Pécsi Tudományegyetem

Műszaki és Informatikai Kar

Mérnökinformatikus szak

SZAKDOLGOZAT

Hibrid hálózatok támadása és védelme, különös tekintettel a Layer 2 technikákra

Készítette: Mayer Roland István

Témavezető: Gyurák Gábor

Pécs

2020

PÉCSI TUDOMÁNYEGYETEM

MŰSZAKI ÉS INFORMATIKAI KAR **Szakdolgozat száma:**

Mérnökinformatikus Szak

T078363/2020.

SZAKDOLGOZAT FELADAT

Mayer Roland

hallgató részére

A záróvizsgát megelőzően szakdolgozatot kell benyújtania, amelynek témáját és feladatait az alábbiak szerint határozom meg:

Téma: Hibrid hálózatok támadása és védelme különös tekintettel a layer2 technikákra

Feladat:

Tervezzen meg és állítson össze egy lokális hálózatot vállalati és ipari ethernet eszközökből. Végezzen sebezhetőség vizsgálatot és valósítsa meg a rendszer elleni támadásokat (főleg L2 szinten). Monitorozza és rögzítse a támadás hálózati forgalmát majd alakítson ki védelmet a támadásokkal szemben.

A szakdolgozat készítéséért felelős tanszék: Rendszer- és Szoftvertechnológiai Tanszék

Külső konzulens: -
munkahelye: -

Témavezető: Gyurák Gábor
munkahelye: PTE-MIK Rendszer- és Szoftvertechnológiai Tanszék

Pécs, 2019. szeptember 24.

Dr. Iványi Péter
szakvezető

HALLGATÓI NYILATKOZAT

Alulírott szigorló hallgató kijelentem, hogy a szakdolgozat saját munkám eredménye. A felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Egyéb jelentősebb segítséget nem vettem igénybe. Az elkészült szakdolgozatban talált eredményeket a feladatot kiíró intézmény saját céljaira térítés nélkül felhasználhatja.

Pécs, 2019. december 29.

.....

hallgató aláírása

Köszönetnyilvánítás

Szeretnék köszönetet mondani elsősorban a családomnak, szüleimnek és testvéremnek, a támogatásuk nélkül nem jutottam volna el idáig. Annak köszönhetően, hogy a születésemtől kezdve hittek bennem és támogattak, eljuthattam idáig.

Szeretném továbbá megköszönni a fáradtságos munkáját minden tanáromnak, oktatómnak az egyetemen, különös tekintettel Gyurák Gábor tanár úrra, aki elvállalta a témavezetői szerepet a szakdolgozatomnál, továbbá szakmai utakkal, látogatásokkal hozzájárult szakmai fejlődésemhez.

Tartalomjegyzék

1. Bevezetés

Manapság már a legkisebbtől a legnagyobbig, a kiskereskedőtől kezdve a nagy multiig elképzelhetetlen egy számítógépes hálózat, kiépített rendszer nélkül a létezés. Az informatika annyira életünk részévé vált, hogy gondolkodás nélkül megbízunk a számítógépekben, rájuk bízzuk otthonunk irányítását okos otthon formájában, engedjük, hogy a pénzünk átutalását végezze, elfogadjuk, hogy a repülő, amire felülünk, annak minden rendszere az informatikától függ. Bele sem gondolunk, milyen veszélyekkel járhat, ha egy kórház berendezéseit megtámadják, ha érzékeny adataink – miket féltve őrzünk – illetéktelen kezekbe kerülnek.

Pedig az informatikai rendszereink egyáltalán nem olyan biztonságosak, mint képzeljük. A legártatlanabb szituáció, a legkisebb mulasztás is okozhat hatalmas károkat. Szakdolgozatom célja, hogy egy működő, kiépített hálózat példájával szemléltessem, milyen apró problémák tudnak nagy károkat okozni, ha egy támadó megtalálja, sőt tudatosan keresi az efféle problémákat. Természetesen a sikeres támadások után a védelmi részre is kitérek, s kijavítom a kiskapukat, melyet egy támadó ki tudna használni a hálózatban.

Fontos dolga egy mérnöknek, hogy megbízható rendszert tervezzen, adjon át, ehhez elengedhetetlen a kellően mély szakmai háttér, a precizitás, illetve egy kicsit a támadó fejével is kell gondolkodni (ezért vannak a penetration tester szakemberek).

2. Alapszituáció

A kiindulási alap az, hogy adott egy vállalatunk, aminek az egyik ágazatának a hálózatát vizsgáljuk. Maga a hálózat alapszinten van konfigurálva, a biztonságra nem különösen ügyelve, csak az alapvető funkciók működnek. Külön vannak kezelve az egyszerű dolgozók és külön a vezetők. A hálózatunk 2 db Cisco 2911-es Routerből, 1 db Phoenix Contact Switchből és 3 db Cisco Catalyst 2960-as switch. Illetve még pár darab hagyományos PC, melyek a dolgozók illetve a vezetők megfelelői, 1-1 szerver és konzolgép. Az egyik dolgozó megnyitott egy ártalmatlannak tűnő, ámde fertőzött e-mailt, aminek a megnyitásával megadja a hozzáférést a támadónak a hálózathoz. Ezután a támadó különböző Layer 2-es támadásokat fog végrehajtani, főként a switchek hibás konfigurációját kihasználva, és megpróbál adatcsomagokat, hálózati adatforgalmat elfogni, működést befolyásoló lépéseket megtenni, vagy egyszerűen a rendszer működését akadályozni. Ezek után a hibás konfigurációkat ki kell javítani, hogy

megakadályozzuk az efféle sérülékenységeket. És persze, nem elég csak elvégezni a javítást, ismét fel kell venni a hacker szürke/fekete kalapját, és újra el kell végezni a támadást, igazolva, hogy a veszélyt valóban megszüntettük, nem tudunk illetéktelenül hozzáférni semmihez.

3. A rendszer specifikálása, problémás részek áttekintése

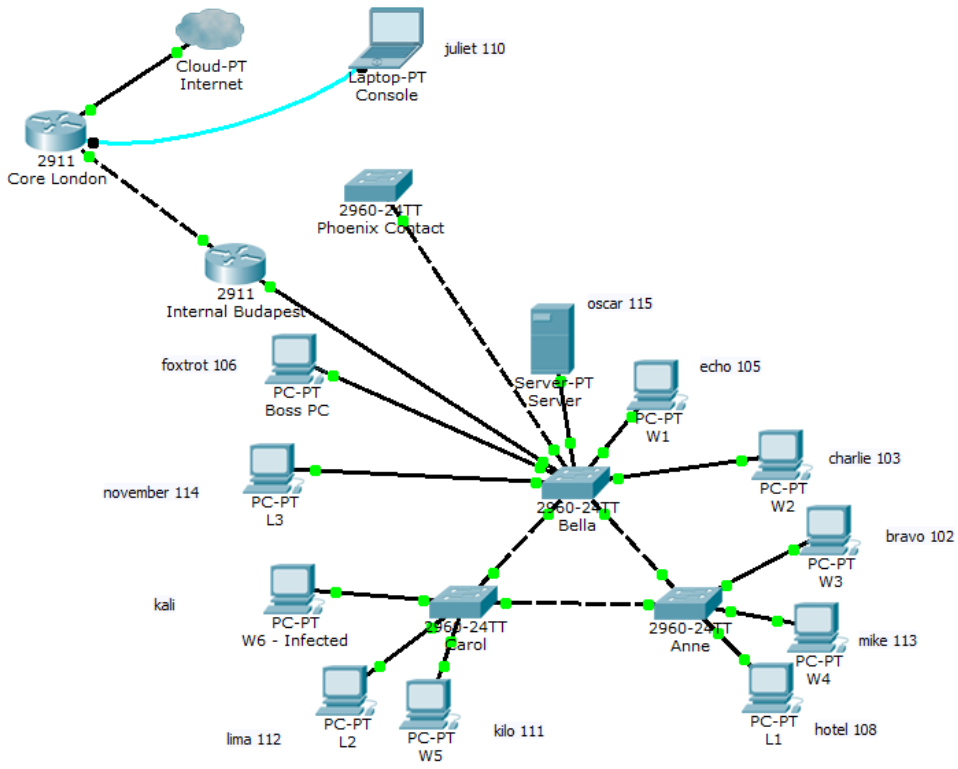
3.1 Tervezési fázis

Mint minden tervezésnél, nagyon fontos, hogy az érdemi munka előtt tekintsünk át minden lehetséges problémát, előre készüljünk fel lehetőleg az előforduló sajátosságokra, ne utólag kelljen a módosításokat végrehajtani. Éppen ebben van a mérnöki munka, hogy megtervezzük a hálózati topológiát, áttekintjük, milyen technológiákat fogunk használni, mit hova szeretnénk csatlakoztatni, így a későbbiekben ezzel már nem lesz probléma (ideális esetben). Magába foglalja ez a Packet Tracer-ben elkészült szimulációt, a topológia megtervezését, vagy az olyan egyszerűnek, és magától értetődőnek tűnő dolgokat, mint a kábelmenedzsment. Azért mégiscsak jobb, ha nem a későbbiekben, mikor már egy vállalati rendszerben akár több ezer kábel van bedugdosva, akkor jövünk rá, hogy egyszerűen már nem fér el.

A technológiai, technikai dolgokon kívül nagyon fontos, hogy a leendő felhasználókat tájékoztassuk arról, mi hogyan működik, pontosan hogyan s hova lehet eszközöket csatlakoztatni, mert mint mindig, most is az emberek a rendszer leggyengébb pontjai. Megtervezhetjük a legszuperebb hálózatot, a legmodernebb eszközökből, ha a felhasználó rossz helyre csatlakoztat valamit, és használhatatlan lesz az egész.

3.2 Topológia

A szükséges eszközöket az előző feladatban már áttekintettük (nem említve az olyan magától értetődő dolgokat, mint UTP kábel, áramellátás, stb.), a felépített hálózati topológia pedig gyűrű lesz. Vannak hátrányok ennél a megoldásnál, például hogy hurokba kerülnek a körbekapcsolt eszközök, amit külön kezelni, megoldani kell, de előnyei is, mint például a redundancia, ami kieső eszköz esetén nagy előny. Nem redundáns hálózatot tervezni elég veszélyes dolog, bármikor bármelyik eszköz meghibásodhat, ami nem redundáns hálózat esetén komoly károkat okozhat a rendszerben. Noha mindent nem old meg ez a topológia sem, ugyanúgy lehet végzetes meghibásodás, azért mégiscsak jobb, mint a semmilyen.



1. ábra A hálózat topológiája (saját kép)

3.3 Az eszközök

IP címek

A rendelkezésünkre álló címek a 192.168.3.0/24-es hálózatból állnak. Szükséges lesz IP-alhálózatokat létrehozni majd, hogy minden különböző hálózat-résznek meglegyen a saját IP tartománya. Illetve az internet felé néző interface DHCP-vel fog külső IP címet kapni.

Phoenix Contact Switch

A vállalati hálózattól különálló helyre érdemes elhelyezni, és vizsgálni az RSTP (Rapid Spanning Tree Protocol) hatásait.

Cisco Catalyst 2960 Switchek

A switchek a VLAN-ok (Virtual LAN) segítségével választjuk ketté a dolgozókat és a vezetőket. Ennek segítségével a különböző portok különböző VLAN-okhoz rendelhetőek, csak a megfelelő portba kell kapcsolni a PC-ket. Mivel a topológiában körbe kapcsolt switchek is szerepelnek, ezért az STP (Spanning Tree Protocol) használata is szükséges lesz.

Cisco 2911 Routerek

A hálózatba kötött két router, a Core és az Internal fogják ellátni a rendszer forgalomirányítását. Az Internal lesz felelős az IP címek kiosztásáért DHCP (Dynamic Host Configuration Protocol) szolgáltatáson keresztül, ami azt jelenti, hogy a hálózatra kötött PC-k IP címét nem kell manuálisan, statikusan beállítani, hanem a router kiosztja dinamikusan. Esetünkben két különálló DHCP pool lesz, egy a vezetőknek, egy pedig a dolgozóknak. A különböző VLAN-okat al-interface-k segítségével kezeli.

A Core router feladat a NAT-olás (Network Address Translation) lesz. Az Internal router irányából érkező IP címeket összefogja, és kifelé, a publikus internet irányába csak egyetlen egy IP címet fog mutatni. A belső hálózat címeit portok szerint lehet azonosítani a hálózaton kívülről.

A két router közötti kommunikációt RIP (Routing Information Protocol) valósítja meg.

Hostok

A hostokkal semmi komoly munka nem lesz, csupán az IP beállításoknál be kell állítani, hogy DHCP-n keresztül kapjon címet, illetve a tűzfalat ideiglenesen ki kell kapcsolni. Ezeken felül teljesen átlagos Windows 8.1-es operációs rendszert futtató asztali számítógépekről van szó.

A fertőzött host

Az e-mail-es vírussal megfertőződött gép egy egyébként dolgozók hálózatába csatlakoztatott gép lesz, ami az esetet leegyszerűsítve egy Kali Linux operációs rendszert futtató laptopként jelenik meg. Természetesen a valójában az egy eredeti, sima dolgozó gépnek kellene lennie, amihez az e-mail-es vírus ad hozzáférést a támadó gépéhez, ami valahol nagyon messze van a rendszertől. A Kali Linux rendelkezik a megfelelő alapfelszereltséggel ehhez, illetve rengeteg további eszköz telepíthető rá szükség esetén.

4. A lehetséges megoldások elméleti háttere

4.1 Hálózatépítés

Maga a hálózat szempontjából a tervezési fázisban a legelső lépcső, hogy tisztában kell lenni a használt protokollok, technológiák természetével, illetve tudnunk kell, pontosan hogyan, s mikor kell használni őket.

VLAN [1]

A VLAN-ok, azaz Virtuális LAN-ok azon elgondolás miatt jöttek létre, hogy az egyre növekvő hálózatokat ne feltétlen kelljen fizikailag csoportokba rendezni (hisz ez sokszor nemhogy kényelmetlen, de kivitelezhetetlen lenne), hanem ezeket a hálózatokat logikai szinten válasszuk külön. Ennek segítségével akár lehetnek különböző hálózatokhoz csatlakoztatott hostok is egy helyen, a VLAN által definiált hálózati logikát megvalósító eszköznek (általában switch) hála a fizikai elrendezéstől függetlenül is szeparálhatóak a különböző LAN-okhoz tartozó eszközök. A VLAN-t megvalósító eszközön egy konfigurációs táblázat segítségével lehet eligazodni a különböző hálózatok között, ami megjegyzi, hogy melyik portot melyik hálózathoz adtuk hozzá. Egy egyszerű példa, mely képes bemutatni a hálózati topológiámon is a VLAN-ok hatalmas előnyét: ha az egyik dolgozót előléptetik vezetővé, akkor ki sem kell költöznie a már szépen belakott irodai környezetéből, hanem „csak” le kell választani az eddigi dolgozó hálózatról, és át kell kapcsolni a másik hálózat számára fenntartott portokba. Ez esetben ez annyit tesz, hogy a rendszergazda elsétál a switch-hez, amire jelenleg csatlakoztatva van, kihúzza a vezetéket, majd átdugja a switch egy másik portjába, ami már a másik VLAN-hoz tartozik.

IEEE 802.1Q [1]

A hálózatok ki vannak építve a VLAN-ok segítségével, megfelelően el vannak szeparálva, viszont még fennáll egy probléma: a különálló hálózatokat összekötő hidak nem tudják az alap Ethernet keretből megállapítani, hogy az éppen érkező csomag pontosan melyik VLAN-ból származik. Ez elég problémás, hiszen e nélkül a különböző hálózatokat nem tudják megkülönböztetni egymástól a hálózatok közti hidak, így pont azt a funkciót veszti el a hálózat, ami miatt létrehozták. Ennek a problémának az áthidalására született meg az IEEE 802.1Q szabvány, amit leginkább dot1Q néven emlegetnek. Ez a végpontokban nem jelenik meg,

csupán a hálózati hidak közötti kommunikációhoz beszúr egy VLAN címkét az eredeti Ethernet keretbe, ezáltal azonosítva, pontosan „ki honnan származik”. Szokás trunkölő- protokollként is hivatkozni rá, hiszen magát a trönkölést segít megvalósítani.

IEEE 802.1D [2]

Az IEEE 802.1D, azaz az STP (Spanning Tree Protocol) szükséges ahhoz, hogy létre tudjunk hozni hurokmentes hálózatot akkor, ha körbe vannak kapcsolva egyes eszközeink a hálózati topológia szerint. Esetünkben ez annyit tesz, hogy az 1. ábrán látható módon kapcsolt switchek közt megszüntetjük a hurkokat, ezáltal a kommunikáció zavartalanul folyhat a huroktól függetlenül. A hálózatban minden VLAN-nak „meg kell mondani”, hogy használhatja a protokollt. Kitüntetett szerepe van a root bridge-nek, amelyen keresztül zajlik a kommunikáció nagy része. Mindig a legkisebb prioritású eszköz a root. Nagyon fontos protokollról van szó, hisz a körbekapcsolással keletkező mizériákat nem véletlenül vállaljuk be, a gyűrű topológia igényli az STP protokollt, de cserébe redundáns lehet a rendszerünk, így nem okoz (megoldhatatlan) gondot, ha egy eszköz meghibásodik.

IEEE 802.1W [3]

Az IEEE 802.1W protokoll az előző, IEEE 802.1D továbbfejlesztése. Ismertebb neve az RSTP (Rapid Spanning Tree Protocol), hatalmas előnye a sima STP-hez képest a sebessége. Míg a hagyományos STP-nek körülbelül fél perc szükségeltetik a működés megkezdéséhez egy indításnál, addig az RSTP-nek elég csupán századmásodpercekben mérhető idő. Visszafele kompatibilis az STP-vel.

DHCP [1]

A Dynamic Host Configuration Protocol használatával nem kell manuálisan IP címet adni a hostoknak, hiszen a DHCP-kiszolgáló oszt neki egyet (a konfigurációtól függ, hogy milyet, természetesen). Mára szinte teljesen kiszorította az elődjeit (RARP, BOOTP). Nagy előnye, hogy minimális konfigurációt igényel, és onnantól a használata egy radio button átállítás a hostokon.

RIP [4]

A Routing Information Protocol egy távolságvektor alapú forgalomirányító protokoll, segítségével a routerek közötti kommunikáció valósítható meg. Nagy előnye, hogy roppant

egyszerű használni, illetve konfigurálni. A hálózat jóságát, azaz metrikáját az alapján méri, hogy hány routeren áthaladva éri el az adott csomag a célt. A maximális elfogadott metrika az 15, efölött már nem képes kezelni a csomagokat.

NAT/PAT

Network Address Translation, melynek célja, hogy átmeneti megoldást nyújtson az IPv4-es címek elfogyására. Segítségével az internetre csatlakoztatott hálózat csupán egy címet használ el, és a belső hálózat védettebb. Két fajtája van, az egyik a statikus NAT, amikor 1 belső IP-hez 1 külső IP tartozik, illetve a dinamikus NAT, amikor N db belső IP-hez tartozik egy M db külső IP-t tartalmazó halmaz egyike.

A Port Address Translation a dinamikus NAT egyik változata, ebben az esetben a külső hálózat egy adott IP címen kommunikál a belsővel, viszont a belső hálózat tagjait portokkal azonosítja. Két változata van ennek is, az egyik, amikor N db belső címet 1 db külső fog össze, illetve amikor ezt az N címet M db külső IP fogja össze.

Nagy hátránya, hogy a külső irányból érkező kommunikáció nehézkes, sokszor nem is lehetséges a hálózattal.

4.2 Támadások

A szakdolgozat témája a Layer 2-re összpontosul, így azon támadások élveznek prioritást. A következőekben felsorolom, és pár mondatban bemutatom a switchek elleni legnépszerűbb/legismertebb támadási formákat, amelyek közül prezentálva is lesz néhány (természetesen a védekezéssel együtt). [4]

CAM table overflow

A Content-Address Memory tárolja a switcheken, hogy melyik porttal melyik MAC-cím kommunikál. Ám ennek a táblának véges a kapacitása, így ha elárasztjuk rengeteg idegen címmel a switchet, akkor idejekorán be fog telni, s ha a telített táblára egy új MAC- cím érkezne, akkor a switch azt már nem tudja kezelni, és lényegében hub-ként funkcionálva elküldi minden portján a csomagot. Ha támadóként ilyenkor egy lehallgató készüléket, úgynevezett sniffert helyezünk el a hálózatban, akkor lehallgatható az adatforgalom.

VLAN double tagging

Ebben az esetben mindkét (vagy több) VLAN fejlécével ellátjuk a küldendő csomagot, így ebben az esetben a VLAN-ok szépen egyesével „lefejtik” a saját fejlécüket, magukra ismernek benne, és továbbítják a kért útvonalon a csomagokat, végül eljut a támadott géphez a csomag.

STP manipulating

Ez a támadás az STP protokollt próbálja megtámadni, olyan módon, hogy a hálózatban olyan alacsony prioritásértéket hirdet magáról, ami miatt a hálózat azt hiszi, hogy ő a root bridge, ezáltal a hálózati forgalom nagy része keresztülfut rajta. Itt is sniffer elhelyezésével lehallgatható a hálózati kommunikáció.

Rouge DHCP

Ebben az esetben azt próbálja meg elhitetni magáról a támadó, hogy ő a DHCP szerver, aki osztja a hostok IP címeit, s beállítja a saját IP címét az alapértelmezett átjárónak. Ennek okán a hálózati forgalom tetszőlegesen befolyásolható, lehallgatható.

DHCP starvation

Hasonló támadás, mint a CAM table overflow, olyan értelemben, hogy itt is a rendelkezésre álló „készletek felélése” a cél. Különböző MAC címekről DHCP kérést intézünk a szerver felé, egészen addig, amíg el nem fogynak a kiosztható IP címek, ezáltal a DHCP szerver funkcionálisan leáll, és nem tudja kiszolgálni a hálózatot.

ARP poisoning

Az ARP (Address Resolution Protocol) feladata, hogy adott IP címekhez a hozzájuk tartozó MAC címeket rendezi egy táblázatba. Ha ebbe a táblázatba sikerül becsempészni egy „nem valós” IP cím - MAC cím párost, akkor a forgalom eltéríthető, lehallgatható.

4.3 Wireshark

A Wireshark egy ingyenes, bárki számára elérhető protokoll-elemző szoftver. Segítségével fogom követni a DHCP kéréseket a támadásoknál, illetve megfigyelni, konkrétan mikor milyen forgalom halad át a hálózaton.

4.4 Yersinia

A Yersinia egy Kali Linuxon futó, Layer 2-es támadásokra specializált szoftver. Rengeteg támadás kivitelezésére jó, például DHCP támadások, 801.1Q és STP protokollok megtámadása. Nevét egy baktériumnemzetségről kapta, ide tartozik a Yersinia pestis is, ami utal arra, hogy a szoftver a „fertőzésre" van kitalálva.

4.5 Védekezés

A gyakorlati részben részletesen ki lesz fejtve mindegyik végrehajtott támadásforma ellen a szükséges védekezés, így itt csak röviden beszélnék róla. A „rendelkezésre álló kapacitást felélő" támadások ellen megfelelő védelmet nyújt a portsecurity, aminek segítségével megmondhatjuk az eszköznek, hogy adott porton maximálisan hányféle MAC címet engedjen át. A Rouge DHCP szerver ellen optimális választás a DHCP snooping, ami megakadályozza, hogy válaszüzenetek érkezzenek az idegen DHCP szervertől, így maga a DHCP kérés el fog jutni a támadóhoz, de IP címet nem fog tudni osztani. [2]

5. A hálózat kiépítése

5.1 Hálózat

A 3.1-es fejezetben bemutatott módon a hálózatot ki kell építeni. A megépítésnél fontos szempont, hogy a jövőben bármikor bővíthető legyen, minimális konfiguráció-módosítással. A portok részletes leírása helyett annyit írnék le, pontosan melyek lesznek használva az adott eszközöknél.

Hostok: hagyományos Ethernet port

Switch-switch kapcsolat: Fast Ethernet 0/21-24 portok

Switch-host kapcsolat: Fast Ethernet 0/1-20 portok

Switch-router és router-router kapcsolat: Gigabit Ethernet 0/0-2 portok.

5.2 IP címek

A rendelkezésünkre álló 192.168.3.0/24 hálózatból létre kell hoznunk alhálózatokat. Adódik a kérdés, hogy pontosan hányat, s ehhez előbb vetnünk kell egy pillantást a topológiára. Az biztos, hogy a két különálló csoportnak, a dolgozónak és a vezetőnek szükséges egy-egy külön alhálózat. Szüksége van egy hálózatra a két router (Intelnal-Core) közötti kapcsolatnak, ez így már három. Viszont kettő fontos port nem kapott még ezzel IP címet, mégpedig a Core router publikus internet felé levő interface, illetve az Internal routeren a switchek felé levő inteface. Az egyszerűség kedvéért az egész hálózatot felosztottam egyenlő, /28-es részekre. Ebből lettek a következő, használt hálózatok:

192.168.3.0/28 – Internal router belső interface-éhez

192.168.3.16/28 – dolgozók alhálózatához

192.168.3.32/28 – vezetők alhálózatához

192.168.3.48/28 – Internal-Core kapcsolathoz

Külső DHCP által osztott cím – Core-publikus internet kapcsolathoz

Íratlan szabály, hogy minden hálózat legelső (vagy a legutolsó) kioszthtató IP címét a routerek kapják, így noha kevés esetben lesz gyakorlati jelentősége, mégis, ehhez szeretném én is tartani magam az egyszerűbb érthetőség miatt. Illetve, amit fontos tudni, hogy minden hálózat utolsó címe a broadcast cím, ami azt jelenti, hogy minden hálózaton levő eszköz megkapja az arra címzett üzeneteket. Így mindegyik alhálózatra marad 14 db IP cím, amit megkaphatnak a hostok. Természetesen, ha ennél nagyobb gépmennyiséget kell csatlakoztatnunk, akkor másképpen kell beosztani a rendelkezésünkre álló hálózatot, de mivel (bőven) nem fogjuk kihasználni a többi rendelkezésre álló címet, ezért a szakdolgozat céljának megfelel, illetve még itt is meglesz a lehetőségünk, hogy több hostot csatlakoztassunk a hálózathoz. A dolgozók és a vezetők alhálózat az Internal router DHCP szolgáltatásától fogják kapni a címeket, míg a többi esetben statikusan kerülnek majd meghatározásra, kivéve az Internal Router publikus internet felé eső interface-ét, ami szintén DHCP által fog IP-t kapni (de nem a Core Router szolgáltatásától természetesen)

5.3 Switchek konfigurálása

Mivel a három körbe kapcsolt switchen néhány apróság kivételével ugyanaz a konfiguráció, ezért az egyiken (a Bellán) szeretném bemutatni az egész konfigurációt, ami érvényes lesz a Bella mellett a Carol illetve a Dorothy switchekre. Természetesen, ami eltérő, vagy plusz konfigurációt igényel, azt az alfejezet végén, de be fogom mutatni.

Először is, mikor elérjük a CLI-t, beállítjuk a switch nevét. Ehhez először be kell lépni a privilegizált módba, majd pedig belépni a konfigurációs terminálba. Itt a hostname megváltoztatásával át is neveztük a switchet.

```
Switch>ena
Switch#conf t
Switch(config) # hostname Bella
```

Ezután létre kell hozni a vlanokat:

```
Bella(config)#vlan 10
Bella(config-vlan)#name worker
Bella(config-vlan)#exit
Bella(config)#vlan 20
Bella(config-vlan)#name leader
Bella(config-vlan)#exit
```

Most, hogy már létrejött a kettő VLAN, hozzá kell rendelni a portokat. Alapjáraton minden switchen van egy 1-es ID-vel rendelkező VLAN, ahhoz van hozzárendelve az összes port. Ha hozzárendeljük a portokat a megfelelő hálózathoz, akkor a csatlakoztatás után már maguktól tudni fogják a hostok, melyik VLAN-hoz tartoznak. Az 5.1-es fejezetben írtam, hogy a switch-host kapcsolatokra a sorszámozás szerinti első 20 Fast Ethernet porton szánom. Nos, ezt még kétfelé kell osztani, így switchenként 10 dolgozó és 10 vezető csatlakoztatható. Emiatt lett az első 10 port a dolgozóké, a második 10 pedig a vezetőké, amiatt, mert a 10-es VLAN kapja a portokat, amiben szerepel a 10, a 20-as pedig, amiben a 20. A portokat access módba is kell állítani, hogy tudja a switch, hogy onnan hostoktól jöhet információ.

```
Bella(config)#int range fa0/1-10
Bella(config-if-range)#switchport mode access
Bella(config-if-range)#switchport access vlan 10
Bella(config-if-range)#exit
Bella(config)#int range fa0/11-20
Bella(config-if-range)#switchport mode access
```

```
Bella(config-if-range)#switchport access vlan 20
Bella(config-if-range)#exit
```

Ezen parancsok után a privilegizált módban kiadott *show vlan brief* parancs megmutatja, hogy immár három VLAN-nal rendelkezünk, az eredeti 1-es azonosítójú mellett megjelent a 10-es és a 20-as is, és felsorolta, melyikhez melyik Fast Ethernet portok tartoznak. 1-10 között dolgozó, 11-20 között vezető, 21-24 közt pedig az eredeti 1-es azonosítójú hálózat látható.

Következő lépésben be kell állítani, hogy a switch tudja kezelni azokat az összeköttetéseket, amelyek nem a hostokkal létesülnek, ezt pedig a trunköléssel oldjuk meg. Ugye nemrégiben a hostokkal kapcsolatba lépő portokat access módba állítottuk, most ugyanarra a parancsra lesz szükségünk szinte, csak access helyett trunk használatával.

```
Bella(config)#int range fa0/21-24
Bella(config-if-range)#switchport mode trunk
Bella(config-if-range)#exit
```

A *show interfaces trunk* paranccsal meg tudjuk tekinteni, mely portokat állítottuk trunk módba. Ennek köszönhetően a router-switch és a switch-switch kommunikáció is működőképessé válik. Ezzel a switchek konfigurációja majdnem teljes egészében megvan, már csak az az apróság hiányzik, hogy a körbekapcsolásból fakadó loop-ot megszüntessük, ehhez pedig a Spanning Tree Protocol használata a szükséges. Mindegyik routeren be kell állítani, hogy minden megadott VLAN esetén működjön. Ehhez a *spanning-tree vlan 10 root primary* parancsot kell kiadni, illetve ugyanezt a 10 helyén 20-al. Ugyanezeket a lépéseket elvégezve a Carol és Dorothy switcheken is sikerrel „üzembe helyeztük” a switchparkunkat.

5.4 Port Mirroring

Be lehet állítani egy olyan funkciót a switcheken, hogy monitorozza az adatforgalmat, és elküldje egy adott porton elhelyezett készülékre. A belső hálózat szélén érné meg egy ilyen eszközt telepíteni. A konfigurációja egyáltalán nem bonyolult: [5]

```
Anne(config)#monitor session 1 source int fa0/22
Anne(config)#monitor session 1 source int fa0/23
Anne(config)#monitor session 1 source int fa0/24
Anne(config)#monitor session 1 destination int fa0/21
```

Az első három sorral megmondtuk, melyik portokról érkező forgalmat szeretnénk figyelni, az utolsó sor pedig azt mutatja meg, melyik portra szeretnénk irányítani a monitorozás eredményét (aminek a végén lehetőleg valami szerver van, ami képes fogadni az adatokat).

5.5 Internal router

A belső router a Gigabit Ethernet 0/0 portján csatlakozik rá a switchek által alkotott hálózatra. Ez a port a hálózatszámítási feladatok elvégzése után statikusan megkapta a 192.168.3.1/28-es IP címet. Ehhez a konfigurációs terminálból egyszerűen be kell lépni az adott port (interface) beállítási részébe (*int gig0/0*) majd hozzá kell rendelni a fentebbi címet (*ip address 192.168.3.1 255.255.255.240*). Ez utóbbi a maszkját jelenti a hálózatnak, amit már /29-el írtam korábban, azt így is ki lehet fejezni, sőt, a Cisco IOS, ami ezeken a rendszereken fut, csak így tudja kezelni.

Miután a switcheket már „betanítottuk” az általunk létrehozott VLAN-ok ismeretére, ugyanezt meg kell tennünk a routerek esetében is, hiszen ha ezt nem tesszük meg, akkor az érkező illetve küldendő csomagokban nem tud különbséget tenni a router, és olyan lenne, mintha a VLAN-ok nem is léteznének. Lássuk, hogyan történik mindez.

```
Internal(config)#int gig0/0.10
Internal(config-subif)#encapsulation dot1q 10
Internal(config-subif)#ip address 192.168.3.17 255.255.255.240
Internal(config-subif)#exit

Internal(config)#int gig0/0.20
Internal(config-subif)#encapsulation dot1q 20
Internal(config-subif)#ip address 192.168.3.33 255.255.255.240
Internal(config-subif)#exit
```

A legelső sor azt jelenti, hogy lépjünk be a Gigabit Ethernet 0/0 10-es azonosítóval rendelkező al-interface-ére, azaz a 10-es ID-vel bíró VLAN-ra. A második sorral beállítottuk a korábban már említett 802.1Q protokollt, ami lehetővé teszi, hogy megkülönböztethetőek legyenek a VLAN-ok. A dot1Q mögötti szám jelenti, hogy melyik VLAN-ra kell alkalmazni a protokollt.

Ezek után csupán beállítottunk egy /28-as IP címet, méghozzá azok közül az elsőt, amit az adott – dolgozó vagy vezető – hálózat számára tartottunk fenn (ne feledjük, az első IP cím a routeré). Ugyanezen lépéseket kellett végrehajtanunk a 20-as azonosítójú hálózattal is, ott is 802.1Q bekapcsolása, első IP cím kiosztása.

Következő feladatunk nem más, mint a DHCP szolgáltatás létrehozása, s beállítása, hogy a hostok kaphassanak IP címeket.

```
Internal(config)#ip dhcp pool worker
Internal(dhcp-config)#network 192.168.3.16 255.255.255.240
Internal(dhcp-config)#default-router 192.168.3.17
Internal(dhcp-config)#dns-server 1.1.1.1
Internal(config)#ip dhcp pool leader
Internal(dhcp-config)#network 192.168.3.32 255.255.255.240
Internal(dhcp-config)#default-router 192.168.3.33
Internal(dhcp-config)#dns-server 1.1.1.1
```

Először be kell lépnünk az adott elnevezésű DHCP poolba, mert ez a protokoll ilyenek szerint kezeli a különböző hálózatok szolgáltatását. Egy eszközön több pool is létrehozható, esetünkben kettőre van szükség, egy a dolgozó és egy a vezető hálózatnak. A fentebb már kiszámolt alhálózatot hozzárendeljük a poolhoz, ezáltal tudtára adjuk a DHCP-nek, miből válogathat a címosztás során. Default routernek mindkét csoportban megadjuk a számukra nemrég létrehozott al-interface IP címét, így minden csomag tudja, végső soron merre kell menni. DNS szervernek alapértelmezetten az 1.1.1.1-et adtam meg. Ezután ha a hostokat statikus IP címről DHCP-re állítjuk, akkor automatikusan meg kell kapniuk a megfelelő címeket.

Ezzel a routernek a hálózat felé „néző” oldalával végeztünk, már csak azt kell beállítanunk, miképp történjen meg az információcsere a Core és az Internal routerek között. Ehhez én a RIP protokollt választottam az egyszerűsége végett.

```
Internal(config)#router rip
Internal(config-router)#network 192.168.3.0
Internal(config-router)#version 2
Internal(config-router)#no auto-summary
```

Ez esetben csak meg kellett adni a hálózatot, amiben a kommunikációt szeretnénk megvalósítani. Viszont, mivel szükségünk van az alhálózatok megfelelő kommunikációjára,

ezért a RIP 2-es verzióját be kell kapcsolnunk, ami támogatja az alhálózatokat. Az alatta látható *no auto-summary* parancs pedig amiatt fontos, hogy a különálló hálózatokat ne mossa össze a rendszer (ez logikus, hiszen épp amiatt hoztuk létre a különálló hálózatainkat, mert egyben nem elégítik ki a szükségeinket).

5.6 Core router

Ezzel a routerrel már kevesebb dolgunk lesz, ugyanis csak egy újdonságot kell beépíteni. Természetesen az Internal-Core közötti kapcsolatot ennél a routernél is el kell látni a RIP protokollal, ez teljesen ugyanúgy történik, mint az előző esetben, így nem írnám le ismét részletesen. Két dolog van, amit még a hálózatunk nem tud. Az egyik, hogy a specifikációban szereplő módon nincsen (lényegi) különbség a két fő alhálózat között, illetve az, hogy senki nem éri el a publikus internetet. Ehhez PAT-ot alkalmaznék access-listekkel (ACL) [6].

Az ACL egy engedélyezési lista, lényegében tűzfalként tud funkcionálni. Megadhatunk neki IP címet, illetve IP tartományt is, és azokat külön engedélyezhetjük, vagy pedig tilthatjuk. Azt szeretnénk ugye elérni, hogy a vezetők csoport elérje a publikus internetet is, míg a dolgozók csak a saját belső hálózatukon keresztül kommunikálhassanak. Tehát az biztos, hogy az ACL-ben engedélyezni szeretnénk a vezetők alhálózatát, viszont mivel másnak nem szeretnénk hozzáférést adni a publikus internethez, ezért nem kell külön minden más hálózatot letiltanunk, hanem elég lesz egyszerre tiltólistára tenni az átengedett listán kívüli összes IP címet. Nézzük, hogy néz ez ki a gyakorlatban:

```
Core(config)#access-list 1 deny any
Core(config)#access-list 1 permit 192.168.3.32 0.0.0.15
Core(config)#access-list 1 permit 192.168.3.49 0.0.0.15
Core(config)#access-list 1 permit 192.168.3.1 0.0.0.0
Core(config)#access-list 1 permit 192.168.3.17 0.0.0.0
Core(config)#ip nat inside source list 1 int gig0/0 overload
```

Az az 1-es azt jelenti, hogy egy standard ACL-t hoztunk létre, 1-es azonosítóval (1-99 között automatikusan standard listát hozunk létre, 100-199 között pedig kiterjesztett listát használhatnánk. Ez utóbbiak sokkal jobban paraméterezhetőek, de számunkra most megfelel a standard lista is). Az első sorral elvégezzük a tiltást, a második sorral pedig megadjuk azon címeket, melyeket szeretnénk átengedni a szűrésen. Ezek a vezető csoport IP címei, illetve

mellé célszerű átengedni a routerek IP címeit is az ACL-en. Fontos, hogy az eddigiektől eltérően nem a hagyományos IP maszkot kell alkalmazni, hanem annak a negáltját, azaz 255.255.255.240 helyett 0.0.0.15-t kell írni. Oda kell figyelni a listák helyes leírására, ugyanis a konfigurációs fájlban olyan sorrendben fog megjelenni az engedélyezés/tiltás, amilyen sorrendben a parancsokat kiadtuk, és esetleges újabb parancsok nem írják teljesen felül a korábbiakat, hanem simán bekerülnek a sorba. Ezért kell odafigyelni, hogy ha esetleg változás történik a listánk felépítésében, akkor a kiadott parancs elé illesztett *no* szócskával kivehetjük a már nem szükséges listát a sorból. Miután megvagyunk az ACL létrehozásával és felparaméterezésével, magát a NAT/PAT-olást kell elvégezni az utolsó sorral, ami annyit mond, hogy az általunk létrehozott ACL alapján végezze a beállítást, a Gigabit Ethernet 0/0 interface-re (a publikus internet felé nézőre) „terhelje rá" az IP címeket. Ha ezzel is megvagyunk, a hálózat összefogása majdnem teljes egészében megtörtént, már csak azt kell megmondanunk a routernek, hogy melyik interface néz a hálózat felé, és melyik az internet felé.

```
Core(config)#int gig0/0
Core(config-if)#ip nat outside

Core(config)#int gig0/1
Core(config-if)#ip nat inside
```

Ezután a router már tisztában lesz azzal, hogy honnan jövő címeket kell összefognia, és merre kell ezt mutatnia.

5.7 Pár szó a vállalati switchről

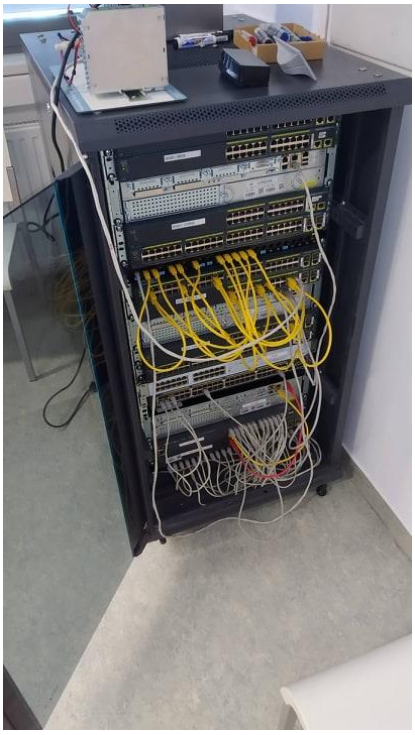
A vállalati switchet értelemszerűen egy vállalati hálózatban használjuk. Sokszor nincsen elég időnk arra, hogy a hálózaton másodperceket várjunk, amíg használható lesz a rendszer, hanem szinte azonnal, a lehető leggyorsabban szükségünk van stabilan működő szolgáltatásokra. Volt már szó az STP-ről, viszont annak egy kisebb hátránya, hogy noha a körbekapcsolt hálózatok problémáját megoldja, körülbelül 35-40 másodperc szükséges neki a stabil felálláshoz. Néha ez nem megengedhető késés, ennek a megoldására született az RSTP (Rapid Spanning Tree Protocol) [7]. Ennek a stabil működéséhez bőven kevesebb, mint egy másodpercre van szükség. Mondanom sem kell, mennyivel nagyobb könnyebbség ez egy olyan hálózatban, ahol nem fér

bele az a körülbelül fél percnyi várakozási idő. Az RSTP úgy lett megalkotva, hogy a „sima” STP-vel visszafele kompatibilis legyen. Hivatalos nevén IEEE 802.1W néven fut.

6. Az első támadás

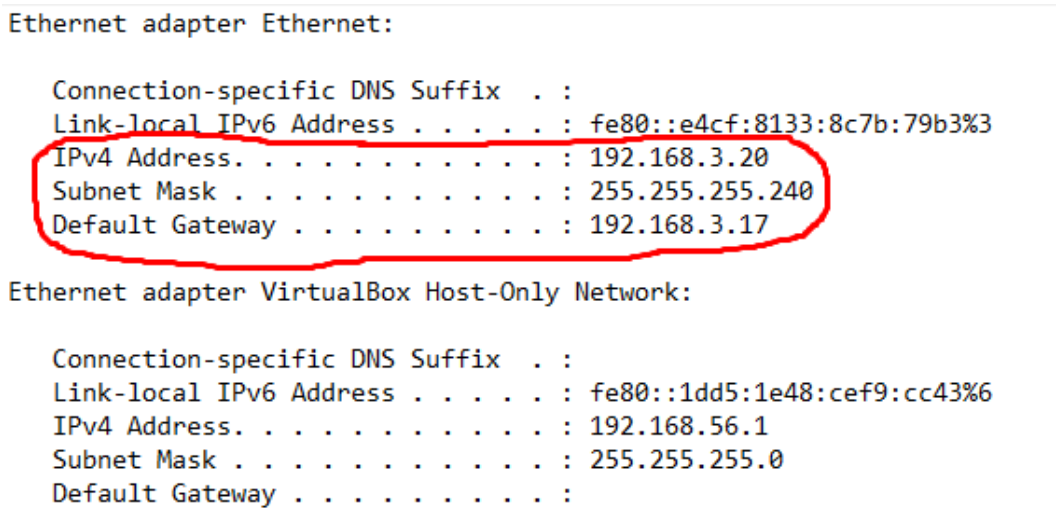
6.1 Előkészületek

A Yersinia egy penetration tester program, ami előre telepítve van Kali Linuxra, főleg Layer 2-es támadásokra specializálódva. Segítségével kivitelezhető a legtöbb kívánt támadásforma. A támadás úgy fog kinézni röviden, hogy a kiépített, felkonfigurált hálózatba csatlakoztatott Kali Linuxos laptop elindítja a támadást. Eközben elindítok egy Wireshark nevű protokoll elemző szoftvert – szintén a Kalin – amivel figyelni fogom a hálózat adatforgalmát, majd, miután a támadások megtörténnek, megpróbálom a hálózat egyes beépített rendszereit, szolgáltatásait használni, például kérni egy IP címet DHCP-vel, küldeni egyszerű csomagokat, amiket ugye a Wiresharkon át látható lesz, mit okozott egy-egy támadás a rendszerben.



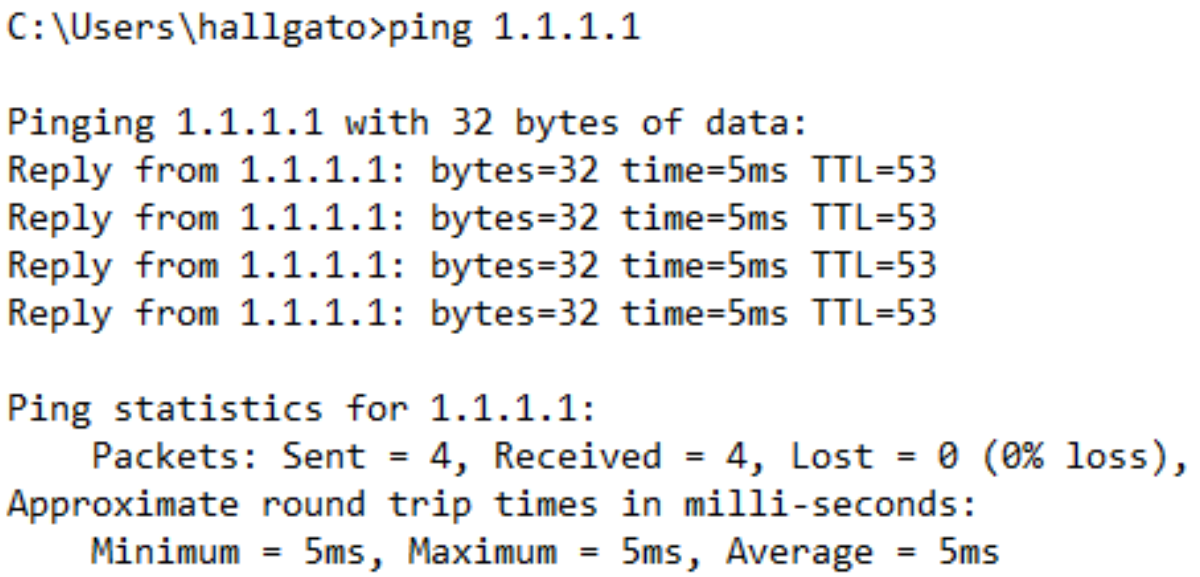
2. ábra A bekötött rack szekrény

A legfontosabb, hogy a hálózat eredeti, támadás előtti állapotában is működőképes, csak éppen nem túl biztonságos. Miután leteszteltem, hogy az ACL azt tiltja, amit kell, és azt engedi át, amit át kell engednie, megpróbálhatom a támadásokat kivitelezni.



3. ábra Egy dolgozó gép a DHCP kérés után - látható, hogy minden a poolban beállítottak szerint történt

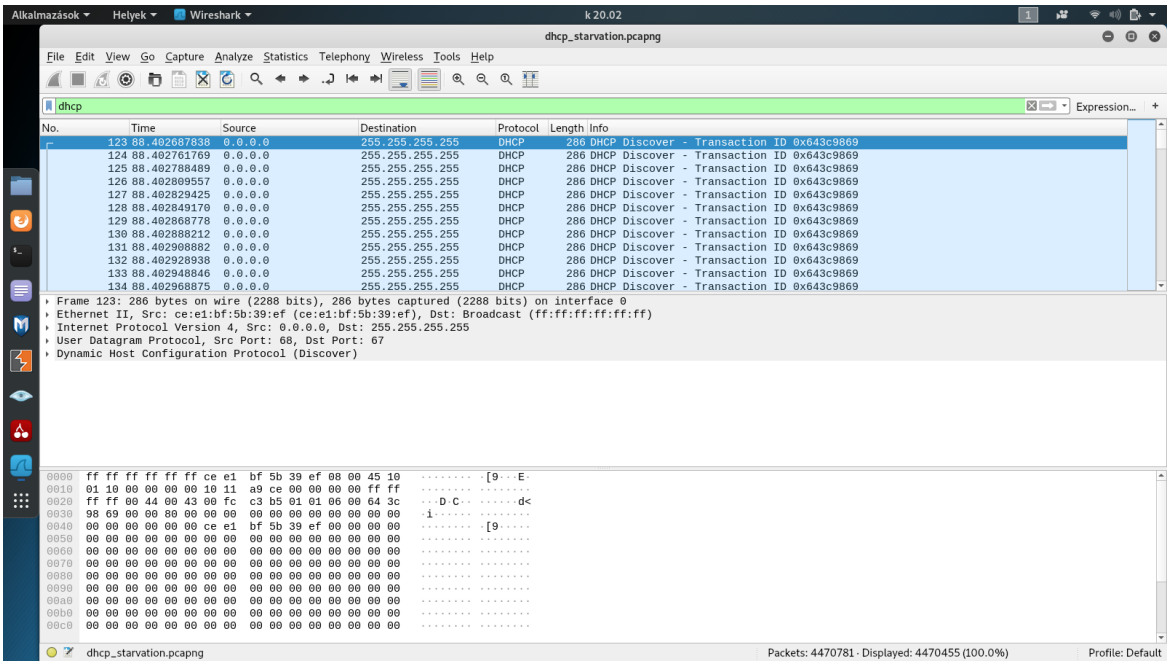
Mivel a Core routeren csatlakozunk a publikus internetre, az ACL-ek be vannak állítva, ezért sikerülni kell pingelésnek, vagy csak egyszerűen internetes weblapok elérésének, amit bizonyít az alábbi fotó:



4. ábra 1.1.1.1 megpingelése, jól látszik, hogy mind a 4 csomagunk megérkezett sikeresen

6.2 Wireshark beállítás

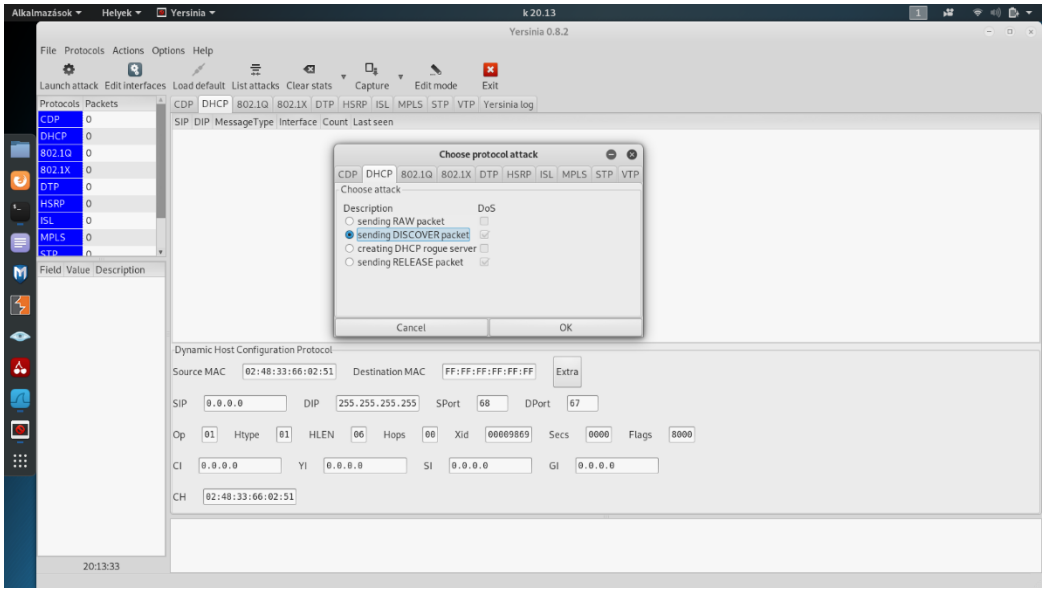
A Wiresharkon komoly beállítás nem szükséges, ahogy csatlakoztatva vagyunk a Ethernet-porton a hálózatra, úgy az eth0 interface hallgatását fogja kezdeményezni, és nekünk pont erre van szükségünk. Ezután kezdhetjük is a lehallgatást bal felül a cápauszony-jellel. Amit meg kell még tennünk, hogy a „filternél" beállítjuk a szűrőt DHCP-re, hisz nekünk az első résznél csak az lesz fontos. Futási időben valami hasonlót fogunk látni:



5. ábra Pillanatkép a DHCP kérésekről

6.3 DHCP starvation

Most, hogy a figyelést beállítottuk, elindíthatjuk a Yersiniában a támadást. Bal felül a Launch attack gombra kattintva kiválaszthatjuk, hogy mit szeretnénk – most DHCP DISCOVER üzenetekkel akarjuk elárasztani a hálózatot. Ezen kívül minden mást a Yersinia fog elvégezni helyettünk, manuálisan lényegében a beállításon kívül semmit nem kell tennünk. Ő el fogja árasztani a kérésekkel a szervert. Ha minden jól sikerül, akkor láthatjuk majd, hogy a teljes DHCP pool fel lesz használva, és noha a Yersinia rendületlenül küldi tovább a DHCP DISCOVER csomagokat, a kiszolgáló nem fog tudni mit kezdeni vele.



6. ábra DHCP DISCOVER üzenetek elindítása

Miután a támadás megtörtént, vessünk egy pillantást arra, hogy miképpen fest a DHCP pool jelenleg. A *(do) sh ip dhcp binding* parancs az Internal routeren a nekünk jelenleg legfontosabb információkat mutatja:

```
Budapest#sh ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration        Type
                    Hardware address/
                    User name
192.168.3.18        0100.270e.263b.31       Dec 05 2019 09:39 AM    Automatic
192.168.3.19        0100.270e.2636.1b       Dec 05 2019 09:40 AM    Automatic
192.168.3.20        0100.270e.2636.20       Dec 05 2019 09:41 AM    Automatic
192.168.3.21        0100.270e.2636.25       Dec 05 2019 09:42 AM    Automatic
192.168.3.22        0194.de80.3bb8.5d       Dec 05 2019 12:25 PM    Automatic
192.168.3.23        0150.b7c3.907e.1e       Dec 05 2019 09:54 AM    Automatic
192.168.3.24        50b7.c390.7e1e          Dec 05 2019 12:11 PM    Automatic
192.168.3.25        e8ec.8f75.97fc          Dec 04 2019 12:22 PM    Automatic
192.168.3.26        6088.690b.9dd1          Dec 04 2019 12:22 PM    Automatic
192.168.3.27        fcid.d808.2a66          Dec 04 2019 12:22 PM    Automatic
192.168.3.28        02c6.2d52.a0f7          Dec 04 2019 12:22 PM    Automatic
192.168.3.29        8a48.6e4e.3302          Dec 04 2019 12:22 PM    Automatic
192.168.3.30        2e2e.b146.60b3          Dec 04 2019 12:22 PM    Automatic
192.168.3.34        0100.270e.263b.2c       Dec 05 2019 11:48 AM    Automatic
192.168.3.35        0100.270e.3012.37       Dec 05 2019 11:46 AM    Automatic
192.168.3.36        0100.270e.3012.31       Dec 05 2019 11:48 AM    Automatic
192.168.3.37        0100.270e.3012.35       Dec 05 2019 09:41 AM    Automatic
192.168.3.38        0100.270e.2636.2f       Dec 05 2019 09:43 AM    Automatic
```

7. ábra A (do) sh ip dhcp binding parancs eredménye az Internal routeren

A parancs felsorolja nekünk a kiosztott IP címeket, és mivel a fertőzött gép a dolgozók között van, ezért ott okozott egy kis galibát. A dolgozók alhálózata a 192.168.3.16/28, ebben a hálózatban az első kioszthető cím a 3.17, ami a router alinterface címe, és látjuk, hogy 3.18-tól egészen 3.30-ig ki vannak osztva az IP címek. A 3.31 azért marad ki a listából, mert mint alhálózat utolsó címe, ezért broadcast címként funkcionál, ami azt jelenti, hogy az erre a címre küldött minden adatot megkap minden az alhálózatra csatlakoztatott eszköz. Magyarul, elfogytak a kioszthető IP címek. Ha jelenleg megpróbálunk csatlakoztatni egy új eszközt a dolgozók alhálozatához, akkor nem fogunk sikerrel járni – és ennek a támadásnak pont ez az egyik alapvető célja. Érdekes egy pillantást vetni arra, hogy a MAC címek mennyire elütnek a valós eszközök ugyanazon azonosítóitól. Alapesetben ebből messzemenő következtetést nem lehet levonni, hisz több száz hosttal rendelkező, több tíz router és switch közös működése közben pár különleges MAC cím nem lesz feltűnő, esetünkben is annak köszönhető ez a nagyon nagy hasonlóság, hogy kevés eszköz van csatlakoztatva a hálózatra. Mármint a valósak közötti hasonlóság, ami miatt szembetűnő a Yersinia különbözősége.

Egy másik ehhez nagyon hasonló támadási módszer a CAM table overflow, amit szinte ugyanúgy kell kivitelezni a Yersiniában, és ugyanúgy kell védekezni ellene.

7. Port Security védelem

7.1 A probléma

A támadásoknál a gondot az okozta, hogy semmiféle védelem nem volt beállítva az ilyen elárasztásos támadási stratégia ellen. A switchek (és a routerek is) buta eszközként funkcionáltak, „ami jött, azt elfogadták”, és ennek okán használhatatlan lett a hálózat (egy része). A megoldás az lehetne, hogy szabályozzuk, hány különféle MAC címet engedünk át egy-egy porton, így ha el is árasztjuk a hálózatot az előző módon, az nem fog kárt okozni a rendszer funkcionalitásában. Fontos, hogy a különféle védelmi megoldásokkal tisztában legyünk, hisz sajnos alapértelmezetten általában nincsenek bekapcsolva ezen védelmi funkciók. [4]

7.2 A portsecurity engedélyezése és beállítása

Mind a három switchen be kell konfigurálni a következőeket, én csak az egyiken fogom bemutatni. Az előző alfejezetben bemutattam, hogy be lehet állítani különböző portokra, hogy mennyi MAC cím engedélyezett, viszont ezt egyesével nem lenne kényelmes megcsinálni egy nagy rendszernél. Ezt úgy értem, hogy külön-külön be kellene lépni minden interface (port) konfigurációs beállításába, be kellene írni, hogy ez meg ez a MAC cím engedélyezett, majd kilépni, és átlépni egy másikba. Ennél sokkal kényelmesebb és jobb megoldás, ha beállítjuk dinamikusan a port security-t, ez annyit jelent, hogy portonként fogjuk megadni mennyi az elfogadott MAC cím az adott portokra. Ha ennél többet kap egy port, akkor automatikusan letilt, és értesítést küld a menedzsment rendszernek. Lássuk, hogy néz ki ez a gyakorlatban [2]:

```
Anne (config-if)# int range fa0/1-20
Anne (config-if)# switchport port-security
Anne (config-if)# switchport port-security maximum 1
Anne (config-if)# switchport port-security violation protect
Anne (config-if)# switchport port-security mac-address sticky
```

Nézzük, melyik sor mit ért védelmi szempontból. Az összes access portra egyszerre szeretném aktiválni, ezért léptem be az 1-20 közötti range-be. A trunk portokra egy kis változtatás elfér majd, így azokról kicsit később. Első lépésben elérhetővé kell tenni magát a port-security szolgáltatást. Ezután meg kell adni a maximum szócska után, hogy egy porton mennyi MAC címet fogadunk el. Ez nem mentődik el a konfigurációs fájlban magától, mivel az alapértelmezett érték az 1 – ilyenkor azt menti el, mintha nem is módosítottuk volna. Érdemes erre odafigyelni, gyakori problémákat okoz, hogy ez a sor ilyen formában nem fog megjelenni a konfigurációs fájlban.

A violation protect szükséges ahhoz, hogy ne léphessük túl a maximum értéket az előbbi beállításban bármiféle jelzés nélkül. Az alapértelmezett beállítás err-disable, azaz alapértelmezetten nem kapunk értesítést a túllépésről.

A mac-address sticky beállítás abban segít, hogy ha egyszer a porton átfut egy MAC cím, azt elmenti a running-config fájlba. Innen már egy egyszerű mentéssel elérhető, hogy a startup-config fájlba is bekerüljön, és a router minden indításánál betöltődjön.

Ezek után a kellő védelem megvan az elárasztásos támadás ellen, már csak le kell tesztelni, hogy valóban úgy működik minden, ahogy szerettük volna. Adjuk ki a do sh(ow) port parancsot, és nézzük, mit láthatunk a beállítások után!

```
Anne(config-if-range)#do sh port
Secure Port  MaxSecureAddr  CurrentAddr  SecurityViolation  Security Action
                 (Count)       (Count)         (Count)
---------------------------------------------------------------------------
      Fa0/1              1            1                  0         Protect
      Fa0/2              1            1                  0         Protect
      Fa0/3              1            0                  0         Protect
      Fa0/4              1            0                  0         Protect
      Fa0/5              1            0                  0         Protect
      Fa0/6              1            0                  0         Protect
      Fa0/7              1            0                  0         Protect
      Fa0/8              1            0                  0         Protect
      Fa0/9              1            0                  0         Protect
     Fa0/10              1            0                  0         Protect
     Fa0/11              1            1                  0         Protect
     Fa0/12              1            0                  0         Protect
     Fa0/13              1            0                  0         Protect
     Fa0/14              1            0                  0         Protect
     Fa0/15              1            0                  0         Protect
     Fa0/16              1            0                  0         Protect
     Fa0/17              1            0                  0         Protect
     Fa0/18              1            0                  0         Protect
     Fa0/19              1            0                  0         Protect
     Fa0/20              1            0                  0         Protect
```

8. ábra A do sh port parancs eredménye

Nagyon fontos beállításokat láthatunk itt. Az első oszlop felsorolja a portokat, ebben nincsen semmi meglepetés, a második oszlop jelzi, hogy mennyi a maximálisan elfogadott cím érték (ezt ugye az alapértelmezett 1-re állítottuk). A harmadik oszlop mutatja, hogy hány MAC cím van csatlakoztatva az adott porthoz, ebből láthatjuk, hogy jelenleg az fa0/1, fa0/2 és az fa0/11 portokhoz van eszköz csatlakoztatva. A negyedik oszlopban láthatjuk azt, hogy egy porton hány értesítést kaptunk arról, hogy túlléptük a megengedett címszámot. Az utolsó oszlop pedig azt jelzi, hogy minden porton be van kapcsolva a violation protect. Védelmi szempontból (egyelőre) ennyit szükséges írnom. Lesz még védelmi megoldás, amit be fogok mutatni, de mivel olyan támadást most nem hajtottam végre, így nincs beépítve a rendszerbe. Ugyanis azt tudniillik, hogy a hálózati rendszerek alapesetben rendelkeznek mindenféle védelmi megoldással, csak engedélyezni kell őket, illetve bekapcsolni.

Az alábbi, 9-es ábrán látható, hogy miután kiépítettük a védelmet, hiába eresztettük rá a támadást ismét a hálózatra, nem töltődött meg a dolgozó DHCP pool (meg igazából egyik sem).

```
Budapest#sh ip dhcp binding
Bindings from all pools not associated with VRF:
IP address          Client-ID/              Lease expiration        Type
                    Hardware address/
                    User name
192.168.3.18        0100.270e.263b.31       Dec 05 2019 09:39 AM    Automatic
192.168.3.19        0100.270e.2636.1b       Dec 05 2019 09:40 AM    Automatic
192.168.3.20        0100.270e.2636.20       Dec 05 2019 09:41 AM    Automatic
192.168.3.21        0100.270e.2636.25       Dec 05 2019 09:42 AM    Automatic
192.168.3.22        0194.de80.3bb8.5d       Dec 05 2019 12:25 PM    Automatic
192.168.3.23        0150.b7c3.907e.1e       Dec 05 2019 09:54 AM    Automatic
192.168.3.24        50b7.c390.7e1e          Dec 05 2019 12:11 PM    Automatic
192.168.3.34        0100.270e.263b.2c       Dec 05 2019 11:48 AM    Automatic
192.168.3.35        0100.270e.3012.37       Dec 05 2019 11:46 AM    Automatic
192.168.3.36        0100.270e.3012.31       Dec 05 2019 11:48 AM    Automatic
192.168.3.37        0100.270e.3012.35       Dec 05 2019 09:41 AM    Automatic
192.168.3.38        0100.270e.2636.2f       Dec 05 2019 09:43 AM    Automatic
```

9. ábra A védelem kiépítése után

8. Egyéb védelmi megfontolások

8.1 Alapértelmezett VLAN

A korábbiakban volt róla szó, hogy alapértelmezetten minden az 1-es ID-val ellátott VLAN-ban található. Mivel ez minden eszközön alapértelmezett, gyakori támadások célkeresztjében áll. Célszerű ezért létrehozni egy tetszőleges ID-vel rendelkező VLAN-t, és azt beállítani alapértelmezettnek:

```
Switch(config)#vlan [tetszőleges_szám]
Switch(config-vlan)#name [tetszőleges_név]
Switch(config-vlan)#exit

Switch(config-if)#switchport mode trunk native vlan [szám]
```

Ezt csak trunk porton lehet beállítani, viszont annak mindkét oldalán szükséges. Ezzel nem az 1-es azonosítóval rendelkező VLAN lesz a hálózatban az alapértelmezett, hanem az a másik, amit csak ezért hozunk létre, VLAN 1-re, hanem előbb meg kellene találnia, mi az alapértelmezett.

8.2 DHCP Snooping

Ez a megoldás szintén egy beépített védelmi funkció, aminek a célja, hogy egyfajta tűzfalként működjön egy megbízható DHCP szerver, és nem megbízható hostok között. Ha helyesen

konfiguráljuk, akkor egy rouge DHCP szerver hálózatba kapcsolását eléggé megnehezíthetjük. A beállítás lépései:

1. Szükség van egy működő, megbízható DHCP szerverre
2. Aktiválni kell a DHCP snoopingot legalább egy VLAN-on (de inkább mindegyiken, ami használatban van). Alapértelmezetten minden VLAN-ra inaktív a funkció.
3. A DHCP szervernek egy **megbízható** interface-re kell kapcsolódnia (tőlünk függ, melyik interface lesz megbízható és melyik nem). Alapértelmezetten minden interface **nem megbízható** állapotban van.
4. DHCP snooping database agent konfigurálása, úgy, hogy az elemek tényleg el legyenek tárolva, és rendelkezésre álljanak egy újraindítás után is.
5. DHCP snooping engedélyezése a globális konfigurációs módban.

Ezek után nincs más dolgunk, mint megnézni a beállítást egy általános példán keresztül. Ez a védelem nincsen beépítve a hálózatba, hiszen Rouge DHCP szervert nem alkalmaztam, de fontosnak érzem bemutatni egy általános esetben. Nézzük, hogy kell beállítani ezt lépésről lépésre:

```
Switch(config)#ip dhcp snooping
Switch(config)#ip dhcp snooping vlan [szám]
Switch(config)#int [megbízható interface]
Switch(config-if)#ip dhcp snooping trust
Switch(config)#ip dhcp snooping database [egy mentési hely]
```

Az első sorral globálisan elérhetővé tesszük a DHCP snooping szolgáltatást. Megbízható DHCP szervert nem kell már létrehozni, hisz azt már korábban megtettük, még a hálózat kiépítésénél. A második sorral megadjuk, hogy melyik VLAN esetén tesszük elérhetővé a snoopingot, tanácsos minden használt VLAN-ra megcsinálni. Ezután be kell lépnünk egy interface-re, amiben megbízunk, hogy valóban az általunk létrehozott, megbízható DHCP szerverrel fogunk találkozni, majd ha ez megvan, be kell állítunk az adatbázisba mentést, ami egy FTP vagy TFTP, vagy bármilyen más szerver lehet. Így már a hálózat belsejéből nem tudunk se IP címet kapni, se átirányítani a forgalmat egy fertőzött gateway-re, mert csak a megbízhatóra beállított interface-n keresztül kaphatunk DHCP szervertől IP címet.

8.3 Dynamic ARP Inspection

ARP snooping támadással lehetőség van arra, hogy az eszközök ARP tábláját megfertőzzük, és ezáltal lehetőségünk van a forgalom elterelésére, például úgy, hogy gateway-nek álcázzuk magunkat és így minden forgalom áthaladhat az eszközünkön, amivel lehallgatni szeretnénk. Ez ellen nyújt védelmet a Dynamic ARP Inspection. Hasonló a beállítása, mint a DHCP Snoopingnak, ugyanúgy meg kell adni, melyik port megbízható, és ugyanúgy VLAN-onként kell megtennünk az engedélyezését.

```
Switch(config)#ip arp inspection vlan [szám]
Switch(config)#int [megbízható interface]
Switch(config-if)#ip arp inspection trust
```

9. Továbbfejlesztési lehetőségek

Továbbfejlesztésként feljegyezhetőek a fentiekben bemutatott, de nem kiépített védelmi stratégiák (VLAN átállítás, DHCP Snooping, Dynamic ARP Inspection), illetve a hálózat felkészítése még sokkal nagyobb eszközszámra. Gondolok itt arra, hogy ne csak 14 hostot tudjon kezelni, hanem például több százat. Ehhez természetesen szükséges lehet további switchek használata.

Egy másik, elképzelhető fejlesztési lehetőség AAA rendszerrel jogosultsági szintek kiépítése a routereken, switcheken.

10. Összefoglalás

Minden vállalati (de igazából bármilyen) hálózatot fontos ellátni a megfelelő védelemmel. E nélkül kiszolgáltatott a rendszerünk minden támadás ellen. Épp ezért, fontos, hogy mindig szakember tervezze meg és építse ki a hálózatot, hogy elkerülhetőek legyenek az ilyen támadások. Ugyanis a legtöbb támadás emberi hanyagságból, mulasztásból lehetséges. Elég csak arra gondolni, amikor meghagyják a default jelszavakat a rendszereknél. De ha megtesszük a szükséges óvintézkedéseket, és az alap, a rendszer által nyújtott védelmeket csak bekapcsoljuk, úgy nagyban megnehezíthetjük a támadók dolgát.

Esetünkben is hasonlóképpen volt, elindultam egy szándékosan sérülékeny, de egyébként jól megtervezett, átgondolt hálózatból, eszközönként. Szépen egyesével építettem ki a rendszert, majd miután ezzel megvoltam, kiépítettem a szükséges védelmeket. A legtöbb időt a tervezési fázis vette el, mert a használt megoldások alapvetően nem bonyolultak, de nagyon alaposan át kellett gondolnom, mit fogok használni, mit hova szeretnék csatlakoztatni.

Fontos szempont, hogy a védelmet mindig a fenyegetettséggel, a veszéllyel arányosan kell kiépíteni. Azt hiszem, ezt nem kell magyarázni, hogy a születésnapi fotóra nem kell feltétlenül minden létező védelmet aktiválni, viszont az Egyesült Államok atomrakétáinak sem lehet az indítókódja a számok 1-től 8-ig (pedig valóban volt ilyen [8], amikor ez az indítókód a kiváló 00000000 több mint két évtizeden át). Esetemben ez azt jelentette, hogy fizikailag csak az adott támadások ellen volt kiépítve a védelem, de azért bemutatásra kerültek a további Layer 2-es védelmi mechanizmusok a leggyakoribb támadásformák ellen. Szükséges a körültekintő, tudatos tervezés, mert mindig az ember a legnagyobb hibafaktor és az amúgy banális hibák megbosszulják magukat minden esetben.

11. Irodalomjegyzék

[1] A. Tanenbaum és D. Wetherall, Számítógép hálózatok, Budapest: Panem Kft., 2013.

[2] O. Santos és J. Stuppi, CCNA Security 210-260 Official Cert Guide, USA: Cisco Press, 2015.

[3] vikrams@aruba, „arubanetworks.com," Hewlett Packard Enterprise Development LP, 7. május 2014.. [Online]. Available: https://community.arubanetworks.com/t5/Controller-Based-WLANs/What-is-the-Rapid-Spanning-Tree-Protocol-802-1w-and-how-do-I/ta-p/181616.

[4] „www.lan.hu," LAN Számítástechnikai Szolgáltató Kft., 17. július 2014.. [Online]. Available: https://www.lan.hu/layer_1.

[5] „miarec.com," MiaRec Inc., [Online]. Available: https://www.miarec.com/knowledge/how-configure-port-mirroring-cisco-catalyst-2960-series#~local-pbx.

[6] G. Gábor, Informatikabiztonság I., Pécs: Pécsi Tudományegyetem, 2015.

[7] jlm, „www.ieee802.org," 26 július 2006. [Online]. Available: http://www.ieee802.org/1/pages/802.1w.html.

[8] „hvg.hu," HVG Kiadó Zrt., 4. december 2013.. [Online]. Available: https://hvg.hu/tudomany/20131204_00000000_usa_atomtaska_inditokodja.

[9] G. Lencse, Számítógép hálózatok, Győr: UNIVERSITAS-GYŐR Nonprofit Kft., 2008.

12. Mellékletek

Konfigurációs fájlok (.txt formátumban):

- anne-config
- bella-config
- carol-config
- core-london-config
- internal-budapest-config
- anne-after
- bella-after
- carol-after

Cisco Packet Tracer-ben készített (nem teljes) szimuláció (.pkt formátumban):

- rendszer-szimu