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 PRŮNIKU NA HRANIČNÍCH ZAŘÍZENÍCH

INTRUSION DETECTION AND PREVENTION SYSTEMS AT BORDER DEVICES

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — Zdenko Bína

AUTHOR

VEDOUCÍ PRÁCE — Ing. Petr Blažek

SUPERVISOR

BRNO 2017

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Zdenko Bína ***ID:*** 147665
Ročník: 3 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Systémy pro detekci a prevenci průniku na hraničních zařízeních

POKYNY PRO VYPRACOVÁNÍ:

Bakalářská práce je zaměřena na detekci a prevenci DDoS útoků v síťové komunikaci. V rámci bakalářské práce prostudujte současný stav problematiky systémy NIDS (Network Intrusion Detection System) a NIPS (Network Intrusion Prevention System). Zvolte dva volně dostupné nástroje (např. SNORT a SURICATA), které zprovozněte na experimentálním pracovišti a nakonfigurujte detekci pro nejméně 5 útoků cílených na odepření síťové služby. U zprovozněných nástrojů se zaměřte na vizualizaci a zpracování záznamů událostí (logů) a porovnejte detekční a mitigační schopnosti u vybraných útoků. Výstupem bakalářské práce bude podrobné porovnání NIDS a NIPS systémů na základě zvolených útoků.

DOPORUČENÁ LITERATURA:

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

[2] CARL, Glenn, et al. Denial-of-service attack-detection techniques. IEEE Internet Computing, 2006, 10.1: 8289.

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

Vedoucí práce: Ing. Petr Blažek
Konzultant:

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

ABSTRAKT

Bakalářská práce se zabývá testováním odolnosti sítě proti DDoS útokům. V teoretické části představuje problematiku těchto útoků a jejich současné trendy, následně se zaobírá systémy IDS a IPS a popisuje zařízení Spirent Avalanche 3100b, určené ke generování síťového provozu.

V praktické části se zabývá konfigurací softwarového webového serveru Apache, běžícího na OS Linux Debian. Podrobuje tento server testům odolnosti proti pěti útokům DDoS. Těm je server vystaven před a po aplikaci systémů NIDS a NIPS, zastoupenými programy Snort a Suricata.

Cílem práce je na základě naměřených výsledků jednotlivých testů porovnat systémy NIDS a NIPS.

KLÍČOVÁ SLOVA

Spirent Avalanche, DoS, DDoS, IDS, IPS, Snort, Suricata

ABSTRACT

This bachelor thesis is focusing on testing the endurance of networks against DDoS attacks. The theoretical part consists of an introduction to the problematics of these attacks and current trends regarding DDoS attacks, focusing on IDS and IPS systems, and Spirent Avalanche 3100b machine, designed to generate network traffic. The practical part is about the configuration of the software web server Apache, which runs on Linux Debian OS, and it is testing this system for endurance against five DDoS attacks. The server is put through attacks before and after application of systems NIDS and NIPS, using Snort and Suricata software.

The goal of the thesis is comparing NIDS and NIPS servers based on the results of testing.

KEY WORDS

Spirent Avalanche, DoS, DDoS, IDS, IPS, Snort, Suricata

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Systémy pro detekci a prevenci průniku na hraničních zařízeních“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně, dne..... .....

Zdenko Bína

Zde bych rád poděkoval vedoucímu své práce panu Ing. Petru Blažkovi za trpělivost, vstřícnost a cenné rady, které mi během psaní bakalářské práce poskytoval.

Obsah

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

Úvod

Už vynalezení prvního počítače byl velký krok vpřed a od tohoto okamžiku vývoj raketově vystřelil. Ze strojů s enormní velikostí a malým výkonem se již dostalo k nesrovnatelně menším a výkonnějším zařízením, z malinkých sítí vznikly obrovské, které dokáží vzájemně propojit všechny kontinenty, a to je jen příklad toho, co vše tento vývoj doposud přinesl. Vývoj, který se stále nezastavuje.

V dnešní době tak máme obrovské sítě, ke kterým se může připojit prakticky kdokoli, kdo vlastní telefon, notebook, počítač, v některých případech již dokonce stačí mít pouze televizi či hodinky. Všechny tyto přístroje mají lidem pomáhat a sloužit. Roste snaha vše co nejvíce digitalizovat a usnadňovat si tak život, jak jen je to možné.

Ovšem každá mince má dvě strany, a i tyto přístroje tak mohou být využity proti svým uživatelům. S trochou snahy lze díky nim získávat citlivá data, napadat servery různých společností nebo vykrádat účty. z těchto a dalších důvodů proto roste snaha podobným aktivitám zabránit nebo je alespoň co nejvíce znesnadnit. Začala tak vznikat různá zabezpečení, která se musí vyvíjet obdobně jako samy sítě.

V této práci se budeme zabývat právě možnostmi takovýchto ochran. Nejprve si rozebereme samotnou problematiku útoků na sítě. Dostaneme se tak k různým typům útoků na sítě, kdy si některé z nich i více přiblížíme. Pozastavíme se u trendů, které v dnešní době v této oblasti panují. Následně se zaměříme na možnosti ochrany. Více si tak rozebereme především systémy detekce a prevence průniku. Některé z nich si popíšeme podrobněji a následně je také vyzkoušíme v simulovaných testech.

To nás přivádí k praktické části této práce, ve které si vyzkoušíme možnosti zařízení Spirent Avalanche 3100b, které nám umožní různé útoky na sítě také simulovat. V našem případě se bude jednat o útoky SYN Flood, ARP Flood, UDP Flood, RST Flood a Unreachable Host. Samotné zařízení je však dost složité, proto jednu samostatnou kapitolu věnujeme jeho obsluze a seznámíme se podrobněji s aplikacemi TestCenter Layer 4-7 Application a TestCenter Result Analyzator, které tuto obsluhu umožňují.

Se zařízením Avalanche je možné simulovat pouze klienty, ale také server. My však využijeme pouze první možnost a nakonfigurujeme webový server Apache na stroji s linuxovým prostředím. Po těchto krocích již budeme moci vyzkoušet vlastní testování.

Vždy je dobré znát limity našeho zařízení, proto si nejprve alespoň některé z nich zkusíme proměřit. Následovat budou testy dříve zmíněných útoků, s cílem zjistit jejich dopady na testovanou síť, a dále zkoušky systémů pro detekci a prevenci těchto útoků. Tyto systémy budeme zkoušet dva a to Snort, který je v oblasti prevence před útoky nejznámější, a také velice oblíbenou Suricatu. Obecnému popisu takovýchto systémů, a oběma zmíněným, věnujeme jednu samostatnou kapitolu. V ní se budeme mimo jiné zabývat také způsoby, jimiž případné útoky nalézají.

Oba systémy je potřeba nejprve zprovoznit, zmíníme tedy, jak provést instalaci, a především jak nakonfigurovat databáze, aby byly schopny reagovat na útoky.

Po odzkoušení systémů vyhodnotíme výsledky měření a porovnáme je mezi sebou, abychom získali lepší přehled rozdílů mezi systémy detekce a prevence průniku.

1. DoS a DDoS útoky

DoS (Denial of Service) i DDoS (Distributed Denial of Service) se řadí mezi síťové útoky. Jejich úkolem je znepřístupnit určitou službu legitimním uživatelům. Toho se dá docílit více způsoby. Dají se tak zaměřit na vyčerpání možností výpočetního výkonu oběti anebo na zahlcení přenosového pásma. Výstižnou definici těchto útoků nabízí Techopedie: „Odepření služby (DoS) je jakýkoli útok, kde se útočník (hacker) pokouší zabránit legitimním uživatelům přistupovat ke službám “.[1]

Rozdíl mezi oběma je pouze v počtu útočících počítačů, kdy při DoS je využíván pouze jediný a při DDoS se do útoku zapojuje větší množství, tedy dva a více strojů. Také sem řadíme útoky DDRoS (Distributed Reflection DoS), kdy počítače zapojené do DDoS navazují komunikaci s dalšími počítači, které se následně zapojí do útoku, aniž by o tom věděli.

Tyto útoky jsou možné na všech vrstvách ISO/OSI modelu, co zahrnuje vše od vytrhnutí kabelu až po cílený útok na konkrétní aplikaci. Více využívané však bývají útoky na nižších vrstvách, protože jsou jednodušší a v případě úspěchu mají větší dopady na oběť. První takový útok byl zaznamenán v roce 1996, kdy byla útokem SYN Flood napadena firma Panix v New Yorku. Jednalo se o útok s podvrhnutými IP pakety, takže útočníky nešlo vystopovat. Při útoku byl systém zahlcován okolo 150 SYN pakety za sekundu a nedokázal tak obsluhovat regulérní požadavky.

Mezi tyto útoky se kromě SYN Flood dále řadí např. Teardrop, UDP flood, ICMP flood, ARP flood,… o těchto i dalších si více povíme v **Typy útoků**.

1.1. Procentuální zastoupení útoků

V této kapitole vycházíme ze statistik společnosti Akamai[2], která se zabývá problematikou útoků DDoS a Web Applications Attacks, tedy útoky na webové aplikace. Na obr. 1 můžeme vidět graf procentuálního zastoupení DDoS útoků z posledního kvartálu roku 2016.

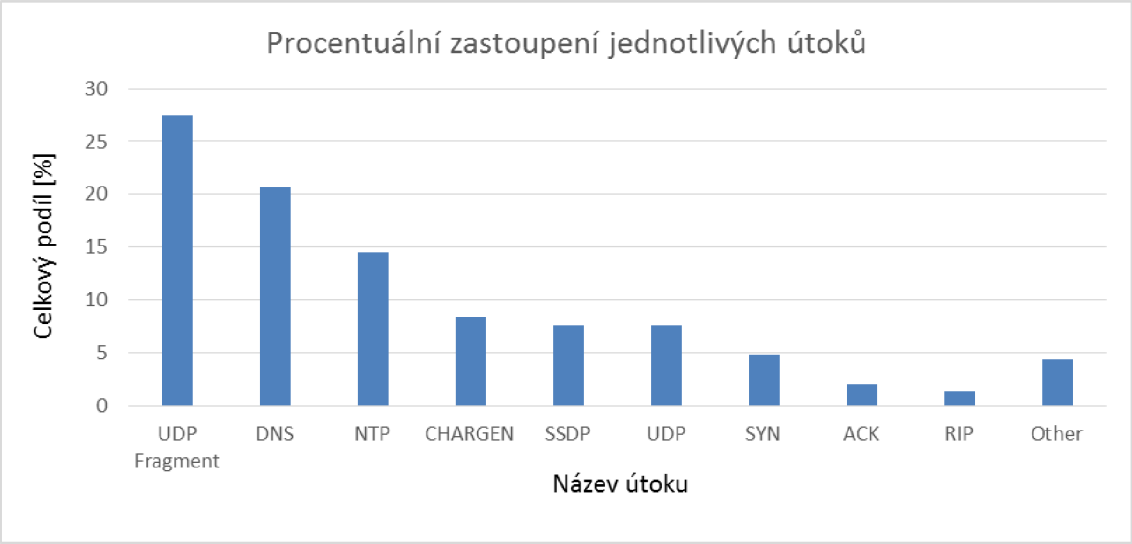
Můžeme vyčíst, že největší zastoupení mají útoky pomocí UDP fragmentů, které tvoří více než čtvrtinu všech útoků. Následují útoky DNS využívané v pětině případů a top tři útoky uzavírá NTP s 14,5 procentním zastoupením. Další útoky již mají

---

[1] https://www.techopedia.com/definition/24841/denial-of-service-attack-dos

[2] https://www.akamai.com/us/en/multimedia/documents/social/q4-state-of-the-internet-security-spotlight-iot-rise-of-300-gbp-ddos-attacks.pdf

zastoupení pod 10 %, jako Chargen s 8,36 %, SSDP s 7,63 %, UDP s 7,61 % či SYN s 4,86 %. Zbývající, ani ne osmi procentní zastoupení, mají na svědomí další útoky (ACK 2 %, RIP 1,36 %, SNMP 0,9 %, TCP Anomaly, TFTP 0,8 %, …).



Obrázek 1 - procentuální zastoupení útoků DDoS ze čtvrtého kvartálu roku 2016

V porovnání se třetím kvartálem došlo k úbytku DDoS útoků o 16 %, co statisticky bývá u posledního čtvrtletí roku běžné. Oproti čtvrtému kvartálu roku 2015 však došlo k čtyřprocentnímu nárůstu všech útoků. Nárůst byl zaznamenán především u enormních útoků, které přesahovali rychlost 100 Gbps. Ty vzrostly z pěti na 12, co dělá nárůst o 140 %. Větší oblibě se pak těší také útoky DDRoS, u nichž byl zaznamenán nárůst o 22 %.

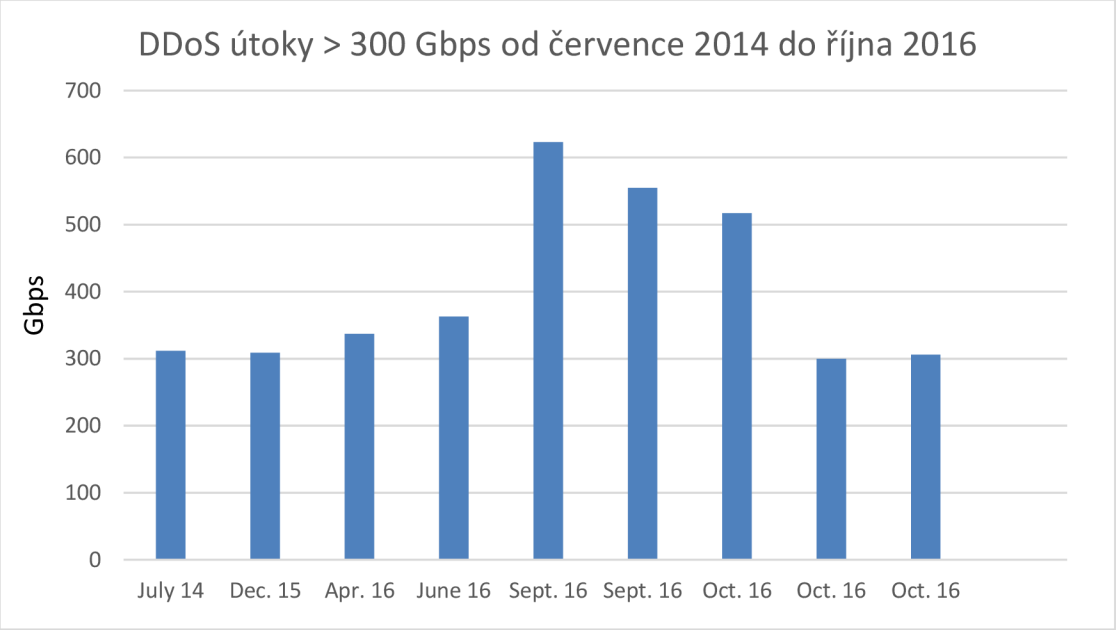
1.2. Největší DDoS útoky

Následující obrázek znázorňuje útoky nad 300 Gbps od července 2014. Je zde očividný nárůst těchto útoků v posledních měsících, neboť hned polovina z nich proběhla v posledním půlroce, a to včetně největšího takto zaznamenaného útoku ze září 2016, který přesahoval dokonce 600 Gbps. Tento útok byl veden na web Briana Krebse, zabývající se internetovou bezpečností. Společnost Akamai nakonec ochranu tohoto webu musela vzdát, protože vyčerpávala příliš mnoho zdrojů a docházelo by k ovlivnění dalších zákazníků. Web tak přešel na celý den do režimu offline. Předpokládá se, že

tento útok byl veden kvůli zjišťování bližších informací o provozovatelích DDoS-for-hire služby vDOS.[3]

Dodatečným prozkoumáním tohoto útoku se přišlo na to, že byl napadený botnetem vytvořeným díky viru Mirai (japonsky budoucnost). Ten napadá chytrá zařízení za pomocí tabulky běžných továrních hesel. Při úspěšném útoku dokáže infikovaná zařízení ovládat a ty se tak stávají součástí botnetu.

Vznikl však také virus Hajime (japonsky počátek), který napadá zařízení obdobně. Ten se však tváří jako ochránce před Mirai, kdy po infikování zařízení zablokuje porty, na které Mirai cílí. V současné době se tak sám označuje jako „white hat“, který chrání zařízení. Není však vyloučeno, že do budoucna se z botnetu tohoto viru také stane hrozba, která bude další sítě napadat.[4]



Obrázek 2 - DDoS útoky přesahující 300 Gbps od července 2014 do konce října 2016

Ani tento útok však nezůstal nepřekonán. Od zmíněného útoku již proběhl další, větší. Jednalo se o útok na francouzskou společnost OVH, poskytující hostingové služby. Útok byl ještě o takřka 80 % mohutnější, co se rovnalo síle útoku okolo 1,1 Tbps. Zapojilo se do něj více než 150000 chytrých zařízení. Konkrétní záměr není znám a k útoku se žádná skupina nepřihlásila. Jednalo se tak hlavně o demonstraci síly.

---

[3] https://www.root.cz/clanky/postrehy-z-bezpecnosti-rekordni-ddos-utok-1-1-tbps/
[4] http://www.osel.cz/9379-cerv-hajime-imunizuje-elektroniku-proti-infekci-mirai.html?typ=odpoved&id_prispevku=153734

Pro tyto útoky již nejsou využívány pouze počítače, ale především IoT (Internet of Things), kdy útočníci ovládnou chytrá zařízení používající Bluetooth. Tato zařízení se totiž čím dál více vyskytují ve zdravotnictví, inteligentních domech i dalších „smart“ zařízeních. Uvádí se, že až 75 % z nich se dá snadno napadnout a ovládnout. Je to způsobeno tím, že většinou využívají přenos hesla v prostém textu bez šifrování a není tak problém tato hesla odchytit a získat kontrolu nad zařízeními. To je přímým důsledkem snahy výrobců o co nejlevnější výrobky. Většina lidí si navíc řekne „kdo by chtěl ovládnout žárovku?!“ a o bezpečnost se tím pádem více nezajímají. Tato žárovka se pak lehce stane součástí botnetu, spolu s dalšími tisíci takovýchto žárovek, a půda pro další útoky je na světě.[5]

Právě kvůli rozvoji IoT se dá předpokládat, že tyto masivní útoky budou stále častější. Je proto potřeba s nimi počítat a hledat účinnou obranu, aby v případě jejich realizace byly dopady co nejnižší.

1.3. Typy útoků

Jak jsme mohli vyčíst, útoků se vyskytuje mnoho a v této kapitole více rozebereme některé z nich. Zajímavé jsou tak pro především ty nejrozšířenější, a také několik dalších, které budeme dále testovat v praktické části.

1.3.1. UDP fragmentace

V případě, že je velikost přenášené buňky větší, než MTU (Maximum Transfer Unit – hodnota udávající maximální velikost přenesené jednotky, u Ethernetu se obvykle rovná 1500 bajtů), daného protokolu nižší vrstvy, musí dojít k fragmentaci, tedy rozdělení buňky na více menších. Ty v hlavičce přenáší také pořadové číslo, díky kterému se dají tyto fragmenty seřadit a sestavit původní velkou buňku.

Při útoku se využívá neočekávaných stavů při opětovném sestavování fragmentů dohromady. Špatným nastavením či označením je útočník schopný docílit několika stavů, aby:

- se pořadová čísla alespoň částečně překrývala a překročila stanovený povolený limit.

[5] Nejedná se samozřejmě pouze o žárovky, jsou zmíněny pouze jako referenční příklad. Dalšími příklady mohou být například kamery, sportovní náramky, chytré hodinky, atd.

- výsledná složená jednotka byla větší než povolený limit.
- chyběla ke kompletaci některá data.
- se zaplnila paměť určená k dočasnému uchování fragmentů čekajících na sloučení.

Napadený systém pak neví, jak se v této situaci zachovat a často dochází k jeho zhroucení.

1.3.2. UDP flood

Útok za pomocí velkého množství UDP datagramů na různé porty. Tímto způsobem útočník zahlcuje šířku pásma. Systém oběti navíc podle portu kontroluje, které aplikaci je datagram vyslán. Když zjistí, že nikomu, zasílá zpět ICMP odpověď o nedosažitelnosti cíle. To musí provést pro každý datagram, takže dojde k vyčerpání prostředků a nemožnosti navazovat další spojení

1.3.3. DNS flood

Obdoba UDP flood se zacílením na konkrétní službu. Snaha o zahlcení DNS serverů, které překládají doménová jména na konkrétní IP adresy. Útoků na tuto službu je však více, například DNS cache poisoning, kdy DNS server přesměruje klienta na špatnou stránku, kterou má pod kontrolou útočník. Na ní se může dále schovávat třeba virus či počítačový červ, případně se jednat o phishingový web s cílem získat citlivá data klientů.

1.3.4. SYN flood

Nejstarší známý útok. Využívá principu navazování TCP spojení označovaného jako „three-way handshake“. Probíhá tak, že útočník vyšle SYN paket pro navázání spojení, který má ovšem podvrženou IP adresu. Oběť paket přijme a vysílá na podvrženou IP adresu odpověď ACK/SYN, kterým potvrzuje přijmutí paketu a snaží se o synchronizaci. Následně čeká na potvrzení navázání spojení, které ovšem nepřichází, a tak posílá paket znovu. Po vypršení stanovené doby bez odezvy druhé strany dochází ke zrušení alokace paměti a záznamu o inicializaci spojení.

V případě velkého počtu takovýchto podvržených žádostí může dojít k alokaci většiny systémových prostředků oběti, co se projeví zpomalením či neschopností zpracovávat další žádosti o navázání spojení

1.3.5. Teardrop

Tento útok využívá položku offset, díky které se zpětně sestavují pakety do původní podoby. Právě zfalšováním offsetu se tak dá docílit, že se budou některé části překrývat a cílový počítač nebude vědět, jak s pakety nakládat, co může vést k zamrznutí nebo restartování stroje.

1.3.6. ICMP flood

Nejčastěji využívá pakety ICMP Echo, které se používají při příkazu ping, tedy ke zjišťování, zda je vzdálené zařízení dostupné. Při útoku se zfalšuje adresa odesílatele a tím se linka začne ucpávat. k ucpávání dochází oběma směry díky ICMP Echo request dotazům směřujícím na oběť a zároveň odpověďmi ICMP Echo reply směřujícími na zfalšovanou adresu.[6]

Útok je snadné provést především v systému Linux, kde se dá nastavit velikost ICMP Echo paketu a také spustit záplavový režim, který následně zasílá ICMP Echo pakety jak nejrychleji umí.[7]

1.3.7.ARP spoofing

Pomocí falšování ARP zpráv se útočník snaží docílit přeposílání provozu na podvrženou stanici. Může tak komunikaci zachytávat anebo ji posílat na neexistující adresu, čím dojde k nemožnosti komunikovat se zařízeními za routerem.

---

[6] ŠTĚRBA, Jan. *Možná ohrožení elektronického bankovnictví.* [online] Praha: Bankovní institut vysoká škola Praha, Katedra informatiky a kvantitativních metod, 2014. Bakalářská práce. Vedoucí práce: Ing. Antonín Vogeltanz. Dostupné z: https://is.bivs.cz/th/20376/bivs_b/Sterba_BP.txt.

[7] https://www.lupa.cz/clanky/denial-of-service-dos-utoky-zaplavove-typy/

2. Systémy detekce a prevence průniku

Kromě firewallu pomáhají síť chránit také systémy detekce a prevence průniku. Ty mají za úkol rozpoznat nežádoucí komunikaci a na základě nastavených pravidel podniknout další kroky. o jaké se jedná si nyní upřesníme.

Systémy IDS

Systém detekce průniku zajišťují pasivní sondy, které analyzují veškerý síťový provoz. V případě, že naleznou nebezpečnou komunikaci, určí stupeň nebezpečí a na jeho základě vydají varování anebo pouze zaznamenají příslušnou aktivitu do logu. V tomto případě je potom na správci bezpečnosti, aby podnikl další kroky k eliminaci nežádoucí komunikace.

Nevýhodou tohoto systému je, že útok zaznamenají až v jeho průběhu a minimálně první paket tak vždy projde.

Systémy IPS

V tomto případě se často hovoří o rozšíření IDS. Na rozdíl od něj IPS dokáže aktivně chránit síť ihned při začátku útoku. V případě nalezení nebezpečné komunikace, mohou aktivně zasáhnout a začít komunikovat s firewallem. Ten na základě těchto podnětů může změnit ACL (Access Conrol List) tak, aby byl nežádoucí provoz potlačil. Nemusí však ihned docházet k takto drastickým krokům. V případě vyhodnocení nízkého rizika dojde pouze k vydání varování a zaznamenání do logu obdobně jako u IDS. Záleží tak na nastavení systému, jakou akci, pro jakou úroveň nebezpečí, mají vykonat.

Problém zde může nastat v případě planého poplachu, kdy systém detekuje hrozbu, která reálně hrozbou není a může tak dojít k odstřihnutí legitimní komunikace, co může vést ke značným škodám. Konfigurace je proto velice citlivá, neboť je žádoucí, aby bylo detekováno co nejméně planých poplachů a zároveň bezpečnost byla co nejvyšší.

2.1. Rozdělení podle umístění

Podle toho, kde v sítí se tyto systémy nacházejí, rozlišujeme dva základní druhy a to Host-based IDS/IPS a Network-based IDS/IPS. Nejedná se však pouze o topologickou změnu. Díky umístění je možné systémy nastavit na sledování rozdílných aktivit a je tak důležité si před nasazením rozmyslet, co chceme kontrolovat.

2.1.1. Host-based

Takto umístěné systémy sledují aktivity na koncových zařízeních jako:

- příchozí a odchozí provoz,
- aktivity uživatelů,
- pokusy o vniknutí,
- různé změny, ať už v souborovém systému, protokolech událostí, aktualitách, otevřených portech, spuštěných službách a aplikacích, …

Kvůli široké škále kontrolovaných aktivit se tyto systémy dále dělí na systémy:

- **Kontrolující souborový systém** – fungují na základě porovnávání určitého stavu (dříve vytvořeného otisku systému, tzv. srovnávací základny) se změnami. Srovnávací základnu je dobré umístit na médium, které je jen pro čtení, aby jej nikdo nepřepisoval, přesto se může stát, že útočník nepozorovaně systému podstrčí srovnávací základnu jinou.

  Protože se některé části systému mění často a jiné málokdy, je potřeba nastavit, které adresáře má systém hlídat. Je tak například zbytečné hlídat adresář Temp, který se mění často, ale za to knihovny jádra, u kterých změny probíhají málokdy, hlídat třeba je.
- **Sledující síťová připojení** – sbírají informace o síťových spojeních a přiřazují je jednotlivým uživatelům a procesům. Díky tomu dokáží reagovat na útok blokováním provozu, ale hrozí u nich riziko falešných poplachů, s následným blokováním legitimní komunikace.
- **Sledující logy** – informace získávají ze systémových logů, které analyzují a v případě podezřelých událostí upozorní správce.

2.1.2. Network-based

Umožňuje sledovat jednotlivé pakety síťového provozu. V síti mohou být umístěny dvěma způsoby:

- Inline, tedy přímo na lince, aby přes ně protékal veškerý provoz, čím se tvoří úzké hrdlo sítě. Díky svému umístění dokáží nežádoucí provoz ihned zablokovat.
- Paralelně v promiskuitním režimu. V podstatě se tak jedná o sondu, na kterou se zrcadlí provoz, který se zde vyhodnocuje. Jakmile se setká s nežádoucím

provozem, nedokáže s touto komunikací sám něco provést a o protiopatření tak požádá router nebo firewall.

Přímo v síti se umísťují před a za firewall/bránu VPN. Mohou být reprezentovány dedikovanými zařízeními pro tyto účely nebo routery a switchy s integrovanými funkcemi, či doinstalovanými síťovými moduly, pro NIDS/NIPS. Sledují příchozí i odchozí provoz na síti a také provoz uvnitř sítě.

2.2. Detekce hrozeb

Pro rozpoznání nežádoucí komunikace se využívá hned několika metod. Každá je založena na jiném základu a dokáže tak rozeznat útoky, které další metody ne. Systémy detekce proto využívají všechny tyto způsoby, k nimž se řadí detekce podle signatur, odchylek a anomálií.[8]

2.2.1. Podle signatur

Spočívá v porovnávání provozu s již známými vzory škodlivého kódu (signaturami). V případě rozdělení takovéto signatury do více paketů je systém schopný výsledný paket zrekonstruovat a škodlivý kód odhalit.

Tímto způsobem jsme schopni spolehlivě najít již známou signaturu, ovšem v případě nového, nepopsaného útoku, není možné tento útok odhalit.

2.2.2. Podle odchylek

Vychází se zde ze standardů, které definují jednotlivé protokoly (konkrétně RFC 4380[9]). V případě nalezení odchylky od běžného provozu, je vydána výstraha, a odchylka dále hlouběji analyzována.

2.2.3. Podle anomálií

Permanentním monitorováním a vyhodnocováním statistik o síťovém provozu je systém schopný odhalit nežádoucí aktivity. Sám vyhledává neobvyklé chování v síti a automaticky odhaluje síťové anomálie, které je schopný aktivně eliminovat. Nemusí

---

[8] REJTA, Lukáš. Bezpečnost síťového provozu: Přehled detekčních metod. [online] Brno: Masarykova univerzita, fakulta informatiky, 2014. Bakalářská práce. Vedoucí práce: Mgr. Tomáš Jirsík. Dostupné z: https://is.muni.cz/th/359202/fi_b/bc.pdf.

[9] http://www.rfc-base.org/txt/rfc-4380.txt.

se však jednat pouze o anomálie způsobené útokem, je schopný zaznamenat také nedostačující výkon či kapacitu sítě, případně špatně nastavené nebo nefunkční zařízení.

Díky této metodě je možné evidovat také dříve neznámý útok, na druhou stranu je zde však zvýšené riziko výskytu falešných poplachů.

2.3. Programy IDS/IPS

S rostoucím počtem útoků na počítačové sítě roste také snaha těmto útokům zabránit. Díky tomu se stále více různých firem zabývá vývojem IDS a IPS. Můžeme zmínit alespoň některé z nich, jako Snort, Surricata, Samhaim, AIDE, Bro NIDS, Fail2ban, OSSEC HIDS, ACARM-ng, …

Právě první dva zmíněné si rozebereme více v následujících řádcích.

2.3.1.Snort

Jedním z nejznámějších NIDS a NIPS programů je Snort. Jedná se o open-source program vytvořený Martinem Roeschem roku 1998. Jeho vývoj pokračuje stále dál pod hlavičkou SourceFire. V současné době je schopný pracovat na všech operačních systémech, jako Windows, Linux, MacOS, a dalších.

Je schopný kombinovat všechny tři dříve zmíněné detekční metody. Kombinuje tak výhody detekce signatur, odchylek, i anomálií. Původně byl však schopný pracovat pouze jako paket sniffer, později již také jako paket logger a NIDS. Stále je možné všech těchto možností využít nastavením správného módu, největších úspěchů však dosahuje NIDS.[10]

- Sniffer mode – mód pro zachytávání paketů a zobrazování na obrazovku. Takto lze zachytávat buď celou komunikaci, nebo pouze od určitého stroje
- Paket logger mode - v podstatě se jedná o rozšíření sniffer mode o ukládání do log souborů[11]

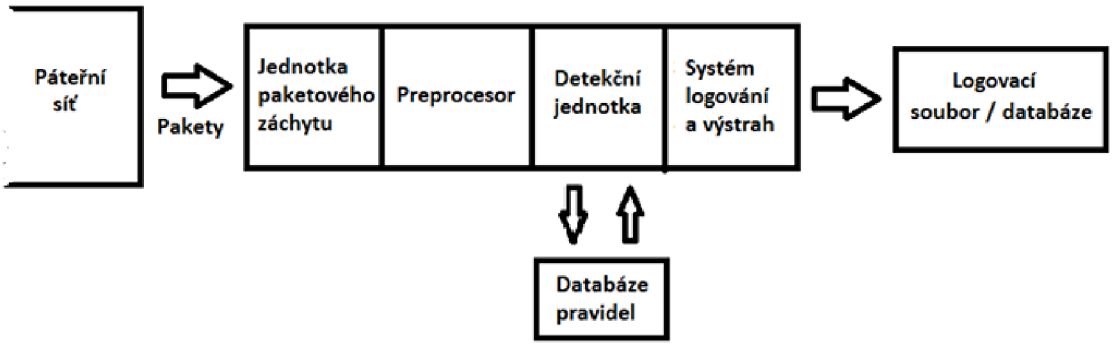
---

[10] MAURIC, Jakub. IDS systém SNORT. [online] České Budějovice: Jihočeská univerzita v Českých Budějovicích, pedagogická fakulta, katedra informatiky, 2009. Bakalářská práce. Vedoucí práce: Ing. Ladislav Beránek, CSc., MBA. Dostupné z: https://theses.cz/id/4lgemh/downloadPraceContent_adipIdno_12546.

[11] ORKÁČ, Radomír. *IDS Snort*. [online] Ostrava: Vysoká škola báňská – Technická univerzita Ostrava, fakulta elektrotechniky a informatiky, 2006. Semestrální projekt do předmětu Směrované a přepínané sítě. Dostupné z: http://www.cs.vsb.cz/grygarek/SPS/projekty0506/Snort.pdf.

- NIDS mode – u tohoto módu probíhá zachytávání paketů a jejich další analýza. Při ní se zachycené pakety porovnávají s předdefinovanými pravidly a na základě vyhodnocení programu dochází k dalším akcím

Samotný program sestává z několika komponent, které na sebe navazují a doplňují se. Patří mezi ně jednotka záchytu paketů, preprocesor, detekční jednotka a systém logování a výstrah. z těchto zmíněných jsou však některé moduly pouze zásuvné, tak aby byly snadno odnímatelné. Konkrétně se jedná o preprocesor, detekční jednotku, výstupní moduly a systém logování a výstrah.



Obrázek 3 - architektura Snortu

Jednotka záchytu paketů

Tato jednotka reprezentuje paket sniffer. Umožňuje naslouchat síťovému provozu a je schopna pracovat s mnoha protokoly, včetně nejrozšířenějších TCP, UDP, ICMP či IPSec. Pakety převádí do pro lidi čitelné formy, či unifikovaného výstupu vhodného pro další zpracování.

Preprocesor

Slouží k vyhledávání nebezpečí a přípravu paketů pro detekční jednotku. Může tak například uspořádat nebo seskupit řetězce do podob známých pro IDS, díky čemu jsou následně správně vyhodnoceny.

Využívá se také pro defragmentaci nutnou v případě přenášení velkých dat, která se obvykle rozdělují do více paketů a před kontrolou je potřeba je nejprve zkompletovat. Pokud tedy nastane situace, kdy se zasílá větší velikost dat, než povoluje MTU, rozdělí se na více datových paketů. Signatura tak může být také rozdělena, čím by

mohli hackeři oklamat IDS. z toho důvodu je paket nejprve celý složen, aby došlo ke správnému detekování hrozeb.

Detekční jednotka

Porovnává data z každého paketu s předdefinovanými pravidly. Vyhodnocuje tak reálné nebezpečí paketu, díky čemu ji můžeme řadit na místo nejdůležitější jednotky Snortu. V případě škodlivého paketu zavádí příslušné protiopatření jako zápis do logu nebo vytvoření výstrahy. Její rychlost je závislá na více faktorech. Mezi ně patří počet porovnávaných pravidel, zatížení sítě, výkon stroje, na kterém běží a rychlost jeho vnitřní sběrnice.[12]

U verzí Snortu 1.x detekční jednotka nalezla první shodu s některým pravidlem, na jejímž základě paket vyhodnotila a postoupila dále, aniž by kontrolovala shody s dalšími pravidly. To mohlo vést k problému, kdy bylo v paketu hrozeb více a první zjištěné mělo nízkou prioritu výstrahy. Paket tak mohl být zařazen mylně. I proto byla detekční jednotka od verze 2.0 kompletně přepsána. V současných verzích se tak můžeme setkat s o mnoho rychlejší detekční jednotkou, která navíc před vytvořením výstrahy porovnává veškerá pravidla a teprve poté vybere vyhovující pravidlo s nejvyšší prioritou.

Systém logování a výstrah

Přímo navazuje na detekční jednotku a dále zpracovává detekované hrozby zapsáním do logu anebo vydáním výstrahy. Samotný log je pouhým textovým souborem, který se v příslušném tvaru ukládá na implicitně nastavené místo anebo díky zásuvným modulům na jiný počítač, email, apod. Takto je možné informovat správce systému o hrozbách v reálném čase.

2.4. Suricata

Dalším rozšířeným open source NIDS a NIPS je Suricata. Ta je vyvíjena společností OISF (Open Information Security Foundation). Její beta verze poprvé spatřila světlo světa v prosinci roku 2009. Oficiálního vydání se dočkala v červenci následujícího roku.[13]

---

[12] POTŮČEK, Petr. Testování firewallů pomocí hardwarového testovacího přístroje. [oniline] Brno: Masarykova univerzita, fakulta informatiky, 2012. Bakalářská práce. Vedoucí práce: doc. RNDr. Eva Hladká, Ph. D. Dostupné z: https://is.muni.cz/th/256710/fi_b/Bakalarska_prace.pdf.

[13] https://suricata-ids.org/

V porovnání se Snortem má některá svá specifika. Jedná se především o podporu více vláknového režimu, díky kterému se dá efektivněji využívat dostupných prostředků. Další předností je rozeznávání protokolů jako takových. Je tak možné nastavit pravidla pro vybraný protokol bez přiřazování určitých portů. Dokáže rozeznat tisíce typů souborů putujících po síti. Během komunikace je schopna vypočítat kontrolní součty u MD5 souborů. Pokud tak máme seznam MD5 hashů, které nemají opustit naši síť, případně které nechceme do sítě vpustit, je Suricata schopna tyto hashe v komunikaci najít.

Umožňuje pracovat s vlastními pravidly, ale je také možné využívat pravidla Snortu.

2.5. Instalace Apache, Snort a Suricata

Tímto tématem se zabývá spousta různých článků na internetu, podle jejichž popisů jsme také postupovali. Uvedeme si zde proto pouze několik základních příkazů a reference na příslušné návody.

2.5.1. Apache

Balíček Apache nám umožní nastavit stanici jako HTTP server. Při instalaci je nejprve potřeba zkontrolovat, jsou-li úložiště aktuální, případně je aktualizovat. To provedeme příkazem

```
sudo apt-get update && sudo apt-get upgrade
```

Standardně se na systému Linux instaluje Apache spolu s databázemi, tzv. LAMP (Linux, Apache, MySQL, PHP). Po kontrole tak můžeme nainstalovat Apache a s ním MySQL a PHP. Zadáváme tak posloupnost příkazů pro instalaci

```
apt-get install apache2,
apt-get install php5 libapache2-mod-php5,
apt-get install mysql-server mysql-client.
```

Stačí už pouze provést restart pomocí

```
/etc/init.d/apache2 restart
```

a HTTP server nám začne běžet. Stačí si už pouze nachystat soubor, se kterým budeme v našich testech pracovat a vložit ho do adresáře /var/www, který je pro Apache přednastavený jako výchozí.[14]

2.5.2.Snort

Instalace systému Snort je hezky popsána přímo na jeho oficiálních stránkách[15]. Není proto důvod instalovat jej jakkoli jinak.

Nejprve je potřeba stáhnout instalační balíčky příkazem wget, pomocí tar je rozbalit a zadat install ke spuštění instalace. Postup je následující:

wget https://snort.org/downloads/snort/daq-2.0.6.tar.gz,

tar xvfz daq-2.0.6.tar.gz,

cd daq-2.0.6,

./configure && make && sudo make install.

Obdobný pro druhý balíček:

wget https://snort.org/downloads/snort/snort-2.9.9.0.tar.gz

tar xvfz snort-2.9.9.0.tar.gz,

cd snort-2.9.9.0,

./configure –enable-sourcefire && make && sudo make install.

Tím je instalace dokončena. Je také možné, ne však nezbytné, stáhnout a aplikovat další pravidla. Důležité ale je upravit soubor /etc/snort/snort.conf pro potřeby našeho měření.

2.5.3.Suricata

Instalace je prakticky stejná jako u Snortu. Stačí si nejprve stáhnout instalační balíček Suricaty, rozbalit jej, a nainstalovat[16]. V našem případě však již byla Suricata nainstalována a aktualizace proběhla již při instalaci Apache, po zadání příkazu

sudo apt-get update && sudo apt-get upgrade.

Také v případě Suricaty je potřeba upravit konfigurační soubory, aby byly vhodné pro naši síť.

---

[14] https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-debian

[15] https://snort.org/

[16] Změna je pouze ve stažení z webu https://www.openinfosecfoundation.org/download/suricata-3.2.2.tar.gz a následné rozbalení a instalace právě tohoto balíčku

3. Testování zátěže

S tímto široce rozšířeným pojmem se můžeme setkat v řadě různých oborů a vždy bude vyjadřovat něco jiného. V obecné rovině slouží ke zjišťování možností daného zařízení či systému. Výstupem je zjištění schopností, limitů, mezí nebo slabin testovaného subjektu. Způsobů testování bývá obvykle více a je si tak potřeba rozmyslet, jakého výstupu chceme dosáhnout, abychom aplikovali směrodatný test.

V našem případě nás budou zajímat schopnosti linuxového serveru. Především se tak zaměříme na otestování jeho možností co do počtu navázaných spojení a transakcí. k tomuto účelu nám poslouží zátěžový generátor Spirent Avalanche 3100b. Díky výsledkům měření budeme schopni stanovit vyhovující počáteční stavy pro další měření tak, abychom server příliš nezatěžovali legitimní komunikací a nezkreslovali se tak další výsledky po aplikaci útoků DDoS.

4. Spirent Avalanche 3100b

Zařízení Spirent Avalanche poskytuje řešení v oblasti testování kapacit, bezpečnosti a výkonnosti síťových infrastruktur, webových aplikací a triple play služeb (přenos TV obrazu, hlasu a dat). Také zajišťuje služby kvality QoS a QoE. Pracuje na 4. – 7. vrstvě referenčního modelu ISO/OSI a je schopné vytvořit až 30 milionů spojení, specifikovat je, testovat DDoS útoky, atd. [17]

Pro obsluhu budeme využívat dvou aplikací a to *TestCenterLayer 4-7 Application* a *TestCenter Result Analyzer*. První zmíněná slouží ke specifikaci jednotlivých testů a následnému měření. Její možnosti jsou rozsáhlejší, proto si ji popíšeme dále v samostatné podkapitole. Druhá aplikace umožňuje interpretaci výsledků těchto testů v přehledné grafické podobě a jejich převod do formátu pdf nebo html.

4.1. TestCenterLayer 4-7 Application

V této aplikaci je možné nastavit parametry testů a sledovat jejich průběh. Aby to bylo možné, je nejprve potřeba vytvořit projekt volbou File/New/Project a zadat pouze název a umístění projektu. Následně můžeme do tohoto projektu začít vytvářet jednotlivé testy. Při jejich tvorbě máme možnost nastavit, jestli bude Avalanche simulovat pouze cílové stanice anebo také server. V našem měření je server zastoupen samostatným zařízením, proto vybíráme pouze simulaci klientů. V posledním kroku vybíráme ze dvou možností testů, Quick nebo Advance. Rozdíl je ve škále nastavitelných prvků, kdy Quick je test se základními parametry a Advance je jeho rozšířením o další parametry jako DDoS, proto ho budeme využívat více.

Po založení testu se nám objeví okno s pěti základními záložkami **Client**, **Content Files**, **Notes**, **Run** a **Results**. Záložky Content Files a Notes pro nás nejsou příliš zajímavé, proto si dále popíšeme pouze další zmíněné.

Nejzajímavější záložkou při nastavování testů je Client. V něm nastavujeme veškeré parametry, kterých je velká spousta, a proto je ještě dále rozdělena na Loads, Actions, Profiles, Network, Subnets, Ports a Associations.

Loads – slouží pro nastavení simulované zátěže, a to jak druhu zátěže, tak jejího průběhu. Nejzajímavější položkou je Specification, kde můžeme vybrat jednu z následujících možností:

---

[17] https://www.infopoint-security.de/medien/spirent_avalanche_3100_datasheet.pdf

- Bandwidth – nastavuje šířku pásma, čím umožňuje určit, kolik dat je možné poslat přes rozhraní za určitý čas.
- BodyBytes – pouze pro simulované servery. Jedná se o generování HTTP žádostí, které od serveru získávají odpovědi dané velikosti.
- Connections – pro nastavení počtu současně udržovaných TCP spojení. Lze využít pro libovolný TCP protokol.
- Connections/second – počet spojení vytvářených za sekundu. Jedná se o jednu z nejpreferovanějších zátěží u výrobců síťových zařízení a také ji budeme využívat pro naše testování.
- SimUsers – neboli Simulated Users, jak název napovídá, jedná se o virtuální uživatele. Cílem je vytvořit a udržet požadovaný počet těchto uživatelů, kteří mají právě jednou vykonat úkon zadaný v Action Listu.
- SimUsers/second – vytváření daného počtu virtuálních uživatelů za sekundu. Oproti SimUsers je test vhodnější pro simulování reálné sítě, protože se nebere v potaz chování serveru, ale dále pokračuje generování požadovaného počtu klientů.
- Transactions – slouží k vytvoření a udržení požadovaného počtu aktivních transakcí. Využívá se pouze pro HTTP a HTTPS protokol.
- Transactions/second – vytváření stanoveného počtu transakcí za sekundu. Obdobně jako u výše zmíněných testů platí, že je vhodnější pro simulování reálné sítě, a proto také, oproti Transaction, více využívané v praxi.

Zbylé položky na této záložce dále pracují s nastavenou zátěží, tj. určují její velikost, dobu trvání, tvar, apod.

Actions – v této záložce můžeme nalézt příkazy, které se mají vykonat. Je rozdělena na záložky Actions, kde vidíme již vytvořené příkazy a Generator, kde můžeme příkazy vytvářet. Stačí vybrat jeden z 19 nabízených protokolů a ihned je vypsán předdefinovaný příkaz, který lze přiřadit k libovolnému profilu v Actions. Po vybrání protokolu je navíc zobrazeno okno s dalšími parametry daného protokolu a je tak možné příkaz dále modifikovat (např. přidat přihlašovací údaje, název souboru, který se má stahnout, aj.).

Do záložky Actions jsme tak schopni nakonfigurovat práci s kterýmkoli z nabízených protokolů. Je zde ovšem možné definovat také další, pokročilejší, příkazy. V našem případě zde budeme konfigurovat základní parametry jednotlivých DDoS

útoků a vygenerované příkazy těchto útoků vkládat do pole Actions. Každý takto vytvořený záznam lze samostatně uložit pod vlastním názvem a posléze jen vybírat, kterou akci chceme v příslušném testu vykonat, díky čemu stačí všechny potřebné akce nadefinovat pouze jednou, a ne po každém testu zvlášť.

Profiles, **Network** – jedná se o další upřesňující nastavení, které v našem testu ale není třeba měnit, proto se těmito záložkami více zabývat nebudeme.

Subnets – slouží pro nastavení jednotlivých sítí, které budeme v testu využívat. Nastavujeme jméno příslušné sítě, rozsah IP adres, masku a adresu sítě. Další parametry, jako třeba výchozí brána nebo MAC adresa, jsou volitelné. s příslušnou licencí se dá vše nastavit i pro IPv6.

Ports – slouží k přiřazení portu IP adrese, přes kterou budeme provádět test. V našem případě se jedná o port 5 a výsledný záznam tak má podobu 192.168.1.2:5,5. Povolujeme zde také DDoS útoky a nastavujeme jejich parametry. Po povolení DDoS se nám zpřístupní podsložka DDoS, v níž jsou tři další karty:

- Attacks – zde je třeba vybrat z 15 útoků ty, se kterými chceme dále pracovat a povolit je. Tím se nám zpřístupní v další kartě, kde je můžeme nastavovat.
- Attacks Variables – umožňuje nastavovat kritéria jednotlivých útoků. Mezi tato kritéria patří například síla útoku, port, na kterém má útok běžet, kolikrát se má opakovat, atd.
- Global Variables – pro nastavení společných parametrů všech útoků. Nastavujeme zde počáteční zdrojovou a cílovou MAC adresu, jejich postupnou inkrementaci, dále počáteční zdrojovou a cílovou IP adresu a nakonec dobu od spuštění testu, kdy se mají začít spouštět útoky.

Associations – slouží pro přiřazení jednotlivých vytvořených profilů dříve zmíněných záložek do příslušného testu. Můžeme zde tak nastavit zátěž, která lze nastavit globálně pro všechny testy anebo přiřazovat jednotlivým testům, dále akce, které se budou vykonávat, zvolený profil, port a síť. Takto sdružené profily je ještě potřeba povolit, aby se při testu spustily.

Další důležitou záložkou je Run. Rozděluje se na další tři části:

- Configure – lze zde nastavit různé rozšiřující logování a statistiky. Také umožňuje povolit rozšiřující nastavení jako DDoS nebo rychlé ukončení testů.
- Monitor – zobrazuje průběh testu. Ten je rozdělen na pět fází:

- Test Preparation – přichystání testu a síťové karty,
- Ramping Up – doba náběhu na počáteční úroveň,
- Steady State – udržování nastavené zátěže na maximální úrovni,
- Ramping Down – doběh testu z nastavené na nulovou úroveň,
- Test Ended – indikace ukončení testu.

Zároveň umožňuje sledovat aktuální stav transakcí, konkrétně o kolik se celkově pokoušelo a kolik z nich bylo úspěšných, neúspěšných a přerušených. Dále zobrazuje uplynulý a zbývající čas a také statistiky druhé vrstvy. Jedná se o odeslané a přijaté pakety/byty a statistiku paketů, kolik se jich za sekundu právě posílá/přijímá, maximálně odesílalo/přijímalo a průměrně odesílá/přijímá.

- Load – zobrazuje průběh testu graficky, ukazuje přednastavené parametry z pole Associations, které byly spuštěny, požadovanou a aktuální zátěž, využití CPU. Zároveň umožňuje měnit hodnotu zátěže v probíhajícím testu.

Poslední záložkou je Results. Zde vidíme jednotlivé již proběhlé testy a můžeme si prohlédnout jejich průběhy a statistiky. Pro to již slouží Spirent TestCenter Result Analyzer, který požadované výsledky přehledně zobrazí v tabulkách a grafech.

5. Praktická část – měření zátěže

Na měření zátěže jsme si vybrali dva testy. Jednak test počtu transakcí za sekundu a dále test počtu připojení za sekundu. Při těchto testech máme možnost odzkoušet základní měření se zařízením Spirent Avalanche 3100b a zároveň si stanovit referenční hodnoty pro složitější testy.

5.1. Testování transakcí za sekundu:

Při tomto testu chceme zjistit, při jaké zátěži bude úspěšnost serveru 100 % a najít hranici, kdy úspěšnost začne klesat. V TestCenteru nastavíme příslušné parametry v položce Client:

- Loads:
  - Specification: Transactions/Seconds
  - Default Time Scale: Second
  - Pattern: Flat
  - Ramp Time: 300
  - Steady Time: 300
- Actions: 1 get http://192.168.2.142/index.html
- Subnets: IP Address (Range): 192.168.2.192-192.168.2.195/24
- Ports: 192.168.1.2:5,5

Položku Height v Client/Loads postupně navyšujeme. Začínáme na hodnotě 150.

I přes nižší počáteční zátěž jsme nejprve naměřili úspěšnost okolo 99 %. Problém nastával ihned při začátku měření, kdy docházelo k vypršení doby spojení, a proto jsme před samotný test vložili 20 sekund dlouhý úsek bez zátěže, který měl nahrazovat první fázi testování, tedy přípravu, aby následně nedocházelo k zahazování paketů kvůli navazování spojení. Po tomto opatření jsme již naměřili úspěšnost 100 %.

Při prvním měření jsme nastavili zátěž na 150 transakcí za sekundu. Úspěšnost byla 100 % a za dobu měření proběhlo 67858 transakcí. Velikost zátěže jsme proto navýšili o 20 na 170 transakcí za sekundu. Počet transakcí stoupl na 76870 a i zde všechny proběhly úspěšně. Při dalším navýšení na 190 transakcí za sekundu stále nedošlo k poklesu úspěšnosti a proběhlo 83622 transakcí. To se však změnilo při 210 transakcích za sekundu. Úspěšnost klesla na 33,5 %. z 94896 transakcí tak bylo jen 31814 úspěšných.

Proměřili jsme proto více úsek mezi 190 a 200 transakcemi. Pro 195 transakcí za sekundu byla úspěšnost stále 100 %, u 198 transakcí už jsme však zaznamenali pokles na 92,5 % úspěšných transakcí. z 89564 transakcí tak bylo úspěšných 82884.

Popsané výsledky máme názorně shrnuty v Tabulka 1.

Transakce za sekundu	Úspěšnost [%]	Celkem transakcí	Úspěšných transakcí	Neúspěšných transakcí
150	100	67858	67858	0
170	100	76870	76870	0
190	100	83622	83622	0
195	100	88134	88134	0
198	92,5	89564	82884	6680
210	33,5	94896	31814	63082

Tabulka 1- výsledky měření zátěže Transactions/Second

5.2. Connections/Second

V dalším měření jsme se zaměřili na proměření možností serveru z hlediska připojení za sekundu, s kterým budeme dále pracovat. Pro toto měření nám stačilo pouze přepnout zátěž v Loads/Specifications z Transactions/Second na Connections/Second a měnit velikost zátěže. Nastavení délky trvání testu na 600 sekund bylo pro naše testy zbytečně dlouhé, proto jsme ji pro další testování zkrátili změnou následujících parametrů:

- RampTime: 30
- Steady: 100

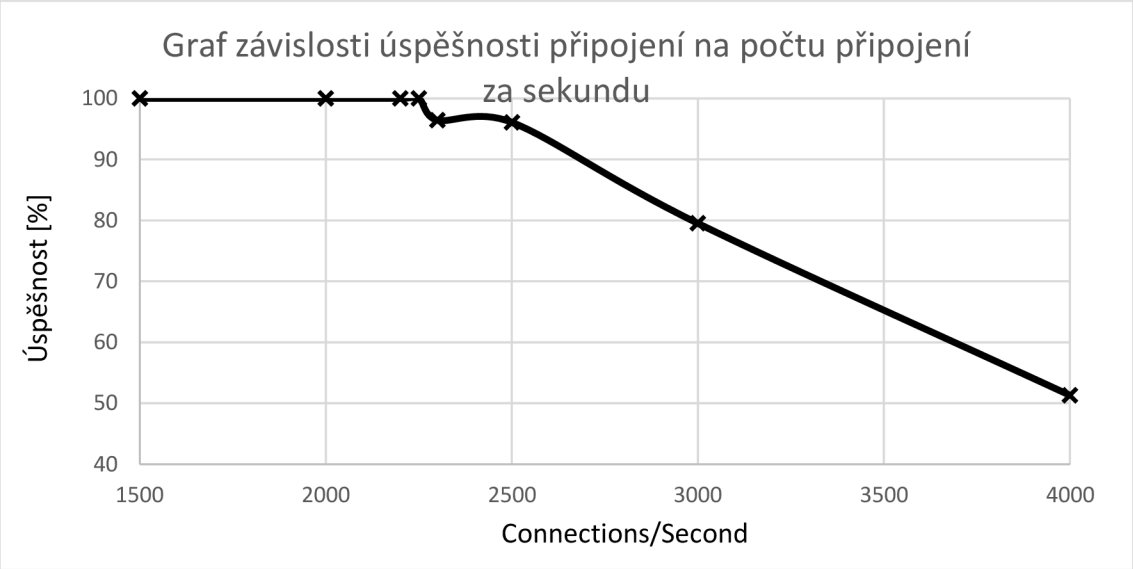
Úsek bez zátěže na začátku testu jsme ponechali, obdobně jako veškerá další nastavení.

Při samotném měření jsme zaznamenali kolísání úspěšnosti až při 2300 připojeních za sekundu. Zvolili jsme proto referenční zátěž pro DDoS útoky na 1500 připojení za sekundu, kde máme jistotu, že server nebudeme přetěžovat legitimní komunikací, ale uvidíme vliv DDoS útoků. Výsledky proměřených zátěží a graf úspěšnosti můžeme vidět v Tabulka 2 a v Obrázek 4. Musíme však brát v potaz, že výsledky odráží celou dobu měření, a ne pouze čas s maximální zátěží. Výsledek je

proto zkreslen prvními 50 sekundami, kdy pouze narůstá počet transakcí na nastavenou úroveň a prakticky se tak zde nevyskytují neúspěšné transakce. s ohledem na dobu trvání fáze Steady tak celou třetinu času zabírá pouze náběh na požadovanou zátěž. Pro naše účely však není toto zkreslení důležité.

Připojení za sekundu	Úspěšnost [%]	Celkem transakcí	Úspěšných transakcí	Neúspěšných transakcí
1500	100	172923	172923	0
2000	100	230417	230417	0
2200	100	253552	253552	0
2250	100	259285	259285	0
2300	96,41	265108	255599	9509
2500	96,08	288181	276883	11298
3000	79,51	345820	274961	70859
4000	51,31	461290	224585	236705

Tabulka 2- výsledky měření zátěže Connections/Seconds



Obrázek 4- Graf závislosti úspěšnosti připojení na nastavené zátěži Connections/Second

6. Testování DDoS:

V těchto měřeních se pokusíme zjistit vliv DDoS útoků na server. Jak již bylo zmíněno, nastavujeme zátěž na pevnou hranici 1500 připojení za sekundu a záložku Load tak nemusíme během testů vůbec měnit. Zajímá nás především počet DDoS paketů za sekundu, při kterém zaznamenáme změny chování serveru.

V našem měření volíme útoky ARP Flood, RST Flood, SYN Flood, UDP Flood a UnreachableHost. Pro měření je potřeba povolit DDoS útoky v záložce Ports a nastavit parametry útoků. Globální parametry pro všechny útoky jsme nastavili následujícím způsobem:

- StartingSourceMACAddress: 05:00:00:00:00:01
- MACAddressIncrement: +1
- StartingDestMACAddress: 00:30:48:5a:c5:ab
- StartingSourceIPAddress: 192.168.20.1
- StartingDestIPAddress: 192.168.2.142
- GlobalStartDelay: 70000

Pro jednotlivé útoky pak nastavujeme další parametry. U všech jsou položky RepeatCount a LocalStartDelay, které také nastavujeme shodně:

- RepeatCount: 50
- LocalStartDelay: 0

Dalšími společnými parametry jsou i PacketsToGenerate a PacketRate, které ovšem určují sílu útoku, která je pro každý útok individuální a v jednotlivých testech je s každým měřením měníme. Obě hodnoty nastavujeme na stejnou úroveň, díky čemu zajistíme, že položka RepeatCount přímo určuje délku DDoS útoku. Další parametry u jednotlivých útoků už stejné nejsou. Měnit je ale nemusíme, protože si vystačíme s předdefinovanými možnostmi.

6.1. ARP Flood

Prvním testovaným útokem byl ARP Flood. Při testování nás zajímá síla útoku a úspěšnost serveru, které také vynášíme do tabulek jednotlivých útoků. Začátek poklesu úspěšnosti zpracovávání požadavků na serveru jsme zaznamenali při síle útoku 300000 paketů/s.

V případě tohoto zlomu není příliš patrné zkreslení způsobené nastavením průběhu testu, ale například při zátěži 700000 DDoS paketů za sekundu program ukázal

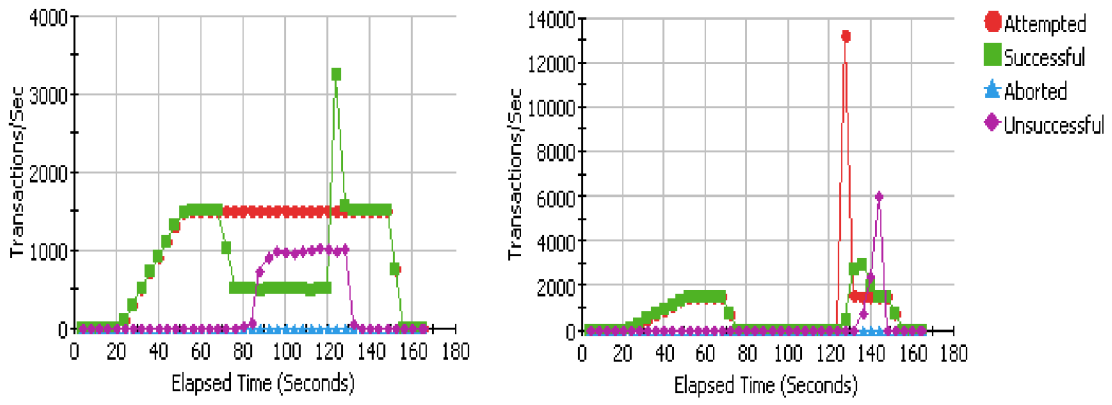
úspěšnost 73,32 %, co bylo na první pohled podezřelé. Po prozkoumání průběhu testu se tato domněnka potvrdila, když v době trvání útoku byl vidět úplný výpadek spojení a úspěšnost serveru se tak rovnala nule. Při těchto měřeních jsou pro nás důležité právě hodnoty naměřené v době, kdy je útok aktivní, proto veškeré výsledky budeme odečítat manuálně z grafů, které nám poskytuje TestCenter Result Analyzer, aby byly co nejpřesnější. U těchto grafů se po najetí kurzorem na vykreslenou křivku zobrazí aktuální hodnoty osy X (doba trvání testu) i Y (počet připojení). Víme, že počet připojení má být 1500, není tak problém dopočítat skutečnou úspěšnost serveru.

Na Obrázek 5 máme zobrazeno porovnání průběhů pro 500000 a 700000 DDoS paketů za sekundu. Útok probíhá od 70. sekundy doby trvání testu následujících 50 sekund. Končí tak právě dvě minuty po začátku testu, který ještě chvíli pokračuje. z vyobrazených grafů na obrázku je tento útok krásně viditelný poklesem úspěšných připojení na začátku útoku a obdobně jejich nárůstem zpět na přednastavenou hodnotu, jakmile útok skončí.

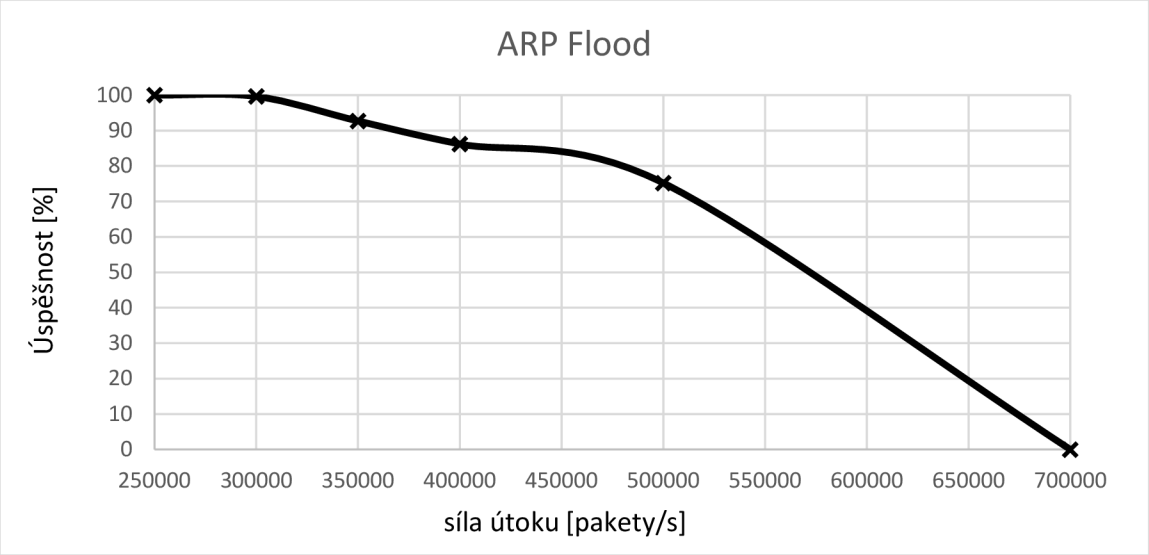
Z naměřených hodnot můžeme vynést graf úspěšnosti serveru v závislosti na síle daného útoku. V grafu vidíme spíše pozvolnější klesání úspěšnosti serveru. Mohlo však dojít ke zkreslení kvůli neproměření úspěšnosti v oblasti blížící se nule. Pravděpodobně tak došlo k úplnému výpadku spojení o něco dříve.

DDoS [paketů/s]	250000	300000	350000	400000	500000	700000
Úspěšnost [%]	100	97	77,9	60,47	32,9	0

Tabulka 3 - úspěšnost serveru při útoku ARP Flood



Obrázek 5 - porovnání průběhu testů pro 500000 (vlevo) a 700000 (vpravo) DDoS paketů za sekundu



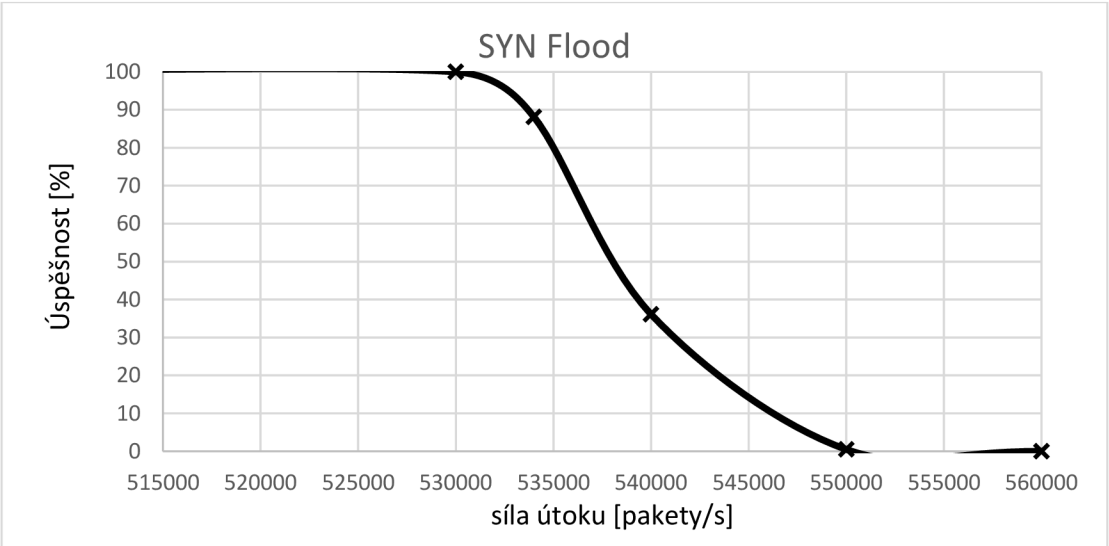
Obrázek 6 - Graf úspěšnosti serveru na počtu paketů DDoS útoku ARP Flood

6.2. SYN Flood

Dalším měřeným útokem je SYN Flood. První změny jsme zaznamenali při 534000 paketech za sekundu. Výsledky si můžeme prohlédnout na Obrázek 7. Při testování zátěže 550000 paketů za sekundu zde zaznamenáváme podobný výpadek spojení jako při ARP floodu, ovšem k nepatrnému navazování spojení stále docházelo, u 560000 již ne.

DDoS [paketů/s]	500000	530000	534000	540000	550000	560000
Úspěšnost [%]	100	100	88,13	36,13	0,5	0

Tabulka 4 - úspěšnost serveru při útoku SYN Flood



Obrázek 7 - Graf úspěšnosti serveru na počtu paketů DDoS útoku SYN Flood

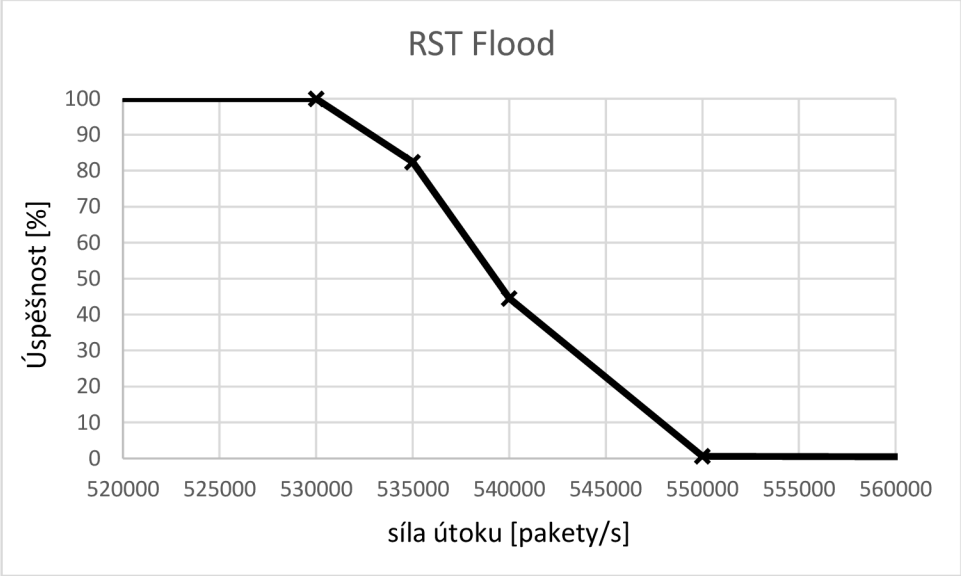
Po vynesení naměřených hodnot do grafu vidíme, že od prvního poklesu úspěšnosti nastává strmý pád úspěšnosti serveru. Po nárůstu počtu DDoS paketů o 30000 za sekundu dochází k úplnému výpadku. V porovnání s ARP Floodem tak vidíme první projevy útoku až při vyšším počtu generovaných paketů, ale za to dopady na server jsou po překonání této hranice mnohem radikálnější.

6.3. RST Flood

Třetím útokem, který vyzkoušíme je RST Flood. Úspěšný byl od 550000 DDoS paketů za sekundu. Při hranici 550000 již server nedokázal obsluhovat komunikaci.

DDoS [paketů/s]	530000	535000	540000	545000	550000	600000
Úspěšnost [%]	100	82,4	44,53	9,3	0,6	0

Tabulka 5 - úspěšnost serveru při útoku RST Flood



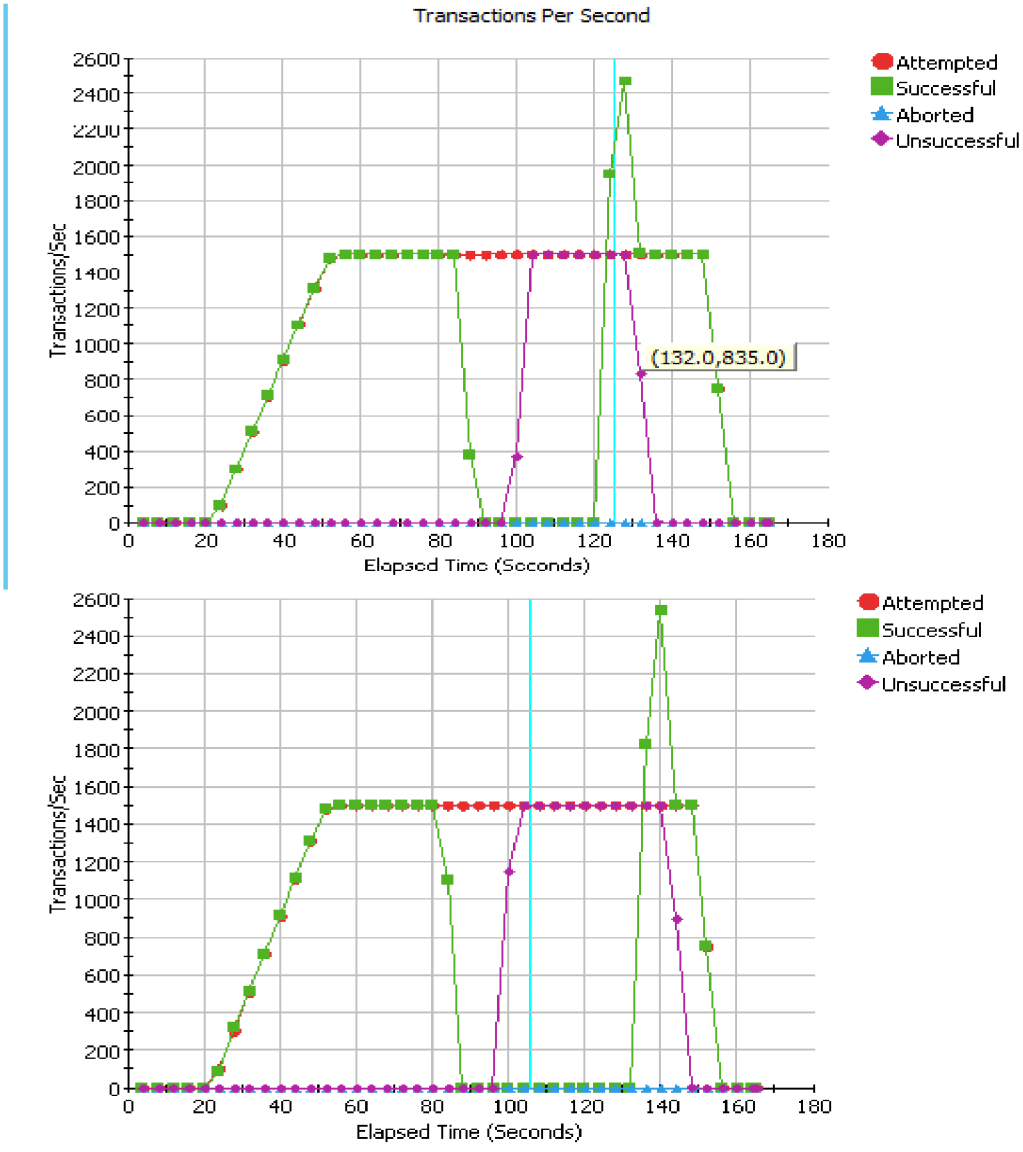
Obrázek 8 - Graf úspěšnosti serveru na počtu paketů DdoS útoku RST Flood

6.4. UDP Flood

Dalším zkoušeným útokem byl velice oblíbený UDP Flood. Při hodnotě 380000 paketů za sekundu jsme poprvé zaznamenali změnu, kdy server rovnou nedokázal obsluhovat legitimní požadavky. Proměřili jsme proto více úsek mezi 365000 a 380000 pakety. Výsledky však ukazují vždy úplnou nemožnost obsluhy. To je pravděpodobně způsobeno

nastavením našeho testování na zátěž Connection/Seconds, která pracuje na protokolu TCP. Protokol UDP využívá jiné porty, díky čemu nedochází k výpadku portů využívaných nastavenou zátěží, co by se projevovalo postupným úpadkem úspěšnosti serveru, ale k vyčerpání zdrojů serveru a nemožnosti obsluhy jako takové. Můžeme však sledovat, že výpadky nenastávají striktně v době zahájení útoku (v 70. sekundě), ale mohou se projevit až o něco později jak vidíme vyobrazeno na Obrázek 9. Můžeme vidět, že se mění také čas, v kterém dojde k obnovení spojení. Pro testy s enormní zátěží dokonce nedošlo ke zpětnému obnovení spojení za dobu trvání testu vůbec.

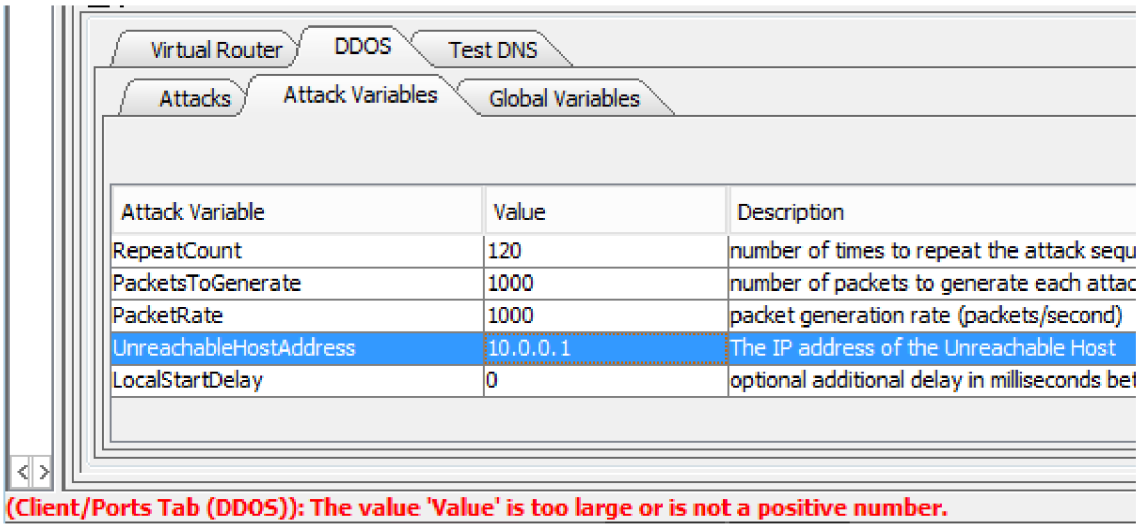
Z těchto důvodů je odečítání úspěšnosti velice nepřesné, proto jej v tomto testu nebudeme vypisovat. Důležitá je pro nás hranice 369000 DDoS paketů za sekundu, při které dochází k prvním výpadkům



Obrázek 9 - srovnání průběhů UDP Flood útoku pro 369000 (nahoře) a 400000 (dole) DDoS paketů za sekundu

6.5. UnreachableHost

Posledním vybraným útokem se stal UnreachableHost. V případě tohoto útoku jsme ovšem narazili na problém s nastavením adresy nedostupného hosta, kdy se nám při zadání jakékoli IP adresy vypsala chyba „(Client/Ports Tab (DDOS)): The value ' Value' is too large or is not a positive number.“. To se nám stalo už při pokusu vůbec upravit přednastavenou adresu, co vidíme vyobrazeno na Obrázek 10. Samotná chyba nastala ihned při zadání znaku tečky do položky Value. Jedná se tak zřejmě o špatně nastavené pole, které umožňuje zadávat jako hodnotu pouze číslo, a ne IP adresu, jak je vyžadováno. z tohoto důvodu nebylo možné požadovaný útok provést.



Obrázek 10 - zobrazení chybové hlášky při nastavování IP adresy cíle útoku

Zhodnocení:

Tímto jsme dokončili testy DDoS útoků bez aplikace IDS a IPS systémů. z výsledků je patrné, že server dokáže nejdéle odolávat útoku ARP Flood, kdy úspěšnost serveru klesá jen pozvolně.

U útoků RST Flood a SYN Flood jsou viditelné podobné křivky úspěšnosti, kdy od prvního poklesu úspěšnosti serveru nastává strmý pokles a po navýšení intenzity útoku o dalších 30000 paketů za sekundu dochází k nemožnosti obsluhy.

Specifickým testem se ukázal být UDP Flood, který vykazoval naprosto jiné průběhy útoků než všechny ostatní útoky. To by však nemělo platit v případě volby zátěže běžící na protokolu UDP.

Poslední vybraný útok se nám bohužel nepodařilo zprovoznit kvůli chybě.

7. Aplikace IDS a IPS systémů

Prvním testovaným systémem bude Snort. Po nastavení konfiguračních souborů jsme vyzkoušeli test, ve kterém jsme spustili referenční zátěž spolu se všemi funkčními útoky. Snort byl nastaven v režimu NIDS. Spuštění v tomto režimu dosáhneme příkazem

snort -c /etc/snort/snort.conf -l /var/log/snort.

Snort na začátku příkazu udává spuštění Snortu. Spojením „-c“ odkazujeme na konfigurační soubor, který má být při spuštění načten a cesta k němu je zastoupena direktivou /etc/snort/snort.conf. „-l“ udává ukládaní do logu. Cesta k tomuto logu následuje, nachází se tedy v adresáři snort.

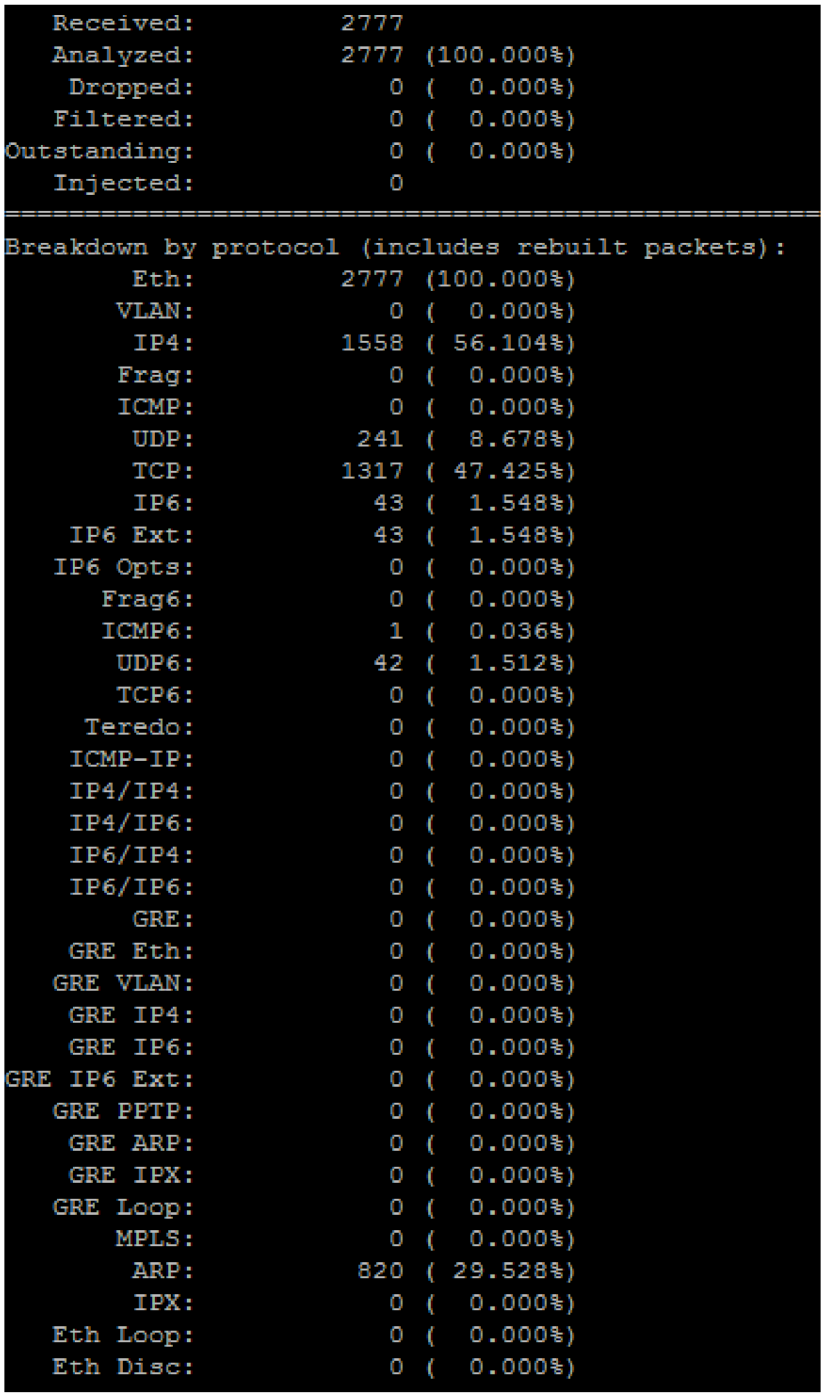
V průběhu testu byl vytvořen logovací soubor ve formátu snort.log.x, kde x označuje vygenerované desetimístné číslo. Tento soubor se dá otevřít samotným snortem při zadání

snort -r snort.log.x.

Dá se však otevřít i v jiných programech, příkladem může být třeba Wireshark. Výsledný log otevřený v programu Snort je vyobrazen je vyobrazen jako *Obrázek 11* a *Obrázek 12*.

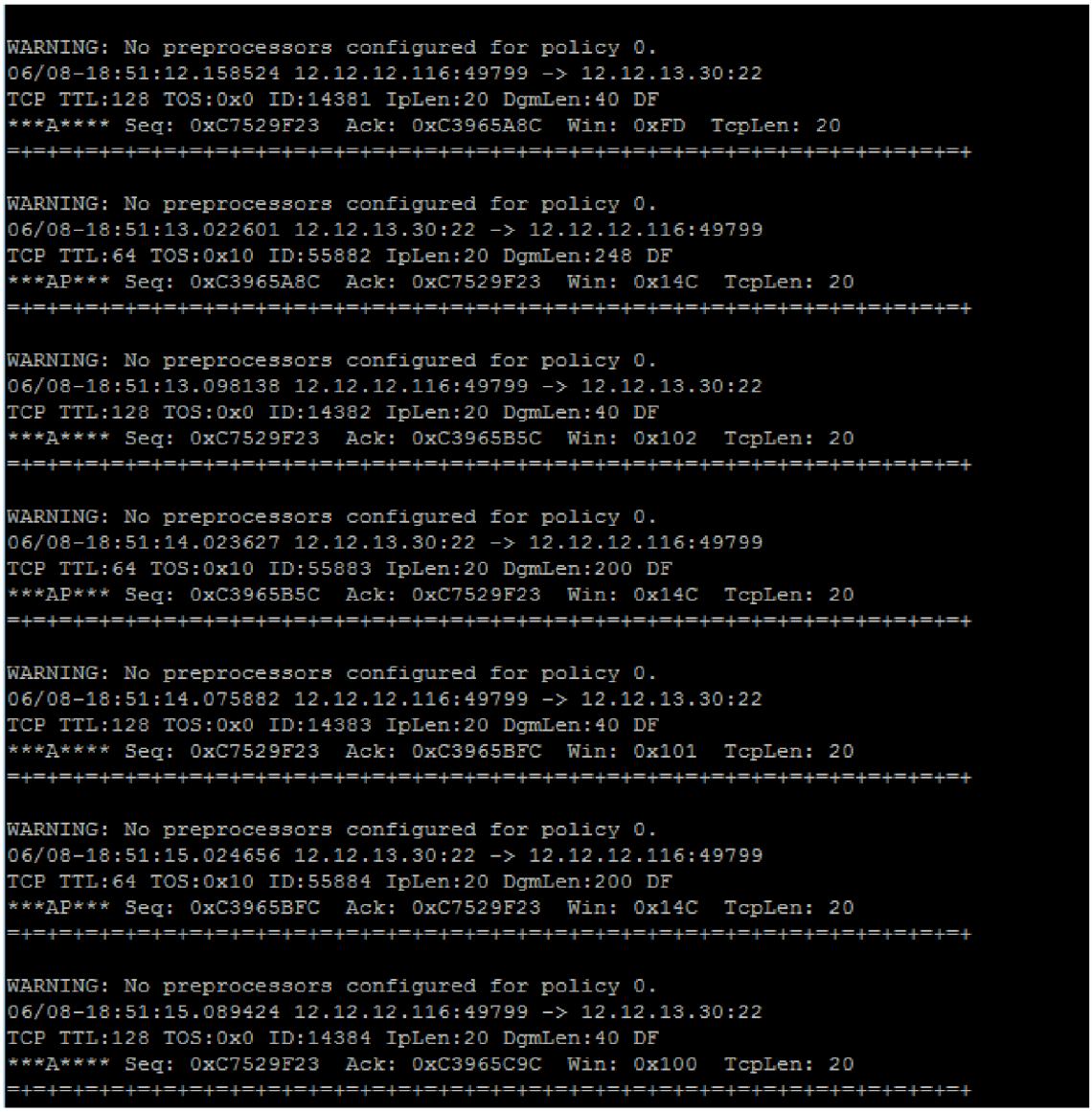
Na prvním je vidět počet zachycených paketů, kterých je 2777 a všechny byly analyzovány. Níže jsou vypsána jednotlivá rozhraní a využívané protokoly, včetně procentuálního zastoupení v komunikaci.

Druhý obrázek zobrazuje výpis některých varování, které byly během provozu zaznamenány. Neřadí se sem však hlášení „Warning, no preprocessors configured for policy 0.“, která jsou způsobena špatným nastavením konfiguračního souboru.



```
   Received:          2777
   Analyzed:          2777 (100.000%)
    Dropped:             0 (  0.000%)
   Filtered:             0 (  0.000%)
Outstanding:             0 (  0.000%)
   Injected:             0
===================================================
Breakdown by protocol (includes rebuilt packets):
        Eth:          2777 (100.000%)
       VLAN:             0 (  0.000%)
        IP4:          1558 ( 56.104%)
       Frag:             0 (  0.000%)
       ICMP:             0 (  0.000%)
        UDP:           241 (  8.678%)
        TCP:          1317 ( 47.425%)
        IP6:            43 (  1.548%)
    IP6 Ext:            43 (  1.548%)
   IP6 Opts:             0 (  0.000%)
      Frag6:             0 (  0.000%)
      ICMP6:             1 (  0.036%)
       UDP6:            42 (  1.512%)
       TCP6:             0 (  0.000%)
     Teredo:             0 (  0.000%)
    ICMP-IP:             0 (  0.000%)
    IP4/IP4:             0 (  0.000%)
    IP4/IP6:             0 (  0.000%)
    IP6/IP4:             0 (  0.000%)
    IP6/IP6:             0 (  0.000%)
        GRE:             0 (  0.000%)
    GRE Eth:             0 (  0.000%)
   GRE VLAN:             0 (  0.000%)
    GRE IP4:             0 (  0.000%)
    GRE IP6:             0 (  0.000%)
GRE IP6 Ext:             0 (  0.000%)
   GRE PPTP:             0 (  0.000%)
    GRE ARP:             0 (  0.000%)
    GRE IPX:             0 (  0.000%)
   GRE Loop:             0 (  0.000%)
       MPLS:             0 (  0.000%)
        ARP:           820 ( 29.528%)
        IPX:             0 (  0.000%)
   Eth Loop:             0 (  0.000%)
   Eth Disc:             0 (  0.000%)
```

Obrázek 11 - výpis využívaných protokolů v logu Snortu



Obrázek 12 - vypsaná varování systému Snort zachycená v logu

ZÁVĚR

Bohužel se nám nepodařilo práci doměřit včas a zajímavé je tak pro nás především porovnání útoků. Z našich výsledků můžeme vyvodit, že nejprůraznějším útokem je UDP Flood, který zaznamenal stoprocentní úspěšnost při nejnižších hodnotách zasílaných DDoS paketů za sekundu.

Útoky SYN Flood a RST Flood mají takřka stejný průběh, a navíc se projevovali při obdobné síle zasílaných paketů. Charakteristickou křivkou se nakonec prezentoval ARP Flood útok. Poslední vybraný útok se nám bohužel vůbec nepodařilo nasimulovat.

V další části jsme měli porovnat systémy Snort a Suricata. Můžeme prohlásit, že se v mnohém podobají, ale každé má svá určitá specifika, která je vyzdvihují a je tak třeba pořádně promyslet, co očekáváme od systému detekce a prevence průniku, abychom nasadili ten správný.

Zdroje:

CARL, Glenn, et al. *Denial-of-service attack-detection techniques.* IEEE Internet Computing, 2006, 10.1: 82-89.

https://www.akamai.com/us/en/multimedia/documents/social/q4-state-of-the-internet-security-spotlight-iot-rise-of-300-gbp-ddos-attacks.pdf.

https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-debian

https://www.lupa.cz/clanky/denial-of-service-dos-utoky-zaplavove-typy/

http://www.osel.cz/9379-cerv-hajime-imunizuje-elektroniku-proti-infekci-mirai.html?typ=odpoved&id_prispevku=153734

https://www.root.cz/clanky/postrehy-z-bezpecnosti-rekordni-ddos-utok-1-1-tbps/

https://suricata-ids.org/

https://www.techopedia.com/definition/24841/denial-of-service-attack-dos

MAURIC, Jakub. *IDS systém SNORT.* [online] České Budějovice: Jihočeská univerzita v Českých Budějovicích, pedagogická fakulta, katedra informatiky, 2009. Bakalářská práce. Vedoucí práce: Ing. Ladislav Beránek, CSc., MBA. Dostupné z: https://theses.cz/id/4lgemh/downloadPraceContent_adipIdno_12546.

ORKÁČ, Radomír. *IDS Snort.* [online] Ostrava: Vysoká škola báňská – Technická univerzita Ostrava, fakulta elektrotechniky a informatiky, 2006. Semestrální projekt do předmětu Směrované a přepínané sítě. Dostupné z: http://www.cs.vsb.cz/grygarek/SPS/projekty0506/Snort.pdf.

POTŮČEK, Petr. *Testování firewallů pomocí hardwarového testovacího přístroje*. [online] Brno: Masarykova univerzita, fakulta informatiky, 2012. Bakalářská práce. Vedoucí práce: doc. RNDr. Eva Hladká, Ph. D. Dostupné z:

https://is.muni.cz/th/256710/fi_b/Bakalarska_prace.pdf.

REJTA, Lukáš. *Bezpečnost síťového provozu: Přehled detekčních metod.* [online] Brno: Masarykova univerzita, fakulta informatiky, 2014. Bakalářská práce. Vedoucí práce: Mgr. Tomáš Jirsík. Dostupné z: https://is.muni.cz/th/359202/fi_b/bc.pdf.

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

ŠTĚRBA, Jan. *Možná ohrožení elektronického bankovnictví*. [online] Praha: Bankovní institut vysoká škola Praha, Katedra informatiky a kvantitativních metod, 2014. Bakalářská práce. Vedoucí práce: Ing. Antonín Vogeltanz. Dostupné z: https://is.bivs.cz/th/20376/bivs_b/Sterba_BP.txt.

ZKRATKY

ACK – Acknowledge

ACL – Access Control List

ARP – Address Resolution Protocol

CPU – Central Processing Unit

DNS – Domain Name System

DDoS – Distributed Denial of Services

DoS – Denial of Service

DDRoS – Distributed Reflection Denial of Service

HTML – HyperText Markup Language

HTTP – Hypertext Transfer Protocol

HTTPS – Hypertext Transfer Protocol Secure

ICMP – Internet Control Message Protocol

IDP – Insurance Data Processing

IDS – Intrusion Detection System

IoT – Internet of Things

IP – Internet Protocol

IPS – Intrusion Prevention System

IPSec – Internet Protocol Security

IPv6 – Internet Protocol version 6

ISO – International Organization for Standardization

MAC – Media Access Control

MTU – Maximum Transfer Unit

NIDS – Network Instrusion Detection System

NIPS – Network Instrusion Prevention System

NTP – Network Time Protocol

OISF – Open Information Security Foundation

OSI – Open Systems Interconnection

PDF – Portable Document Format

QoE – Quality of Experience

QoS – Quality of Service

RFC – Request for Comments

RIP – Routing Information Protocol

SNMP – Simple Network Management Protocol

SSDP – Simple Service Discovery Protocol

SYN – Synchronization

TCP – Transmission Control Protocol

TFTP – Trivial File Transfer Protocol

UDP – User Datagram Protocol

VPN – Virtual Private Network