.1.1, 10.1.1.2]) nebo negace (např.!10.1.1.2), kdy jsou brány v potaz všechny IP adresy kromě uvedené. Hodnoty mohou být také ve formě proměnných, což lze vidět v následujícím příkladu. Tyto proměnné vždy začínají znakem $ a jejich hodnoty jsou součástí jiného samostatného souboru. Jednotlivé formy pro definici zdrojových a cílových adres lze také kombinovat. Velmi často se také používá hodnota *any*, která zastupuje jakoukoliv adresu [8].

Hodnoty portů jsou umístěny ihned za zdrojovou, respektive cílovou adresou (označeno na následujícím příkladu modrou barvou). Hodnoty portů také (většinou) definují o jaký typ komunikace se jedná. Při psaní pravidel se většinou klade větší důraz na port cílový. Tyto hodnoty opět mohou být v různých formách jako konkrétní číslo (např. 80), rozsah čísel (např. [80: 82]), skupina čísel (např. [80, 81, 82]), negace (např. [80:100,!99]) a nebo proměnných jako je v případě příkladu na zdrojovém kódu 3 - *$HTTP_PORTS*. I v tomto případě lze využít hodnotu *any*, pro zastoupení všech portů [8].

```
tcp $EXTERNAL_NET any -> $IIS_WEB_SERVERS $HTTP_PORTS
```

Poslední součástí hlavičky je určení směru komunikace, která má být testována (označeno červenou barvou). Ten je udáván pouze pomocí znaků -> a <>, kdy první z nich znázorňuje testování na proudu paketů jedním směrem a druhý proudu paketů oba směry [8].

2.1.3 Parametry

Jak již bylo zmíněno dříve, jedná se o tu část, která se nachází v závorce a jednotlivé elementy, které jsou specifikovány klíčovým slovem (keyword), jsou od sebe odděleny středníkem. Jde o tu část, která udává celému pravidlu jeho požadované vlastnosti na detekci provozu. Existují dva typy těchto elementů, a to ty, které mohou mít nějakou proměnou (např. msg: "Tohle je název pravidla";), nebo ty, které již žádnou další proměnou nemají (např. nocase;) a sami o osobě udávají nějakou vlastnost danému pravidlu. Všechny tyto elementy mají v rámci pravidla danou pozici a pořadí a jakákoliv změna by změnila význam celého pravidla. Těchto elementů určených klíčovým slovem existuje obrovské množství a v této práci jsou představeny pouze některé z nich, konkrétně ty, které porovnávají obsah dat paketu (payload). Je to z důvodu jejich implementace do nástroje pro analýzu pravidel. Souhrn všech možných elementů lze najít v oficiální dokumentaci pro IDS/IPS systém Suricata [8].

Mezi jedny z nečastějších, nejdůležitějších a nejvíce vyskytujících se elementů, patří elementy týkající se testování obsahu dat paketu (skupina Payload keyword). Jedná se o element *content*, který slouží k definování obsahu, na který mají být data paketu testována, jestli se v něm daný obsah vyskytuje, například *content:"..priklad.com.."*;. V pravidle může být těchto elementů i více. Pracují na principu porovnávání jednotlivých bajtů dat paketu s danou hodnotou. Daná hodnota toho elementu může mít různé podoby, ať už klasický textový řetězec, hexadecimální podobu i jejich kombinace [8].

V základním nastavení tento element bere ohled na velká a malá písmena. To lze však jednoduše změnit vložením přepínače *nocase* do signatury. Dalším modifikátorem tohoto elementu (content) je element *depth*, pomocí kterého lze nastavit kolik bajtů od počátku dat paketu bude testováno na daný obsah. Obsahuje absolutní hodnotu, tj. číselnou proměnnou např. *depth:4;* znamená, že obsah dat paketu bude testován na daný content pouze do 4 bajtu. Dalším modifikátorem je element *offset*, který obsahuje číselný parametr, který určuje absolutní hodnotu, a to od kterého bajtu od začátku dat paketu se budou bajty porovnávat, např. *offtes:4;* určuje, že content bude porovnáván až od 4 bajtu [8].

Existují také modifikátory s relativní hodnotou, konkrétně *distance* a *within*. Oba elementy se používají v případě pravidla s více content elementy, kdy první z nich udává vzdálenost od prvního content elementu, ve které mají být porovnávány bajty na obsah content elementu druhého, například *content:"abcd"; content:"efgh"; distance: 3;* znamená, že v případě nalezení shody v datech paketu s obsahem prvního content elementu, jsou následující 3 bajty vynechány a ve zbylých datech jsou další bajty porovnávany s obsahem druhého content elementu. Hodnota distance elementu může být i záporná, a tudíž se mohou oba obsahy i překrývat. Druhý element *within* udává hodnotu, jak daleko je druhý obsah elementu content vzdálen od prvního, respektive kolik dalších bajtů bude od prvního nalezeného obsahu následně porovnáváno [8].

V rámci této skupiny elementů sem patří také důležitý element *pcre*, který umožňuje využít regulární výrazy (Perl Compatible Regular Expressions). Velmi

často se využívá spolu s elementem content. Obsahuje také mnoho přepínačů, které mohou měnit význam regulárního výrazu, celý jejich seznam včetně popisu je uveden v oficiální dokumentaci [8].

Jak již bylo napsáno, skupin elementů existuje velké množství a jsou členěny do skupin, které jsou odvozeny z oblastí, kterých se týkají, například IP, TCP, UDP, ICMP, HTTP, soubory, DNS, SSL/TLS a mnoho dalších [8]. Tyto elementy kopírují možnosti provozu a nastavení v těchto jednotlivých oblastech, které mohou tyto specifické komunikace nabývat. Signatura pro detekci může tedy být založena nejen na detekci dat paketu ale i mnoha ostatních aspektech, jako například kódu a typu ICMP provozu, určité cílové IP adrese, konkrétním typu DNS dotazu, konkrétním typu Kerberos zprávy a mnoho dalších, které lze spolu i kombinovat, samozřejmě v závislosti na povaze provozu a chtěném výsledku.

2.1.4 Metadata

Metadata mají pouze informační charakter a nemají vliv na analýzu paketu. Mezi metadata se řadí elementy [8]:

- *msg* (message) – definuje název pravidla např. msg:"ET MALWARE Turkojan C&C Initial Checkin (ams)";
- *sid* (signature ID) – přiřazuje pravidlu jedinečné ID číslo např. sid:30002;
- *rev* (revision) – udává verzi daného pravidla, tj. počet modifikací např. rev:2;
- *gid* (group ID) – používá se k přidělení nového ID různým skupinám signatur
- *classtype* – určuje kategorii pravidla a jeho prioritu, která je určena k příslušné kategorii v souboru classification.config, např. classtype:trojan-activity;
- *reference* – odkazuje na původ signatury (důvod vzniku), kdy první proměnná je typ odkazu a za ním samotný odkaz např. reference:cve,2022-30525;
- *priority* – definováno automaticky na základě classtype, kdy lze tento element i přepsat vlastním vepsáním přímo do pravidla – rozsah 1-255 (1 je nejvyšší priorita), pravidla s nejvyšší prioritou jsou testována jako první
- *metadata* – tento element umožňuje přidat další, nefunkční informace
- *target* – specifikuje, která strana je cílem útoku, v případě alertu se tedy připíše k názvu, např. target:[src_ip|dest_ip]
- patří sem také elementy ohledně data vytvoření a aktualizace pravidla a další

Všechny tyto elementy mají v rámci signatury pevně stanovené místo, kde se nacházejí, kdy například element *msg* se nachází na prvním místě závorky, zatímco ostatní elementy na jejím konci, v pořadí od konce závorky *rev, sid, classtype* [8].

2.2 Software pro analýzu pravidel

Jedná se o nástroj analýzu pravidel určených především pro systém Suricata. Je možné jej použít i pro pravidla systému Snort, která sice mají malé odlišnosti v syntaxi, avšak základní analýza, na kterou je i tento nástroj určen, je shodná.

2.2.1 Požadavky na software

Celkově by tento analytický nástroj měl poskytovat podporu při analýzách bezpečnostních událostí a incidentů, kdy jeho úkolem je urychlit analýzu zaznamenané komunikace a k ní příslušného detekovaného pravidla.

Nástroj by měl tedy umožňovat práci se samotnými pravidly, především výběr daného pravidla z celé jejich databáze, zobrazit jej a pro lepší přehlednost vybrat některé atributy, především ty, které se týkají obsahové časti. Ty by následně měly být testovány na uživatelem vloženém textovém řetězci. Nástroj následně v případě shody zvýrazní dané atributy pokud se v tomto řetězci nachází. Poté analytik musí provést kontrolu spolu s ostatními atributy. Vybrané atributy mohou mít různou podobu, kdy mohou být také v HEX formátu a tudíž by zde měla být také funkce převodu do čitelného textu pro lepší přehlednost.

Vzhledem k faktu, že nástroj by měl pracovat s daty a také s textem, tak bude napsán v programovacím jazyku Python, který je vzhledem k požadavkům plně dostačující.

2.2.2 Realizace

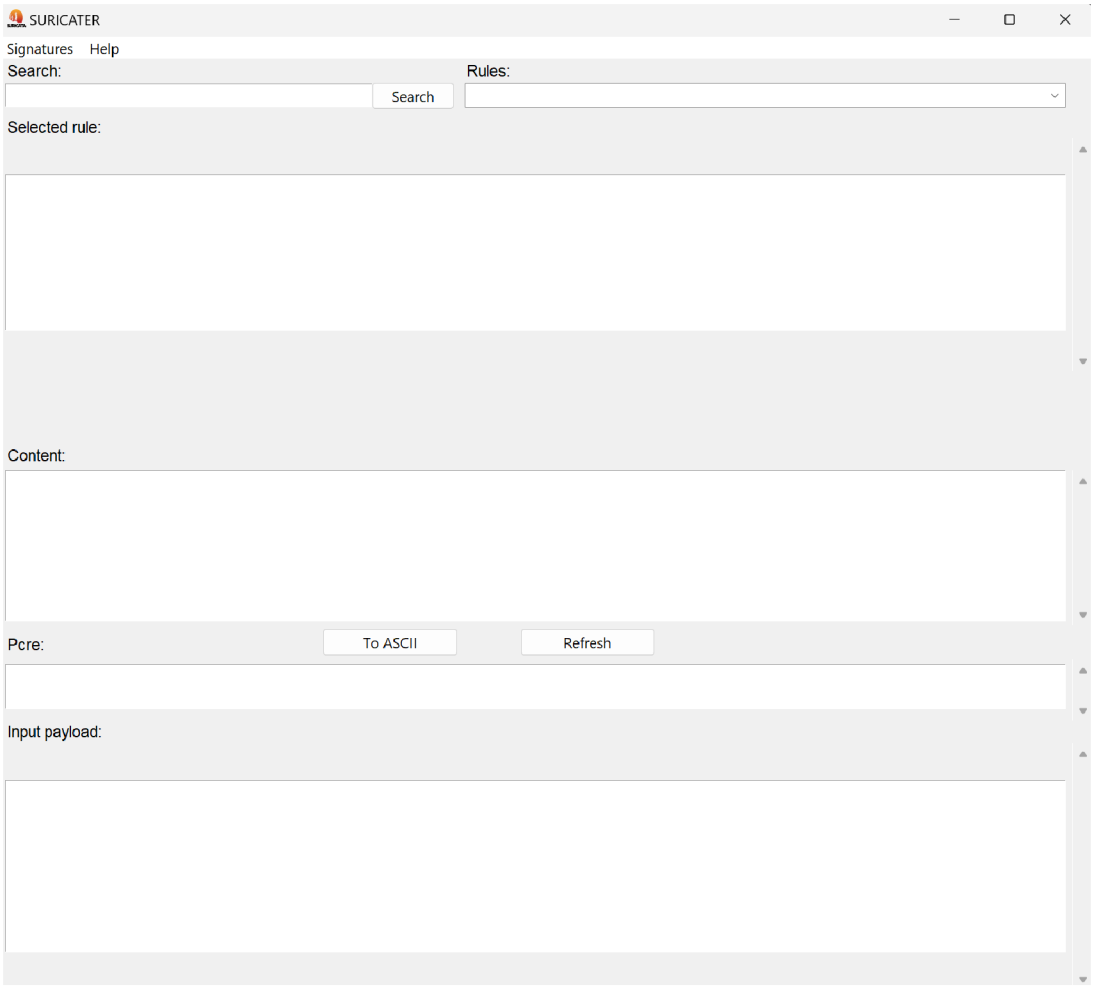
V rámci samotné realizace bylo využito programovacího jazyka Python s knihovnou Tkinter, která je určena pro tvorbu GUI (Graphical User Interface) aplikací. Vzhledem k zaměření nástroje byl zvolen název Suricater a na následujícím obrázku 8 lze vidět samotný nástroj a rozložení jeho prvků.

V tomto základním pohledu lze vidět dvě záložky a několik polí. Tato pole jsou určena k extrakci daných parametrů z pravidla, což má zlepšit jeho přehlednost. Z něj jsou extrahovány parametry týkající se detekce daných znaků na různých místech v payloadu (popis těchto parametru je podrobněji popsán v předchozích kapitolách).

Po rozkliknutí první záložky *Signatures*, jde vidět odkaz na výběr databáze s pravidly (*Choose*) a tlačítko pro export (*Export*) pravidel, konkrétně těch, které se shodují s filtrem v poli *Search* – jsou exportována ve formátu *.rules*.

V druhé záložce, *Help*, je poté oficiální dokumentace k popisu a vysvětlení všech možných parametrů, které se mohou v pravidle vyskytovat. V záložce *Info* je potom informace o samotném nástroji.

Po nahrání databáze pravidel lze v rozbalovacím menu *Rules*, vybrat dané pravidlo, které lze najít také pomocí pole *Search*, které umožňuje fulltextové vyhledávání skrze obsah všech pravidel v databázi. Ta lze vyhledávat pomocí všech možných textových řetězců, ať už konkrétních dat paketů, názvů nebo jejich částí, jedinečných čísel (SID) a dalších možností. Lze tak vyhledat konkrétní pravidlo nebo celou skupinu pravidel, které lze následně exportovat do samostaného souboru. Kliknutím na konkrétní pravidlo v rozbalovacím menu *Rules*, lze následně zobrazit extrahované parametry z daného pravidla.

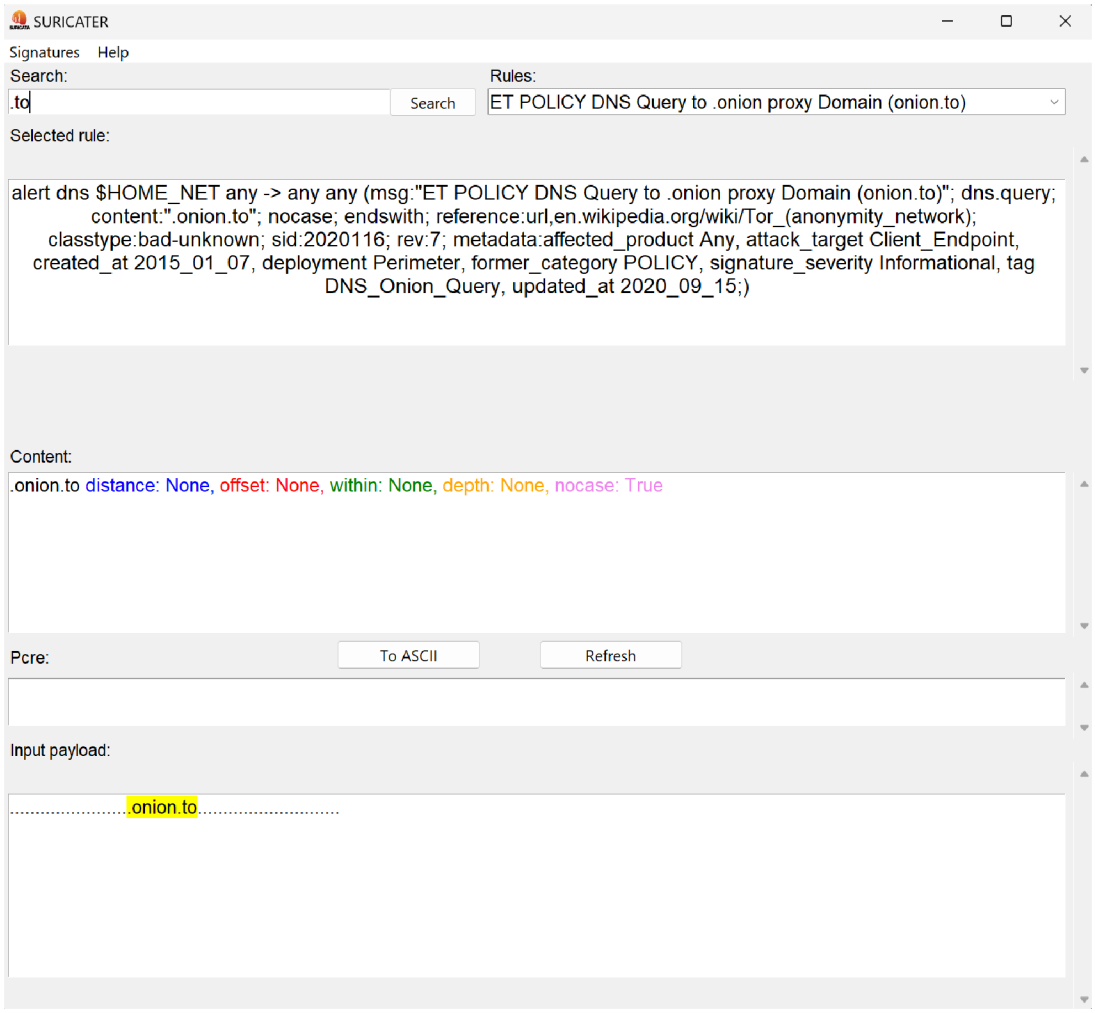


Obrázek 8: Nástroj Suricater

U položky *Content*, lze vidět také dvě tlačítka, kdy se jedná o přepínání z HEX hodnot do ASCII a zpět. Tato funkce je zde především z důvodu, že se v pravidlech vyskytují HEX hodnoty definující vzor, který má být hledán. Extrahované jsou pouze ty parametry, které se přímo týkají nebo nějakým způsobem definují, kde v datech paketu se má daný vzor nacházet.

Samotné testování se provádí v posledním okně s názvem *Input payload*, kde se vloží požadovaný textový řetězec, na kterém se má případná shoda vzoru

zvýraznit, tak jako to lze vidět na následujícím obrázku 9, kde bylo pro ukázku vybráno pravidlo detekující přístup na TLD (Top Level Domain) *.to*.



Obrázek 9: Analýza pomocí nástroje Suricater

3 Implmenetace IDS/IPS do sítě

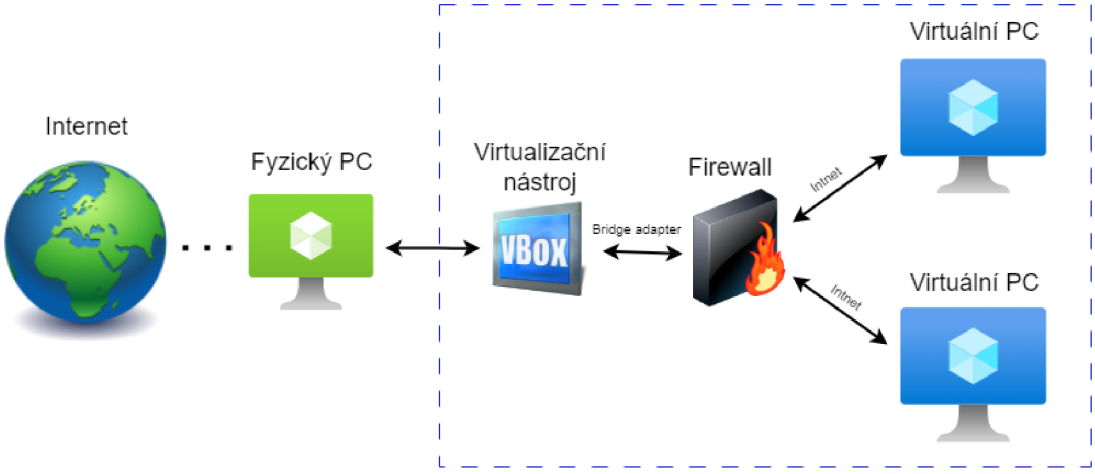
Tato kapitola se zabývá jak virtuální, tak i reálnou implentací IDS/IPS systému Suricata.

3.1 Virtuální implementace

V této kapitole je obsažena virtuální implementace IDS/IPS Suricata na firewallu OPNsense. Vstupními požadavky pro testování je instalace virtualizačního nástroje, v tomto případě se jedná o VirtualBox verze 7.0.6 s alespoň jedním nainstalovaným OS. V této implementaci však jde o dva OS, konkrétně Ubuntu 22.04 LTS a Kali Linux 2023, které představují koncové stanice uživatelů zapojených v jedné síti za firewallem. Celkové zapojení lze vidět na následujícím obrázku 10.

V rámci této implementace podle následujícího obrázku je také potřeba správně nastavit síťová připojení jednotlivých zařízení. Na firewallu je potřeba nastavit dvě síťová připojení, kdy první z nich je připojeno k adaptéru, který využívá ovladač síťové karty fyzického počítače k připojení do internetu (Bridge adapter) a druhé připojení je pouze do vnitřní sítě (na obrázku označeno jako Intnet), která je tvořena pouze virtuálními počítači a simuluje například domácí síť. Na koncových počítačích je nutné tato síťová nastavení nastavit tak, že budou připojeny pouze do vnitřní sítě.

Samotný IDS/IPS Suricata je poté spuštěn přímo na OPNsense firewallu, což je popsáno níže v textu.



Obrázek 10: Virtuální implementace IDS/IPS Suricata na firewallu

3.1.1 OPNsense

V rámci této implementaci je potřeba ve virtualizačním nástroji zprovoznit virtuální firewall OPNsense. Jedná se o open-source firewall, který je založen

na operačním systému FreeBSD. Jeho první verze byla vydána v roce 2015 a vychází z firewallu pfSense, respektive m0n0wall. Tento firewall disponuje velkým množstvím funkcí, díky kterým dokáže konkurovat drahým komerčním řešením [12]. Některé funkce OPNsense [12]:

- Intrusion Detection and Prevention
- Dvou faktorová autentizace do systému
- VPN (OpenVPN, IPsec)
- Vestavěné reportovací a monitorovací nástroje
- Forward Caching Proxy s podporou Blacklistu
- Netflow exportér
- Podpora mnoha rozšíření
- A mnoho dalších funkcí.

3.1.2 Instalace a nastavení IDS/IPS

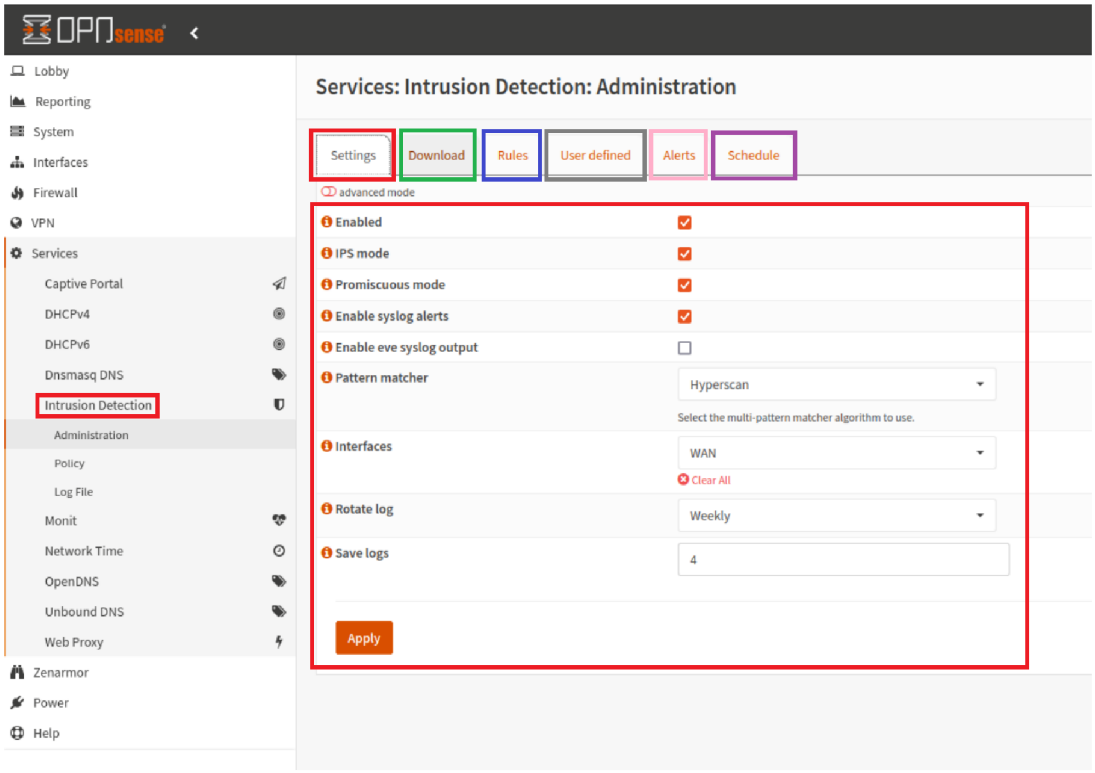
K úspěšné instalaci OPNsense firewallu je potřeba si stáhnout ISO soubor obrazu disku. Ten lze najít přímo na oficiálních stránkách firewallu.

Předpokladem je mít také nainstalován virtualizační software a v něm si vytvořit nové příslušné virtuální zařízení se správnou konfigurací, tzn. minimální požadavky vydavatele jsou [12]:

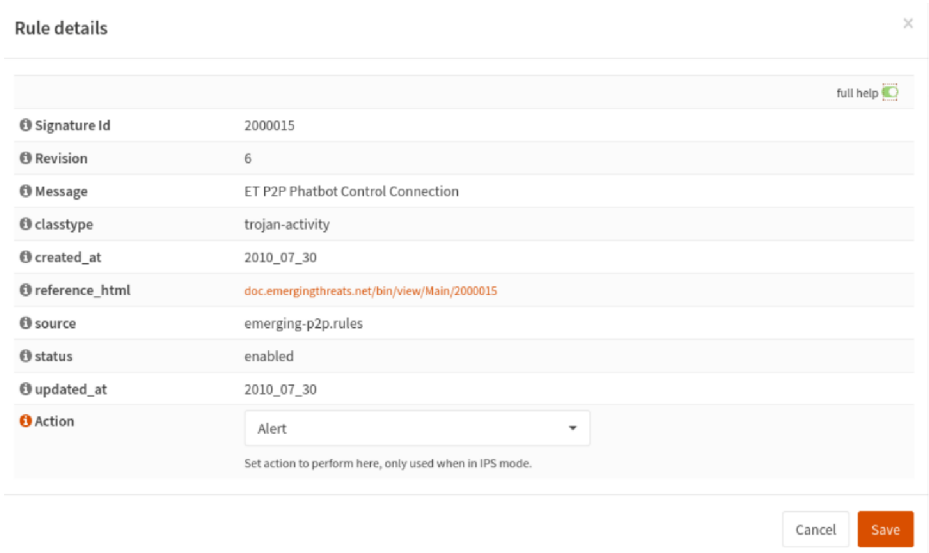
- minimálně dvoujádrový procesor s taktovací frekvencí 1 GHz
- velikost RAM alespoň 2 GB
- velikost úložiště alespoň 4 GB.

Samotné IDP/IPS je velmi jednoduché na tomto firewallu aktivovat. V tomto případě se jedná o povolení služby v nabídce firewallu (na obrázku 11 označeno červenou barvou). Zde je nutné povolit také IPS mód, promiskuitní režim a dále volitelně syslog upozornění. Lze také zvolit druh zachytávacího algoritmu, který byl v tomto případě nastaven na hyperscan, kdy se podle oficiální dokumentace jedná o nejlepší volbu [12]. Je také potřeba stáhnout a povolit zvolené dostupné balíky pravidel, které se nachází na další záložce (na obrázku 11 označeno zelenou barvou). V záložce pravidla (na obrázku 11 označeno modrou barvou), lze pak zobrazit jednotlivá pravidla a jejich detailní informace, jako je zachyceno na obrázku 12. Zde je možné také jednotlivá pravidla povolovat, zakazovat a nastavovat jejich akci v případě detekce, to znamená, že buď bude vytvořeno pouze upozornění, nebo v případě nastavení módu IPS, bude komunikace ukončena. Mezi pravidly lze také vyhledávat a to pomocí filtrovacího okna.

Firewall OPNsense umožňuje také jednoduchou tvorbu vlastních pravidel (na obrázku 11 označeno šedou barvou). Je zde také možnost nastavení plánovače úloh (na obrázku 11 označeno fialovou barvou), kdy lze tuto funkci využít například na pravidelnou aktualizaci pravidel, k pravidelnému restartu služeb nebo zasílání reportů. V poslední záložce, upozornění (na obrázku 11 označeno růžovou barvou), lze vidět následně záznamy, které byly detekovány na základě pravidel.



Obrázek 11: Aktivace IDS/IPS



Obrázek 12: Detail pravidla

V rámci nabídky pro IDS/IPS (obrázek 11) lze také vidět možnost nastavení politik, kdy zde lze definovat politiky pro nainstalovaná pravidla. Tyto politiky zde jde také přidat, spravovat nebo mazat u jednotlivých pravidel. Tato funkce se hodí především v rámci sítě, kde je záměr více kontrolovat, která pravidla budou použita, za jakých podmínek, jaké budou mít bezpečnostní skóre a další.

V této virtuální implementaci bylo vytvořeno vlastní pravidlo, které však nebylo možné udělat přímo pomocí grafického prostředí firewallu, kde lze zvolit například zdrojové a cílové IP adresy, SSL otisk, akci, to je vytvořit upozornění nebo zakázat komunikaci a popis pravidla, ale bylo nutné jej napsat např. v textovém editoru a pomocí vzdáleného (SSH) přístupu jej vložit přímo do daného adresáře s pravidly OPNsense firewallu.

Pravidlo mělo za účel detekovat skenování rozsahu sítě z interní sítě, konkrétně [13]:

```
alert tcp $HOME_NET any -> 10.0.1.1 any(msg:"Custom:
   NMAP SYN SCAN DETECTED"; flags:S; priority: 5;
 threshold: type threshold, track by_src, count 50,
   seconds 1;classtype:attempted-recon; sid:1234;)
```

Kromě samotného pravidla, je potřeba také XML soubor, který se přímo pomocí SSH přístupu nahraje do příslušného adresáře *(/usr/local/opnsense /scripts/suricata/metadata/rules)* OPNsense firewallu. Příklad XML souboru je na obrázku 13. Tento soubor je určen k tomu, že dojde ke zviditelnění mezi balíky pravidel, které lze stáhnout a nainstalovat. V tomto případě je pravidlo stáhnuto ze zařízení s IP 10.0.1.10 a jedná se o soubor s názvem rule.rules, který obsahuje vytvořené pravidlo. Po instalaci již jej lze vidět mezi nainstalovanými pravidly (obrázek 14). Aby mohlo pravidlo být nainstalováno, musí být také spuštěn http server, například pomocí příkazu *python3 -m http.server 80* na zařízení a přímo

z adresáře s pravidly, ze kterého bude stahování probíhat – v tomto případě na zařízení s již zmíněnou IP 10.0.1.10 [13].

```
-<ruleset documentation_url="http://docs.opnsense.org/">
   <location url="http://10.0.1.10/" prefix="rule"/>
  -<files>
     <file description="rule rules">rule.rules</file>
     <file description="rule" url="inline::rules/rule.rules">rule.rules</file>
   </files>
 </ruleset>
```

Obrázek 13: XML soubor

Obrázek 14: Nainstalované pravidlo

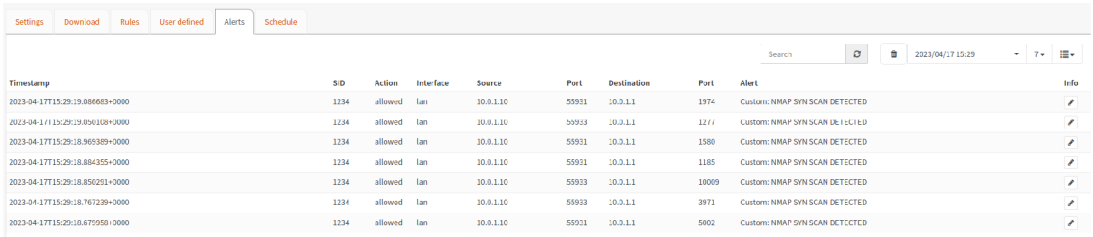
Nyní již lze přistoupit k samotnému testu a spustit skenování sítě pomocí příkazu *sudo nmap -sT -Pn 10.0.1.1*, kdy skenovaná IP adresa je přímo adresa firewallu. Na obrázku 15 jde vidět výsledek tohoto skenu [13].

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sS -Pn --top-ports 500 10.0.1.1
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-16 14:30 EDT
Nmap scan report for OPNsense.localdomain (10.0.1.1)
Host is up (0.00083s latency).
Not shown: 496 filtered tcp ports (no-response)
PORT    STATE SERVICE
22/tcp  open  ssh
53/tcp  open  domain
80/tcp  open  http
443/tcp open  https
MAC Address: 08:00:27:9E:60:40 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 3.72 seconds
```

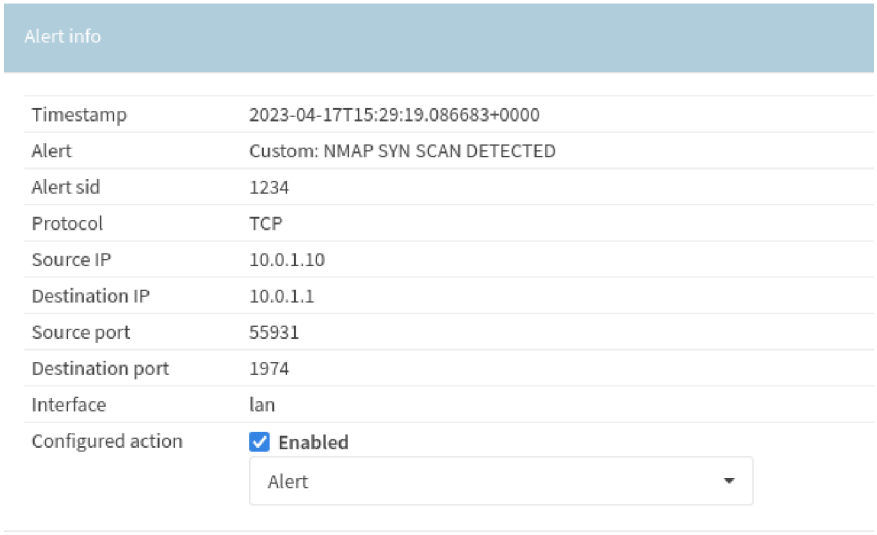
Obrázek 15: Skenování pomocí nástroje nmap

Po skončení skenování lze následně vidět přímo ve firewallu upozornění na detekeci, konkrétně v záložce alerts (obrázek 16).



Obrázek 16: Jednotlivé upozornění

Detailní informace po rozkliknutí jednoho z upozornění lze vidět na následujícím obrázku 17.



Obrázek 17: Detailní informace

Tato implementace je vhodná především pro testování a ladění systému před samotným nasazením do reálného provozu. V této implementaci lze také jednoduše testovat jednotlivá specifická pravidla při jejich tvorbě, kdy jsou pravidla tvořena „na míru“ a je potřeba otestovat jejich funkčnost, například při výskytu nového malware lze takto v izolovaném prostředí testovat chování tohoto softwaru a zároveň vytvořit a doladit pravidlo pro jeho spolehlivou detekci.

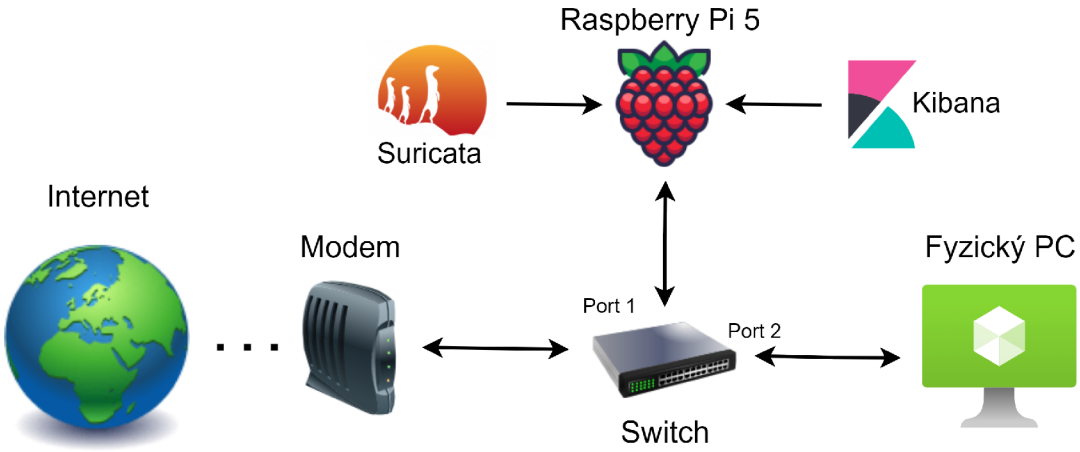
Při tvorbě těchto pravidel je velmi důležité dbát na správnou syntaxi a strukturu pravidla. V případě, kdy nastane při aplikaci pravidel jakýkoliv problém, tak lze vidět záznamy – logy v příslušné záložce a na základě těchto záznamu lze dekódovat, kde příslušný problém nastal. V samotném firewallu je možné také nainstalovat sady přednastavených pravidel, které jsou zdarma dostupná a nemusí se tedy tvořit pouze pravidla vlastní. Samotné spuštění funkce IDP/IPS a jeho běh má také za následek větší vytížení paměti a procesoru a je tedy nutné brát v potaz i spouštěná pravidla a jejich množství. Alternativou může

být volně dostupné rozšíření Zenarmor, které poskytuje více funkcí, je jednodušší na správu a instalaci a také má přehlednější výstup dat.

Závěrem lze dodat, že tento firewall je vhodné použít jak v domácí síti, tak i v rámci malé firemní sítě jako jsou například hotely, restaurace apod., kde je možné danou síť přizpůsobit přímo podmínkám dané firmy.

3.2 Reálná implementace

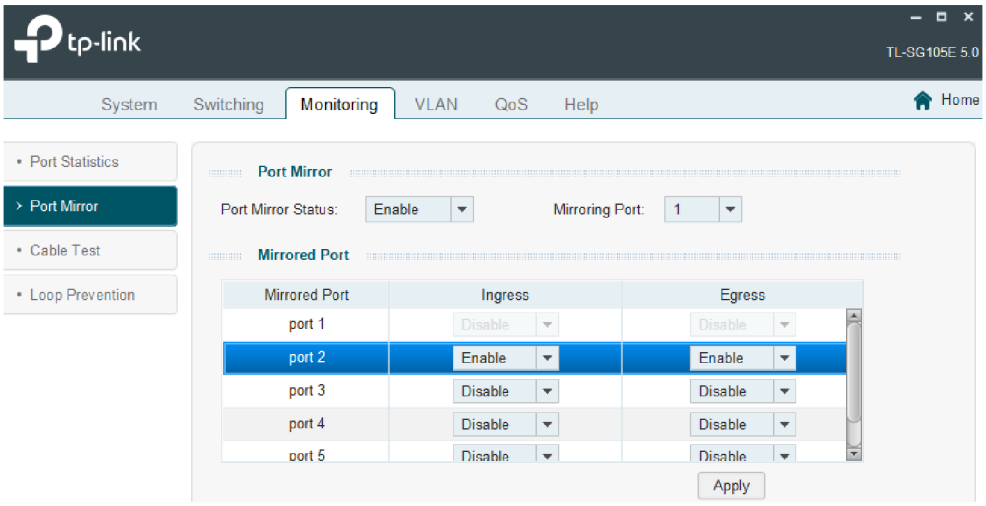
V rámci reálné implementace bylo využito open-source softwaru, konkrétně IDS/IPS Suricata pro detekci anomálií v síťovém provozu a Kibana (součást ELK stack) pro vizualizaci dat. Fyzické zařízení pro běh tohoto softwaru bylo využito Raspberry Pi 5, které je vybaveno 64bitovým procesorem Arm Cortex-A76 s frekvencí 2,4 GHz a 8 GB pamětí RAM [14]. Celkové schéma zapojení jde vidět obrázku 18.



Obrázek 18: Reálného zapojení

3.2.1 Hardware

Součástí zapojení byly také síťové prvky. Jako první prvek v síťové komunikaci směrem od poskytovatele internetu byl zapojen modem. Do něj byl do jednoho z portů připojen switch, konkrétně 5portový switch, který musí disponovat funkcí zrcadlení portu, zde konkrétně zrcadlení portu 2 do portu 1. V tomto případě se jednalo o switch TP-link TL-SG105E. Tato funkce umožňuje kopírovat veškerý provoz běžící skrze port 2 na port 1. To bylo možné velmi jednoduše nastavit v konfiguračním prostředí switche, tak jako je to možné vidět na následujícím obrázku 19. K portu 1 bylo následně připojeno Raspberry Pi 5 s již zmíněným softwarovým vybavením, které tuto komunikaci dále zpracovává a vyhodnocuje. K portu 2 byl připojen klasický počítač, který generoval různý provoz.



Obrázek 19: Konfigurace zrcadlení portu

3.2.2 Software

Hlavním operačním systémem na Raspberry Pi 5 bylo Ubuntu verze 23.10. Jednotlivé softwarové vybavení, ať už Suricata nebo Kibana, je dostupné volně ke stažení na jednotlivých oficiální stránkách, kde je však nutné vždy zvolit správnou verzi daného balíčku, tj. v tomto případě pro verzi ARM64. Toto vybavení bylo nainstalováno a nakonfigurováno podle oficiálních dostupných návodů [16] [17].

Software Suricata zde tedy sloužil jako hlavní nástroj pro detekci podezřelého provozu. V tomto případě byla Suricata pouze v režimu IDS. Samotná instalace na operační systém Ubuntu je velice jednoduchá a je možné jej nainstalovat také pomocí repozitáře, tak jako je uvedeno v oficiální dokumentaci [16]. Následně je důležité upravit konfigurační soubor *suricata.yaml*, kdy tento soubor slouží k celkovému nastavení a definování různých parametrů ať už celé sítě, tak i samotné sondy Suricata. Lze zde také definovat proměnné, které je možné následně použít v samotných pravidel, například již předdefinovanou proměnou je rozsah domácí sítě:

HOME_NET: "[192.168.0.0/16,10.0.0.0/8,172.16.0.0/12]".

V tomto souboru se také definují cesty k souborům s pravidly, která jsou uložena v textovém souboru s příponou *.rules*. Lze zde také nastavit mnoho dalších parametrů, jako velikost paketů, množství současně zpracovávaných paketů, nastavení Eve formátu (Extensible Event Format), kde se definují, jaké všechny atributy budou spolu se samotným záznamem uloženy – jedná se například o rozšířené atributy HTTP, SMTP nebo DNS. Dále je možné v tomto souboru zapnout funkci samotného ukládání dané zachycené komunikace do formátu PCAP (Packet Capture Data). Obecně lze tedy pomocí tohoto souboru nastavit celý chod a chování sondy.

Velmi důležitým prvkem pro následnou implementaci a vizualizaci dat v Kibaně, který se zde také nastavuje, je cesta, kde budou jednotlivá vytvořená

upozornění uložena. Ta se ukládají ve formě logů, tedy v textovém formátu v souboru s příponou *.log*, avšak i tento výstup je plně konfigurovatelný. Celkově soubor *suricata.yaml* tvoří velmi důležitou část celého softwaru a zde byl pouze výčet těch parametrů, které byly v rámci této implementace vyzkoušeny. Mnohem více parametrů včetně jejich popisu jsou uvedeny v oficiální dokumentaci [17].

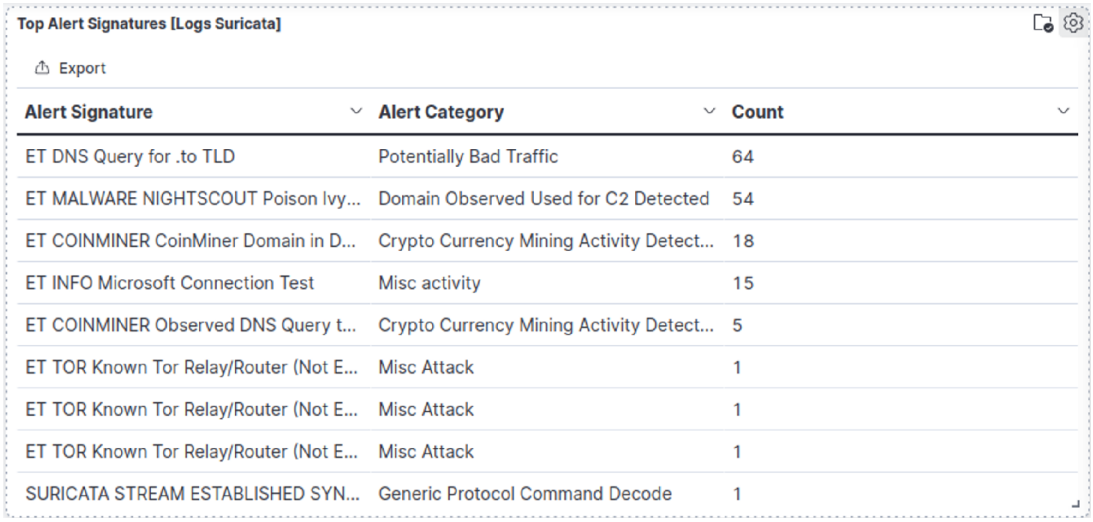
Pro vizualizaci dat sondy bylo využito softwaru Kibana z balíku ELK Stack (Elasticsearch, Logstash, and Kibana). Jedná se o open-source software k vizualizaci dat ve webovém prohlížeči. Samotnou integraci Suricata IDS lze provést několika způsoby. Pro větší objemy dat (množství logů), respektive provoz a celkově více zařízení ve větších sítích bude nejvhodnější mít vlastní hardwarové vybavení jak pro Kibanu, tak i pro sběr jednotlivých logů (syslog server) – záznamů a upozornění a s tím i využití ostatních nástrojů v rámci balíku ELK Stack. Samotné sondy Suricata by v těchto sítích byly umístěny v různých částech sítě a bylo by jich také větší množství.

Pro domácí síť však plně dostačuje lokální umístění tohoto softwaru spolu se Suricata IDS na samotném Raspberry Pi 5, které ještě bylo doplněno o disk s velikostí 256 GB. Vše bylo zprovozněno pouze na tomto zařízení, což také celou implementaci zjednodušilo, především z pohledu ukládání a čtení jednotlivých souborů (logů) a celého vzájemného propojení obou programů. Na základě těchto faktů byla zvolena implementace pouze pomocí rozšíření, které lze do Kibany jednoduše přidat a nakonfigurovat. Konkrétně pomocí Filebeat a modulu pro Suricatu [15]. Toto rozšíření se postará o samotné parsování (rozdělení dat na jednotlivé atributy) jednotlivých záznamů a také o distribuci dat mezi sondou Suricata a Kibanou. To se standardně děje na základním portu 5601. V rámci této implementace je nutné dodržet konfiguraci pro procesory typu ARM64 [15].

3.2.3 Testování

Pro samotné testování monitoringu domácí sítě bylo v rámci sondy Suricata využito volně dostupného balíku pravidel – ET open. Jak již bylo zmíněno dříve, tak se jedná se o volně dostupný ruleset od společnosti Proofpoint, kdy jsou tato pravidla zaměřena na detekci obecnějších anomálií, zranitelností, úniku informací a dalších hrozeb v síti. Balík obsahuje desítky tisíc pravidel, která jsou logicky členěna do jednotlivých samostatných souborů, především podle typů komunikace a druhů útoků. Lze tedy aktivovat pouze některé z těchto pravidel, v závislosti na požadavcích na danou síť. Také je možnost tvorby vlastních pravidel, což je zmíněno především v kapitole 2.2. Pro představu například v domácí síti je žádoucí sledovat únik platebních údajů, což detekují například pravidla *ET POLICY Credit Card Number Detected in Clear*, přístupy na známé škodlivé stránky - *ET MALWARE HTTP Request to a known malware domain (regicsgf.net)*, popřípadě detekovat možný adware na některém ze zařízení - *ET ADWARE_PUP Binet (randreco.exe)*.

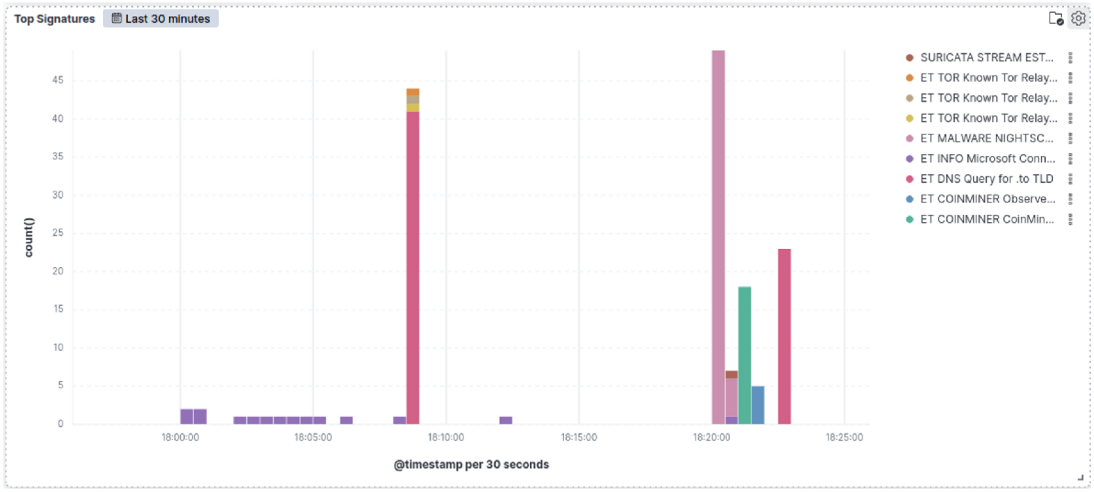
V tomto případě byl na zařízení připojeném ke switchi generován podezřelý provoz, především využití prohlížeče TOR, přístup na podezřelé TLD *.to*, možná těžba kryptoměn a další, tak jako je možné vidět na následujícím obrázku 20.



Alert Signature	Alert Category	Count
ET DNS Query for .to TLD	Potentially Bad Traffic	64
ET MALWARE NIGHTSCOUT Poison Ivy...	Domain Observed Used for C2 Detected	54
ET COINMINER CoinMiner Domain in D...	Crypto Currency Mining Activity Detect...	18
ET INFO Microsoft Connection Test	Misc activity	15
ET COINMINER Observed DNS Query t...	Crypto Currency Mining Activity Detect...	5
ET TOR Known Tor Relay/Router (Not E...	Misc Attack	1
ET TOR Known Tor Relay/Router (Not E...	Misc Attack	1
ET TOR Known Tor Relay/Router (Not E...	Misc Attack	1
SURICATA STREAM ESTABLISHED SYN...	Generic Protocol Command Decode	1

Obrázek 20: Seznam detekovaných pravidel

V rámci vizualizace dat v Kibaně je možné v tomto programu definovat vlastní, téměř jakékoliv zobrazení jakýchkoliv dostupných požadovaných dat. Lze si tak definovat vlastní nástěnku s vlastními požadovanými a sledovanými údaji a grafy, přesně podle potřeb a požadavků uživatele. Příklad časového zobrazení jednotlivých upozornění lze vidět na obrázku 21.



Obrázek 21: Časové zobrazení detekovaných pravidel

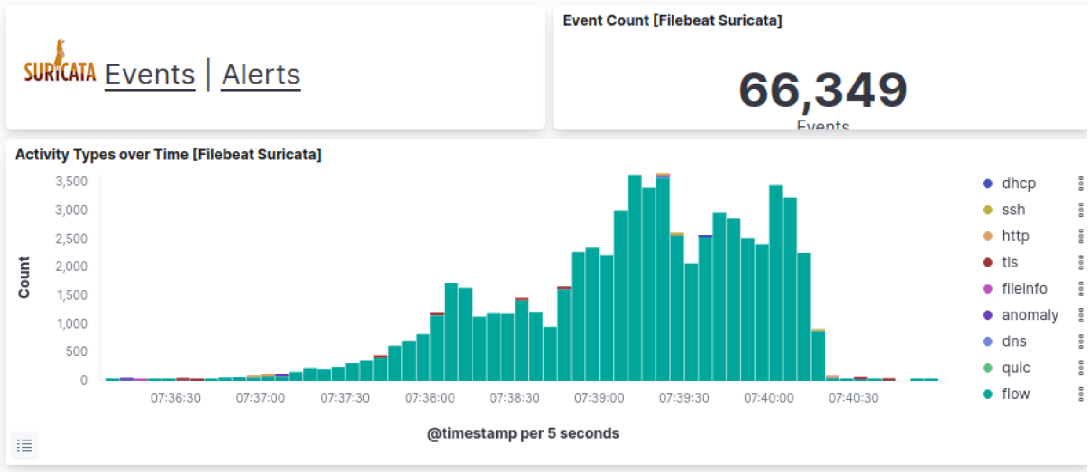
V Kibaně je také možné si jednotlivé záznamy otevřít a zobrazit tak možné detaily zaznamenaného logu, včetně základních atributů jako je zdrojová a cílová adresa a port, čas, tak i rozšířených atributů, které však musejí být povoleny a definovány v konfiguračním souboru sondy, například DNS atributů nebo dat paketu, jako je zobrazeno na dalším obrázku 22. Lze zde také zobrazit nejen logy detekované pravidly, ale i jednotlivé logy, které nebyly detekovány a nevytvořily upozornění. To je výhodné především pro vyšetřování dané komunikace, kdy lze

tedy porovnávat vytvořená upozornění s těmito logy a určit tak, co předcházelo dané podezřelé komunikaci.

t suricata.eve.in_iface	eth0
t suricata.eve.payload	ACQb3wEAAAEAAAAAAAAAEcG9vbAloYXNodmF1bHQDcHJvAAABAAE=
t suricata.eve.payload_printable	.$.....pool.hashvault.pro.....
t suricata.eve.pkt_src	wire/pcap
# suricata.eve.stream	1

Obrázek 22: Detaily logu

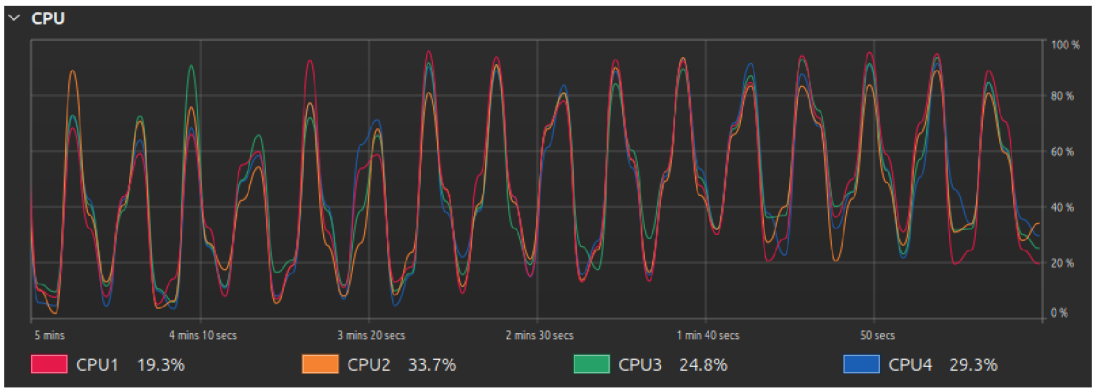
Bylo provedeno také jednoduché testování výkonu, konkrétně jestli bude mít vliv větší počet jednotlivých událostí, které jsou testovány na shodu s pravidly, na výkon Raspberry Pi 5. Události byly generovány pomocí zařízení v síti s využitím grafického nástroje *Zenmap*, který slouží jako grafické ovládání pro nástroj *Nmap*. Zenmap obsahuje již několik uložených profilů pro skenování, kdy v tomto testu byl použit profil *Intense scan, all TCP ports*, který provede sken všech TCP portů. Celkový příkaz tedy byl *nmap -p 1-65535 -T4 -A -v scanme.nmap.org*. Jak lze vidět na obrázku 23, tak tento sken dokáže v jeden okamžik vygenerovat až několik tisíc událostí, což je nad rámec běžného provozu v domácí síti.



Obrázek 23: Vygenerované události

V softwaru Kibana (Elasticsearch) bylo nastaveno automatické obnovení dat po 10 sekundách, což se ukázalo, že samotné zobrazování dat má větší vliv na využití zdrojů zařízení, než běh sondy Suricata. Jak lze vidět na následujícím obrázku 24, tak vždy v případě začátku nahrávání dat byly zdroje využity téměř na maximum s následným poklesem, což byla doba, kdy data nebyla nahrávana k vizualizaci. V případě zkoušení nižších hodnot obnovy docházelo k menším propadům mezi začátkem nahrávání dat a poklesem, a v případě nastavení obnovy po 1 sekundě již k poklesům docházelo minimálně. Zdroje tak byly využívány na

maximum téměř neustále. Vliv na tento výkon má také vytvoření buď mnoha menších nebo většího grafu, který musí zpracovávat obrovské množství záznamů a může tak docházet ke zpomalení a ovlivnění výkonu počítače. Opět je tedy vhodné volit grafy pouze v rozumné míře. Celkově je tedy vhodnější pro sítě s větším provozem, mít jednotlivé nástroje odděleny na vlastních zařízeních.



Obrázek 24: Využití CPU

V rámci této implementace bylo ověřeno, že lze pomocí dostupných zařízení vytvořit systém pro monitoring domácí sítě. Bylo také ověřeno, že Raspberry Pi 5 je již dostatečně výkonné pro běh jak sondy Suricata, tak i pro vizualizaci samotných dat v rámci implementace pro domácí síť. Pro vylepšení výkonu by bylo vhodné použít a aplikovat pouze určitá vybraná pravidla, především ta, která nás zajímají, místo celého balíku všech pravidel. Na výkon má také vliv konfigurace samotné sondy pomocí konfiguračního souboru *suricata.yaml*, kdy v případě, že bude uchováváno v rámci logů velké množství informací (atributů), které sonda detekuje, tak i tohle zpracování bude mít vliv na celkový výkon. Je tedy vhodné přesně určit atributy, které jsou užitečné a vhodné pro uchování.

Pro běžného uživatele, který nemá specifické potřeby pro monitoring domácí sítě však bude výhodnější využít již některý z open-source firewallů, jako například již zmíněný OPNsense anebo využít celé zařízení s již vestavěnou funkcí IDS/IPS jako je například Ubiquiti UniFi Dream Machine [18].

Závěr

Tato bakalářská práce se zabývá celkovým popisem a návrhem IDS/IPS pravidel, jejich implementací a také shrnutím dostupných řešení těchto systémů. Lze tedy jednoduše tvrdit, že je dostupné větší množství různých těchto systémů, kdy je lze také rozdělit do několika kategorií podle koncového použití na NIDS, HIDS a Hybrid IDS, a také podle způsobu detekce na IDS s detekcí založené na pravidlech a anomáliích. Mezi zástupce HIDS patří například řešení OSSEC a je tedy určeno pro umístění přímo na koncové zařízení. Mezi NIDS a také potažmo hybridní IDS lze zařadit univerzální řešení Suricata, popř. Snort. Všechny zmíněné IDS/IPS systémy také patří do kategorie detekce pomocí pravidel, zatímco ZEEK (Bro) je založen na detekci pomocí anomálií.

V rámci tvorby a analýzy pravidel, byla představena IDS/IPS pravidla a jejich základní parametry, které definují celé pravidlo a určují, co se v dané komunikaci musí vyskytovat, aby mohlo být vydáno upozornění, popř, celá komunikace zablokována, což se děje v režimu IPS. Byla také představena tvorba jednoduchého pravidla, konkrétně pro detekci skenování ve vnitřní síti pomocí nástroje NMAP. To bylo následně implementováno ve virtuální implementaci. Součástí této kapitoly byla také tvorba nástroje Suricater pro jednodušší analýzu daných pravidel, který slouží především analytikům při vyšetřování incidentů. Jak již nástroj napovídá, je určen především pro formát pravidel používajících se u IDS/IPS systému Suricata, avšak lze jej využít také pro systém Snort, kde jsou pravidla až na malé odchylky velmi podobná. Nástroj extrahuje určité parametry z pravidla a ty následně testuje na vložených datech paketu z konkrétní zkoumané komunikace. V případě shody je daný parametr zvýrazněn přímo v okně nástroje. Jeho úkolem je tedy rychleji a přehledněji určit, co způsobilo danou výstrahu, popř. blokaci komunikace.

V rámci implementace je kapitola rozdělena na virtuální a reálnou implementaci. V prvním případě se jednalo o vytvoření virtuální sítě ve virtuálním prostředí, kdy byl k detekci využit open-source firewall OPNsense, který disponuje také možností IDS/IPS Suricata. V tomto řešení zde bylo nahráno již zmíněné pravidlo a následně bylo z jednoho virtuálního počítače provedeno skenování sítě. Tento sken byl poté zachycen tímto firewallem a bylo vytvořeno upozornění. Jedná se tedy o velmi jednoduché řešení, jak na implementaci, tak i na konfiguraci, které plně dostačují pro menší sítě.

V reálné implementaci bylo využito již fyzických zařízení, konkrétně hlavním zařízení, které sloužilo pro běh IDS/IPS Suricata a také pro vizualizaci dat bylo zařízení Raspberry Pi 5. Jedná se o levné, avšak dostatečně výkonné zařízení, které dokáže opět v menších sítích spolehlivě tento systém realizovat. Samotné Raspberry Pi 5 bylo zapojeno pouze v režimu IDS a pomocí veřejně dostupných pravidel, tak bylo možné v reálném čase sledovat výstrahy a možnou podezřelou komunikaci v síti.

Conclusions

This bachelor thesis deals with the overall description and design of IDS/IPS rules, their implementation, and a summary of available solutions for these systems. It is therefore simple to assert that there is a larger number of different systems available, which can also be divided into several categories according to their end use into NIDS, HIDS and Hybrid IDS, and also according to the method of detection into IDS with rule-based and anomaly-based detection. Representatives of HIDS include the OSSEC solution, which is intended for placement directly on the end device. Universal solutions such as Suricata or Snort can be included among NIDS and also hybrid IDS. All mentioned IDS/IPS systems also belong to the category of rule-based detection, while ZEEK (Bro) is based on anomaly detection.

In the context of creating and analyzing rules, IDS/IPS rules and their basic parameters were introduced, defining the entire rule and determining what must occur in a given communication in order for a warning to be issued or the entire communication to be blocked, which occurs in IPS mode. The creation of a simple rule, specifically for the detection of scanning in the internal network using the NMAP tool, was also introduced. This was then implemented in a virtual implementation. This chapter also includes the creation of the Suricater tool for easier analysis of these rules, which is primarily used by analysts investigating incidents. As the tool suggests, it is mainly designed for the format of rules used in the Suricata IDS/IPS system, but it can also be used for the Snort system, where the rules are very similar with minor deviations. The tool extracts certain parameters from the rule and then tests them on the inserted packet data from the specific communication being examined. In the case of a match, the parameter is highlighted directly in the window of the tool. Its task is therefore to determine more quickly and clearly what caused the warning or blocking of communication.

In terms of implementation, the chapter is divided into virtual and real implementation. In the first case, it was about creating a virtual network in a virtual environment, where the open-source firewall OPNsense, which also has the IDS/IPS Suricata option, was used for detection. In this solution, the mentioned rule was uploaded and a network scan was performed from one virtual computer. This scan was then captured by this firewall and a warning was created. It is therefore a very simple solution, both in terms of implementation and configuration, which is fully sufficient for smaller networks.

In the real implementation, physical devices were used, specifically the main device, which served to run IDS/IPS Suricata and also for data visualization was the Raspberry Pi 5 device. It is a cheap, yet sufficiently powerful device that can reliably implement this system in smaller networks. The Raspberry Pi 5 itself was only involved in IDS mode and using publicly available rules, it was possible to monitor warnings and possible suspicious communication in the network in real time.

A Obsah přiloženého datového média

`bin/`
: Složka obsahuje spustitelný soubor a dokumentaci.

`doc/`
: Obsahem této složky je text práce ve formátu PDF a všechny soubory potřebné pro vygenerování PDF dokumentu textu (v ZIP archivu).

`rules/`
: Součástí složky je několik příkladů souborů s pravidly.

`src/`
: Složka obsahuje zdrojový kód nástroje spolu s potřebnými soubory.

`readme.txt`
: Instrukce pro spuštění a použití nástroje.

Literatura

[1] EC-COUNCIL. *Certified Network Defender (CND) Version 2* [online]. 2nd ed. International Council of E-Commerce Consultants (EC Council), 2020 [cit. 2023-07-11]. Dostupné z: https://bookshelf.vitalsource.com/books/9781635675641

[2] BAKER, ANDREW R a JOEL ESLER. *Snort IDS and IPS Toolkit* [online]. 1. Burlington: MA: Syngress Publishing, c2007 [cit. 2023-07-14]. ISBN 978-1-59749-099-3.

[3] ASHTARI, Hossein. Intrusion Detection System vs. Intrusion Prevention System: Key Differences and Similarities. In: *Spiceworks* [online]. Austin Texas, 2022 [cit. 2023-07-14]. Dostupné z: https://www.spiceworks.com/it-security/network-security/articles/ids-vs-ips/

[4] OSSEC. *OSSEC* [online]. OSSEC Project Team, c2010-2021 [cit. 2023-07-17]. Dostupné z: https://www.ossec.net/docs/

[5] *OSSEC HIDS* [online]. OSSEC Project, c2019 [cit. 2023-07-17]. Dostupné z: https://ossec-documentation.readthedocs.io/en/latest/index.html#

[6] ZEEK. *ZEEK* [online]. The Zeek Project, c2019-2021 [cit. 2023-07-17]. Dostupné z: https://docs.zeek.org/en/master/

[7] Suricata. *Suricata* [online]. Open Information Security Foundation (OISF), c2016-2023 [cit. 2023-07-17]. Dostupné z: https://docs.suricata.io/en/suricata-6.0.13/index.html#

[8] *Suricata* [online]. Boston: OISF, c2023 [cit. 2023-07-21]. Dostupné z: https://suricata.io/

[9] HOOVER, C. Comparative Study of Snort 3 and Suricata Intrusion Detection Systems. *Computer Science and Computer Engineering Undergraduate Honors Theses* [online]. Fayetteville [cit. 2023-07-22]. Dostupné z: https://scholarworks.uark.edu/csceuht/105

[10] *Snort* [online]. San Jose: Cisco, c2023 [cit. 2023-07-22]. Dostupné z: https://www.snort.org

[11] Downloads. *Snort* [online]. San Jose: Cisco, c2023 [cit. 2023-09-04]. Dostupné z: https://www.snort.org/downloads#rules

[12] *OPNsense* [online]. Middelharnis: Deciso, c2015-2023 [cit. 2023-07-07]. Dostupné z: https://opnsense.org

[13] LS111 CYBER SECURITY EDUCATION. Suricata IDS/IPS Installation on Opnsense - Virtual Lab Building Series: Ep3. In: *Youtube* [online]. [cit. 2023-07-07]. Dostupné z: https://www.youtube.com/watch?v=TPKLu4a3A4E

[14] RASPBERRY PI LTD. *Buy a Raspberry Pi 5*. Online. Raspberry Pi. Dostupné z: https://www.raspberrypi.com/products/raspberry-pi-5/. [cit. 2024-05-17].

[15] *Suricata module*. Online. Elastic. C2024. Dostupné z: https://www.elastic.co/guide/en/beats/filebeat/current/filebeat-module-suricata.html. [cit. 2024-05-17].

[16] *Suricata User Guide*. Online. Nenalezený vydavatel. C2016-2024. Dostupné z: https://docs.suricata.io/en/latest/. [cit. 2024-05-17].

[17] *Kibana Guide*. Online. Elastic. C2024. Dostupné z: https://www.elastic.co/guide/en/kibana/current/index.html. [cit. 2024-05-17].

[18] *Dream Machine*. Online. Ubiquiti Store United States. C2024. Dostupné z: https://store.ui.com/us/en/products/udm. [cit. 2024-05-17].