VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

SYSTÉMY PRO DETEKCI A PREVENCI PRŮNIKŮ
INTRUSION DETECTION AND PREVENTION SYSTEMS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Andrej Pitschmann**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Zdeněk Martinásek, Ph.D.**
SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Andrej Pitschmann ***ID:*** 173725
Ročník: 3 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Systémy pro detekci a prevenci průniků

POKYNY PRO VYPRACOVÁNÍ:

Hlavním cílem práce je navrhnout a vypracovat laboratorní úlohu seznamující studenty s principem IDS/IPS (Intrusion Detection/Prevention Systems) systémů. Prostudujte současný stav problematiky, nainstalujte a zprovozněte nejméně dvě volně dostupná řešení (např. SNORT nebo SURICATA) a nakonfigurujte detekce vybraných útoků (nejméně 5 vybraných typů DDoS o různých silách), které jsou cíleny na standartní serverové služby (HTTP, HTTPS, FTP atd.). U zprovozněných systému se zaměřte na vizualizaci a zpracování záznamů událostí (logů). Porovnejte detekční popřípadě mitigační schopnosti zprovozněných systémů a zaměřte se na diskuzi uživatelského rozhraní a visualizaci činnosti systémů. Navrhněte a vypracujte laboratorní úlohu seznamující studenty se základní instalací a konfigurací IDS/IPS cílené na DDoS útoky (zvolte jeden systém). Součástí práce bude komplexní návod laboratorní úlohy pro studenty a do elektronické přílohy vytvořte video návod, který obsahuje instalaci a následnou realizaci laboratorní úlohy.

DOPORUČENÁ LITERATURA:

[1] SCARFONE, Karen; MELL, Peter. Guide to intrusion detection and prevention systems (idps). NIST special publication, 2007, 800.2007: 94.

[2] BEAL, Vangie. Intrusion detection (IDS) and prevention (IPS) systems. 2005.

Termín zadání: 1.2.2017 ***Termín odevzdání:*** 8.6.2017

Vedoucí práce: Ing. Zdeněk Martinásek, Ph.D.
Konzultant:

doc. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Cieľom úlohy je vytvoriť laboratórnu úlohu pre študentov a oboznámiť ich s problematikou IDS a IPS systémov. Pre tento účel sme vybrali voľne šíriteľný program Snort. Oboznámime sa s jeho inštaláciou a konfiguráciou v dvoch rôznych režimoch. Prvý je režim NIDS v ktorom si definujeme jednoduché pravidlo a otestujeme ho. Využijeme aj doplnkové programy, ktoré pracujú so Snortom. Zachytenú komunikáciu si zobrazíme v grafickom prostredí Base. Pokračovať budeme konfiguráciou NIPS, ktorá na rozdiel od predošlej rozhodne aká bude ďalšia akcia zachytených paketov. Na základe tejto časti vytvoríme zložitejšie pravidlá pomocou ktorých budeme testovať útok na FTP server a budeme sa snažiť obmedziť dopad DoS útoku na náš Server. Po testovaní budeme vytvárať laboratórnu úlohu, ktorá zoznámi študentov o konfigurácii programu, jeho režimoch a o vytvorení vlastných pravidiel, ktoré otestujú vo virtuálnom prostredí.

KĽÚČOVÉ SLOVÁ

Bezpečnosť, Dos, Hping3, IDS, IPS, NIDS, Snort, Pravidlá, Útok

ABSTRACT

The aim of the task is to create a laboratory exercise for students, and acquaint them with the issue of IDS and IPS systems. For this purpose we have selected the open source program Snort. We will learn about its installation and configuration within two distinguished modes. First one is the NIDS mode in which a simple rule will be defined and tested. Moreover, the additional programs working with Snort will be used. The intercepted communication will be displayed in graphic environment Base. Furthermore, we will continue with the NIPS configuration, which contrary to the previous one will decide the further action of intercepted packets. Based on this part, we will create more complex rules which we will used to test an attack on FTP server and we will try to limit the impact of DOS attack on our server. Following the testing we will be creating a laboratory exercise that will acquaint students with the program's configuration, its modes and the creation of own rules that they will test in the cyber environment.

KEYWORDS

Attack, DoS, Hping3, IDS, IPS, NIDS, Rules, Security, Snort,

PITSCHMANN, Andrej *Systémy pro detekci a prevenci průniků*: semestrálny projekt. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2017. 55 s. Vedúci práce bol Ing. Zdeněk Martinásek, Ph.D.

VYHLÁSENIE

Vyhlasujem, že som svoj semestrálny projekt na tému „Systémy pro detekci a prevenci průniků“ vypracoval(a) samostatne pod vedením vedúceho semestrálneho projektu, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor(ka) uvedeného semestrálneho projektu ďalej vyhlasujem, že v súvislosti s vytvorením tohoto semestrálneho projektu som neporušil(a) autorské práva tretích osôb, najmä som nezasiahol(-la) nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý(-á) následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

POĎAKOVANIE

Rád bych poděkoval vedoucímu diplomové práce panu Ing.Zdenkovi Martináskovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

POĎAKOVANIE

Výzkum popsaný v tomto semestrálnom projektu bol realizovaný v laboratóriách podporených projektom SIX; registračné číslo CZ.1.05/2.1.00/03.0072, operačný program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .

podpis autora(-ky)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OBSAH

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ÚVOD

V tejto práci bude vytvorené vlastné virtuálne prostredie v ktorom bude simulovaný DoS útok pomocou Hping3 nástroja z OS Kali Linux, ktorý je vytvorený za účelom penetračných testov. Dôležitou úlohou je zachytiť komunikáciu, analyzovať ju a rozhodnúť či je bezpečná pre našu sieť. Na zachytávanie komunikácie bude použitý voľne dostupný program Snort. Zoznámi nás s jeho inštaláciou a jeho základnou konfiguráciou. Samotný program súbory ukladá aj v binárnych súboroch, takže ho rozšírime o doplnkový program Barnyard2, ktorý umožňuje ich čítanie. Všetky zachytené dáta sa budú ukladať v databáze MySQL, ktorá bude s programami spolupracovať. Aby bolo možné zachytiť komunikáciu, musíme mať definované určité pravidlá. Tie sa budú sťahovať od výrobcu alebo budeme vytvárať vlastné. Po spracovaní zachytenej komunikácie sa budú výsledné dáta zobrazovať v grafickom rozhraní nadstavby Base kde budú podrobne rozdelené podľa typu, počtu, zdrojovej IP, cieľovej IP a podobne.

V druhej časti sa bude konfigurovať Snort do IPS módu, teda nielen že zachytí ale aj vyhodnotí či sa jedná o škodlivú komunikáciu alebo nie. Všetko bude závisieť od pravidiel, ktoré budeme vytvárať. Na všetky pravidlá prevádzky sa bude robiť aj príslušný test aby sme zistili či je pravidlo napísané správne. Nakoniec bude vyvolaný samotný Dos útok na FTP server a kde budeme sledovať ako pracuje Snort.

Záverečná práca bude prílohe obsahovať laboratórnu úlohu pre študentov, ktorí sa oboznámia s inštaláciou a konfiguráciou Snortu, jeho nadstavieb, vyskúšajú si IDS aj IPS mód a hlavne budú tvoriť vlastné pravidlá. Všetky pravidlá budú počas práce testované a zobrazované na VM Snort. Na USB kľúči budú priložené tri virtuálne prostredia na ktorých sa celá práca uskutoční.

1 INTRUSION DETECTION SYSTEM

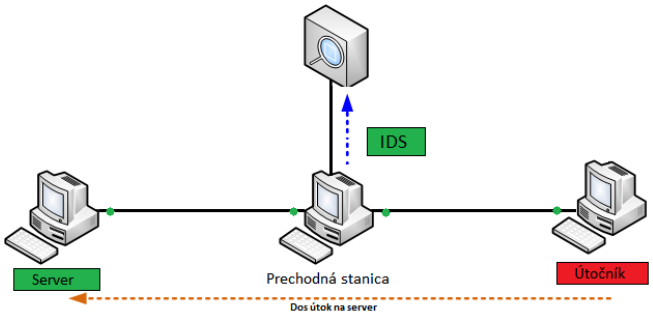
IDS systém je sada techník a metód, ktoré zachytávajú podozrivé aktivity v sieti a na koncových užívateľoch. Po zachytení nežiadúcej komunikácie ihneď informuje administrátora siete. Pre jednoduchý príklad môžeme IDS prirovnať k alarmu v budove. Po vniknutí nežiadúcej osoby nevie zabrániť jej vniknutiu ale vie to ohlásiť majiteľovi. IDS prehliada obsah sieťovej komunikácie a snaží sa odhaliť náznak útoku [9]. IDS porovnáva prieniky podľa dvoch kategórií :

1. **signature-based** – každý útok nesie so sebou určitú signatúru ako počítačový vírus, ktorý je odhalený detekčným softvérom. Snažíme sa nájsť dátové pakety, ktoré obsahujú akúkoľvek signatúru. Na základe pravidiel a porovnávaní je systém schopný nájsť a zaznamenať podozrivú aktivitu načo v prípade zhody reaguje alarmom [9].
2. **anomaly-detection** – súčasť detekcie, ktorá sa zaoberá paketovou anomáliou v hlavičke protokolu. V niektorých prípadoch má viac výsledkov práve táto metóda v porovnaní so signature-based IDS [9].

Systém detekcie prienikov podľa umiestnenia rozdeľujeme na tri základné spôsoby. Netwok-based, Host-based a spojením týchto dvoch umiestnení vznikne Distrbuted IDS. Každé umiestnenie má svoje výhody ktoré budú ďalej popísané [9].

1.1 Network-based IDS

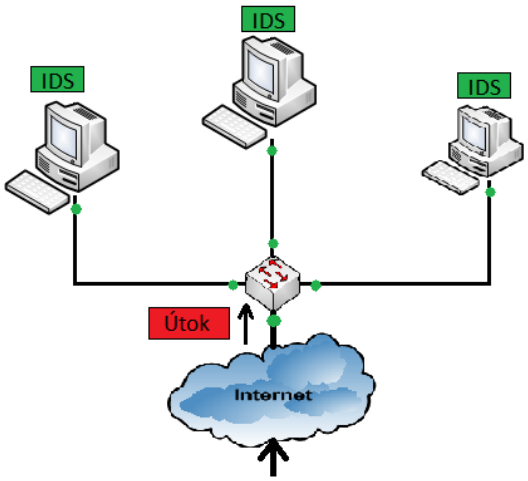
Musí byť umiestnené na veľmi dobré strategické miesto v sieti v ktorej hľadá príznaky aby ich mohlo porovnať so svojou databázou. Sleduje tok siete pomocou promiskuitného módu sieťovej karty hostiteľského systému. Pri identifikácii útoku informuje administrátora. NIDS pracuje v režime on-line aj off-line. On-line pracuje so sieťovým tokom dát v reálnom čase. Analyzujú sa ethernetové pakety na ktoré sú aplikované pravidlá a rozhoduje sa či sú škodlivé alebo nie. Off-line režim pracuje s uloženými dátami, ktoré prechádzajú ďalšími procesmi a overuje sa ich spoľahlivosť. Príklad zapojenia v sieti je zobrazený na obrázku 1.1 [7].



Obr. 1.1: Príklad topológie so zapojením Network-Based IDS

1.2 Host-based IDS

Obstaráva ochranu koncovej stanice, čím je väčšinou užívateľ PC. Pri útoku alarmuje používateľa alebo administrátora. Dokáže identifikovať druhy útokov, ktoré by NIDS nebol schopný. Nevýhoda tohto systému vzniká pri útoku na sieťovú vrstvu. Okrem odstavenia celého systému zlyhá aj HIDS pretože je jeho súčasťou. Príklad zapojenia v sieti je zobrazený na obrázku 1.2 [7].



Obr. 1.2: Príklad topológie so zapojením Host-Based IDS

1.3 Distributed IDS

Kombinácia predošlých dvoch spôsobov. V sieti sa nachádzajú senzory, ktoré sledujú tok dát, pričom o tom informujú riadiaci systém. [5].

1.4 Funkcie IDS

1. **Monitorovanie**
   Systém monitorovania oznamuje, že cieľové objekty boli zmenené alebo upravené. Používajú nato pokročilé kryptografické algoritmy. IDS hlási akúkoľvek zmenu súboru alebo prihlásenia. Môžeme spomenúť konkrétny softvér Tripwire. Monitoruje ciel a zmenu okamžite ohlási [6] .

2. **Heuristika**
   Slovo heuristika sa vzťahuje k časti umelej inteligencie. Na analýzu prichádzajúcich dát sa používa skriptovací jazyk. Narušenie zachytávame na základe vzniknutých anomálií v určitom časovom intervale [6].

3. **Poplach**
   V preklade Alert sa tiež radí medzi určitý druh upozornenia. Pri zachytení nezvaného hosťa o tom informuje sieťového administrátora formou vyskakovacieho okna, výpisu v konzole alebo poslania správy. Poplachy sú uložené v zaznamenaných súboroch alebo databáze kde si ich môžme neskôr prezrieť [9].

4. **Falošný poplach**
   Výstraha generovaná pri zhode s pravidlom. Nemusí sa však jednať priamo o útočníka. Môže k tomu dôjsť pri nesprávnej konfigurácii keď používateľ vyšle správu, ktorá je zhodná s pravidlom čo vedie k falošnému poplachu. Predchádzať tomuto problému môžeme pomocou modifikácie existujúcich pravidiel alebo ich vypnutím. Čo je v niektorých prípadoch jednoduchšie [9].

5. **Záznamy**
   Zaznamenaná správa je uložená do vybraného súboru. Samozrejme umiestnenie môžeme sami zmeniť. Uložený záznam môže byť v podobe textového reťazca ale aj v binárnej forme. Pre čítanie binárnych súborov existujú ďalšie programy, ktorými sa budeme zaoberať v kapitole 5.4.4 [9].

2 INTRUSION PREVENTION SYSTEM

IPS systém je softvér, ktorý obsahuje všetky schopnosti IDS a navyše umožňuje zastaviť komunikáciu pri vzniknutom incidente. Vie identifikovať prieskumnú aktivitu, ktorá by mohla znamenať možný útok. Napríklad nejaký útočný nástroj na vytváranie sieťových útokov, forma malware, červy, ktoré vykonávajú obyčajný prieskum činnosti hostiteľa pre následné útoky. Tieto prieskumné činnosti by malo byť IPS schopné odhaliť a upozorniť administrátora, ktorý bude schopný vykonať kroky pre zmenu bezpečnostných opatrení. Prieskumná činnosť je často realizovaná na internete a detekcia už vo vnútri siete. Tak ako pri IDS, rozdeľujeme IPS do niekoľkých kategórií umiestnenia v sieti [13] .

2.1 Netwok-Based IPS

Monitoruje jednotlivé segmenty siete alebo zariadenia pričom analyzuje sieťové, transportné a aplikačné protokoly pred aktivitou ohrozujúcou sieť [13].

2.2 Host-Based IPS

Monitoruje charakteristiky len jedného užívateľa a udalosti, pri ktorých by mohlo dochádzať k podozrivej aktivite. Dobrý príklad pre host-based IPS sú káblové, poprípade bezdrôtové prenosy dát jedného užívateľa, jeho systémové prihlásenia, spustené procesy, prístupy k súborom a ich modifikácia, zmeny v systéme a aplikáciách [13]. Monitorovanie na základe umiestnenia môžeme rozdeliť na:
- **Server** okrem monitorovania OS serveru sa dohliada aj na niektoré spoločné aplikácie.
- **Klient** monitoruje sa OS užívateľa(stolný PC alebo notebook) ale aj bežiace aplikácie ako internetový prehliadač alebo FTP prenos
- **Aplikačné služby** sú taktiež monitorované ako napríklad webový server, databázový server alebo DNS server

2.3 Wireless IPS

Monitoruje bezdrôtovú komunikáciu v sieti a analyzuje bezdrôtové protokoly pre zistenie podozrivej aktivity. Wi-fi útoky vyžadujú umiestnenie útočníka alebo zariadenia v blízkosti bezdrôtovej sieti. Mnoho sietí sú nakonfigurované pre verejnosť čo umožňuje prístup komukoľvek. Wi-fi senzory majú rovnakú základnú úlohu ako v Network-based IPS ale fungujú veľmi odlišne z dôvodu zložitosti monitorovania

bezdrôtových sietí. Wireless IPS pracuje na základe vzorkovania sieťovej prevádzky. Pracujeme v dvoch frekvenčných pásmach a to 2,4 GHz a 5 GHz. Nieje možné aby senzor pacoval v celom frekvenčnom pásme v rovnakom čase. Pri monitorovaní jedného kanálu ostatné frekvencie vypne. Čím dlhšie sleduje jeden kanál tým je vyššia pravdepodobnosť že sa škodlivý malware dostane do siete cez iný kanál [13].

2.4 Network Behavior Analysis

Označuje sa aj ako NBA, skúma sieťovú prevádzku alebo štatistiky sieťového toku pre zistenie nezvyčajných tokov dát ako napríklad útok DoS, formu vírusu alebo porušenia oprávnení. Skúmané údaje v toku dát sú dôležité pre detekciu narušenia [13]. Preveruje sa:
- Zdrojová a cieľová IP adresa
- Zdrojový a cieľový TCP a UDP port
- Počet paketov a veľkosť prenesených dát v jednom spojení
- Časové údaje začatia a skončenia prenosu

2.5 IPS odhaľovanie chýb

1. **Signature-based** – Veľmi efektívna metóda pri odhaľovaní už známych hrozieb ale neúčinná pri odhaľovaní predošlých hrozieb len v inej forme. Napríklad ak je odhalený malware pod určitým názvom súboru po jednoduchej zmene v názve nebude odhalený. Signature-based porovnáva tok paketov v reálnom čase so zoznamom signatúr obsahujúci porovnávacie operácie [13].
2. **Statistical anomaly-based** – je proces porovnávania definícií aká prevádzka je považovaná za normálnu a takú kde sú výrazné odchýlky. Obsahuje profily reprezentujúce normálne správanie všetkých užívateľov, sieťového pripojenia alebo aplikácie. Profily boli vytvorené pri pozorovaní charakteristických znakov normálnej prevádzky v určitom čase. Môžu obsahovať vyťaženie HW, počet poslaných emailov užívateľa a podobne. Na základe týchto parametrov sa potom rozoznávajú anomálie. Hlavnou výhodou je že vedia odhaliť predošlé útoky v nepoznaných formách [13].
3. **Stateful Protocol Analysis** – Je proces porovnávania vopred určených profilov ako sa majú a nemajú jednotlivé protokoly používať. Stavové určenie znamená, že IPS je schopné pochopiť a vystopovať sieťové, transportné a aplikačné protokoly a ich aktuálny stav v sieti [13].

3 ÚTOKY NA SIEŤOVÚ INFRAŠTRUKTÚRU

Veľmi dôležitá problematika v dnešnom elektronickom svete sú kybernetické útoky. Musíme brániť, aby sme neprišli o cenné osobné informácie, ktoré by mohol niekto zneužiť. Všetci si uvedomujeme hrozby z rozľahlej časti internetu ale riziko nesú aj vnútorné LAN siete. Stávajú sa čoraz viac nebezpečnejšie [15].

3.1 Denial of Service

V jednoduchosti znamená znemožnenie prístupu služieb akýmkoľvek spôsobom, od vytiahnutia kábla zo zariadenia ktorý poskytuje službu alebo využitie nenápadnej slabiny v aplikácií. Pri analýze rizík potrebujeme brať do úvahy všetky zariadenia v sieti, ktoré hrajú rolu poskytovateľa služieb. Okrem serverov a databázy musíme rátať aj so smerovačmi, zálohovými servermi, autentizačnými servermi, prepínačmi a podobne. Pri každom zariadení musíme brať na vedomie ich možnosť zraniteľnosti a ich riziká. V niektorých prípadoch ide útočníkom skôr o narušenie činnosti siete, ako o získanie informácií. Rozdiel medzi DoS a DDoS útokom je v počte použitých staníc na útok, ktoré pomocou infikovaných počítačov útočia na obeť. Použijú master-zombie[1] stanicu, ktorá ovláda všetky slave-zombie[2] stanice [2, 12].

3.1.1 DoS útoky

Rozdelenie DoS útokov podľa vrstvy OSI modelu je pre lepší prehľad rozdelené v tabuľke 3.1.

1. **Fyzická vrstva**
   Pri nedostatočnom zabezpečení miestnosti, kde máme umiestnenú serverovňu sa môže stať, že sa útočník dostane až dnu a môže vypnúť zo siete alebo poškodiť niektoré zo zariadení. Zneprístupní nám tak internetovú službu alebo nás úplne odstaví od pripojenie na internet.

2. **Spojová vrstva**
   ARP Spoofing – Druh útoku, kde útočník zo seba urobí DG. Cez tento bod začnú počítače komunikovať a páchateľ bude zachytávať celý dátový tok. Pri útoku sa nebude komunikácia preposielať ale sa bude zahadzovať a nikto sa nebude môcť pripojiť do siete. Mnohé firewall brány dokážu tento spôsob útoku zachytiť. Útoku môžeme predchádzať tak, že na počítači nastavíme statické položky v ARP cachce [2][15].

[1] Master zombie – napadnutý počítač, ktorý ovláda iné stanice
[2] Slave zombie – označovaný ako otrok, poslúcha master zombie

3. **Sieťová vrstva**

   TTL záplava – Ide o reflektívny útok pri ktorom sa využívajú hodnoty životnosti paketu TTL. Životnosť paketu zabezpečuje, aby sa sieť nezahltila. Pri nastavení zdrojovej IP adresy odosielateľa zvolíme IP adresu obete a nastavíme malé TTL. Dáta rozpošleme na náhodné cieľové stanice. Po vypršaní životnosti odosielateľ obdrží informáciu o vypršaní od všetkých cieľov čo spôsobí zahltenie [1].

4. **Transportná vrstva**

   Syn Flood – Útočník prevezme kontrolu nad niekoľkými užívateľmi a prikáže im spojiť sa so serverom. Sfalšuje sa zdrojová IP adresa, ktorú nasmerujeme na obeť. Po nadviazaní spojenia s cieľom sa začnú posielať TCP/IP SYN pakety ktoré obsahujú zlú zdrojovú IP adresu, teda server sa nedočká odpovede. Na každý paket odpovedá SYN/ACK paketom pre spoľahlivé nadviazanie komunikácie. Web server udržuje dátovú štruktúru pre každú SYN žiadosť, ktorá čaká odpoveď a pri veľkej prevádzke nastáva zrútenie služby [1, 2].

9. **Aplikačná vrstva**

   Vrstva koncových užívateľov, takže obsahuje veľkú škálu možností rôznych útokov. Pre príklad si uvedieme DHCP útok.

   DHCP útok – Dynamické prideľovanie IP adries, ktoré podobne ako DNS služba pracuje na princípe žiadosť a odpoveď. Zo zoznamu prideľuje adresy na zariadenia, ktoré si o to požiadajú. Po vyčerpaní voľných IP adries nebude schopný prideliť všetkým užívateľom IP adresu a tak nebudú mať prístup do siete [2].

Tab. 3.1: Tabuľka rozdelenia vybraných útokov podľa vrstvy OSI modelu

Vrstva	**Útok**
Aplikačná	Fork Bomb
	HTTP záplava
	DHCP útok
Transportná	P2P útok
	SYN záplava
Sieťová	ICMP záplava
	Ping záplava
	Smurf
Spojová	ARP spoofing
Fyzická	Vypojenie kábla

4 SNORT

Snort je open-source IDS/IPS nástroj, ktorý monitoruje tok paketov a zaznamenáva našu sieťovú komunikáciu v reálnom čase. Môže vykonať paketovú analýzu a je schopný odhaliť rôzne druhy útokov. Všetky tieto úkony vykonáva na základe určitých pravidiel sieťovej prevádzky[14].

Signatúri
Vo svete bezpečnosti, toto slovo dostalo niekoľko definícií v priebehu niekoľkých rokov. Signatúra je súčasť určenej detekčnej metódy, ktorá sa zakladá na rozlišovaní značiek alebo charakteristík vhodných na zranenie systému. Príznaky sú špeciálne vytvorené na odhalenie známych hrozieb. Tento typ technológie používa aj Antivírusový softvér aby chránil užívateľov pred hrozbami. Ako sme už zistili tento typ ochrany má obmedzené možnosti a vírus môže infikovať obeť predtým ako je príznak zapísaný [14].

Pravidlá
Rôzne metódy na vykonávanie detekcie, ktoré si narozdiel od signatúr zakladajú na aktuálnej zraniteľnosti. Vytvorenie vlastných pravidiel vyžaduje pochopenie ako to v sieti funguje [14].

Zraniteľnosť
Veľa ľudí sa pokúšalo zistiť čo vlastne slovo zraniteľnosť znamená a ako je spojená s bezpečnosťou. Vyjadruje myšlienku, že zraniteľnosť je nejaká slabina, ktorá vznikla vo väčšine prípadov neúmyselne. Hlavne keď je implementovaná alebo použitá správne útočník sa môže zmocniť vyšších privilégií, ktoré regulujú lokálne chránené operácie kde by boli ohrozené údaje alebo dôveryhodnosť. Zahŕňa to všetko od pretečenia pamäte alebo na špecifické nedostatky, ktoré značia jednoduchosť sociálneho inžinierstva. Zraniteľnosti sa vyskytujú v každom zariadení už od samých začiatkov modernej doby [14].

Protokol
Každý počítač musí komunikovať v sieti prostredníctvom niečoho. Používa na to sadu pravidiel , ktorá sa nazýva protokol. Môžeme si to uviesť na príklade.

Klient pošle systému Client HELLO správu, dostáva odpoveď Server HELLO. Obe strany si vymenia kľúčovú informáciu aby zistili aký typ šifrovania používajú. Následne obe strany začnú šifrovanú komunikáciu pomocou protokolov, ktoré nesú správy [14].

DAQ
Celým názvom Data Acquisition library pre riadenie vstupu a výstupu paketov. Pri štarte vyberieme typ DAQ a mód režimu v ktorom bude pracovať Snort. DAQ môžeme nastaviť cez príkazový riadok alebo ho uložiť v konfiguračnom súbore `snort.conf`. Po inštalácii Snortu je možné vypísať si dostupné typy DAQ príkazom `snort --daq-list`. Všetky sú uložené v zložke `usr/local/lib/daq`. Niekedy je možné že príkaz nezobrazí všetky typy, ktoré Snort obsahuje [14].

4.0.2 Písanie pravidiel

V tejto časti si vysvetlíme základy pre tvorbu pravidiel, ktoré budeme neskôr aplikovať do našej virtuálnej siete. Nebude spomenuté úplne všetko ale len základy. Každé pravidlo sa skladá z niekoľkých častí [10]:

1. Akcia: Akcia vyjadruje čo bude spravené s detekovaným paketom. Možnosti sú:

1. Alert: Upozorní nás na detekovný paket
2. Log: zaznamená paket
3. Pass: ignoruje detekovaný paket
4. Drop: zahodí paket
5. Reject: blokuje paket a zaznamená ho vo forme logu.

2. Protokol: V tejto časti definuje jeden zo 4 protokolov a to ICMP, IP , TCP, UDP. Momentálne sa uvažuje aj o pridaní viacerých protokolov ako RIP, OSPF, IPx, ARP a ďalšie.

3. IP adresa a port: Definuje zdrojovú IP adresu a port z ktorej komunikácia prichádza. V základnom nastavení v tejto časti píšeme `$EXTERNAL_NET` a `any`. Any znamená že očakáva komunikáciu zo všetkých portov.

4. Smer komunikácie: Obsahuje znaky komunikácie ako:
`->` – prichádzajúca komunikácia do našej siete
`<-` – odchádzajúca komunikácia do siete
`<>` – obojstranná komunikácia

5. Cieľová adresa a port: V tejto časti definujeme nami zadnú IP adresu, teda 10.0.20.0/24 ako `$HOME_NET`.

6. Doplňujúce možnosti: Obsahuje základné informácie o pravidle, napríklad správu ICMP paket bol práve zachytený. Okrem toho zaznamenáva aj obsah paketov,

napríklad nesúce dáta, značky, veľkosť, typ a podobne [10].

4.0.3 Možnosti pravidiel

Pre každý paket nesúci informácie si nastavíme podmienky, za ktorých ho budeme vedieť lepšie rozoznávať. Spravíme to na základe toho čo v sebe obsahuje [10].

Správa:, skrátene msg obsahuje čokoľvek čo si sen napíšeme. Slúži ako správa pre administrátora. Objaví sa pri zhode s pravidlom. Jej syntax je nasledovný: `msg:"správa"`.

Referencia môže obsahovať napríklad url adresu, ktorá odkazuje na externé systémy. V tejto úlohe nebudeme využívať túto možnosť. Zapisuje sa v tvare [10]:

`reference:<id system>,<id>; [reference:<id system>, <id>;]`

GID: Celým nazvom generator ID identifikuje, ktorá časť snortu generuje udalosť pri spustení pravidla.

`gid:<generator id>;`

SID: Identifikuje jedinečné Snort pravidlo. Ak sa v pravidlách nachádzajú dva pravidlá s rovnakým SID, Snort výhlási chybu a nepustí sa. Tomuto prípadu sa musíme naučiť predchádzať.
`<100` – rezetvované pre neskoršie použitie
`100-999 999` – pravidlá distribuované Snortom
`>1000 000` – lokálne pravidlá
Zapisujeme v tvare:

`sid:<snort rules id>;`

REV: označuje revíziu Snort pravidla spolu s ID a spolu zabezpečujú stálu aktuálnosť pravidla. Malo by sa písať spolu s SID. Zapisujeme v tvare:

`rev:<revision integer>;`

CLASSTYPE: Hlavným dôvodom tejto doplnkovej možnosti je kategorizácia praividiel podľa typu ohrozenia. Snort obsahuje základné typy útokov rozdelené do tabuľky 4.1 od priority ohrozenia najviac 1 po najmenej 4 [10]. Zapisujeme v tvare:

`classtype:<class name>;`

V tejto bakalárskej práci sa nestretneme s vysokým ohrozením ale len so strednou triedou ohorozenia. Všetky závisí od toho ako zostavíme pravidlo. Do tabulky 4.2 si v skratke zhrnieme niektoré doplňujúce možnosti pri tvorbe pravidiel pre Snort.

Tab. 4.1: Tabuľka klasifikácie paketov [10]

Klasifikácia	Popis	Priorita
policy-violation	Porušenie osobných údajov	Vysoká
inappropriate-content	Nepovolený obsah	Vysoká
denial-of-service	Potencionálny DoS útok	Stredná
suspicious-login	Podozrivé prihlásenie	Stredná
icmp-event	Zistená ICMP udalosť	Nízka
tcp-connection	Vytvorené TCP spojenie	Veľmi nízka

Tab. 4.2: Doplnkové možnosti pri tvorbe pravidla [10]

	Možnosť	Popis
1.	MSG	Obsahuje správu pre administrátora
2.	REFERENCE	Umožňuje použiť externý systém
3.	GID	Identifikuje časť Snortu, ktorá bude použitá pre generovanie udalosti
4.	SID	Jedinečný identifikátor pravidla
5.	REV	Revízne číslo
6.	CLASSTYPE	Klasifikácia ohrozenia

4.0.4 Získavanie paketov

Vo verzii Snortu 2.9 sa objavila novinka s názvom DAQ, teda Data Acquisition library pre spracovanie vstupu a výstupu paketov. V predchádzajúcej časti sme si povedali niečo o typoch DAQ a kde sa nachádzajú. Na obrázku v zakomentovanej časti konfiguračného súboru Snortu na obrázku 5.11 vidíme aj vzor ako by mohla naša konfigurácia vyzerať. Vidíme tam typ získavania paketov ako napríklad afpacket, dump a príslušný mód alebo miesto pre aktuálne DAQ moduly. Základne je to nastavené na `/usr/local/daq` [10][14].

V prípade, že nemáme možnosť editovať konfiguračný súbor, môžme všetky podstatné parameter napísať do príkazu. V tomto prípade INLINE módu s knižnicou X spustíme snort ako:

```
Snort -c /etc/snort/snort.conf -I ens38:ens39 -daq X --daq-mode
inline --daq-var buffer_size_mb=1024
```

Ako sme už videli vyššie, DAQ delíme na niekoľko možností:
PCAP: je základné nastavenie Snortu, ktoré umožnuje čítať zachytené pakety aj v prípade, že nieje pripojený ku ďalšiemu zariadeniu.

Obr. 4.1: Zobrazenie základnej konfigurácie so vzormi DAQ

AFPACKET: v tomto prípade musíme určiť viac ako jedno rozhranie na ktorom bude Snort počúvať. Rozhrania musia byť oddelené dvojbodkou. Snort može počúvať aj viac ako na dvoch rozhraniach ale musi byť v poradí ens38:ens39:ens40:ens41. Nemôžu byť poprehadzované ľubovoľne.
NFQ: je nova schopnosť spracovávania paketov na základe IP tabuliek.
DUMP: metóda, ktorá umožňuje testovanie INLINE módu vo viacerých možnstiach. Vzužíva PCAP pre zachytávanie paketov.

4.0.5 Filtre

Nové verzie Snortu od 2.8.5 okrem iného obsahujú aj nové funkcie ako sú rôzne typy filtrov ako sú detekčné filtre, rýchlostné filtre a filtre udalostí. Ich úlohou je kontrolovať tvorbu toku, spracovávanie alebo zaznamenávanie do predom určených zložiek. Zastupujú radu predchádzajúcich funkcií ktoré obsahovali staršie verzie tohto voľne dostupného programu [10][14].

1. **Detekčný filter** – Nový druh filtra, ktorý nahrazduje „trashold“, ktoré limituje počet vyhlasených alarmov počas zachytávania komunikácie. Detečný filter definuje mieru prekročenia vzťahujúce sa na určité pravidlo podľa zdrojovej alebo cieľovej adresy, ktoré následne generuje vhodnú akciu. Zapisuje sa v tvare:

   ```
   detection_filter: \ track <by_src|by_dst>, \ count , seconds ;
   ```

   nastavenie podľa zdrojove/cieľovej IP: stopuje komunikíciu na základe zdrojovej alebo cieľovej IP adresy.
   Count: maximálny počet pokusov zachztených z jednej IP za 1 sekundu
   Seconds: časový interval za ktorý stúpne počet pokusov
2. **Rýchlostné filtre** – Filtre poskytujú užívateľom pridať akciu, ktorá rozhodne o ďalšej akcii počas určitého časového intervalu pokiaľ je rýchlosť pripojovania

prekročená. Na jedno pravidlo môže byť definovaných aj viac pravividiel. Vyhodnocujú sa v poradí v akom sa nachádzajú v konfiguračnom súbore. Syntax pravidla:

```
rate_filter \ gen_id , sig_id , \ track <by_src|by_dst|by_rule>,
\ count , seconds,\new action,\timeout
```

```
rate_filter \
        gen_id 20, sig_id 20, \
        track by_src/by_dst, \
        count 2, seconds 1, \
        new_action drop/reject/pass, timeout 5
```

Obr. 4.2: Nastavenia Rate filtra v konfigurácii

zdrojová/cieľová IP: rýchlosť je aplikovaná na unikátnu zdrojovú/cieľovú IP alebo na odkazujúce pravidlo.
Count c: maximálny počet pripojení z jednej unikátnej IP za jednu sekundu. Hodnota musí bzť kladná.
Seconds s: časový limit, počas ktorého stúpne počet prístupov
New action: nahradzuje pôvodnú akciu v pravidel novou, může to byť alert, drop, reject, pass
Timeout: doba počas ktorej nieje dostupný prístup

3. **Udalostnný filter** – Filter, ktorý obmedzuje počet pripojení na základe definovaného pravidla Formát tohto filtru je nasledovný:

```
event_filter \ gen_id , sig_id , \ type <limit|threshold|both>,
\ track <by_src|by_dst>, \ count , seconds
```

```
event_filter \
        gen_id xx, sig_id xx, \
        type limit/threshold/both, track by_src/by_dst,  \
        count 10, seconds 5
detection_filter
```

Obr. 4.3: Nastavenia Event filtra v konfigurácii

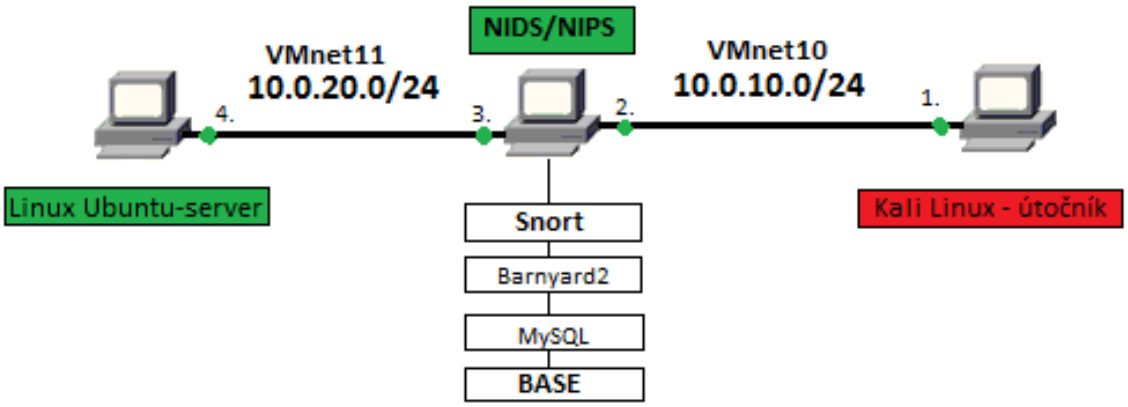
SID,GID: unikátne číslo príznaku a generátor [10][14].
Typ: obsahuje možnosti limit,treshold a both. Tieto typy zaznamenajú maximálny počet pokusov o pripojenie a následne ignorujú všetko ostatné počas určitého časového intervalu.

5 PRAKTICKÁ ČASŤ

Vo virtuálnom prostredí budeme simulovať jednoduchú lokálnu sieť, ktorá bude vytvorená v prostredí softvéru WMware Workstation 12 a bude obsahovať tri stanice. Stredná stanica nastavená pre detekciu útokov bude spájať dve rôzne podsiete medzi ktorými bude nastavené statické smerovanie pre lepšiu možnosť odchytávať celú komunikáciu smerujúcu na server. Topológia siete je zobrazená v časti Topológia siete obrázok A.1. Prvý bude Server, ktorý bude inštalovaný na OS Linux Ubuntu 16-04 64-bit. Tam bude spustený Apache2, webová aplikácia pre tvorbu HTTP serveru a FTP. Na ďalšom virtuálno stroji bude inštalovaný program Snort taktiež spustený na OS Linux Ubuntu 16-04 64-bit. Bude obsahovať mnoho iných stiahnutých programov na kompletné spracovanie uložených súborov zo Snortu v. 2.9.8.3 spolu s pravidlami Pulled Pork v. 0.2.7. Barnyard2 v. 2.1.14 bude komunikovať s MySQL v. 14.14 databázou verzie. O grafické spracované súbory sa postará užívateľský výstup Base. Po základnej konfigurácii Snortu ako NIDS budeme po otestovaní pokračovať konfiguráciou ako NIPS. V tejto časti sa jedná hlavne o dobre napísané pravidlá na základe ktorých sa bude prepúšťať komunikácia. V poslednej časti budeme pracovať ako útočník na tretej stanici s OS Linux Kali. Vytvoríme DoS útok na FTP server. 2016 64-bit.
Na konci v prílohe bude vypracovaná laboratórna úloha pre študentov, ktorá ich oboznámi s problematikou NIDS/NIPS. Bude obsahovať základnú inštaláciu a konfiguráciu Snortu s jeho nadstavbami. Otestujeme pravidlá vytvorené od výrobcu ale aj vlastné, ktoré aplikujeme do našej virtuálnej topológie.

5.1 Topológia siete

Vytvoríme jednoduchú LAN sieť kde umiestnime 3 stanice. Stanice sa nachádzať v dvoch rozdielnych podsieťach, takže na všetkých staniciach budeme musieť nastaviť statické cesty k ďalšiemu skoku. Okrem statickej konfigurácie budeme musieť na stanici Snort povoliť preposielanie Ipv4 paketov ako na obrázku 5.2 aby sme boli schopní plnej komunikácie od stanice Server po útočníka. Adresy budeme priraďovať postupne pre každú stanicu v konfiguračnom súbore rozhraní:
`:~$ sudo vim /etc/networks/interface`.
Všetky adresy sú zobrazené v adresnej tabuľke 5.1.



Obr. 5.1: Topológia zapojenia našej virtuálnej siete s NIDS/NIPS systémom

```
:~# vim /etc/sysctl.conf
27 # Uncomment the next line to enable packet forwarding for IPv4
28 net.ipv4.ip_forward=1
```

Obr. 5.2: Povolený konfiguračný riadok pre-posielania paketov na VM Snort

5.2 VIM editor

Pri editácii súborov Snortu a jeho nadstavieb sa pre efektívnejšie pohybovanie v obsiahlych zložkách musíme lepšie naučiť pracovať s editorom VIM. V tabuľke 5.2 sú umiestnené základné príkazy pre ukladanie, orientáciu, a modifikovanie konfiguračných súborov. Samozrejme existujú aj iné editory ako napríklad gedit, nano. My budeme v tejto bakalárskej práci používať editor VIM [8].

5.3 Hping3 manuál

Pre simulovanie DoS útoku budeme potrebovať vedieť ako pracuje program pomocou ktorého budeme útok vytvárať. Jedná sa o jednoduchý program s množstvom funkcií od pomalého posielania ICMP, TCP a UDP paketov po čo najrýchlejšie posielanie. Okrem iného obsahuje aj možnosti pre modifikáciu paketov ich veľkosť, dĺžku, možnosť nastaviť cieľový port, zmenu zdrojovej IP adresy a mnoho ďalšieho. V tabuľke 5.3 sú spomenuté základne parametre, ktoré budú potrebné pre neskoršie účely. Pre rozsiahlejší manuál stačí zadať príkaz `man hping3` na VM Kali Linux. [11].

Tab. 5.1: Tabuľka IP adries

Stanica	**Rozhranie**	**IP adresa**
SERVER	4.	10.0.20.6/24
NIDS	3.	10.0.20.7/24
	2.	10.0.10.7/24
ÚTOČNÍK	1.	10.0.10.6/24

Tab. 5.2: Tabuľka používaných príkazov vo VIM

	Príkaz	**Popis**
1.	i	insert mód, môžeme zapisovať do súboru
2.	ESC	opustenie predošlého režimu
3.	:wq	uloženie a ukočenie, môže byť použité aj oddelene
4.	:#	# presun na číslo riadku
5.	:set number	označenie riadkov
6.	gg	nastavenia kurzora na začiatok súboru
7.	G	nastavenie kurzora na koniec súboru
8.	/#	#vyhľadávanie
9.	%d	vymaže všetky znaky v súbore
10.	q!	vynútené opustenie súboru

5.4 Snort NIDS

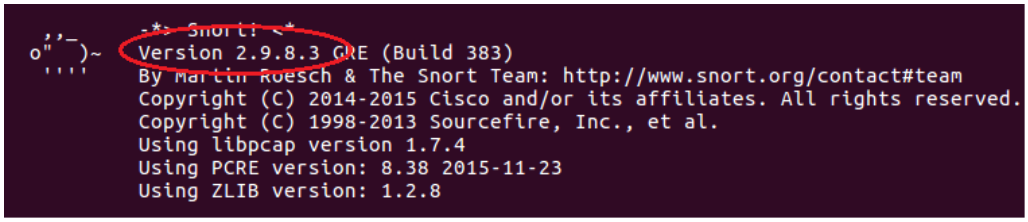
V tejto časti popíšeme samotnú konfiguráciu Snortu ako NIDS. Jeho výstupy budú v binárnej forme čo znamená že to nebudeme vedieť prečítať bez pomoci určitého softvéru. Použijeme preto program Barnyard2, ktorý je preto ako stvorený. Spolupracuje s databázou MySQL do ktorej ukladá všetky záznamy pre neskoršie použitie. Keď bude všetko bez problémov uložené v databáze, ľahko to zobrazíme v grafickom rozhraní BASE, ktorý do nej bude pristupovať. Jeho výstupom bude veľký prehľad o uloženej komunikácii prehľadnou formou podla zdrojovej a cieľovej IP, typu prenášaného protokolu a podobne[3].

5.4.1 Inštalácia

Pred samým začiatkom musíme aktualizovať systém aby sme sa vyhli neočakávaným problémom. Stiahneme si Snort zo stránky výrobcu spolu so Snort DAQ v. 2.0.6 knižnicami, ktoré potrebujeme pre zachytávanie paketov. Po inštalácií otestujeme funkčnosť príkazom `:~$ /usr/bin/snort -V` a výstup musí vyzerať ako na obrázku 5.3.

Tab. 5.3: HPING3 stručný manuál

	Príkaz	**Popis**
1.	- -fast	rýchle posielanie paketov (10/s)
2.	- -flood	posielanie paketov najrýchlejšie ako môžeš
3.	-0	posielanie RAW IP
4.	-1	posielanie ICMP paketov
5.	-2	UDP mód
6.	-a	ukradnutá IP adresa
7.	-s	zdrojový port
8.	-p	cieľový port
9.	-S	Synchronizačné značenie
10.	-c	počet paketov
11.	-d	veľkosť paketov (byt)



Obr. 5.3: Výpis spustenia Snortu a jeho aktuálna verzia

5.4.2 Konfigurácia NIDS

Z bezpečnostných dôvodov si vytvoríme užívateľa Snort ,ktorý bude patriť do novo vytvorenej skupiny Snort. Pre správnu konfiguráciu je potrebné vytvoriť niekoľko súborov a adresárov, ktoré prepojíme s konfiguračným súborom `snort.conf`. Sú to súbory obsahujúce stiahnuté pravidlá, nami vytvorené pravidlá a zložky pre archiváciu dátových tokov. Pre lepší prehľad a uľahčenie práce do budúcna si skopírujeme všetky potrebné konfiguračné súbory na jedno miesto z ktorého budeme Snort spúšťať.

Editujeme hlavný konfiguračný súbor `snort.conf`. Je to veľmi obsiahly súbor v ktorom vyhľadáme cesty k pravidlám a zakomentujeme ich pomocou mriežky. Tieto pravidlá neexistujú a tak by nám program hlásil chyby pri načítavaní týchto súborov. V časti `ipvar HOME_NET` na riadku 45 napíšeme adresu siete aj s prefixom aby vedel o aký rozsah siete sa bude prioritne starať. Nachádzajú sa tu aj zle nastavené

cesty k adresárom, ktoré sme nedávno vytvorili. Opravíme ich a uložíme aktuálnu konfiguráciu [3].

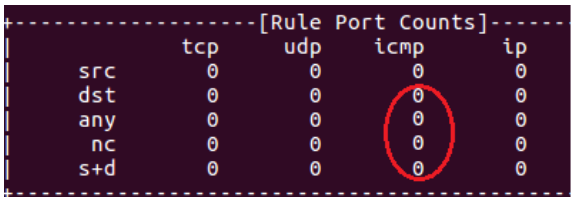
Ak sme všetko urobili správne a Snort je dobre nakonfigurovaný, tak po spustení to bude vyzerať nasledovne(pracujeme na rozhraní ens39). Príkaz pre sputenie a následný výpis konfigurácie:

```
# snort -T -c /etc/snort/snort.conf -i ens39
```

→Snort successfully validated the configuration!

→Snort exiting

Vidíme že v tabuľke pravidiel máme samé nuly. To znamená, že ešte žiadne pravidlo nebolo definované.



Obr. 5.4: Výpis tabuľky pravidiel po aktualizovaní Snortu

5.4.3 ICMP pravidlo

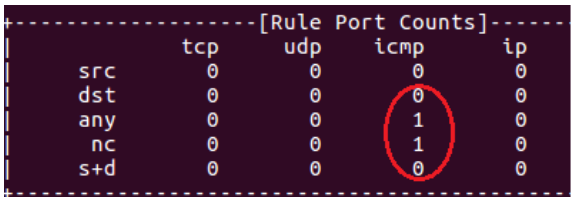
V tejto časti spomenieme základne pravidlo pre zachytenie dostupnosti stanice na druhej strane. Celkovú tvorbu pravidiel si rozoberieme v neskoršej časti Tvorba pravidiel .Pri štarte Snort načíta konfiguračný súbor `snort.conf`, ktorý bude odkazovať na sadu pravidiel „local rules“ v ktorej vytvoríme jednoduché a základné ICMP pravidlo. Bude zachytávať všetky ICMP pakety, ktoré prechádzajú rozhraním. Jedná sa o klasický ping, ktorý zisťuje dostupnosť stanice na druhej strane [3].

Odkomentujeme pravidlá local rules v konfiguračnom súbore Snortu. Tie teraz budeme editovať a testovať. Vložíme do nich jednoduchý riadok ktorý definuje nové pravidlo, ktoré obsahuje akciu ktorá sa má vykonať, zdrojovu IP a port, smer komunikácie, ciľovú IP a port a doplňujúce nastavenia. Všetko bude vysvetlené v ďalšej časti.

```
alert icmp any any -> $HOME_NET any (msg:"ICMP test detected";
GID:1; sid:10000001; rev:001; classtype:icmp-event;)
```

Znova spustíme test ako v predošlom prípade. Teraz by mala tabuľka vyzerať ako na obrázku 5.5

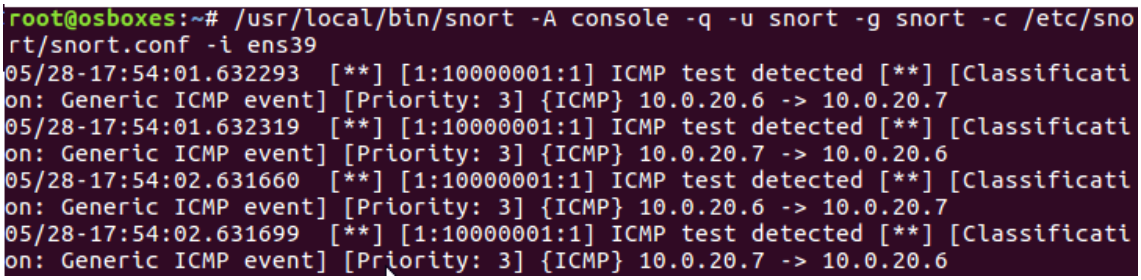
Vidíme že na obrázku nám pribudli dve jednotky v riadku any a nc v stĺpci ICMP. To znamená, že pravidlo je nakonfigurované a už len čaká na otestovanie. Pingneme



Obr. 5.5: Výpis tabuľky pravidiel po aktualizovaní Snortu

rozhranie z hociktorého klienta v sieti napríklad zo servera `ping 10.0.20.6`. Aby sme videli zachytené pakety vo forme upozornenia v terminály ako na obrázku 5.6 musíme spustiť Snort s parametrom `-A cosnsole` ako užívateľ snort v skupine snort, špecifikovať rozhranie a zadať správnu cestu ku konfiguračnému súboru.

```
# /usr/local/bin/snort -A console -q -u snort -g snort -c
  /etc/snort/snort.conf -i ens39
```



Obr. 5.6: Poplach v termináli pri zachytený ICMP paketov

Všetka komunikácia ako ju vidíme v terminály sa uložila do základne definovanej zložky `\var\log\snort\snort.log.xxx`. Ak komunikáciu nechceme ukladať, spustíme príkaz s parametrom N.

5.4.4 Barnyard2

Ako sme už spomínali Barnyard2 je program, ktorý číta binárne záznamy Snortu a ukladá ich do databázy. V tomto kroku budeme inštalovať aj MySQL databázu spolu s úpravou jej konfigurácie. Pri inštalácií MySQL balíka od nás bude žiadať root heslo. Zvolíme si ROOTPASSWORD. Najprv zmeníme konfiguračný súbor Snortu tak, že doplníme riadok ktorý bude hovoriť programu, že výstup má zapísať v binárnej podobe. [3].

```
(r.520) output unified2: filename snort.u2, limit 128
```

Barnyard2 potrebuje vedieť kde sa nachádza MySQL čo závisí na architektúre systému. Používame 64-bitovú verziu takže použijeme príkaz:

```
# ./configure --with-mysql --with-mysql-libraries=/
  usr/lib/x86_64-linux-gnu|
```

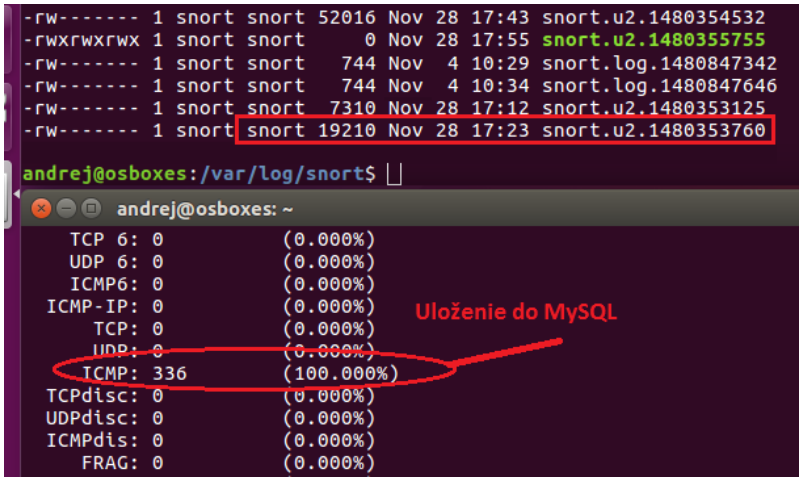
Rovnako ako predtým skopírujeme potrebné konfiguračné Barnyard2 súbory na miesto `/etc/snort/`. Prihlásime sa do MySQL databázy ako užívateľ root. Použijeme heslo, ktoré sme zadávali pri inštalácii. Vytvoríme novú databázu Snort. Ako zdroj použijeme databázu zo stiahnutého súboru: `barn2/schemas/create_mysql`. Vytvoríme používateľa Snort a definujeme mu heslo SNORTPASSWORD. Pridelíme mu pravidlá na základné editovanie databázy.

Znova upravíme konfiguračný súbor Barnyard2 tak, aby ukladal dáta do databázy Snort.

```
output database: log, mysql, user=snort password=SNORTPASSWORD
dbname=snort host=localhost
```

Otestujeme si generovanie správ zo zachytenej komunikácie. Spustíme Snort aj Barnyard2 v jednom príkaze a sledujeme uložené dáta v súbore ako na obázku 5.7: `/var/log/snort/snort.u2.xxx`

```
# barnyard2 -c /etc/snort/barnyard2.conf -d /var/log/snort-f
  snort.u2 -w /var/log/snort/barnyard2.waldo -g snort -u snort
```



Obr. 5.7: Ukážka uloženia paketov v binárnych súboroch

Na obrázku sú zaznamenané len ICMP pakety, pretože sme zadefinovali len jedno pravidlo pre komunikáciu. Ako vidíme môžme zachytávať aj TCP segmenty a UDP datagramy ale aj IPv6 pakety.

5.4.5 PulledPork

Pulled Pork je doplnkový skript, programovaný v jazyku Perl pomocou ktorého stiahneme najnovšie pravidlá zo stránky výrobcu. Stiahneme si samotný Pulled Pork a do konfiguračného súboru `pulledpork.conf` pridáme správne cesty k pravidlám podobne ako v prípade `snort.conf`. Po aktualizovaní pridá do tabuľky viac ako 20 tisíc nových pravidiel pre rôzne druhy známych útokov. Touto cestou sme aktualizovali pravidlá Snortu, ktorý bude vedieť rozoznať nové hrozby [3].

5.4.6 BASE

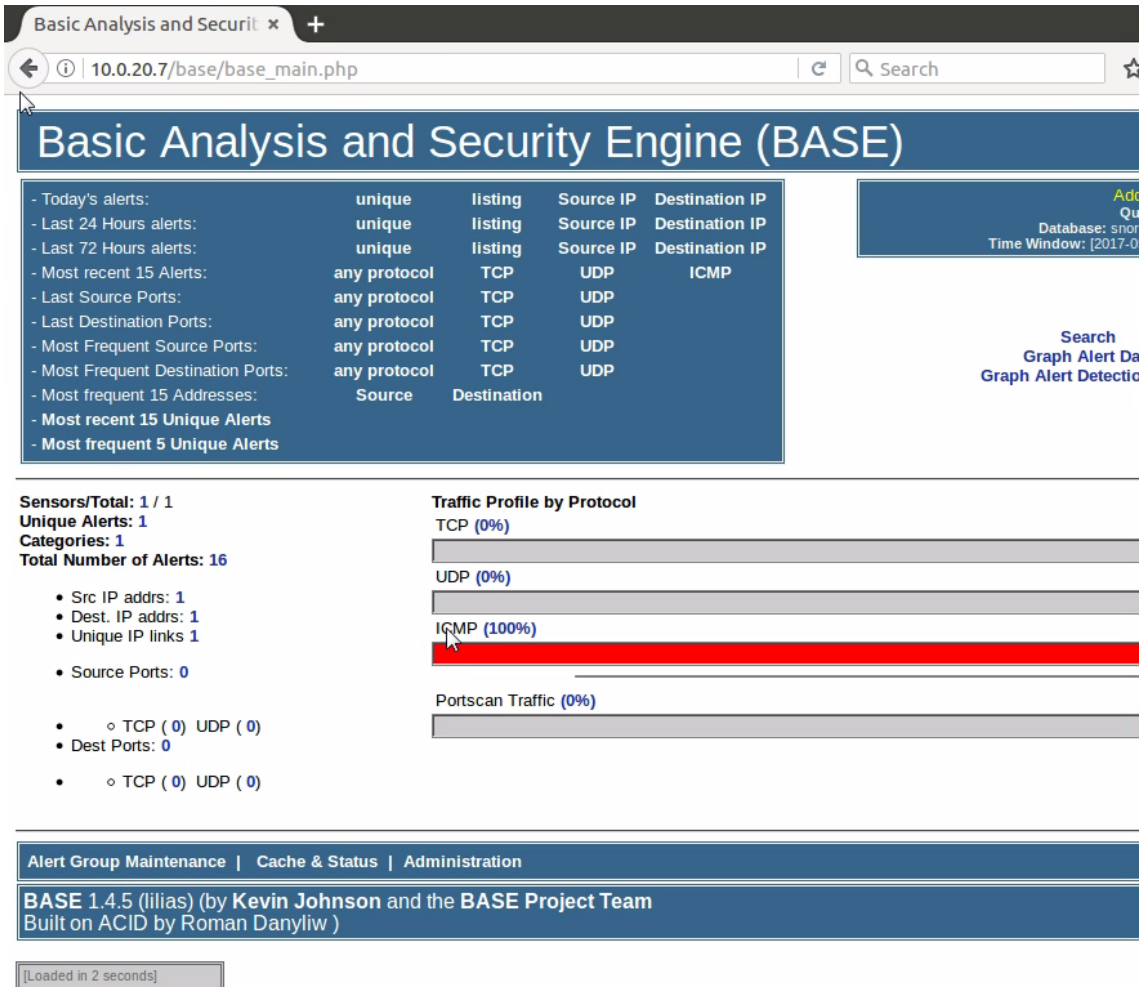
Poslednou nadstavbou detekčného programu Snort je BASE fungujúci za pomoci Apache2. Jedná sa o grafické zobrazenie zachytených paketov na našom rozhraní. Informácie čerpá z databázy v ktorej je všetko uskladnené a každou detekciou aktualizované. Ako väčšina programov, potrebujeme nainštalovať potrebné pre-rekvizity potrebné pre bezproblémový chod grafického rozhrania. Následne prejdeme na samotnú inštaláciu BASE. Pre rôzne verzie Ubuntu je rozdielne uloženie koreňového súboru pre Apache2. Pre Ubuntu 14 a viac je to `/var/www/html/base`. Po inštalácii je potrebné zmeniť určité parameter v konfiguračnom súbore `base_conf.php` podľa obrázka 5.8 [4].

```
 50 $BASE_urlpath = '/base';
 80 $DBlib_path = '/var/adodb//';
102 $alert_dbname   = 'snort';
103 $alert_host     = 'localhost';
104 $alert_port     = '';
105 $alert_user     = 'snort';
106 $alert_password = 'snortpassword';
456 // $graph_font_name = "Verdana";
457 //$graph_font_name = "DejaVuSans";
458 // $graph_font_name = "Image_Graph_Font";
459 $graph_font_name = "";
```

Obr. 5.8: Editovaný konfiguračný súbor Base

Po úspešnom nainštalovaní rešartujeme službu Apache2 sa vieme pripojiť na grafického rozhrania tak, že do internetového prehliadača zadáme adresu `10.0.20.7/base/`. Zobrazí sa základná stránka, ktorú potrebujeme ešte nastaviť. Urobíme to tak, že v pravom rohu klikneme na „Create BASE“ a následne „Main page“. Ak sú zachytené data správne, všetko sa nám zobrazí v prehliadači ako na obrázku 5.9.[4]

Na obrázku už nakonfigurovaného grafického rozhrania BASE vidíme červeným zachytené pakety ICMP z našich predchádzajúcich testov. Okrem iného sú tam aj kolónky pre protokoly TCP, UDP. V ľavej hornej časti vidíme časovú prehľadnú časť, kde sú detekcie rozdelené od posledných 24 do 72 hodín, môžeme si IP adresy rozdeliť podľa zdrojovej alebo cieľovej a dokonca BASE obsahuje možnosť zoradenia si najčastejšie sa vyskytujúcich upozornení. V pravej časti si môžeme nehať vykresliť

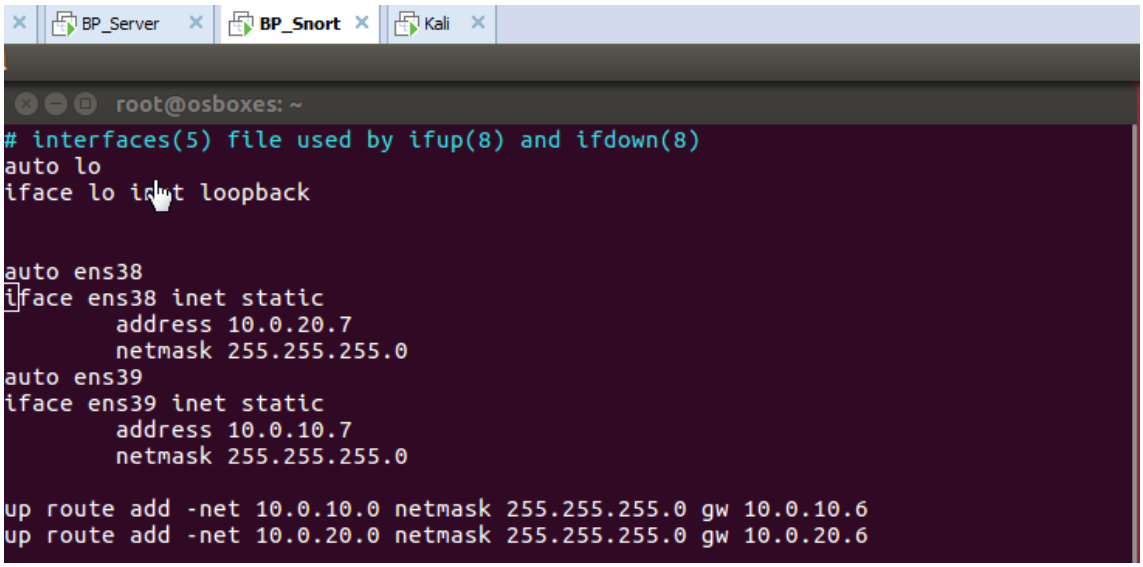


Obr. 5.9: Base grafické rozhranie

aj graf zachytených upozornení či použiť rozšírené vyhľadávanie. Parametre vyhľadávania sú podobné ako pri písaní pravidla. Môžeme vyhľadávat na základe senzoru, signatúri, clasifikácie, priority a podla času zachytenia.

5.5 Snort NIPS

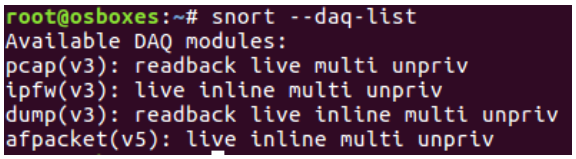
V predchádzajúcej časti sme si predstavili snort ako NIDS, teda zachytené pakety nám len zobrazí ale nič s nimi neurobí. V tejto časti sa bližšie pozrieme na Snort pracujúci v INLINE móde. V tomto móde pracuje ako most pre dva rozhrania. Každé z nich pracuje s inou podsieťou presne ako v našom prípade. Na obrazku 5.10 je zobrazená konfigurácia naších dvoch rozhraní na VM Snort.



Obr. 5.10: Konfigurácia rozhranní Snortu

Na obrázku vidíme dva rozhrania ktoré patria do dvoch rozdielných podsietí s príslušnou konfiguráciou. Na spodku sa nachádzajú dve statické cesty pre každú podsieť. Oba rozhrania musíme prepnúť do promiskuitného módu, aby sme mohli odchytávať úplne celú komunikáciu. Použijeme na to príkaz:
`ifconfig ens38 promisc up` a `ifconfig ens39 promisc up`

Samozrejme v iných prípadoch môžeme prepojiť aj viac rozhraní s viac podsieťami. Rozhrania vždy pracujú spolu napr. ens38 s ens39, ens40 s ens41 a opačne. Nikdy nie ens38 a ens41 a podobne! Pri inštalácii Snortu sme na začiatku inštalovali aj potrebné knihovne DAQ. Zadaním príkazu `snort --daq-list` ich vypíšeme.

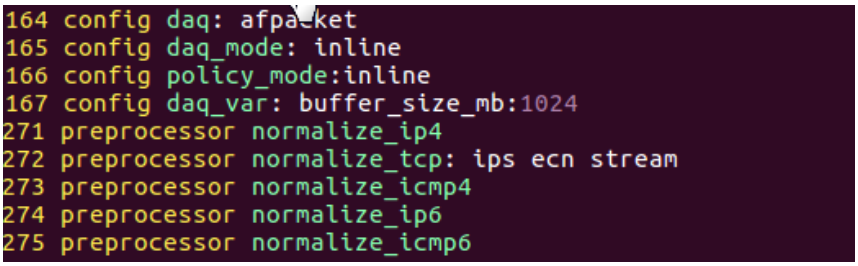


Obr. 5.11: DAQ list Snortu

Z obrázku 5.11 vidíme možnosti konfigurácie Snortu v INLINE móde. Pre našu úlohu budeme potrebovať afpacket. Okrem iného sú tam aj PCAP, IPFW a DUMP.

5.5.1 Afpacket

Opäť budeme konfigurovať základný súbor Snortu a to Snort.conf uložený v `/etc/snort/`.



Obr. 5.12: Konfigurácia inline módu

Z obrázku vidíme, že mód je nastavený na inline, daq sme zvolili afpacket a veľkosť buffera nastavíme na 1024Mb. To je základná hodnota pre ukladanie paketov v pamäti. Musíme sa uistiť, že všetky riakdky s preprocesormi na obrázku sú odkomentované.

5.5.2 Testovanie úvod

Po nastavení INLINE módu sme pripravený ho otestovať. V časti testovanie budeme písať vlastné pravidlá pre komunikáciu, budeme to testovať a pozorovať. Každé napísané pravidlo musí plniť svoju úlohu. Pre doplňujúce overenie budeme používať program Wireshark spustený na stanici server. V nasledujúcej praktickej časti budeme testovať protokoly ICMP, TCP a UDP.

Pre spustenie Snortu budeme používať príkaz:

```
sudo /usr/local/bin/snort -A console -Q -c /etc/snort/snort.conf
-i ens38:ens39
```

Kedže sa v príkaze nachádza prepínač `-A console`, budú sa nám všetky detekcie ukladať na disk vo formáte do zložky `/var/log/snort.log.xxx`. Použili sme aj prepínač `-Q` čo hovorí Snortu aby sa spustil v INLINE režime. V príkaze vidíme spomenuté aj naše dva rozhrania, ktoré tvoria most medzi dvomi rozhraniami od serveru po útočníka. V prípade že chceme sledovať komunikáciu medzi 4 rozhraniami, príkaz bude nasledovný:

```
\sudo snort -T -c /etc/snort/snort.conf -Q -i ens38:ens39::ens40
 :ens41
```

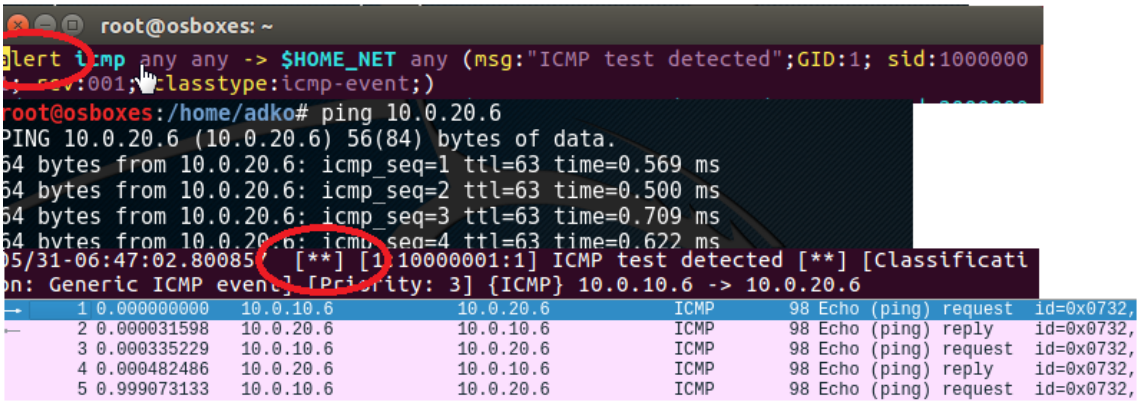
V prípade, že nechceme ukladať záznami na disk, použijeme prepínač `-N`:

```
sudo /usr/local/bin/snort -A console -Q -c /etc/snort/snort.conf -i
ens38:ens39 -N
```

Pravidlá budeme vždy editovať v súbore `/etc/snort/rules/local.rules`. Na strane servera stiahneme Wireshark príkazom `apt-get install Wireshark`. Spustíme ho v príkazovom riadku ako `sudo Wireshark` a zvolíme rozhranie ens38. Na strane útočníka budeme používať len príkazový riadok a nástroj Hping3. Jeho stručný maunál je popísaný vyššie v časti Hping3 manuál 5.3.

5.5.3 Testovanie ICMP

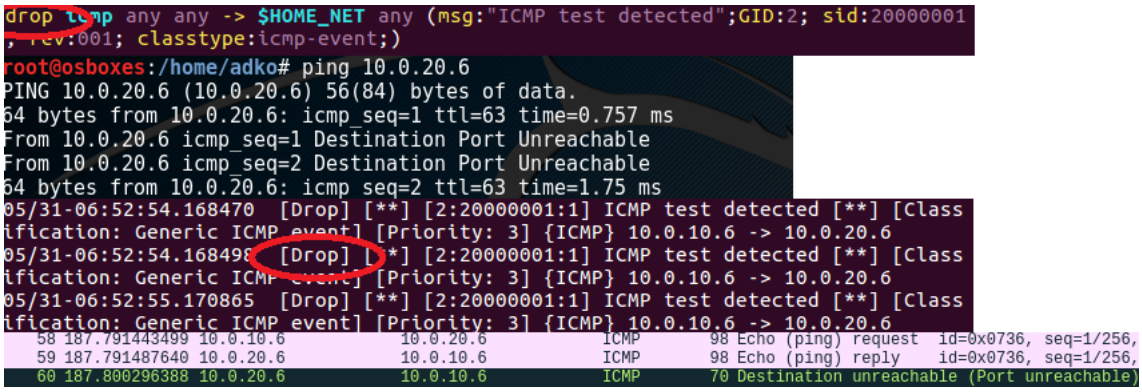
Obrázok 5.13 je rozdelený na 4 časti testovania. Prvá časť sa venuje písaniu pravidla v Snorte, kde chceme aby nám vyskočilo upozornenie keď Snort zachytí ICMP paket. Druhá časť je ping Serveru z útočníka, tretia je samotné zobrazenie alertu a posledný obrázok je detekcia Wiresharku. Vidíme že všetky ICMP pakety sa dostali tam kam mali a útočník dostal odpoveď o dostupnosti serveru. Hviezdičky označujú, že žiadna akcia nebola vykonaná len prebieha upozornenie. V Ďalších prípadoch bude nahradená inou akciou.



Obr. 5.13: Zobrazenie upozornenia pre pakety ICMP paket

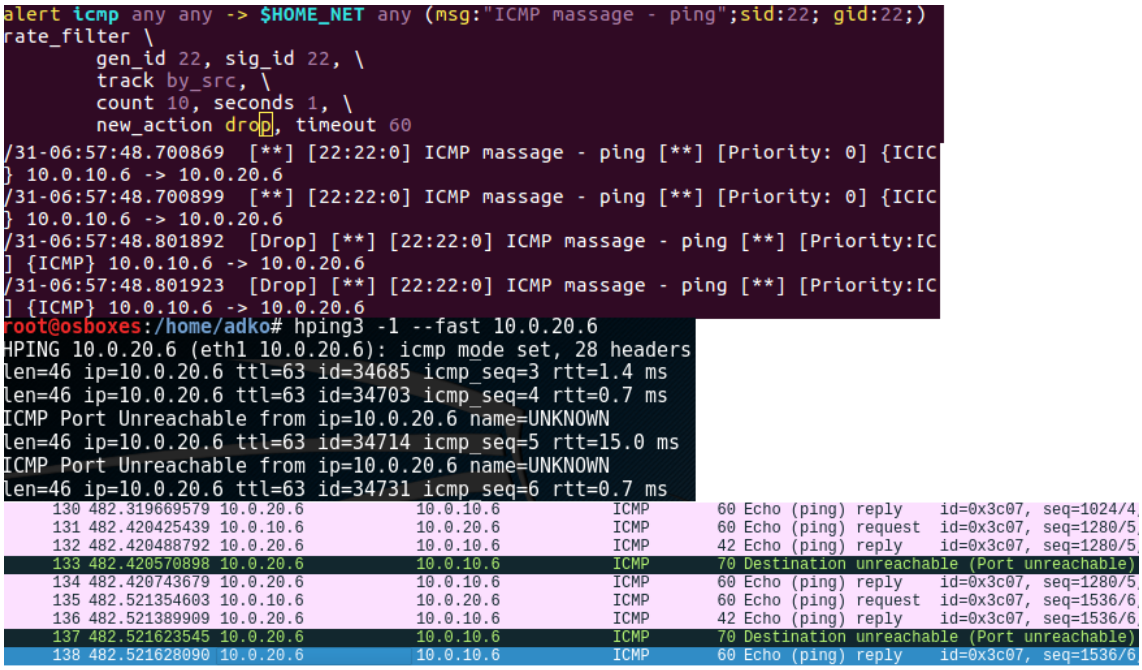
Ďalší obrázok 5.14 sa bude venovať rovnakému typu protokolu ale budeme všetky pakety zahadzovať. Opäť bude obrázok rozdelený na 4 časti kde bude zobrazený postup. Zmeníme typ akcie z alert na drop a použijeme ping na Server z útočníka. Už v príkazovom riadku vidíme, že stanica je nedostupná. V snorte nám vybehnú upozornenia nato, že nám dropuje ICMP pakety a nakoneic to potvrdíme vo Wiresharku na Serveri.

Na poslednom obrázku 5.14 bvidíme použitie filtra a aplikovanie na pravidlo sid 22. Tento filter sa vzťahuje konkrétne len k tomuto jednému pravidlu. Snort normálne prepúštal ICMP pakety až pokiaľ nedetegoval viac ako 10 paketov za 1 sekundu. Potom pristúpil na novú akciu, ktorá mu prikazovala zahodiť všetky pakety



Obr. 5.14: Zobrazenie zahadzovania paketov pre ICMP paket

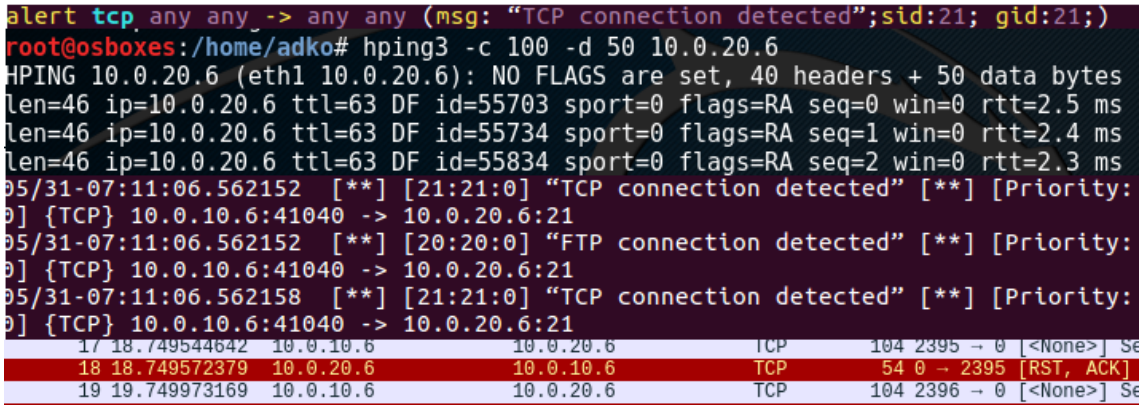
a počkať 60 sekúnd pre opätovné povolenie ICMP paketov. V OS Kali sme použili rýchlejšiu metódu posielania paketov z dôvodu ukážky filtra nakonfigurovanom v pravidlách Snortu. Pre overenie sme späť použili Wireshark na stanici Server.



Obr. 5.15: Zobrazenie práce rate filtra

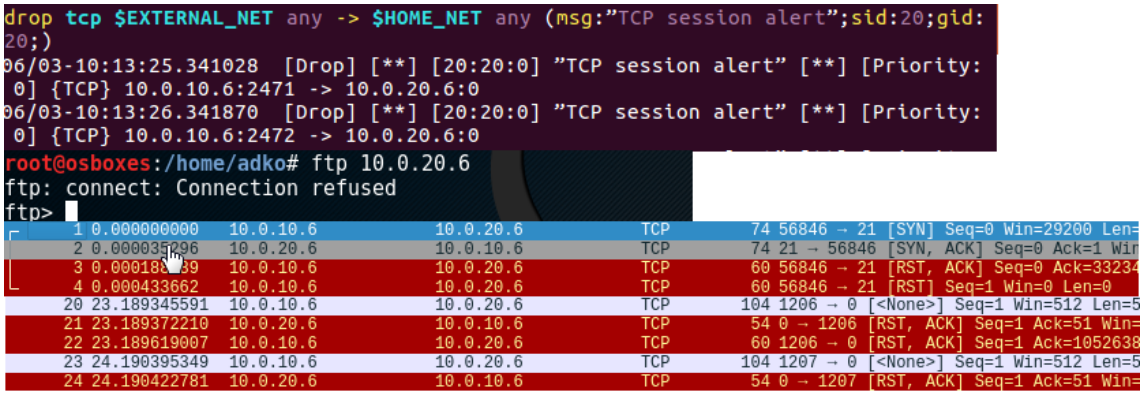
5.5.4 Testovanie TCP

V ďalšom testovaní sa pozrieme ako pracuje TCP protokol. Pre tento typ protokolu budeme musieť pre testovanie trošku upraviť príkaz v OS Kali. Spustíme ho len s definovaním počtu paketov a ich dĺžky. Zmes obrázkov sa opätovne skladá zo 4 častí kde definujeme pravidlo, otestujeme ho a zachytíme. Viac obrázok 5.16.



Obr. 5.16: Testovanie TCP spojenia

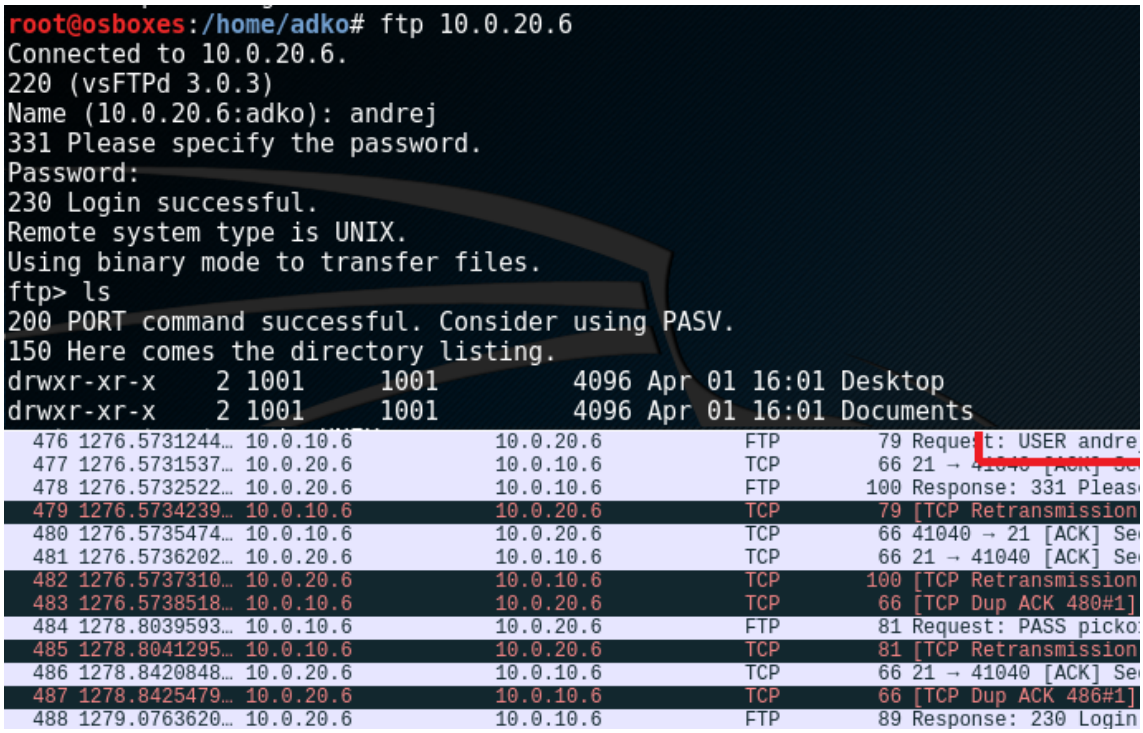
Tak ako pri ICMP aj tu sa pokúsime zahodiť všetky TCP spojenia. Pre ukážku po zadefinovaní pravidla sa skúsime pripojiť na FTP server z Kali linux. Uvidíme ako bude spojenie prebiehať. Editujeme pravidlo a zmeníme akciu na drop. Na obrázku vidíme ako Snort zahadzuje všetky spojenia aj s naším pokusom o nadviazanie FTP. Je to z toho dôvodu, že pravidlo je určené pre všetky porty aj vrátane portu 21. FTP nám oznámi že pripojenie je odmietnuté. Viac obrázok 5.17.



Obr. 5.17: Testovanie zahadzovania TCP spojenia

5.6 DoS útok

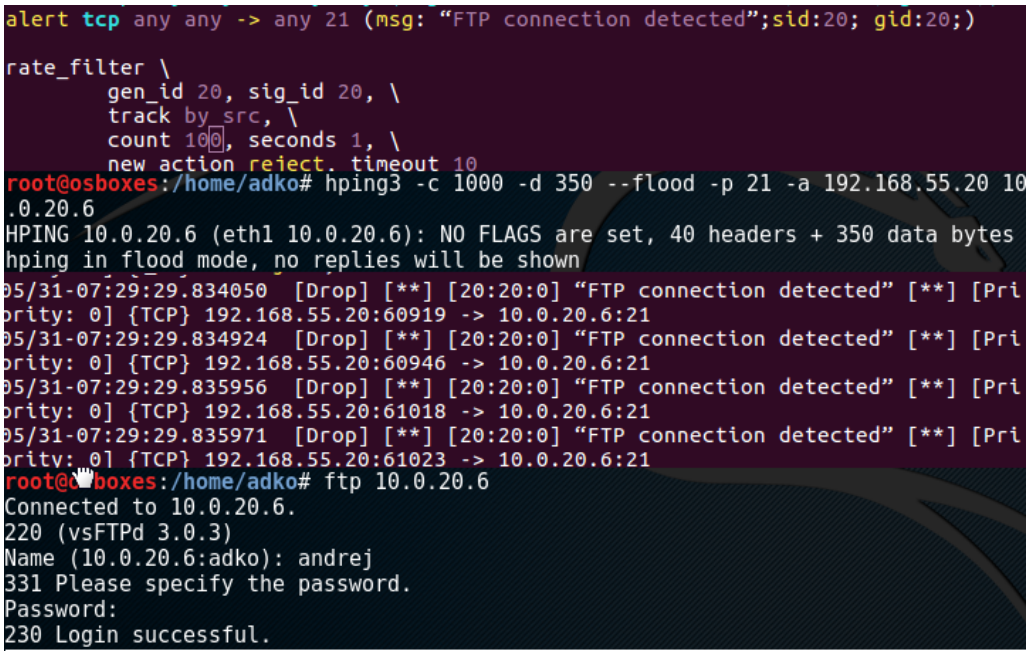
Ako posledný bod tejto práce budeme simulovať DoS útok na FTP server. Pozrieme sa ako prebieha prihlasovanie na FTP pred definovaním pravidiel a ako po. Po pripojení na FTP zadáme meno užívateľa a jeho heslo. Po tomto autentizačnom procese sme schopní pohybovať na na jeho serveri. Viď obrázok 5.18. Pod pripojením je aj záznam z Wiresharku kde je vyznečené prihlaovacie meno a heslom. Posledný riadok potvrdzuje úspešné pripojenie.



Obr. 5.18: Pripojenie na FTP pred útokom

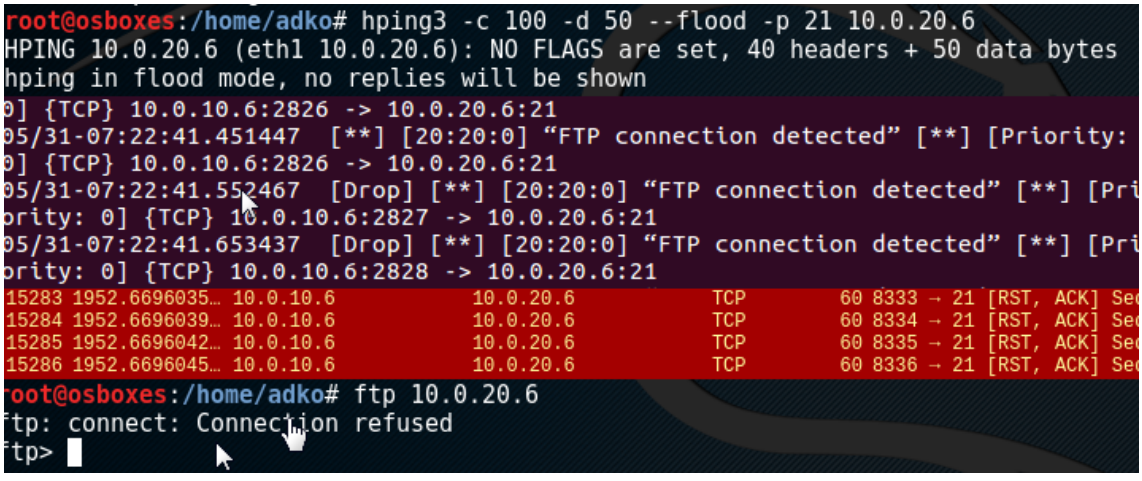
Definujeme si pravidlo, ktoré bude zamerané na FTP port 21. Podľa obrázka 5.19 nastavíme aj rate filter, ktorý sme testovali v predošlých prípadoch. Nastavíme ho na maximálne 100 relácií za 1 sekundu z unikátnej IP adresy. Potom nasleduje pauza 10 sekúnd. Ak sa prevádzka ukľudní tak po 10 sekunách sa pôvodná IP adresa môže opäť pokúsiť o vytvorenie relácie.

Vo výpise Snortu nevidíme riadok pre úspešné pripojenie pretože obsahoval mnoho riadkov, v ktorých to nieje možné nájsť. Pre útok sme použili ukradnutú IP adresu 192.168.55.20 aby sme videli, že Snort nezahodí celú komunikáciu na porte ale len opakované pokusy o pripojenie z jednej IP adresy. Z adresy 10.0.10.6 teda pôvodnej, bolo možné sa pripojiť na Server. Trvalo to o niekoľko sekúnd dlhšie pretože sme zahltili skoro celú linku TCP spojeniami. Nakoniec po zadaní mena a hesla sme sa úspešne dostali na server.



Obr. 5.19: Útok na FTP server a spustenie pravidla s ukranutou IP adresou

Ďalej skúsime použiť pre útok pôvodnú IP adresu ktorá bola nakonfigurovaná na začiatku. Uvidíme, ako sa nám podarí pripojiť na server pri tomto pokuse. Obrázok 5.20 opisuje postup pripojnia.



Obr. 5.20: Útok na FTP server a spustenie pravidla s pôvodnou IP adresou

Pravidlo ostáva rovnaké ako v predošlom prípade. Na obrázku 5.20 vidíme, že Snort zahaduje všetky TCP spojenia od prekročenia povoleného limitu spojení z jednej IP adresy a teda aj náš pokus o pripojenie na FTP server. Hláška odoprenia pripojenia je rovnaká ako v predošlom prípade kde sme testovali TCP spojenie pri celkovom zahadzovaní TCP.

6 ZÁVĚR

Cieľom bakalárskej práce bolo vypracovať laboratórnu úlohu pre študentov a oboznámiť ich s problematikou NIDS a NIPS. V práci boli vytvorené vlastné virtuálne prostredia z troch virtuálnych strojov kde útočník vytvoril DoS útok na stanicu Server. Celá komunikácia prechádzala cez stanicu Snort, ktorá sa po správnej konfigurácii vedela rozhodnúť či sa jedná o priamy DoS útok. Snort dokázal detekovovať útok z nástroja Hping3 na FTP server a zmierniť jeho dopad na takú mieru, že server FTP bol ďalej prístupný.

V práci boli nakonfigurované dva rôzne režimy. NIDS pre detekciu prichádzajúcich paketov bez možnosti novej akcie a NIPS pre možnosť zahodiť paket ak sa jedná o škodlivý tok dát. Komunikácia sa zachytávala podľa predom určených pravidiel, ktoré sme si sami nakonfigurovali. Pravidlá boli písané pre protokoly ICMP, TCP a UDP. Pre čítanie binárnych súborov, ktoré Snort používal ako výstup sme konfigurovali aj nadstavbu Barnyard2. Pre lepší prehľad toku dát sme všetko zobrazili v grafickej nadstavbe softvéru Base, kde sme mohli vidieť detaily o zachytených paktoch ako zdrojová alebo cieľová IP. Taktiež bola k dispozícii aj časová os k danému útoku.

Posledným bodom práce bolo vypracovanie laboratórnej úlohy pre študentov, ktorá ich oboznámi s problematikou NIDS/NIPS. Zoznamuje študentov s topológiu virtuálnej siete aj s jej konfiguráciou. Obsahuje aj nastavenie oboch režimov spolu so samostatnou úlohou pre vypracovanie vlastných pravidiel podľa zadania. Pre rýchlejšiu orientáciu a prácu so súbormi je priložený aj stručný návod pre editor VIM a základný manuál pre nástroj Hping3.

LITERATÚRA

[1] CID, D. *Layer 7 DDOS – Blocking HTTP Flood Attacks* [online] poslední aktualizace 2.6.2014 [cit. 13. 11. 2016]. Dostupné z URL: <http://www.lupa.cz/clanky/denial-of-service-utoky-reflektivni-a-zesilujici-typy/>.

[2] ČMELÍK, M. *Seznamte se – DoS a DDoS útoky* [online] poslední aktualizace 3.7.2013-12:08 [cit. 13. 11. 2016]. Dostupné z URL: <https://www.security-portal.cz/clanky/seznamte-se-%E2%80%93-dos-ddos-%C3%BAtoky>.

[3] DIETRICH, Noah *Snort 2.9.8.x on Ubuntu.* poslední aktualizace 2015 [cit. 4. 12. 2016]. Dostupné z URL: <http://sublimerobots.com/2015/12/>.

[4] DIETRICH, Noah *Snort 2.9.9.x on Ubuntu.* poslední aktualizace 2017 [cit. 3. 6. 2017]. Dostupné z URL: <http://sublimerobots.com/2017/01/>.

[5] EINWECHTER, N. *An Introduction To Distributed IDS* [online] poslední aktualizace 7.1.2002 [cit. 27. 11. 2016]. Dostupné z URL: <https://www.symantec.com/connect/articles/introduction-distributed-intrusion-detection-syst>.

[6] ENDORF, Carl F., Eugene SCHULTZ a Jim MELLANDER *Detekce a prevence počítačového útoku,*[online] Praha: Grada, 2005. ISBN 80-247-1035-8 [cit. 27. 11. 2016]. Dostupné z URL: <https://books.google.cz/books?id=AWYduASeDlEC&lpg=PP1&pg=PP1&hl=sk#v=onepage&q&f=false>.

[7] KRUEGEL, Christopher, Richard. LIPPMANN a Andrew CLARK. *Recent advances in intrusion detection* 10th international symposium, RAID 2007, Gold Coast [i.e. Coast], Australia, September 5-7, 2007 : proceedings. New York: Springer-Verlag, c2007. 347s. ISBN 9783540743194.

[8] KYSILKA, P. *Editor VIM prakticky* poslední aktualizace 28.4.2003 [online] [cit. 6. 6. 2017]. Dostupné z URL: <http://www.abclinuxu.cz/clanky/navody/editor-vim-prakticky-i>.

[9] REHMAN, R. *Intrusion detection systems with Snort: advanced IDS techniques using Snort, Apache, MySQL, PHP, and ACID. Upper Saddle River* Prentice Hall PTR, c2003. 275s. ISBN 0131407333 [cit. 12. 11. 2016].

[10] ROESCH, M; GREEN. CH *SNORT Users Manual 2.9.9* [online] [cit. 4. 6. 2016]. Dostupné z URL: <http://manual-snort-org.s3-website-us-east-1.amazonaws.com/>.

[11] SANFILIPPO, S. *Hping3-Linux man page* [cit. 12. 11. 2016].

[12] SCAMBRAY, Joel, George KURTZ a Stuart MCCLURE *Hacking bez tajemství.* 2002, 2. aktualiz. vyd. Praha: Computer Press. Komunikace a sítě. 627s. ISBN 80-7226-644-6. [cit. 10. 11. 2015].

[13] SCARFONE, Karen; MELL, Peter *Guide to Intrusion Detection and Prevention Systems (IDPS)* Computer Security Resource Center February 2007. 127s. National Institute of Standards and Technology.

[14] *Snort FAQ* [online] [cit. 4. 6. 2016]. Dostupné z URL: <`https://www.snort.org/faq`>.

[15] ZUČÁK, M. *Sieťové riziká a útoky na lokálnej sieti* [online] poslední aktualizace 27.3.2010-20:42 [cit. 13. 11. 2016]. Dostupné z URL: <`http://www.secit.sk/sk/content/sietove-rizika-utoky-na-lokalnej-sieti`>.

ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

ARP	Address Resoluiton Protocol
DAQ	Data Acquisition library
DG	Default Gateway
DNS	Domain Name System
FTP	File Transfer Protocol
GUI	Graphical User Interface
HIDS	Host-Intrusion Detection System
HTTP	Hypertext transfer protocol
HW	Hardware
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IP	Internet Protocol
IPS	Intrusion Prevention System
LAN	Local Area Network
NBA	Network Behavior Analysis
NIDS	Network-Intrusion Detection System
PC	Personal Computer
SQL	Structured Query Language
TCP	Transmission Control Protocol
TLS	Transport Layer security
TTL	Time To Live
VRT	Vulnerability Research Team

ZOZNAM PRÍLOH

A LABORATORNÍ ÚLOHA

A.1 Teoretický úvod:

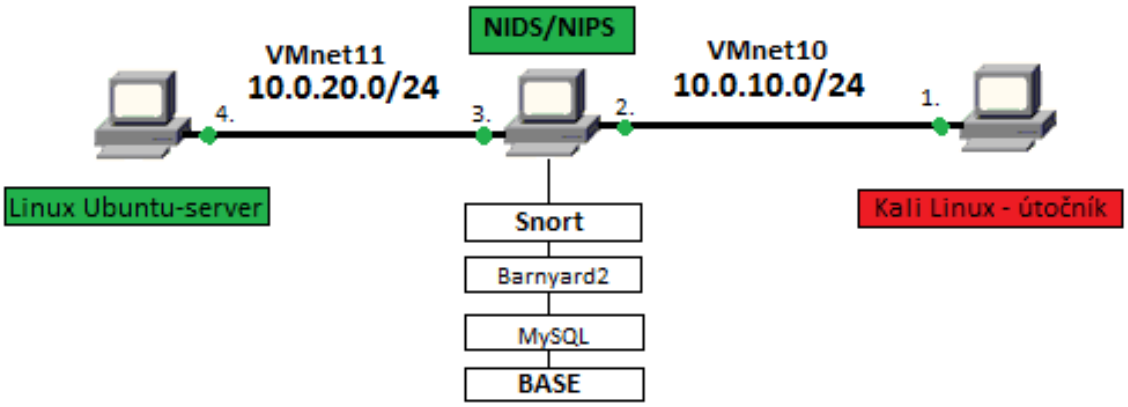
V této laboratorní úloze se zaměříme na konfiguraci Snortu a jeho nástaveb. Laboratorní cvičení obsahuje tři virtuální stroje pro WMware verze 12.

1. Server – Linux Ubuntu
Osahuje nainstalovaný FTP server, Apache2 pro tvorbu http serveru. Pro zachytávání komunikace slouží program Wireshark.

2. Útočník - Kali Linux
Neobsahuje žádné doplňkové služby. Na této stanici využíváme pouze příkazový řádek pro nástroj Hping3 pro tvorbu DoS útoku.

3. Snort – Linux Ubuntu
Obsahuje základní instalaci Snortu a jeho nástavby.

Virtuální prostředí jsou zapojeny následovně:



Obr. A.1: Topológia zapojenia našej virtuálnej siete s NIDS/NIPS systémom

Prostřední stanice obsahuje Snort, Barnyard2, databázi MySQL a grafické rozhraní BASE. Krajní stanice se nacházejí ve dvou různých podsítích, tedy Snort pracuje jako most mezi dvěma podsítěmi. Pro detailnější náhled si konfiguraci můžeme ověřit v souboru `/etc/network/interfaces`.

Snort – program pro detekci paketů pomocí senzoru na vybraném rozhraní.

Barnyard2 – program pro čtení binárních souborů do kterých se ukládají data.

MySQL – spolupracuje s programem Barnyard2 tak, že veškerá zachycená komunikace se ukládá do databáze pro pozdější náhled.

BASE – poslední nástavba programu Snort, která umožňuje vidět veškerou zachycenou komunikaci Snortu v grafickém prostředí. Spolupracuje s databází MySQL.

Pro lepší orientaci v obsáhlých konfiguračních souborech je přiložena tabulka A.1 s nápovědou k práci s editorem VIM.

Tab. A.1: Tabuľka používaných príkazov vo VIM

	Príkaz	**Popis**
1.	i	insert mód, můžeme zapisovat do souboru
2.	ESC	opuštění předchozího režimu
3.	:wq	uložení a ukočenie, může být použito i odděleně
4.	:#	# přesun na číslo řádku
5.	:set number	označení řádků
6.	gg	nastavení kurzoru na začátek souboru
7.	G	nastavení kurzoru na konec souboru
8.	/#	#vyhledávání
9.	%d	vvymaže všechny znaky v souboru
10.	q!	vynucené opuštění souboru

Při tvoření útoku budeme používat nástroj Hping3, který budeme spouštět z virtuální stanice Kali linux jako útočník. Manuál tohoto nástroje je poměrně dlouhý, proto tento laboratorní úkol obsahuje i základní možnosti příkazů v tabulce A.2, které budeme potřebovat.

Tab. A.2: HPING3 stručný manuál

	Príkaz	**Popis**
1.	- -fast	rychlé posílání paketů (10/s)
2.	- -flood	posílání paketů nejrychleji jak můžeme
3.	-0	posílání RAW IP
4.	-1	posílání ICMP paketů
5.	-2	UDP mód
6.	-a	ukradená IP adresa
7.	-s	zdrojový port
8.	-p	cílový port
9.	-S	synchronizační značení
10.	-c	počet paketů
11.	-d	velikost paketů

A.1.1 Psaní vlastních pravidel

Pravidlo Snortu se skládá ze 6 základních částí zobrazených na obrázku A.2. Všechny jsou potřebné pro bezchybné spuštění Snortu.

1. Akce – Akce vyjadřuje co bude uděláno s detekovaným paketem. Možnosti jsou:
   (a) Alert: Upozorní nás na detekovaný paket
   (b) Log: zaznamená paket
   (c) Pass: ignoruje detekovaný paket
   (d) Drop: zahodí paket
   (e) Reject: blokuje paket a zaznamená ho ve formě logu.

2. Protokol – definuje jeden ze 4 protokolov: ICMP, IP , TCP, UDP

3. IP adresa a port – Definujeme zdrojovou IP adresu a port z které komunikace přichází.

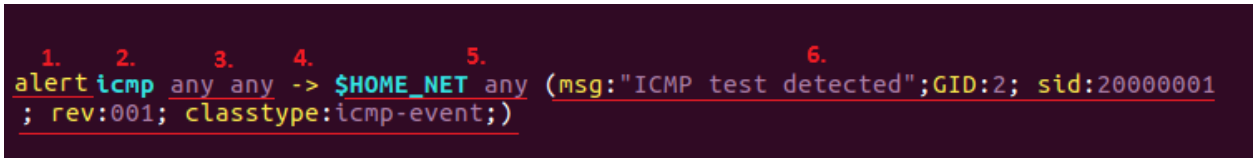
4. Směr komunikace

Obsahuje znaky komunikace jako:
   (a) -> přicházející komunikace do naše sítě
   (b) <- odcházející komunikace do sítě
   (c) <> oboustranná komunikace

5. Cílová adresa a port – Definuje naši cílovou IP adresu a port.

6. Doplňující možnosti –
   (a) Zpráva pro odpovídající pravidlo: msg:“ “
   (b) SID - jednoznačný identifikátor pravidla. Může být jen jeden
   (c) GID - generátor ID

(d) REV - revizní číslo
(e) Class-type – typ klasifikace komunikace, 1 je nejvíce 4 nejméně.



Obr. A.2: Základné pravidlo rozdelené na časti.

Pozn. Každá část je oddělená **středníkem** a doplňující možnosti jsou v **závorkách!**
Další kroky konfigurace nás seznámí jak Snort pracuje, jak fungují nástavby a grafické zobrazení zachycené komunikace. Postupně projdeme všemi částmi konfigurace NIDS (Network Intrusion Detection System) po NIPS (Network Intrusion Prevention System). Umístění Snortu při obou částech bude na stejném místě v síti na rozdíl od HIDS (Host-based Detection system) kde se Snort umisťuje na koncového uživatele. Snort v NIPS módu může kontrolovat i bezdrátovou komunikaci. Není však možné aby Snort pracoval v celém frekvenčním pásmu ve stejnou dobu.

NIDS (Network Intrusion Detection System) Je sada technik a metod které zachycují komunikaci na určeném uzlu s následným upozorněním např. ve formě alarmu. Neprovádí žádnou akci zahození.
NIPS (Network Intrusion Prevention System) Vylepšený systém IDS, který kromě oznámení o zachycení komunikace i vyhodnotí zda se jedná o útok nebo běžnou komunikaci. Na základě toho rozhodne zda komunikaci z IP adresy propustí nebo ne.

A.2 Postup práce:

A.2.1 Konfigurace jako NIDS:

1. Otestujeme, zda je Snort nainstalovaný správně. Použijeme na to příkaz sudo `/usr/bin/snort -V`. V příkazovém řádku nám vypíše verzi Snortu.

2. Upravíme konfigurační soubor s umístěním: `/etc/snort/snort.conf`. Na 45. řádku definujeme naši IP adresu a prefix sítě pro Snort aby věděl, v jakém rozsahu se budeme pohybovat. Na řádcích XY jsou nastaveny cesty k dalším souborům, bez kterých nelze Snort spustit. Pro informaci si je můžeme prohlédnout.

3. Od řádku 545 jsou umístěny **include**, které zahrnují soubory s pravidly. Všechny jsou okomentované pomocí #. Odkomentujte jen první řádek local.rules. Do tohoto souboru budeme psát vlastní pravidla a testovat je.

4. Začneme s jednoduchým testem ICMP paketů. Editujeme zatím prázdný soubor `/etc/snort/rules/local.rules`, kde vytvoříme jednoduché pravidlo:

```
alert icmp any any -> $HOME_NET any (msg:"ICMP test detected
"; GID:1; sid: 1; rev: 1; classtype:icmp-event;)
```

Na základě teoretického úvodu víme, z čeho se má pravidlo skládat. Do zprávy pro pravidlo si můžeme vložit libovolný text.

5. Snort spustíme příkazem :

```
/usr/local/bin/snort -A console -q -u snort -g snort -c /etc/snort/
snort.conf -i ens39
```

```
-A console vypíše do konzole zachycenou komunikaci
-q tichý mód
-c cesta ke konfiguračnímu souboru
-i rozhraní na kterém se Snort spouští
```

Pingneme server ze stanice útočníka a sledujeme odezvu. Všechna komunikace se defaultně ukladá do `/var/log/snort/snort.xxx`. Pro ukládání Snortu do binární formy musíme opět modifikovat konfigurační soubor Snortu. Na řádek 520 přidáme: `output unified2: filename snort.u2, limit 128`.

Barnyard2 tyto záznamy sesbírá a uloží do databáze MySQL. Při instalaci databáze jsme zvolili root heslo rootpass. Aby program věděl, jakým způsobem přistupovat, musíme v jeho konfiguračním souboru přidat řádek, který mu říká, kde ho má uložit, pod jakým uživatelem, heslem a do jaké databáze.

```
output database: log, mysql, user=snort password=snortpassword
dbname=snort host=localhost
```

Pro spuštění Barnyardu2 použijeme příkaz:

```
sudo /usr/local/bin/barnyard2 -c /etc/snort/barnyard2.conf -d
/var/log/snort -f snort.u2 -q -w /var/log/snort/barnyard2.waldo
 -g snort -u snort -a /var/log/snort/archived_logs
```

Spuštění tohoto příkazu trvá velmi dlouho cca 20 minut. Nakonec chceme všechna data zobrazit v grafickém provedení. Na to nám bude sloužit BASE. Pro správnu funkčnost i zde musíme modifikovat konfigurační soubor `/var/www/html /base/base.conf` jako v předešlých případech. Výsledná konfigurace bude vypadat následovně ako na obrázku A.3:

```
 50 $BASE_urlpath = '/base';
 80 $DBlib_path = '/var/adodb//';
102 $alert_dbname   = 'snort';
103 $alert_host     = 'localhost';
104 $alert_port     = '';
105 $alert_user     = 'snort';
106 $alert_password = 'snortpassword';
456 // $graph_font_name = "Verdana";
457 //$graph_font_name = "DejaVuSans";
458 // $graph_font_name = "Image_Graph_Font";
459 $graph_font_name = "";
```

Obr. A.3: Base grafické rozhranie

Následne restartujem Apache2: `sudo service Apache2 restart`
Spustíme webový prohlížeč a do vyhledávače zapíšeme adresu 10.0.20.7/base. Naběhne nám defaultní stránka, kde musíme kliknout na "create BASEä Main page. Nakonec se nám zobrazí grafické rozhraní, kde vidíme zachycenou komunikaci, kterou nám Barnyard2 uložil do MySQL a odkud si ji čte BASE.

6. Obeznáměte se s grafickým prostředním programu Base.

A.2.2 Konfigurace jako NIPS

1. Vypíšeme si dostupné metody zachytávání paketů: snort –daq-list budeme pracovat s afpacket.

2. Zapneme rozhraní do promisc módu, aby zachytával všechnu komunikaci ifconfig ens3X promisc up

3. Editujeme Snort.conf následovně ako na obrázku A.4
4. Snort spustíme příkazem:

```
sudo /usr/local/bin/snort -A console -Q -c /etc/snort/snort.conf
-i ens38:ens39
```

`-Q - inline mód`

při nastavení -A console bude Snort ukládat všechna data do `snort.log.xxx`

```
164 config daq: afpacket
165 config daq_mode: inline
166 config policy_mode:inline
167 config daq_var: buffer_size_mb:1024
271 preprocessor normalize_ip4
272 preprocessor normalize_tcp: ips ecn stream
273 preprocessor normalize_icmp4
274 preprocessor normalize_ip6
275 preprocessor normalize_icmp6
```

Obr. A.4: Konfigurácia inline módu

A.3 Vlastní pravidla

1. Vytvoříme pravidla pro zachytávání TCP, UDP a ICMP protokolů a otestujeme je pomocí Kali Linuxu. Pod pravidlem je umístěný příkaz pre Hping3 na ověření funkčnosti:

Upozorní na ICMP protokol z vnější sítě do domácí na kterýkoliv port:

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP massage -
ping"; classtype:icmp-event;sid:1;gid:1;)
test - hping3 -1 10.0.20.6
```

Upozorní na ICMP protokol z vnější sítě do domácí na kterýkoliv

```
port s TTL 63:
alert icmp any any -> $HOME_NET any (msg: "ICMP with TTL=63"; ttl:
10;sid:5;gid:5;)
test Kali - ping 10.0.20.6
```

Upozorní na IP protokol z vnější sítě do domácí na kterýkoliv port:

```
alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"RAW IP packet";
sid:6;gid:6;)
test Kali - hping3 -c 10 -d 5 -0 --flood 10.0.20.6
```

Upozorní na UDP protokol z vnější sítě do domácí na kterýkoliv port:

```
alert udp $EXTERNAL_NET any -> $HOME_NET any (msg:"UDP flood";sid:
7;gid:7;)
test Kali - hping3 -2 --fast 10.0.20.6
```

Upozorní na TCP protokol z vnější sítě do domácí na port 21:

```
alert tcp any any -> $HOME_NET 21 (msg:"FTP connection";sid:7;
gid:7;)
test Kali - hping3 -c 100 -d 50 -p 21 --flood 10.0.20.6
```

Upozorní na TCP protokol z vnější sítě do domácí na kterýkoliv port v rozmezí 1 až 1023:

```
alert tcp any any -> $HOME_NET 1:1023 (msg:"known port flood";sid
:10;gid:10;)
test Kali - hping3 -c 100 -d 50 -p 1000 --flood 10.0.20.6
```

Upozorní na TCP protokol z vnější sítě do domácí na kterýkoliv port s paketem větší než 300 bytů:

```
alert tcp any any -> $HOME_NET any (dsize: > 300; msg: "Large size
IP packet detected";sid:15;gid:15;)
test Kali - hping3 -c 100 -d 305 --flood 10.0.20.6
```

Upozorní na TCP protokol z vnější sítě do domácí na kterýkoliv port s označením P jako PUSH:

```
alert tcp any any -> $HOME_NET any (flags: P; msg: "PUSH flag
detected";sid:17;gid:17;)
test Kali - hping3 -c 100 -d 50 -P --flood 10.0.20.6
```

Upozorní na UDP protokol z vnější sítě do domácí na kterýkoliv port:

```
alert udp $EXTERNAL_NET any -> $HOME_NET any (msg:"UDP flood";sid:7;
gid:7;)
test Kali - hping3 -2 --fast 10.0.20.6
```

2. Použijeme filtry, které spojíme s pravidly na základě SID. K jednomu filtru můžeme připojit více pravidel.

pravidlo a filtr s povolením nejvíc jedno TCP spojení za 5:

```
alert tcp any any -> $HOME_NET 21 (msg:"TCP session FTP connection";
sid:21; gid:21;)
event_filter \
        gen_id 21, sig_id 21, \
        type limit, track by_src,  \
        count 1, seconds 5
test - hping3 -c 1000 -d 500 -p 20 --fast -a 192.168.5.5 10.0.20.6
```

nejvíce 10 TCP připojení z jedné IP na port 21 za 1 sekundu x zahození, pauza 10 sekund:

```
alert tcp any any -> $HOME_NET 21 (msg: "FTP spojenie zaznamenané";
sid:20; gid:20;)
rate_filter \
        gen_id 20, sig_id 20, \
        track by_src, \
        count 10, seconds 1, \
        new_action drop, timeout 10
test - hping3 -c 1000 -d 500 -p 21 --flood -a 192.168.5.5 10.0.20.6
```

Při posledním pravidle zkusíme otestovat připojení na FTP server. V příkazu pro útočníka je nastavena ukradená IP adresa, takže během útoku, jak budou zahazované TCP spojení z adresy 192.168.5.5. Zkusíme se připojit z původní adresy 10.0.10.6. Snort by měl zablokovat jen ukradenou IP adresu a zbylé propustit. Nakolik je linka dost vytížená připojení na FTP 10.0.20.6 může trvat trošku déle.

A.4 Samostatná práca

V samostatné práci budeme modifikovat pravidla Snortu. Předešlé pravidla zakomentujeme. Vytvoříme pravidlo a otestujeme:

a) Zahodí všechny TCP spojení z adresy 192.168.96.15
b) Propustí protokol TCP na port 80 se zprávou: "komunikace na http portu"
c) Vypíše zachycený paket s velikostí nad 400 bytů
d) Vypíše upozornění na TCP spojení se značkou S jako synchronizace na portu větším než 50
e) Na internetu vyhledejte možnost zápisu pravidla které nás upozorní, že uživatel vyhledává slovíčko Snort na internetu. (nápověda: Content)
f) Vytvořte pravidlo a rate filtr, který bude zachycovat ICMP pakety podle zdrojové IP, maximálně 5 spojení za 1 sekundu, po překročení zahodí a nastaví timeout 60 s.

ODPOVĚDI:

```
A,
drop tcp 192.168.96.15 any -> $HOME_NET any (msg:"TCP session
dropped";sid:1;gid:1;)
hping3 -c 100 -d 50 --flood 10.0.20.6

B,
pass tcp any any -> $HOME_NET 80 (msg:" komunikácia na http porte
";sid:2;gid:2;)
hping3 -c 100 -d 50 -p 80 --flood 10.0.20.6

C,
alert tcp any any -> $HOME_NET any (dsize: >400; msg: "Paket vačší
ako 400 bytov";sid:3;gid:3;)
hping3 -c 100 -d 405 --flood 10.0.20.6

D,
alert tcp any any -> $HOME_NET 50:  (flags: S; msg: "SYNC flag
detected";sid:4;gid:4;)
hping3 -c 100 -d 50 -S --flood 10.0.20.6

E,
alert tcp any any -> 10.0.20.6 any (content:"snort"; msg: "snort
word matched";sid:5;gid:5;)
vyhledat v Mozille na Kali Linux : 10.0.20.7/snort

F,
alert icmp any any -> $HOME_NET any (msg:"ICMP massage - ping"
;sid:6; gid:6;)
rate_filter \
       gen_id 6, sig_id 6, \
        track by_src, \
        count 5, seconds 1, \
        new_action reject, timeout 60

hping3 -c 1000 -d 500 -p 20 -1 --fast 10.0.20.6
```