**УНИВЕРЗИТЕТ „Св. Кирил и Методиј“**

**ПРИРОДНО МАТЕМАТИЧКИ ФАКУЛТЕТ - СКОПЈЕ**



**Изградба на оптимални модели на правила за системи за откривање на напади**

**- дипломска работа -**

Кандидат,Ментор,

Александар Христовски Проф. д-р Ванчо Чабуковски

Скопје, 2018

***Апстракт:*** *Во денешно време компјутерите и компјутерските мрежи се навлезени во секој аспект од животот. Тоа ја прави темата на компјутерската безбедност од витално значење. Често безбедноста на една мрежа зависи од оптималното распоредување на ресурси кои ги имаме на располагање. Овие ресурси вклучуваат хардвер, софтвер, како и едуциран кадар за менаџирање и надгледување на мрежата. Системите на откривање на напади се дел од системите кои се користат за обезбедување на една мрежа и нивната основна и најважна функција е континуирано надгледување на таа мрежа, барајќи потенцијални закани и известување на одговорните луѓе доколку пронајдат такви. Навременото фаќање на напад на мрежа на одредена организација може да направи разлика меѓу непрекината работа и комплетен пропаст на мрежата пропратен со загуби на капитал и доверба. Исто како повеќето системи, системите за откривање напади зависат од нивната конфигурација. Откривањето напади ваквите системи го вршат со споредување на пакети кои поминуваат низ мрежата со множество правила, и пакетите кои не се совпаѓаат со дефиницијата за одобрени пакети со овие правила се сметаат за потенцијално опасни аномалии. Многу чест случај е администраторите на мрежи да ги користат сите, или барем поголем број од правилата кои доаѓаат со систмот за откривање напади. Ова е лоша пракса која може да резултира со отфрлени пакети, неанализирани поминати пакети или премногу лажни алармни пораки од системот. Според тоа, важно е администраторот на одредена мрежа да ги разбира потребите кои таа мрежа би ги имала од системот за откривање напади, и да ги конфигурира правилата соодветно. Целта на оваа дипломска работа е предложување принципи по кои би можел да се води админситраторот за да дојде до една ваква оптимална конфигурација.*

***Клучни зборови:*** *Компјутерска безбедност, мрежна безбедност, IDS, системи за откривање напади*

***Abstract:*** *At present times, computers and computer networks are ever present in all aspects of life, thus, the field of computer and network security is of paramount importance. Oftentimes, the problem of computer and network security hangs on optimal use of available resources. Resources, in this sense of the word, entail hardware, software as well as personnel trained in network management and monitoring. Intrusion detection systems are part of overall network security systems and are primarily responsible for continuous network monitoring for potential threats. If a threat is found, the system should alert the people responsible to it. Dealing with such threats in a timely fashion could mean the difference between uninterrupted workflow and complete network failure bearing capital and trust losses. Like most systems, intrusion detection systems are only as good as their configuration. Intrusion detection is usually performed by testing whether network packages conform to a defined set of rules. If they do not, the IDS regards them as potentially hostile anomalies. It is a common practice for network administrators to use all, or at least a large subset of the predefined rules that come with the IDS. Such a practice can lead to excessive package rejection, analysis skipping a portion of the packages, or a large number of false positives. Therefore, it is of great importance that the administrator understands the network requirements regarding an IDS, and configure the rule set appropriately. This paper lays out certain principles and guidelines that would lead to optimal configuration of intrusion detection systems.*

***Keywords:*** *Computer security, network security, IDS, intrusion detection systems*

Содржина

[1. Вовед 5](#__RefHeading___Toc189_3320784199)

[2. Мрежни протоколи 11](#__RefHeading___Toc191_3320784199)

[2.1 IP (Internet Protocol) 11](#__RefHeading___Toc193_3320784199)

[2.2 TCP (Transmission Control Protocol) 14](#__RefHeading___Toc197_3320784199)

[2.3 UDP (User Datagram Protocol) 18](#__RefHeading___Toc203_3320784199)

[3. Техники и алатки за напад 20](#__RefHeading___Toc207_3320784199)

[3.1 nMap 24](#__RefHeading___Toc209_3320784199)

[3.2 Nessus 24](#__RefHeading___Toc211_3320784199)

[3.3 Metasploit 25](#__RefHeading___Toc213_3320784199)

[4. Техники и алатки за откривање и заштита од напади 26](#__RefHeading___Toc215_3320784199)

[4.1 Firewall 26](#__RefHeading___Toc217_3320784199)

[4.2 IDS/IPS 27](#__RefHeading___Toc219_3320784199)

[4.3 Snort 29](#__RefHeading___Toc221_3320784199)

[4.4 Suricata 30](#__RefHeading___Toc223_3320784199)

[5. Тестирање и анализа за оптимални IDS правила за различни типови на скенирање 31](#__RefHeading___Toc225_3320784199)

[5.1 Откривање на хостови 31](#__RefHeading___Toc227_3320784199)

[5.2 Скенирање на порти 36](#__RefHeading___Toc229_3320784199)

[5.3 No Ping Scan и генерирање на оптимални IDS/IPS правила 43](#__RefHeading___Toc231_3320784199)

[5.4 TCP Connect Scan и генерирање на оптимални IDS/IPS правила 44](#__RefHeading___Toc233_3320784199)

[5.5 Xmas Scan и генерирање на оптимални IDS/IPS правила 46](#__RefHeading___Toc235_3320784199)

[5.6 FIN Scan и генерирање на оптимални IDS/IPS правила 48](#__RefHeading___Toc237_3320784199)

[5.7 NULL Scan и генерирање на оптимални IDS/IPS правила 49](#__RefHeading___Toc239_3320784199)

[5.8 UDP Scan и генерирање на оптимални IDS/IPS правила 51](#__RefHeading___Toc241_3320784199)

[5.9 SYN Scan и генерирање на оптимални IDS/IPS правила 52](#__RefHeading___Toc243_3320784199)

[6. Креирање на оптимален модел на IDS правила за различни мрежни околини 55](#__RefHeading___Toc245_3320784199)

[6.1 Мрежи на мали и средни организации и соодветни правила 55](#__RefHeading___Toc247_3320784199)

[6.2 Мрежи на IT организации и соодветни правила 58](#__RefHeading___Toc463_3320784199)

[6.3 Мрежи на средни и големи организации и соодветни правила 60](#__RefHeading___Toc251_3320784199)

[7. Заклучок 64](#__RefHeading___Toc781_3320784199)

[7. Користена литература 66](#__RefHeading___Toc675_3320784199)

**Листа на слики**

Слика 1: Мрежа со минимална безбедност

Слика 2: Мрежа со висока безбедност

Слика 3: IP протокол header

Слика 4: TCP протокол header

Слика 5: UDP протокол header

Слика 6: No ping скенирање

Слика 7: Snort запис за No Ping скенирање

Слика 8: Suricata запис за No Ping скенирање

Слика 9: TCP Connect скенирање

Слика 10: Snort запис за TCP Connect скенирање

Слика 11: Suricata запис за TCP Connect скенирање

Слика 12: Xmas скенирање

Слика 13: Snort запис за Xmas скенирање

Слика 14: Suricata запис за Xmas скенирање

Слика 15: FIN скенирање

Слика 16: Snort запис за FIN скенирање

Слика 17: Suricata запис за FIN скенирање

Слика 18: NULL скенирање

Слика 19: Snort запис за NULL скенирање

Слика 20: Suricata запис за NULL скенирање

Слика 21: UDP скенирање

Слика 22: Snort запис за UDP скенирање

Слика 23: Suricata запис за UDP скенирање

Слика 24: SYN скенирање

Слика 25: Snort запис за SYN скенирање

Слика 26: Suricata запис за SYN скенирање

Слика 27. Пример мрежа на мала организација

Слика 28. Пример мрежа на IT организација

Слика 29. Пример мрежа на голема организација

# 1. Вовед

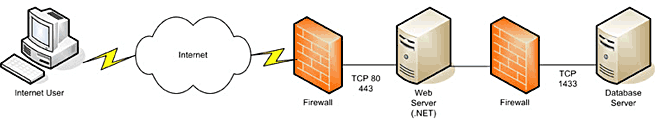
Безбедност на информации е тема која се занимава со доверливоста, интегритетот и достапноста на информации [2] (кратенката CIA – Confidentiality, Integrity, Availability е најпознатиот термин кој ги енкапсулира клучните концепти на безбедноста на информации). Иако оваа тема се занимава со заштита на податоци запишани на било каков медиум, денес терминот безбедност на информации е речиси синоним со терминот компјутерска безбедност. Во овој контекст поимот доверливост подразбира недостапност на одредени информации на субјекти коишто немаат пристап кон тие информации. Оваа дефиниција, покрај на заштитата од високо технички напади се однесува и на секојдневни ситуации, како на пример, заборавање на лаптоп на јавно место или праќање на елекронска пошта која содржи чувствителни информации на погрешна адреса со што се прекршува доверливоста на информации. Вториот клучен концепт, интегритетот на податоци значи одржување на податоците во целост, како и оневоможување легални или нелегални учесници во комуникацијата да прават неприметни промени на податоците. Денес за оваа цел најчесто се користи инфраструктура со јавен клуч (PKI – Public Key Infrastructure) која нуди хеширање, кодови за автентикација на порака (MAC – Message Authentication Code) и други начини за осигурување на интегритет на податоци. За да податоците бидат од корист, тие мора да се достапни кога има потреба од нив. Концептот на достапност на податоци подразбира дека компјутерските системи кои ги чуваат и обработуваат податоците, како и безбедносните системи, за нивна заштита функционираат беспрекорно и се занимаваат со превенција на прекин на овие сервиси потенцијално предизвикан од широк дијапазон на причини, од природни непогоди и грешки во пренос на елекрична енергија до Denial Of Service (DOS) и Distributed Denial Of Service (DDOS) напади.

Во последните години концептот на непорекливост (non-repudiation) е често додаван на претходно спомнатите три клучни концепти на безбедноста на информации. Непорекливоста е концепт кој потекнува од правото и гарантира дека учесникот во комуникацијата кој праќа одредена порака не може да одрекува дека ја пратил пораката, а доколку пораката е успешно пренесена, непорекливоста го оневозможува примачот на пораката да одрекува дека ја примил таа порака. Непорекливоста, слично како и концептот на интегритетот се постигнува со користење алатки од криптосистеми со јавен клуч (PKI).

Како екстензија на моделот на доверливост, интегритет и достапност, односно CIA триадата често се користи соодветно именуваната Хексада на Паркер. Оформена во 1998 година од Дон Б. Паркер, овој модел ги разбива трите клучни концепти на CIA моделот во шест. Ги додава следните три концепти: сопстевност, автентичност и корисност. Концептот на сопственост (или контрола) се занимава со потенцијална штета на корисници на информации кога ќе се изгуби доверливоста на одредена информација. Концептот на автентичност е суштински многу сличен на концептот на интегритет, но поспецифично се занимава со докажување авторство на одредена порака. Концептот на корисност е екстензија на концептот на достапност. Како пример, вообичаено се наведува имање пристап до одредени енкриптирани податоци, но немање на клуч за декрипција. Паркер го гледа ова како ситуација во која податоците кои се потребни се достапни во одредена форма, и така го задоволуваат третиот критериум на CIA моделот, но не се корисни во таа форма.

Во корпоративна околина, работни позиции кои се занимаваат со безбедност на информации се главен директор на безбедност (Chief Security Officer – CSO), главен директор на информациска безбедност (Chief Information Security Officer – CISO), безбедносен инжењер, аналитичар на информациска безбедност, администратор на безбедносни системи како и консултант за компјутерска безбедност.

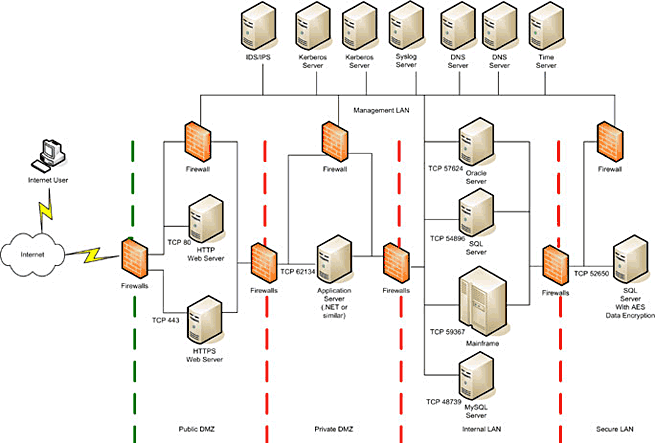
Поставување на безбедна мрежа е често прашање на финансиски средства кои можат да се потрошат за оваа цел. Основна алатка на мрежната безбедност се firewall-и кои филтрираат дозволен од недозволен сообраќај. Во класичниот едноставен случај е еден web server и еден сервер за бази на податоци. Во овој случај минималното често прифатливо решение вклучува поставување firewall-и на двете логични позиции на заштита, односно еден пред веб серверот и еден пред интерната мрежа којашто се состои од серверот со бази на податоци, како што е прикажано на слика 1.

*Слика 1. Мрежа со минимална безбедност*

Со ова релативно евтино класично решение, напаѓачот мора да компромитира само еден сервер пред да има пристап до веб апликациите на системот. Но, бидејќи единствена заштита се мрежни firewall-и, ова решение не ја штити мрежата од напади на апликативно ниво, така што напаѓачот може да експлоатира buffer overflow или injection ранливост во потенцијално несигурна апликација. Додатно на ова, работата на firewall системите зависи од квалитетот на нивната конфигурација и цената на овој систем расте со цената на вработениот администратор кој ги конфигурира и одржува овие firewall-и.

Општо е прифатено дека безбедноста на мрежата се зголемува со зголемувањето на чекорите кои потенцијален напаѓач мора да ги помине за да ја компромитира. Мрежа која се смета за безбедна денес вклучува користење на DMZ (demilitarized zone), исто така познато како мрежа на периматарот, која ги содржи и експонира сервисите на мрежата кон екстерна, недоверлива мрежа (вообичаено интернет). DMZ зоните можат да се поделат на јавна и приватна зона, со тоа дека јавната ја содржи логиката на корисничниот интерфејс, додека приватната ја содржи процесирачката логика на апликации. Покрај DMZ зони, безбедна мрежа треба да содржи и:

* IDS, односно системи за детекција на напади, најчесто NIDS (Network Intrusion Detection System) кој би служел за мониторирање на сообраќајот во сите критични делови на DMZ и безбедниот LAN.
* Time Server, односно сервер за синхронизација, чија примена во главно е осигурување на сите сервиси за логирање.
* Сервери за логирање кои служат за собирање информации од сите критични системи како firewall-и, рутери, IDS-ови и други мрежни уреди.
* Firewall-и, чии правила мора да се соодветно конфигурирани да пропуштаат само сообраќај кој е соодветен.
* DNS сервери за интерна употреба, со тоа што сите пакети за надвор од мрежата се делегирани на DNS серверите на провајдерот. Со ова се обезбедува строго мониторирање на интерните DNS серверите со што се спречуваат напади како DNS poisoning на пример.
* Безбедни LAN сервери, односно сервери за складирање податоци кои треба да се придржуваат до две клучни правила и тоа: се што се складира е енкриптирано и минимален број луѓе и апликации имаат строго контролиран пристап до нив.
* Kerberos сервери, преку кои сите апликациски сервиси се автентицираат меѓусебно.



*Cлика 2. Мрежа со висока безбедност*

Како додатни безбедносни мерки се препорачува користење на подмрежи (IP subnets) на соодветни места, виртуелни локални мрежи (VLAN) базирано на потенцијалниот ризик од очекуваниот сообраќај, виртуелни машини кои би му дозволиле на хост-базиран IDS (HIDS) да исклучи сервер кој смета дека е компромитиран и др.

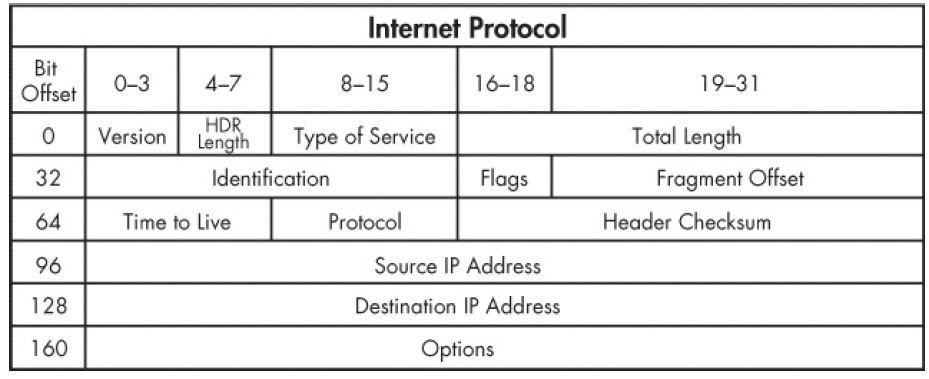
Темата на оваа дипломска работа се системите за откривање напади (IDS), односно конфигурацијата на правилата со кои овие системи работат. Овие системи често се оставени да ги користат сите, или голем број од преддефинираните правила со кои доаѓаат, и пишувањето на сопстевни правила адаптирани на одредена мрежа е занемарено. Преку пишување на вакви правила кои ги земаат во предвид специфичните барања на една мрежа може да се зголеми ефикасноста на системот за откривање напади. Понатаму се прикажани неколку општи случаи кои би можеле да служат како правци во кои би се постигнала оптимална конфигурација на правила.

# 2. Мрежни протоколи

## 2.1 IP (Internet Protocol)

Internet Protocol-от е примарниот протокол за комуникација преку интернет, и е одговорен за пренесување датаграми преку мрежни граници [5]. Во рамките на една мрежа уредите можат да комуницираат со користење на хардверски (MAC) адреси, но кога целиот уред е надвор од просторот на колизија (позади switch или рутер), за комуникација се потребни IP адреси. Доколку е потребно да прати пакет надвор од локалната мрежа, пакетот се енкапсулира во IP датаграм и сепраќа на Default Gateway (вообичаено рутер). Рутерот користејќи табели и префикси одлучува на кој излез да го препрати пакетот, или доколку нема запис во табелата за тоа одредиште, или одредиште со тој префикс, пакетот го препраќа на неговиот Default Gateway и постапката се повторува и се пропагира низ различни рутери додека пакетот не стигне на рутерот одговорен за одредишната адреса кој го отпакува и го праќа да одредишниот компјутер.

### IP Header

На слика 3 е прикажан IP протокол header-от, при што:

*Слика 3. IP протокол header*

* Version – ја означува верзијата на интернет протоколот
* IHL, Internet Header Length – ја означува големината на header-от, изразена во words од 32 битови. Минималната големина на header-от е 20 битови, амаксималната е 160.
* Differentiated Services Code Point/Type of Service – поле кое се користи во главно од понови сервиси кои бараат синхорни податоци како Voice over IP.
* Explicit Congestion Notification е поле кое е наменето да означува моментална искористеност на мрежата и е опционална функционалност која единствено се користи кога сите учесници во комуникацијата како и мрежата преку која комуницираат ја подржуваат
* Total Length – поле кое ја означува вкупната големина на пакетот, вклучувајќи ги header-от и податоците. Минималната вредност е 20 (header без податоци), додека максималната е 65.535.
* Identification - е поле кое се користи за идентификација на група фрагменти на еден IP датаграм.
* Flags – ги содржи вредностите на 3 IP header знаменца, имено:

1. bit 0: Резервиран, секогаш е 0
2. bit 1: Don’t Fragment (DF)
3. bit 2: More Fragments (MF)

Ако DF полето е поставено на 1, а фрагментирање е неопходно пакетот е отфрлен.

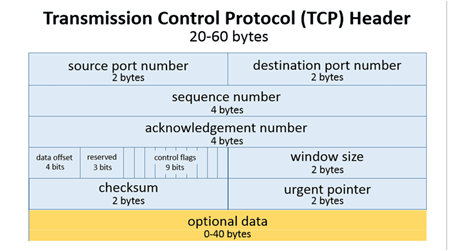
* Fragment Offset – го содржи offset-от нa еден фрагмент во однос со почетокот на оригиналниот, нефрагментиран IP датаграм.
* Time To Live – е поле чија вредност се намалува со секој поминат рутер, и е наменето за превенција на ситуација каде што еден пакет се врти во круг меѓу повеќе рутери и со тоа непотребно ја оптеретува мрежата. Доколку TTL полето достигне 0, пакетот е отфрлен. Во пракса, често се користи како hop count, односно за броење низ колку рутери поминал пакетот за да стигне на одредишната адреса.
* Protocol – поле кое содржи код на протоколот на податочниот дел на IP датаграмот.
* Header Checksum - е поле кое содржи 16 битен checksum на IP header-от. При пристигање кај рутер, рутерот ја пресметува checksum вредноста и проверува дали одговара со запишаната. Доколку не одговара, пакетот е отфрлен. Поради тоа што секој рутер ја намалува вредноста на TTL полето, checksum вредноста се променува на секој траснпорт
* Source Address - поле кое ја содржи адресата на праќачот на пакетот. Поради NAT оваа вредност може да се промени.
* Destination Address - поле кое ја содржи одредишната адреса на пакетот, и исто како Source Address, може да се промени поради NAT.
* Options – ова поле речиси никогаш не се користи

## 2.2 TCP (Transmission Control Protocol)

TCP, или Transmission Control Protocol, е стандард за комуникација на Transport нивото на OSI моделот [5]. TCP воспоставува врска меѓу учесниците во комуникацијата која останува активна додека трае комуникацијата (connection-oriented протокол), и нуди доверлива размена на информации.

### TCP Header

На слика 4 е прикажан TCP протокол header-от кој има варијабилна големина помеѓу 20 и 60 бајти поделени во:

******

*Слика 4. TCP протокол header*

* Source port — 16 битови, портата на компјутерот кој праќа пакет
* Destination port — 16 битови, портата на компјутерот кој прима пакет
* Sequence number — 32 битови, идентификациониот број на секвенцата
* Acknowledgement number — 32 битови, идентификациониот број на ACK пакетот
* Data Offset — 4 битови, ја специфицира големината на TCP header-от изразена во 32-битни words.
* Reserved — 3 битови, наменети за можна идна употреба, и се подесуваат на нула
* Контролни знаменца — 9 битови, од кои шест се во употреба, имено:  
  1. URG — индицира дали Urgent pointer-от има улога;
  2. ACK — индицира дали Acknowledgement полето има улога; секој пакет освен иницијалниот SYN пакет треба да го има ова знаменце подесено на 1;
  3. PSH — ова знаменце му кажува на примачкиот компјутер да ги пренесе податоците прочитани во buffer-от на соодветната апликација;
  4. RST — ова знаменце подесено на 1 ја ресетира конекцијата;
  5. SYN — ова знаменце би требало да е подесено на 1 само во првиот пакет пратен од било кој од учесниците во комуникацијата и служи за синхронизација на бројот на секвенцата;
  6. FIN — ова знаменце го обележува последниот пакет испратен од праќачот.
* Window size — 16 битови, ја специфицира големината на прозорот, односно количината на информации вообичаено изразена во бајти која праќачот може да ја прати пред да добие ACK одговор.
* Urgent pointer — 16 битови, се користи доколку URG знаменцето е поставено на 1 за да му каже на примачот на пакетот кој дел од пакетот да го смета за итен. Итниот дел на пакетот се смета од почетокот на податоците до Urgent pointer-от.
* Опции — од 0 до 320 битови, се користат за подесување опции како максимална големина на сегмент, големина на прозор, селективно барање на ACK пакети, timestamp итн. Најчесто се користат преддефинирани опции и овој дел од header-от е нулти.

### TCP врска

TCP врската се воспоставува на следниот начин:

1. Иницијаторот на врската праќа пакет со SYN знаменцето поставено на 1, и случајно избран број како број на секвенца.
2. Другиот учесник праќа пакет на иницијаторот во кој ACK и SYN знаменцата се поставени на 1, Acknowledgement полето го содржи очекуваниот број на следната секвенца, односно бројот на моменталната секвенца инкрементиран за еден, a Sequence полето е поставено на бројот на моменталната секвенца, односно и исто како полето во пакетот на иницијаторот.
3. Иницијаторот праќа пакет без податоци со истиот ACK број како пакетот во претходиот чекор, и инкрементиран број на секвенца. Бидејќи пакетот не содржи податоци, промачот на пакетот не го инкрементира неговиот број на секвенца. После овој чекор, врската е воспоставена.
4. Преку воспоставената врска, учесниците разменуваат пакети, и со секој примен пакет учесниците го инкрементираат бројот на секвенцата и ACK бројот за големината на пакетот изразена во бајти
5. Откако еден од учесниците ќе одлучи да ја заврши врската испраќа пакет со FIN знаменце поставено на 1 и соодветен број на секвенца за тој пакет (еднаков на ACK бројот од претходниот пакет).
6. Откако еден учесник ќе добие FIN пакет, тој праќа ACK пакет назад со соодветен Acknoledgement број, и својот FIN пакет со соодветен број на секвенца.

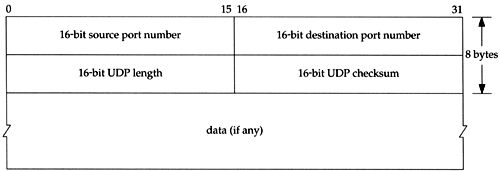
По добивањето на пакетот од претходниот чекор, иницијаторот на затворањето на врската враќа ACK пакет број на секвенца инкрементиран за еден од бројот на секвенца во претходниот пакет, и со ова конекцијата е успешно завршена.

## 2.3 UDP (User Datagram Protocol)

User Datagram Protocol е формализиран во 1980 година од Дејвид П. Рид и е еден од основните интернет протоколи. UDP нуди комуникација без одржување константна врска помеѓу учесниците, има минималистички пристап кој не вклучува handshaking механизам, не вклучува механизам за детекција или поправање на грешки, ги експонира апликациите на грешки во пренос и не нуди гаранции за истиот, односно не е доверлив протокол [5]. Се применува во случаи кога брза испорака на пакетите е поважна од точна, гаранатирана испорака. Апликации за streaming на видео или аудио преку интернет, Domain Name System, Network Time Protocol, IP tunneling, Remote Procedure Call, Network File System, DHCP, Trivial File Transfer Protocol, IPTV, VoIP, online игри, Routing Information Protocol се дел од апликациите кои работат преку UDP.

### UDP Header

1. Header-от на UDP датаграм (слика 5) има 8 бајти, подеднакво поделени во 4 полиња, и тоа:



*Слика 5. UDP протокол header*

* Број на изворна порта -се подразбира дека доколку настане потреба примачот на датаграмот може да се обрати на изворната апликација на оваа порта. Доколку ова не е возможно, ова поле треба да содржи нула.
* Одредишна порта - која е привремена порта (ephimeral port) доколку одредиштето е клиентот, или добро позната порта доколку одредиштето е серверот.
* Должина - поле кое ја означува должината на UDP header-от и податоците во датаграмот. Минималната должина е 8 бајти, , а максималната е 65507 бајти доколку имплементацијата работи врз Ipv4 и поголема ако работи врз IPv6.
* Checksum - поле кое може да се употреби за проверка на грешки во header-от и податоците. Се подесува на нули доколку не се користи.

# 3. Техники и алатки за напад

Компјутерските напади можат да бидат класифицирани по повеќе критериуми [1]. Базирано на легалноста и мотивацијата на нападот се разликуваат:

* white hat, односно penetration testing напади, кои се комплетно легални и нарачани од нападнатата страна со цел да се тестира безбедноста на нивниот систем,
* black hat, односно нелегани напади кои можат да бидат мотивирани од личен профит, активизам, љубопитство и слично, како и
* gray hat, кои се ненарачани, нелагални напади со цел да се предупреди нападнатата страна за одредена ранливост која постои кај нea.

Класификација на база на кој е напаѓачот разликува:

* компјутерски воен акт, каде што напаѓачот и нападнатата страна се држави или нации, компјутерски криминал, каде што напаѓачот е индивидуа или организација или
* компјутерски тероризам каде што повторно напаѓачот е индивидуа или организација.

Без разлика на класификацијата, секој напад се базира на наоѓање ранливост кај нападнатата страна и користење техники за експлоатирање на таа ранливост. За оваа цел напаѓачот може да напише свои, или да искористи веќе напишани алатки. Пишувањето свои алатки значително би ја отежнало работата на напаѓачот со оглед на тоа дека бара многу фундаментално знаење за тоа како работи техниката која напаѓачот сака да ја искористи, а додатно на тоа е, најчесто многу непрактичен избор бидејќи за повеќето техники веќе постојат добро напишани и оптимизирани алатки.

Напад на една мрежа вообичаено трае повеќе месеци и може да се подели во седум дискретни чекори [2]:

1. Истражување (reconnaissance)

Во оваа фаза напаѓачот ги рагледува информациите околу организацијата која ја напаѓа кои може да ги добие на комплетно не интрузивен начин. Вообичаено ова ги опфаќа јавно достапните информации, како и било кои информации околу структурата на мрежата кои може да ги добие “од далеку”, како адреси на веб сервери, mail сервери итн. Социјален инжинеринг е вообичаено единствена техника која се користи во овој чекор.

1. Скенирање

Скенирањето користи за идентификација на т.н. “слаба точка” која може да се искористи за влез во мрежата и тука е првиот момент кога нападот може да биде забележан од страна на напаѓачката страна. Овој чекор понатаму може да се подели во скенирање на инфраструктурата на мрежата кое определува хостови кои се дел од неа и за каква намена, скенирање на порти и сервиси кои се активни на таа мрежа и скенирање за ранливости на активните сервиси. nMap е алатка која ги покрива сите три подфази на скенирањето, а Nessus и SAINT се најпознатите скенери за ранливости на мрежата

1. Пристап и негова ескалација

Под претпоставка дека напаѓачот нашол ранливост која може да ја експлоатира, во овој чекор првпат добива “присуство” на нападнатата мрежа. Бидејќи речиси секогаш целта на компјутерскиот напад бара високи привилегии на мрежата, одкога ќе биде присутен на мрежата, напаѓачот би морал да работи на ескалација на неговите привилегии. Техниките користени во оваа фаза се експлоатирање на најдени ранливости, пробивање на лозинки и социјален инжињеринг. Моментално Metasploit е најпознатиот софтвер за експлоатација кој ја автоматизира голем дел од работата потребна за да се постигнат овие цели, но во идеален случај, напаѓачот би можел да најде недокументирана ранливост и да напише свој код за нејзино експлоатирање и ова значително би ги намалило шансите пристапот да биде забележан од безбедносните системи на нападнатата мрежа. Сепак, ова е ретко возможно.

1. Ексфилтрација

Ексфилрација е процесот на наоѓање начин да се пренесат (вообичаено, неприметно) податоци од нападнатата мрежа на уреди поседувани од напаѓачот. Ова често може да се постигне со едноставно копирање на друг сервер, но некогаш потребни се ad hoc мерки.

1. Одржување на пристапот

Откако ќе добие пристап и привилегии на мрежата, често, напаѓачот би сакал да си обезбеди континуиран, незабележлив пристап. Ова се постигнува со користење root kits, тројански коњи и сличен софтвер. Metasploit пакетот содржи автоматизирани начини на инсталирање ваков софтвер, иако, повторно, авторски код на напаѓачот би било многу потешко за безбедносните системи на мрежата да го забележат.

1. Напад

Во овој чекор целта е прекин на работата вршена од нападнатата мрежа. Прикриеноста на нападот прекинува да биде важна на напаѓачот, и целта е да се предизвика што позначителна и непоправлива штета на функцијата на нападнатата организација.

1. Криење на нападот

Класичните начини на прикривање на нападот и присуството се чистење на системските записи, како и записите направени од IDS и други безбедносни системи, иако некои мрежи бараат ad hoc решенија за оваа цел.

## 3.1 nMap

nMap (“Network Mapper”) е бесплатен продукт со отворен код, наменет за истражување на мрежа и сервиси кои работат во неа [6]. Користи IP пакети (raw IP packets) за да одреди хостови кои се активни на мрежата и сервиси кои тие ги нудат, и поддржува напредни техники кои дозволуваат мапирање на мрежи со имплементирани IP филтри, firewall-и, рутери, повеќе механизми на скенирање на порти (вклучувајќи TCP и UDP), детекција на оперативни системи, детекција на верзии на софтвер, ping sweep и други.

## 3.2 Nessus

Nessus е најкористениот скенирачни софтвер за ранливости во светот. Скенирање со овој софтвер открива:

* Ранливости кои напаѓач може да ги искористи за да добие пристап и контрола врз одреден систем.
* Погрешна, односно, не безбедна конфигурација на софтвер
* Користење преддефинирани лозинки, често користени лозинки, празни лозинки.
* Можност за Denial Of Service преку користење специјално направени пакети врз имплементација на TCP/IP протоколи.

Покрај penetration testing, Nessus често се користи за подготовка на систем за да исполни стандарди за користење платежни картички како и други финансиски услуги. Работи така што открива отворени порти користејќи соодветен скенер за понатаму да пробува да изврши експлоатирачки код врз нив. Nessus важи за квалитетен скенер на ранливости, меѓу другото поради константно одржуваната база на експлоити со која располага

## 3.3 Metasploit

Еден од најпознатите проекти во компјутерската безбедност, Metasploit содржи голема база експлоити, софтвер за проверка на ранливост, програми за одржување на пристап како тројански коњи и root kit-ови и со ова на корисниците им дозволува да:

* Одберат експлоит од колекцијата од над 1677 експлоити за различни оперативни системи.
* Опционално да проверат дали целниот систем е ранлив на тој експлоит.
* Да одберат каков код да се изврши на целниот систем доколку експлоатирањето се изврши успешно (payload).
* Да одберат техника на кодирање така што кодот од претходниот чекор би останал незабележан од IDS/IPS.
* Да го извршат експлоатирањето преку Metasploit.

Додатно на ова Metasploit нуди и пакет наречен “Command shell” кој служи за извршување команди на целниот систем, пакет “Meterpreter” кој му нуди опции за префрлување податоци и отворање VNC врска со целниот компјутер, како и опција за креирање динамичен код за тројански коњи, вируси и root kit-ови коишто потенцијално можат да останат незабележани од безбедносните системи на целната мрежа

# 4. Техники и алатки за откривање и заштита од напади

## 4.1 Firewall

Firewall е безбедносен систем кој контролира доаѓачки и поаѓачки сообраќај на мрежата. Овој систем традиционално е заинтересиран во header на пакетите кои пробуваат да поминат низ него, користи база на правила за да разликува вообичаен од злонамерен сообраќај и има способност да ги отфрли пакетите кои не спаѓаат во одобреното множество пакети [1]. Во последно време се јавуваат Next Generation Firewalls (NGFW) кои можат да се конфигурираат да служат како IPS (Intrusion Prevention Systems). Функционално firewall-ите можат да се класифицираат како:

* пакет филтри, кои работата на второто и третото ниво на OSI моделот и пропуштаат или отфрлаат пакети базирано на информации добиени од header-от на тие пакети,
* инспектори на состојба на врски, работат на шесто ниво на OSI моделот и имаат способност “поинтелигентно” да расудуваат и да внимаваат на состојбата на врската меѓу внатрешниот и надворешниот учесник во комуникацијата и
* proxy firewall-и кои работат на апликативно ниво на OSI моделот, се посветени уреди кои стојат на влезот, односно излезот на мрежата и го добиваат целиот сообраќај во било која насока, и го препраќаат одобрениот.

Многу firewall-и денес спаѓаат во повеќе од една од овие класи. Овие системи се основна заштита на било која мрежа и како такви држат најголем дел од пазарот на безбедносни уреди. Меѓу најкористените организациски решенија се: Fortinet FortiGate, Cisco ASA, Sophos UTM, pfSense и други.

## 4.2 IDS/IPS

IDS (Intrusion Detection System), односно систем за откривање на напади е харверски или софтверски систем кој надгледува активност на одредена мрежа (NIDS - Network Intrusion Detection System) или компјутерски систем (HIDS -Host Intrusion Detection System). Доколку работи врз еден компјутерски систем, IDS-от во главно внимава на запишување и читање на податоци. Мрежниот систем за откривање напади ги анализира header-от и содржината на IP пакетите и според тоа одлучува дали станува збор за нормален сообраќај или малициозна активност. Доколку открие малициозна активност системот го известува администраторот за тоа вообичаено преку внесување запис во SIEM (Security Information and Event Management) систем. Разликувањето “добар” од “лош” сообраќај се прави или врз база на множество правила или преку откривање аномалии.

Во првиот случај администраторот на мрежата внесува правила на тоа што тој очекува да биде нормален сообраќај и системот го алармира за сите пакети кои не спаѓаат во множеството пакети одобрени од овие правила. Ваквиот систем е ефективен колку и неговата конфигурација, и не е ефективен спрема нови напади. Класичен пример на ваков систем е Snort, како и поновиот Suricata, кој исто така е систем за превенција на напад.

Принципот за разликување сообраќај преку откривање аномалии користи хевристика и техики од машинско учење за да “научи” што е нормален сообраќај на една мрежа преку “множесто за тренинг”. Доколку “тренингот” заврши успешно, ваков систем би можел да го известува администраторот за секоја девијација од стандардниот сообраќај низ мрежата. Предноста на овој принцип е можност да се откријат иновативни напади кои инаку не би можеле да се запишат како правило. Негативните страни се тоа дека често е тешко да се обезбеди прочистено “тренинг множество”, како и големата појава на лажни аларми (false positives). Најпознатиот систем за откривање напади кој го користи принципот на откривање аномалии е Bro.

Системи за превенирање на напади (IPS -Intrusion Prevention System), исто така познати како активни системи за откривање напади, се всушност системи за откривање напади со способност да отфрлат пакети кои не спаѓаат во одобреното множесто пакети. Главната разлика помеѓу систем за превенирање на напади и firewall е тоа дека овие системи покрај информациите во header на пакетите ја разгледуваат и нивната содржина, иако со појавувањето на понови firewall-и, оваа разлика ја снемува [3]. Меѓу попознатите системи за превенирање напади се: Suricata, IBM Security Intrusion Prevention System, Cisco FirePower NGIPS.

## 4.3 Snort

Snort може да се користи како packet sniffer (еквивалент на tcpdump), packet logger, корисно за водење евиденција за сообраќај на мрежа и решавање проблеми поврзани со проточност, како и комплетен IDS систем. Оригинално напишан од Мартин Роеш во 1998 година, моментално на Snort работи Sourcefire, компанија чиј основач е Роеш а е во сопственост на Cisco.

Snort има јазик со едноставна, но ефективна синтакса за пишување правила. Класично Snort правило ја има следната форма:

акција протокол екстерна-адреса екстерна-порта <-насока-> интерна-адреса интерна-порта опции,

каде акција содржи инструкција што да прави Snort во случај да фати соодветен пакет; eкстерната адреса и екстерната порта се наоѓаат надвор од мрежата која ја мониторира Snort , додека интерната адреса и порта се наоѓаат во мрежата; насоката означува дали пакетот е примен од мрежата или се испраќа од мрежата;во опции често се внесува пораката која треба да ја запише Snort во случај да фати соодветен пакет, идентификациски број на правилото, број на ревизии, како и проверка на содржината на пакетот.

На пример:

log tcp 192.168.0.52/24 any → 192.168.0.24 22 (msg: “interen SSH pristap”;)

ќе внесе запис “interen SSH pristap” секојпат кога некој ќе прати TCP пакет од локална IP адреса на порта 22 на 192.168.0.24, додека правилото:

alert tcp !192.168.0.52/24 any → 192.168.0.24 22 (msg: “eksteren SSH pristap”;)

ќе запише alert со содржина “eksteren SSH pristap” секојпат кога ќе фати TCP пакет од адреса која не спаѓа во рангот на IP адреси од 192.168.0.0 до 192.168.0.255 до порта 22 на 192.168.0.24.

## 4.4 Suricata

Suricata е IDS/IPS систем, односно систем за детекција и систем за превенција на напад со отворен код. Развиван континуирано од OISF (Open Information Security Foundation), првата верзија е стандардизирана во јули 2010.

Принципот на пишување правила во Suricata е речиси идентичен со принципот на пишување правила за Snort, со тоа дека Suricata има додадени акции за да ја исполни функцијата на систем за превенција на интрузии.

# 5. Тестирање и анализа на оптимални IDS правила за различни типови на скенирање

## 5.1 Откривање на хостови

Откривањето на хостови преставува процес кој ни помага да ги забележиме активните адреси од поголем ранг на адреси [6]. Во продолжение ќе се задржиме на неколку различни форми на откривање хостови на една мрежа:

* List Scan (-sL) е најпримитивната форма на откривање на хостови на една мрежа која ги листа сите хостови на една мрежа без разлика дали се достапни или не, преку reverse DNS резолуција на нивните адреси. Корисно за осигурување на кои хостови припаѓаат на една мрежа преку нивните имиња, како и можно добивање додатни информации преку имињата. Пример, ако сме заинтересирани на мрежата на ПМФ, и во листата добиеме хост со име fw.pmf можеме да препоставиме дека тоа е firewall-от на мрежата. Бидејќи овој тип на скенирање не праќа никаков пакет на хостовите на мрежата, не може лесно да се открие скенирањето. Поради исти причини, повискоки фунционалности како скенирање на порти, детекција на оперативен систем или ping скенирање не може да се комбинираап со List Scan.
* No Port Scan (-sn), (или ping scan, -sP во претходни верзии на nMap) е опција која се користи на иста намена како List Scan, но наместо reverse DNS резулуција на имиња на постигнува оваа цел со праќање пакетни на целните хостови. За разлика од List Scan, ова скенирање ги листа само достапните хостови и може да се искористи за брзо истражување без да привлече многу внимание. Истотака може да се конфигурира да изврши traceroute и различни nMap скрипти. Без додатни конфигурации -sn праќа ICMP echo request (ping), TCP SYN пакет на порта 443, TCP ACK пакет на порта 80 и ICMP timestamp request. Ако се скенира ethernet мрежа се праќаат ARP requests освен ако не се специфицира опцијата –send-ip.
* No Ping (-Pn) (-P0 или -PN во претходни верзии на nMap)е опција која го скока мрежното откривање целосно. Вообичаено nMap извршува скенирање на порти, детекција на верзии и сите побарани скенирања на хостови во бараната мрежа кои претходно нашол дека се достапни. Со -Pn, nMap ќе изврши се што е побарано на сите адреси во мрежата без разлика дали таа IP адреса е активна. За адреси на локална ethernet мрежа, ARP скнирање сепак ќе биде извршено, освен ако –disable-arp-ping или –send-ip е специфицирано.
* TCP SYN Ping (-PS <port-list>) е опција која праќа празен TCP пакет со SYN знаменце поставено на 1. Ако не е специфицирана друга порта, пакетот се праќа на порта 80. Можеме да специфицираме други порти со нивно наведување непосредно после -PS, пример –PS22-25,80,113,1050,35000 ќе прати TCP SYN пакет на портите 22, 23, 24, 25, 80, 113, 1050, 35000. SYN знаменцето му кажува на хостот дека скенирачкиот компјутер пробува да воспостави врска. Доколку соодветната порта на хостот е затворена, хостот праќа RST пакет назад до скенирачниот компјутер, а доколку е отворена хостот праќа SYN/ACK пакет, на што кернелот на скенирачниот компјутер ќе одговори со RST бидејќи кернелот ќе го смета SYN/ACK пакетот за неочекуван. Во фазата на мрежно откривање на nMap не му прави разлика дали примил RST или SYN/ACK пакет назад бидејќи двете значат дека хостот е активен.
* TCP ACK Ping (-PA <port-list>) е ping скен, сличен на TCP SYN Ping скен, со тоа дека со ова скенирање ACK знаменцето на TCP пакетот е поставено на 1. Предефинираната порта е 80, и исто како со -PS скенирањето може да се конфигурира на било која друга порта или серија порти. Бидејќи ACK знаменцето е поставено на 1, активен хост секогаш би требало да врати RST пакет, бидејќи ова би му било неочекуван пакет на невоспоставена врска. Комбинација на -PS и -PA скенирања ги максимизира шансите за поминување на пакети преку firewall. Честа конфигурација (вградена во Linux Netfilter/iptables) на firewall блокира SYN пакети освен до јавно пристапни сервиси како mail или web сервери и во овој случај TCP ACK скенирањето може да помина, додека TCP SYN скенирањето не. Се доаѓа до спротивна ситуација со друга се почеста конфигурација на firewall-и, во која се отфрлаат пакети кои не одговараат на моменталната состајба на врската, и со оваа конфигурација TCP SYN пакет би поминал, додека TCP ACK пакет не.
* UDP Ping (-PU <port-list>) праќа празен UDP пакет на порта 40125, освен ако друга порта не е специфицирана. На некои порти пакет до податоци специфични за протокол може да зголемат шанси да се добие одговор. Содржината на пакетот може да се подеси со користење на –data, --data-string и –data-length опциите. Ако се наиде на затворена порта на хостот се добива ICMP Port Unreachable пакет назад. Ова сигнализира дека хостот е активен, додека друг тип на ICMP пакет како host/network unreachable или TTL exceeded или не добивање никаков пакет назад индицираат дека хостот не е достапен. Доколку портата е отворена повеќето сервиси само го игнорираат пакетот и не праќаат ништо назад. Поради оваа причина предефинираната порта е 40125, порта која речици никогаш не се користи. Главната придобивка на користење на ова скенирање е дека ги поминува филтрите за TCP порти.
* SCTP INIT Ping (-PY <port-list>) е скенирање кое користи SCTP INIT пакет, односно првиот чекор од SCTP four-way handshake. Доколку не се специфицира друга порта, се користи порта 80. Доколку оваа порта е затворена се добива ABORT пакет назад, а доколку е отворена се добива INIT-ACK. Двете сигнализираат дека хостот е активен.
* ICMP Ping Types (-PE, -PP, -PM) се опции кои ја користат стандардната ping алатка за откривање на мрежа. Опцијата -PE праќа ICMP type 8 (echo request) и очекува тип 0 (echo reply) назад. Многу хостови и firewall-и денес ги блокираат овие пакети (спротивно на RFC 1122) и поради оваа причина ICMP скенирања се несигурни за скенирање мрежи преку интернет, но се користата за мониторирање локлни мрежи. ICMP timestamp и subnet mask барања се реализираат преку -PP и -PM соодветно.
* IP Protocol Ping (-P0 <protocol-list>), опција која праќа пакети од специфицираните протоколи во протокол листата. Листата предефинирано содржи ICMP (протокол 1), IGMP (протокол 2) и IP-in-IP (протокол 4) пакети. Може да се додадат TCP (протокол 6), UDP (протокол 17), SCTP (протокол 132) и други. Овој метод бара одговори користејќи го соодветниот протокол, или ICMP protocol unreachable пакети, и двете индицираат дека хостот е активен.
* ARP Ping (-PR) е често употребувана опција за скенирање локални ethernet мрежи. На повеќето LAN мрежи, особено тие кои користат рангови на приватни адреси, мнозинството IP адреси не се употребуваат во даден момент. Кога nMap праќа пакет како ICMP echo request, оперативниот систем мора да ја најде одредишната хардверска (MAC) адреса која одговара на дадената IP адреса. Ова е често проблематично поради тоа што соодветните библиотеки на опертативните системи не се напишани со очекување дека ќе треба да се користат за правење голем број ARP барања за кратко време. ARP скенирањето ги ги прави оптимизираните алгоритми на nMap одговорни за ARP барања. Доколку добие повратен одговор, на nMap не му се потребни IP-базираните ping пакети бидејќи веќе знае дека тој хост е активен. За Ipv6 (опција -6), -PR го користи IPv6 еквивалентот на ARP, Neighbor Discovery.
* --diable-arp-ping се користи доколку сакаме да го превенираме nMap од користење на ARP или ND ping во истражувањето на мрежата.
* --traceroute се користи за да се добие комбинација на порта и протокол најверојатна да стигне до целниот хост. Работи така што праќа пакети со низок TTL кој постепено се зголемува до првиот одговор кој не е ICMP Time Exceeded пакет.
* -n му кажува на nMap никогаш да не прави reverse DNS резолуција на активните IP адреси, и може значително да го скрати времето потребно за скенирање.
* -R му кажува на nMap секогаш да прави reverse DNS резуолуција на целните IP адреси, без разлика дали се активни.
* --system-dns, специфицира користење на системскиот DNS resolver со сериски, еден по еден повици за разлика од преддефинираниот паралелен nMap resolver. Оваа опција може значително да го зголеми времето потребно за скенирање и се користи само доколку се најде грешка во resolver-от на nMap.
* --dns-servers е опција која се користи за рачно подесување на DNS серверите кои ги користи nMap, односно преддефинираните DNS-ови на системот.

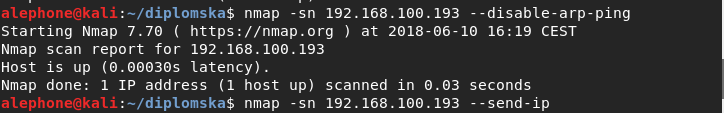
## 5.2 Скенирање на порти

Скенирањето на порти е вообичаено првиот интрузивен чекор на еден напад. Преку скенирање на порти напаѓачот може да добие информации околку состојбата на портите на хостовите, сервисите кои работат на нив, како и многу подетална слика на инфраструктурата на мрежата [4]. Неколку типични форми на скенирање на порти се дадени во продолжение:

* TCP SYN Scan (-sS) е најпопуларниот nMap скен. TCP SYN скенот е брз и релативно тивок. Праќа SYN пакети, и очекува SYN/ACK пакет ако портата е отворена или RST пакет ако портата е затворена. Доколку не добие никаков одговор или ICMP unreachable одговор, портата се смета за филтрирана. Било кој пакет добиен како одговор на SYN скен се отфрла, така што во случајот да се добие SYN/ACK, TCP врската никогаш не е воспоставена, и на ова се должи брзината и релативната тешкотија за детекција на овој тип на скенирање.
* TCP Connect Scan (-sT) во главно се користи кога TCP SYN скен не е опција, обично поради недостаток на привилегии на корисникот да праќа raw пакети, или ако скенирањето се извршува од Windows кој нема подршка за raw пакети. TCP Connect скенирањето го користи оперативниот систем за воспоставување на врска со хостот на портата за која е заинтересиран, е полесен за детекција од TCP SYN скенот, како и побавен.
* UDP Scan (-sU), праќа UDP пакет на секоја целна порта. За некои порти како 53 (DNS), 161/162(SNMP) пакетот содржи информации специфични за протоколот со цел да зголеми рата на добиени одговори, но за повеќето порти пакетот е празен освен ако не се назначи поинаку. Ако се добие ICMP Port Unreachable Error (тип 3, код 3), портата се смета за затворена, додека други ICMP грешки (тип 3, код 0, 1, 2, 9, 10 или 13) резултираат во маркирање на портата како филтрирана. Доколку сервис одговори со UDP пакет, портата се смета за отворена. Ако не се добие никаков одговор портата е класифицирана како отворена|филтрирана, односно, се смета дека портата можеби е отворена, но пакет филтри ја блокираат комуникацијата. Детекција на верзии (-sV) може да се искористи за да се диференцираат отворени од филтрирани порти.
* SCTP INIT Scan (-sY) е сквивалент на TCP SYN скенирање во SCTP протоколот. Овој скен праќа INIT пакет за започнување на врска и очекува INIT-ACK одговор доколку портата е отворена или ABORT доколку портата е затворена. Доколку не добие одговор, или добие ICMP Unreachable Error, портата се смета за филтрирана.
* TCP NULL (-sN), FIN (-sF), Xmas (-sX) скенирања го користат дизајнот на TCP за диференцирање отворени од затворени порти. Според RFC 793, доколку одредишната порта е затворена, за секој дојдовен пакет кој не е RST пакет, треба да се прати RST одговор, а понатаму вели дека доколку некако се добие пакет во кој ниеден од SYN, ACK или RST битовите не е поставен на 1 тој треба да се отфрли. NULL скенирањето ги поставува сите знаменца на 0, FIN скенирањето го поставува FIN битот на 1, додека Xmas скенирањето ги поставува FIN, PSH и URG битовите на 1. Доколку одредишниот систем го следи RFC 793, за било кој пакет кој не ги содржи SYN, RST или ACK битовите ќе биде одговорено со RST доколку портата е затворена или нема да биде одговорено воопшто доколку портата е отворена (односно пакетот ќе биде отфрлен). Предностите на ова скенирање се тоа дека пакетите можат да поминат одредени firewall-и, и во некои случаи скенирањето е уште потешко за детекција од TCP SYN скенирање. Негативната страна на ова скенирање е тоа дека многу системи не го следат RFC 793 и одговараат на вакви пакети со RST без разлика дали портата е отворена, што прави сите порти да се маркираат како затворени. Меѓу овие системи се Windows, многу Cisco уреди, BSDI и IBM OS/400. Истотака овие скенирања не можат да разликуваат отворени порти од филтрирани порти бидејќи не можат да знаат дали пакетот е отфрлен од сервисот или од firewall-от.
* TCP ACK Scan (-sA) е скенирање со кое се праќаат пакети со ACK знаменце поставено на 1 и не служи за откривање отворени порти туку за мапираше на правила на firewall, и одредување дали тие внимаваат да состојбата на врската. Без разлика дали портата е отворена или не, одговорот би бил RST и портата е маркирана како unfiltered. Порти кои не враќаат одговор, или враќаат ICMP пораки за грешка се маркираат како filtered.
* TCP Window Scan (-sW) е речиси идентичем како TCP ACK скенот, со разликата дека Window Scan-от користи имплементациски детал во некои системи за да разликува оворени од затворени порти преку испитување на Window полето на повратните RST пакети. Некои системи враќаат позитивна вредност во Window полето за отворени порти и 0 за затворени. Сепак ова се многу мал број на системи и Window скенирањето не е често доверливо.
* TCP Maimon Scan (-sM) е тип на скенирање именувано по Уриел Маимон, неговиот пронаоѓач и е идентично со TCP NULL, FIN и Xmas скенирањата, освен тоа дека пакетот кој се праќа ги има FIN и ACK битовите подесено на 1. Според RFC 793 на ова треба да биде одговорено со RST пакет, но Маимон забележал дека многу системи наследници на BSD само го отфрлаат пакетот доколку портата е отворена.
* --scanflags е опција со која може да се дизајнираат TCP пакети со знаменца поставени по избор. Пример, --scanflags URGACKPSHRSTSYNFIN праќа пакет со сите знаменца поставени на 1. Додатно на поставувањето на знаменцата може да се специфицира и базичен тип на скенирање како -sA или -sF што му кажува на nMap како да ги интерпретира резултатите. Доколку не се специфицира поинкау TCP SYN скенирањето се користи како базичен тип.
* SCTP COOKIE ECHO (-sZ) e SCTP скенирање кое го искористува тоа дека SCTP имплементации би требало да отфрлат COOKIE ECHO пакети на отворени порти без одговор, но да одговорат со ABORT доколку портата е затворена. Предноста на ова скенирање е тоа дека не е толку очигледно како SCTP INIT скенирање, како и тоа дека може да помина низ firewall-и кои не внимаваат на состојбата на врската. Негативната страна е дека ова скенирање на може да разликува меѓу оворени и филтрирани порти.
* TCP Idle (Zombie) Scan (-sI <zombie host>[:probe-port], поинаку познато како слепо скенирање, е најтивкиот начин на скенирање најден досега благодарение на тоа дека ниеден пакет до одредишниот компјутер не се праќа од скенирачкиот компјутер. Секој IP пакет во себе содржи поле за идентификација (IP ID), и кај многу имплементации ова поле е интегер кој се инкрементира за еден со секој пратен пакет. Idle скенирањето работи така што праќа пакет (SYN/ACK вообичаено со nMap) кон трет компјутер (zombie хост), и очекува пакет назад (RST пакет кој го праќа третиот компјутер како одговор на неочекуваниот SYN/ACK). Овој одговор го содржи IP ID бројот на zombie хостот. Откако ќе го добие одговорот скенирачкиот компјутер креира SYN пакет со IP header изворна адреса на zombie хостот и го праќа до одредишниот компјутер. Доколку SYN пакетот наиде на отворена порта, одредишниот компјутер му враќа SYN/ACK на zombie хостот мислејќи дека треба да се отвори TCP врска со него. Поради тоа што за zombie хостот SYN/ACK пакетот е неочекуван, тој праќа RST кон одредишниот компјутер, и со тоа си го инкреметира IP ID полето за еден. Доколку SYN пакетот наиде на затворена порта, одредишниот компјутер праќа RST пакет на zombie хостот кој е неочекуван и игнориран, односно zombie хостот не одговара никако и неговиот IP ID останува непроменет. Доклку SYN пакетот наиде на филтриран порт кај одредишниот компјутер, тој е отфрлен и не се одвива никаква комбинација меѓу одредишниот компјутер и zombie хостот, што пак го остава неговото IP ID поле непроменето. Понатаму, скенирачкиот компјутер повторно праќа SYN/ACK пакет кон zombie хостот, на кој zombie хостот му одговара со RST пакет кој го содржи моменталниот IP ID. Ако сега IP ID полето е за еден поголемо од што беше во првиот чекор значи дека zombie хостот не пратил пакет кон одредишниот компјутер и моментално скенираната порта е затворена или филтрирана. Спротивно, ако IP ID полето е поголемо за два, значи дека моментално скенираната порта кај одредишниот компјутер е отворена. Две големи предности на ова скенирање се анонимноста (ниеден пакет пратен кон одредишниот компјутер не ја содржи адресата на скенирачкиот компјутер) и можноста да се најде zombie хост чии пакети можат да поминат повеќе правила и филтри на firewall-от и така скенирањето да врати повеќе информации. Наоѓање zombie хост што ги исполнува критериумите (да има IP ID поле кое глобално се инкрементира за еден со секој пратен пакет, и да биде моментално мрежно неактивен така што други пакети не би го инкрементирале IP ID бројачот) може да биде проблематично. Друга негатина страна на ова скенирање е тоа дека не може да разликува затворени од филтрирани порти.
* IP Protocol Scan (-s0) е скенирање кое овозможува дознавање кој IP протоколи (TCP, ICMP, IGMP итн.) работат на одредишниот компјутер. Го постигнува ова со итеративно праќање IP пакети со различни вредности во протокол полето во IP header-от, и бара ICMP Protocol Unreachable Error (тип 3, код 2) во одговорот за да го маркира протоколот како затворен. Доколку добие ICMP Port Unreachable (тип 3, код 3) го маркира протоколот како отворен, а други ICMP Unreachable одговори (тип 3, код 0, 1, 9, 10 или 13) значат дека протоколот е филтриран. Сите овие одговори значат дека ICMP протоколот е активен.
* FTP Bounce Scan (-b <FTP relay host>) е скенирање кое се базира на постара спецификација на FTP протоколот (RFC 959) која овозможува FTP врски преку proxy. Ова дозволува да му кажеме на FTP сервер да го препрати нашиот фајл на друг компјутер. Грешките кои ги добиваме како одговор обично специфицираат дали портата на одредишниот компјутер е отворена или затворена. Иако често користена опција во времето кога е првпат вградена во nMap, во 1997 година, денес речиси воопшто не се користи бидејќи е речиси невозможно да се најде FTP сервер кој го дозволува ова.

## 5.3 No Ping Scan и генерирање на оптимални IDS/IPS правила

Праќаме No Ping Scan со оневозможен ARP Ping (за да го присилиме nMap да прати IP пакет на локален ethernet) на адресата 192.168.100.193 со наредбата nMap -sn 192.168.100.193 --disable-arp-ping (слика 6)

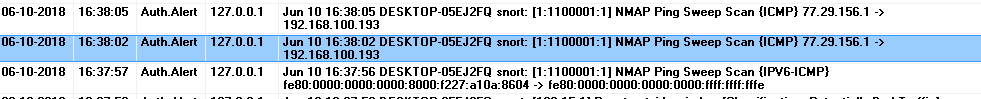
*Слика 6. No ping скенирање*

Во конфигурацијата на Snort го додаваме правилото:

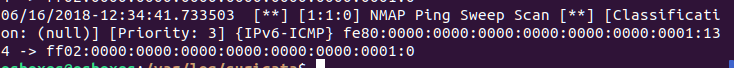
аlert icmp any any -> 192.168.100.193 any (msg:"NMAP Ping Sweep Scan"; dsize: 0; sid:1100001; rev: 1)

Соодветното, во конфигурацијата на Suricata го додаваме правилото:

alert icmp any any -> any any (msg:"NMAP Ping Sweep Scan"; sid:1;)

******кое му кажува на IDS-от да запише alert со порака “NMAP Ping Sweep Scan” доколку добие ICMP пакет на адресата 192.168.100.193 на било која порта со податочен сегмент со големина 0 (слика 7 и 8).

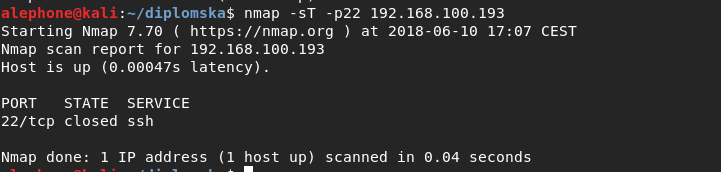
*Слика 7. Snort запис за No Ping скенирање*



*Слика 8. Suricata запис за No Ping скенирање*

## 5.4 TCP Connect Scan и генерирање на оптимални IDS/IPS правила

Можеме да го скенираме целиот компјутер (192.168.100.193 во овој пример) со TCP Connect Scan така што му кажуваме на nMap да воспостави TCP врска на различни порти со наредбата: nmap -sT 192.168.100.193 (слика 9). Портата 22 (ssh) е често од интерес и е дел од преддефинираниот -sT скен

*Слика 9. TCP Connect скенирање*

Во Snort конфигурацијата го додаваме правилото:

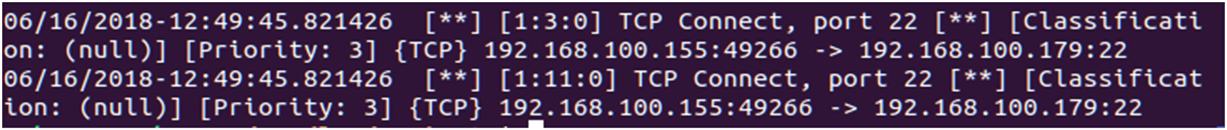
alert tcp any any -> 192.168.100.193 22 (msg:"TCP Connect, port 22"; sid:1100001; rev: 1)

Соодветното правило за Suricata:

alert tcp any any -> any 22 (msg:"TCP Connect, port 22"; sid:3;)

******кое му кажува на IDSот да запише alert доколку се оствари TCP врска на порта 22 со поракта "TCP Connect, port 22" (слика 10 и 11). Ова правило е корисно за сите адреси на кои не се очекува SSH врска. Доколку се очекува SSH врска од одредени адреси, тие може да се наведат и пропуштат.

*Слика 10. Snort запис за TCP Connect скенирање*



*Слика 11. Suricata запис за TCP Connect скенирање*

## 5.5 Xmas Scan и генерирање на оптимални IDS/IPS правила

Во конфигурацијата на Snort ги додаваме правилата:

alert tcp any any -> any 22 (msg:"Xmas Scan, faten na port 22"; flags: FPU; sid:1100003; rev: 1)

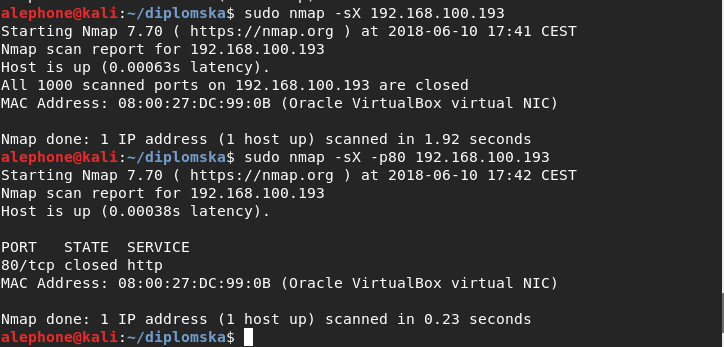
alert tcp any any -> any 80 (msg:"Xmas Scan, faten na port 80"; flags: FPU; sid:1100004; rev: 1)

Соодветните Suricata правила:

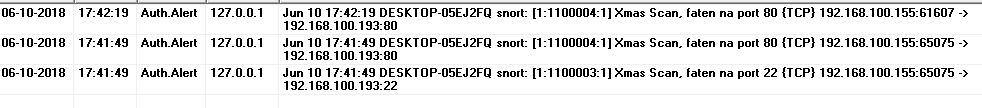
alert tcp any any -> any 22 (msg:"Xmas Scan, faten na port 22"; flags: FPU; sid:4;)

alert tcp any any -> any 80 (msg:"Xmas Scan, faten na port 80"; flags: FPU; sid:5;)

Овие правила му кажуваат на IDSот да запише alert (слика 13 и 14) во случај да наиде на TCP пакет со FIN, PSH и URG знамеца поставени на 1, а SYN, ACK, RST знаменца поставени на 0, односно пакет кој нема никаква употреба во TCP врска, освен Xmas скенирање. Во овој случај nMap заклучува (неточно) дека сите порти се затворени бидејќи кернелот ги отфрла овие пакети без да врати RST без разлика дали портата е отворена или не.

Користиме sudo поради тоа што за Xmas, FIN и NULL скенирања мораме да креираме raw IP пакети со соодветни знаменца, а само root корисник има пристап до raw пакети

*Слика 12. Xmas скенирање*

******Користиме sudo поради тоа што за Xmas, FIN и NULL скенирања мораме да креираме raw IP пакети со соодветни знаменца, а само root корисник има пристап до raw пакети

*Слика 13. Snort запис за Xmas скенирање*

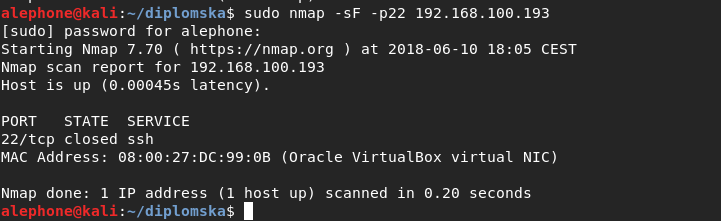
******

*Слика 14. Suricata запис за Xmas скенирање*

## 5.6 FIN Scan и генерирање на оптимални IDS/IPS правила

Кај FIN скенирањето сите TCP битови освен FIN се поставени на 0. Со nMap праќаме FIN пакет на адреса 192.168.100.193 на порта 22 со наредбата

nmap -sF -p22 192.168.100.193



*Слика 15. FIN скенирање*

За да го фатиме овој пакет на порта 22 (или било која друга порта со заменување на port вредноста) во конфигурацијата на Snort го додаваме правилото

alert tcp any any -> any 22 (msg:"FIN Scan, faten na porta 22", flags: F; sid:1100004; rev: 1)

Соодветно, во конфигурацијата на Suricata:

alert tcp any any -> any 22 (msg:"FIN Scan, faten na porta 22"; flags: F; sid:6;)

Со ова, Snort или Suricata ќе запишува alert со порака "FIN Scan, faten na porta 22" (слика 16 и 17) секој пат кога ќе добие ваков пакет

******

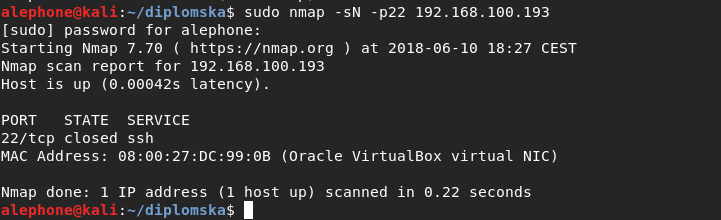
*Слика 16. Snort запис за FIN скенирање*



*Слика 17. Suricata запис за FIN скенирање*

## 5.7 NULL Scan и генерирање на оптимални IDS/IPS правила

Сите знаменца (SYN, ACK, RST, URG, PSH, FIN) кај NULL скенирањето се поставени на 0. Праќаме пакети со наредбата sudo nmap -sN 192.168.100.193



*Слика 18. NULL скенирање*

За да фатиме вакви пакети во Snort (слика 19) конфигурацијата додаваме правило:

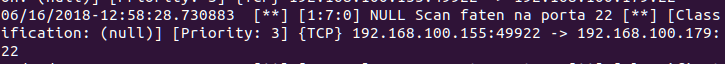
alert tcp any any -> any 22 (msg:"NULL Scan faten na porta 22"; flags:0; sid:1100004; rev: 1)

Додека во конфигурацијата на Suricata (слика 20):

alert tcp any any -> any 22 (msg:"NULL Scan faten na porta 22"; flags:0; sid:7;)

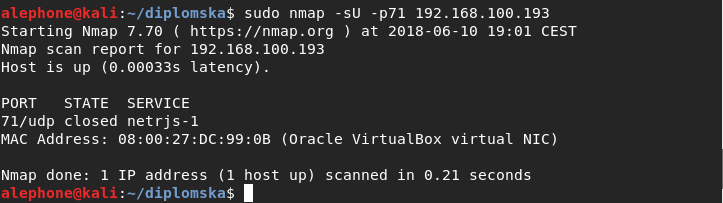
Со ова IDSот бара пакети со сите знаменца поставени на 0, и запишува alert доколку наиде на такво.

*Слика 19. Snort запис за NULL скенирање*

****** *Слика 20. Suricata запис за NULL скенирање*

## 5.8 UDP Scan и генерирање на оптимални IDS/IPS правила

Ова скенирање проверува дали има UDP сервис на одредена порта на активен хост. Наредбата во nMap за проверка на порта 71 на хост 192.168.100.193 е nmap -sU -p71 192.168.100.193 (слика 21). Повторно користиме sudo за да му дозволиме на nMap да креира raw пакети.

*Слика 21. UDP скенирање*

Во конфигурацијата на Snort можеме да внесеме правило:

alert udp any any -> any any (msg:"UDP Scan"; sid:1100005; rev: 1)

А во конфигурацијата на Suricata внесуваме:

alert udp any any -> any 71 (msg:"UDP Scan"; sid:8;)

Ова правило ќе опфати UDP скенирање на сите порти. Проблемот со ова правило е дека вообичаено било која мрежа има голем волумен на UDP сообраќај и ќе предизвика огромен број на alert-и. Ова може да се подобри со внесување на адреси од кои не е сомнително да се добие UDP пакети (пример, адресите кои спаѓаат на локалната мрежа), но ова остава простор напаѓач да искористи инаку доверлив компјутер за да го изврши скенирањето (иако, веројатноста за ова е ниска, исто како и веројатноста UDP скенирање да открие нешто што напаѓач кој е веќе во мрежата не може да го дознае со други методи). Друго решение е правилото да слуша на порта на која вообичаено не добиваме UDP сообраќај, но напаѓач сепак би ја скенирал.

Пример правило на порта 71 во Snort е

alert udp any any -> any 71 (msg:"UDP Scan"; sid:1100005; rev: 1)

Истото правило во Suricata

alert udp any any -> any 71 (msg:"UDP Scan"; sid:8;

******

*Слика 22. Snort запис за UDP скенирање*

******

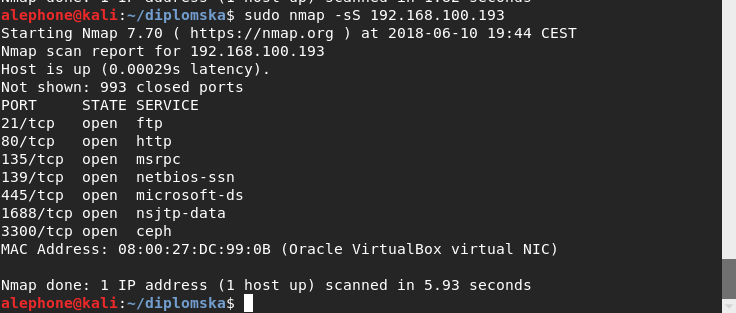
*Слика 23 Suricata запис за UDP скенирање*

## 5.9 SYN Scan и генерирање на оптимални IDS/IPS правила

SYN, или полу-отвореното скенирање, наречено така поради тоа што никогаш не воспоставува целосна TCP врска со одредишниот компјутер е познатото, тивко скенирање. Во nMap може да го искористиме овој тип на скенирање на адресата 192.168.100.193 со наредбата:

nmap -sS 192.168.100.193

Во овој случај повторно ни требаат root привилегии (или sudo) за да ни овозможат да не го користиме системскиот connect повик кој би ја комплетирал врската. Наместо тоа nMap креира SYN пакет и го анализира одговорот (SYN/ACK, RST или никаков) за да ја одреди состојбата на портата (отворена, затворена или филтрирана)



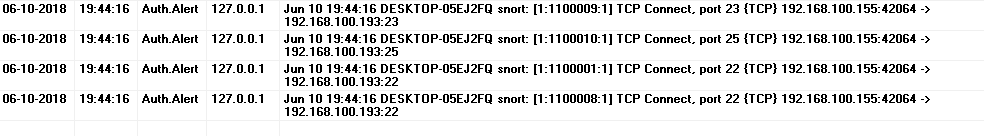
*Слика 24. SYN скенирање*

Во конфигурацијата на Snort додаваме еквивалентни правила како за TCP Connect скенирањето:

alert tcp any any -> any 22 (msg:"TCP Connect, port 22"; sid:1100008; rev: 1)

alert tcp any any -> any 23 (msg:"TCP Connect, port 23"; sid:1100009; rev: 1)

alert tcp any any -> any 25 (msg:"TCP Connect, port 25"; sid:1100010; rev: 1)

******Овие правила запишуваат alert-и за скенирањето

*Слика 25. Snort запис за SYN скенирање*

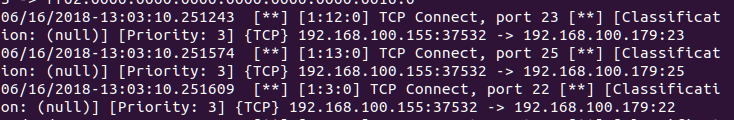
Во конфигурацијата на Suricata додаваме еквивалентни правила:

alert tcp any any -> any 22 (msg:"TCP Connect, port 22"; sid:11;)

alert tcp any any -> any 23 (msg:"TCP Connect, port 23"; sid:12;)

alert tcp any any -> any 25 (msg:"TCP Connect, port 25"; sid:13;)

Suricata еквивалентиот запис е прикажан на слика 26.

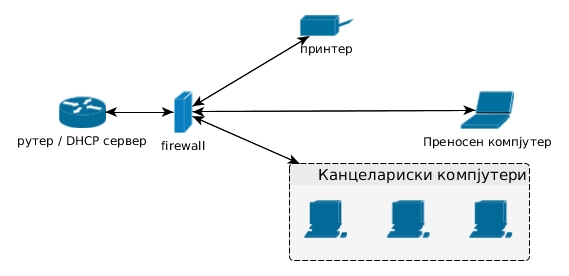
*Слика 26. Suricata запис за SYN скенирање*

SYN скенирањето може да помине незабележано од лошо конфигурарани (и вообичаено, постари) firewall-и кои обрнуваат внимание на состојбата на врската, но бидејќи тука употребените Snort правила едноставно ги забележуваат влезните TCP пакети од одредени адреси на одредени порти (било која надворешна адреса, порта 22, 23 и 25 во примерот), Snort и Suricata ги забележуваат SYN пакетите пратени од nMap.

# 6. Креирање на оптимални модели на IDS правила за различни мрежни околини

## 6.1 Мрежи на мали и средни организации и соодветни правила

Компјутерските мрежи на мали и средни организации најчесто со состојат од еден излез кон надворешна мрежа односно рутер кој е истотака и DHCP сервер и има конфигурирана Access Control List, што му овозможува да служи како примитивен firewall, канцелариски компјутери, статични и преносни, како и принтер или слични уреди. Како сите овие елементи се поврзани е прикажано на слика 27.



*Слика 27. Компјутерска мрежа на мала организација*

Во мрежа на ваква организација можеме да претпоставиме дека има мал број на мрежни протоколи кои е потребно да функционираат, како и мал број на отворени порти на кои треба да го следиме сообраќајот. Рутерот би ја имал улогата на Dynamic Host Configuration Protocol (DHCP) сервер, и ова би бил најефективниот начин за хостовите да добијат IP адреси [5]. DHCP работи на портите 67 на серверска страна и 68 на клиентска страна користејќи UDP пакети, па би можеле да внимаваме на пакети од други протоколи на овие порти, и под претпоставка дека 192.168.0.1 е адресата на рутерот, додека мрежата ни се наоѓа на мрежата 192.168.0.0/24 би можеле да искористиме:

alert tcp any any -> 192.168.0.1 67 (msg:"TCP Connect, port 67 na 192.168.0.1"; sid:..;)

alert icmp any any -> 192.168.0.1 67 (msg:"ICMP paket, port 67 na 192.168.0.1"; sid:..;)

alert tcp any any -> any 68 (msg:"TCP connect, port 68"; sid:..;)

alert icmp any any -> any 68 (msg:"ICMP paket, port 68"; sid:..;)

Овие четири правила би запишале alert во случај соодветните DHCP порти на клиентите и на рутерот да добијат TCP или ICMP пакет.

ICMP Echo протоколот би бил корисен за диагностика на мрежата при нејзино поставување или одржување, и како таков може да биде оставен активен, и со тоа портата седум отворена. Како и да е, надвор од повремената администраторска употреба, пакет добиен на оваа порта може да значи енумерација на мрежата или припремање на напад. Иако може да направи лажни аларми, доколку администраторот оцени дека има потреба да се надгледува мрежата подетално, би можел да искористи правила како:

alert ip any any -> any 7 (msg:"Ping!"; sid:..;)

Вакви правила би го алармирале администраторот во случај на ping на некој хост во мрежата, и на голем број техники за откривање на мрежата.

На портата 631 функционира Internet Printing Protocol, како и CUPS (Common Unix Printing Protocol) и би била веројатно отворена доколку постои потреба од принтање преку мрежа. IPP и CUPS користат TCP пакети за пренос на информации за принтање, и UDP пакети за наоѓање слободен принтер на мрежата. Соодветно администратор може да тестира за пакети од друг тип, или пакети кои доаѓаат надвор од мрежата со правилата:

alert icmp any any -> any 631 (msg:"ICMP paket na porta 631"; sid:..;)

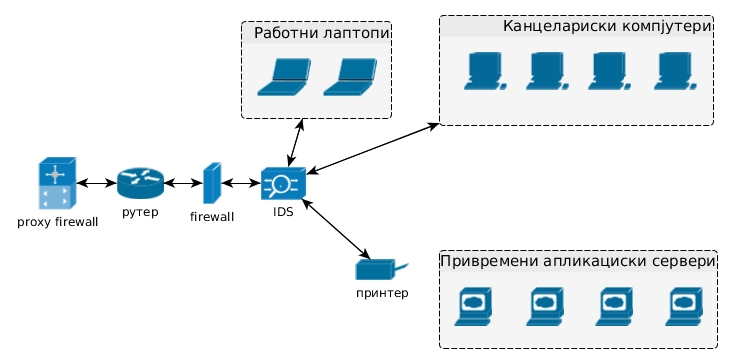
alert ip !192.168.0.1/24 → any 631 (msg:"Paket od nadvor or mrezata na porta 631"; sid..;)

Доколку постои потреба за печатење (принтање) преку интернет, истото ниво на безбедност би се постигнало со назначување на променлива, пр. $ALLOWED\_IPS со вредност, множеството на дозволени екстерни адреси и интерната мрежа. Би се добило правилото:

alert ip !$ALLOWED\_IPS any → any 631 (msg:"Paket od nedozvolena adresa na porta 631"; sid..;)

## 6.2 Мрежи на IT организации и соодветни правила

Во мрежата на организација која се занимава со информатични технологии би бил присутен firewall кој може да е посебен уред, или вграден во рутерот кој е излез до надворешна мрежа, работни компјутери, како и апликациски сервери кои би служеле во процесот на развивање софтвер. На слика 28 е претставува класична топологија на ваква мрежа.



*Слика 28. Компјутерска мрежа на IT организација*

Организации кои се занимаваат со информатички технологии, особено програмерски канцеларии би имале значително различна стратегија за одржување безбедност на мрежата. Често е веб-базирани апликации, посебно во време на развиток и тестирање, додека не е посветено доволно внимание на безбедноста на кодот да се хостираат на мрежата. Ова ја прави очигледна потребата од proxy firewall, кој работи на апликативно ниво на OSI моделот и би можел да ја штити апликацијата од напади како Cross Site Scripting и SQL инјекции [1]. Поради ваквите апликации би требало да има константна комуникација помеѓу програмерскиот тим и администраторот, така што системите за откривање и системите за превенирање напади би можеле да се конфигурираат соодветно на портите на кои работат апликациите и протоколите кои ги користат. Додатно на тоа, портата седум која се користи за ICMP Echo протоколот би се очекувало да биде многу почесто користена во ваква организација и нејзино мониторирање може да е многу помалку исплатливо. FTP протоколот истотака може да е често користен, така што, портата 21 (или 22, за sftp протоколот) би требало да биде мониториран, но во главно за ICMP или UDP пакети, кои би наговестувале испитувања за статусот на портата и сервисот кој работи на неа:

alert icmp any any -> any 21 (msg:"ICMP paket na porta 21"; sid:..;)

alert udp any any -> any 21 (msg:"UDP paket na porta 21"; sid:..;)

Портата 22 би можела да биде отворена за SSH и sftp сервисите и на неа може да се надгледува за UDP, ICMP пакети, како и пакети од недозволени адреси:

alert icmp any any -> any 22 (msg:"ICMP paket na porta 22"; sid:..;)

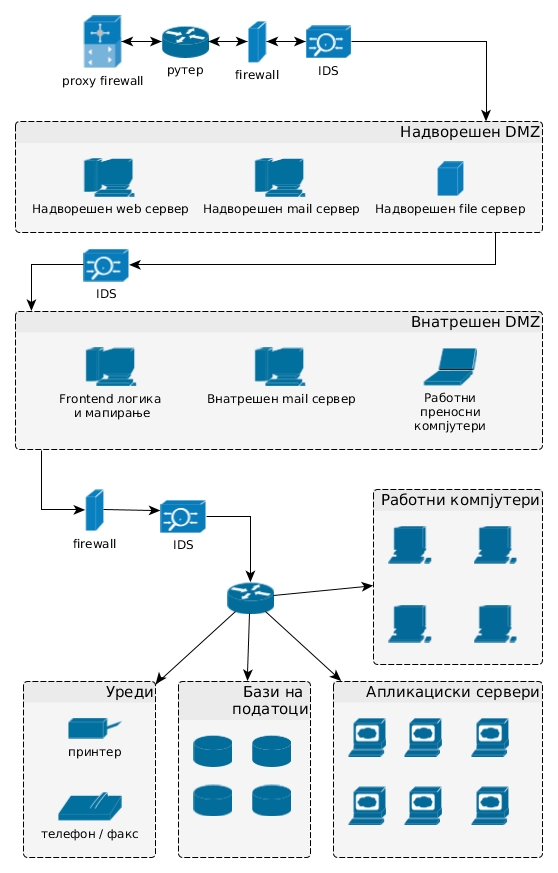
alert udp any any -> any 22 (msg:"UDP paket na porta 22"; sid:..;)

alert ip !$ALLOWED\_IPS any -> any 22 (msg:"Paket od nedozvolena adresa na porta 22"; sid:..;)

Соодветните порти за користените системи за менаџирање со бази на податоци би требало да бидат отворени за интерната мрежа, и документацијата на користениот систем би требало да биде разгледана за да се имаат во предвид протоколите кои се дел од нормалната функција.

## 6.3 Мрежи на средни и големи организации и соодветни правила

Средните и големи огранизации би имале корист од следење на правилата за DHCP, ICMP Echo и IPP/CUPS исто како помалите огранизации. Веројатно е дека поголем број вакви организации би имале софтвер и апликации кои работат на нивната мрежа, пример, универзитетски кампус би имал системи за учење на далечина, софтвер за администрација на ресурси и книговодствен софтвер би нашле честа примена во средни и поголеми организации. Топологија на мрежа на ваква организација е прикажана на слика 29.



*Слика 29. Компјутерска мрежа на голема организација*

Покрај задолжителното присуство на firewall и често корисниот систем за откривање напади, поделба во виртуелни мрежи (VLAN) и постоење на демилитаризирана зона (DMZ) би биле релативно евтини, но ефективни безбедносни чекори. Како додатна предност, ова би му дозволило на администраторот да знае точно каков пакет би можел да очекува на кој сегмент од мрежата, така што тој би можел да очекува пакети кои соодветствуваат на портите и протоколите кој ги користи книговодствениот софтвер само во мрежата на која работи книговодството и така се секој сегмент каде што овој принцип е применлив. За разлика од организациите кои се занимаваат со информатика, бројот на сервиси кои би работеле на мрежата не се менува често. Консултирање со прирачниците за сервисите кои би работеле постојано на мрежата би требало да го овозможи администраторот да напише соодветни правила за таа мрежа.

Реално е дека средна или голема огранизација би имала потреба од често префлување бинарни датотеки. Да кажеме во случајот на универзитетски кампус дека имаме адреси од канцелариите на професорите сместени во променлива $TEACHERS, и дека само тие би можеле да додаваат датотеки преку FTP на системот за учење на далечина, чија адреса е сместена во $ELEARNING. Правилото:

alert ip !$TEACHERS any → $ELEARNING 21(msg:”Unauthorized FTP”; sid:..;)

би го известило администраторот за обид за воспоставување FTP врска до системот за учење на далечина од адреса која не припаѓа на професор.

Слично, доколку адресите на сметководителите се сместени во променлива $BOOKKEEPING, и тие се единстевни кои смеат да праќаат POST request-и на веб-базираниот софтвер за сметководство чија адреса е сместена во $BKKP\_SW, правилото:

alert ip !$BOOKKEEPING any → $BKKP\_SW 80(msg:”Unauthorized POST ”; content:”POST”; nocase; sid:..;)

би го известило администраторот доколку дојде таков request од недозволена адреса.

Средна или голема организација би можела да има сопствен DNS (лоциран на $DNS) за интерни потреби, и администраторот би можел да согледа пакети пратени од надвор од организацијата (чии адреси се сместени во $ORG) со правилото

alert ip !$ORG any → $DNS any (msg:”Foreign packet”; sid:..;)

како и пакети пратени од било каде на порта различна од портата 53, користена од DNS со правилото:

alert ip any any → $DNS !53(msg:”Packet at wrong port”; sid:..;)

Доколку организацијата има сопствен mail (кој работи на адресата сместена во $MAIL) сервер, би сакале да ги надгледуваме и портите 25 (Simple Mail Transfer Protocol), како и 110 (Post Office Protocol, version 3), кои се користат во праќање и примање електронска пошта:

alert udp any any → $MAIL 25(msg:”UDP paket”; sid:..;)

alert icmp any any → $MAIL 25(msg:”ICMP paket”; sid:..;)

alert udp any any → $MAIL 110(msg:”UDP paket”; sid:..;)

alert icmp any any → $MAIL 110(msg:”ICMP paket”; sid:..;)

Овие правила ќе пропуштат TCP пакети, со кои работат SMTP и POP3 на соодветните порти, но ќе го алармираат администраторот за пакети на други протоколи.

Ваквите принципи не би биле соодветни за телекомуникациски мрежи кои користат PPP и други протоколи, но би можеле да се искористат во сегменти на мрежата кои функционираат како вообичаена мрежа. Фирми кои нудат хостинг услуги, би имало потреба да додадат многу специфични правила во зависност од нивната инфраструктура. Истотака, појавата и големата популарност на cloud архитектури значително ја менува големата слика на безбедноста. Од аспект на организацијата корисник, работата да се чува мрежата безбедна се намалува рапидно со зголемување на бројот на сервиси и апликации кои работат на cloud. Од аспект на операторите на cloud платформата, безбедноста на имплементацијата станува многу подинамично прашање кое многу зависи од платформата.

# 7. Заклучок

Системите за откривање напади му даваат на администраторот на одредена компјутерска мрежа дополнителна шанса да превенира напад во првите негови чекори, како и најбрзо да ја санира штетата направена од веќе извршен напад. Од аспект на организација, ова може да спречи големи загуби на капитал, работен тек и доверба. Функционалноста на ваков систем, односно вредноста која би ја допринел на одредена организација многу зависи од неговата конфигурација и администрација. Системите за откривање напади базирани на множество правила вообичаено доаѓаат со голема група преддефинирани правила и е честа пракса оваа група правила да се остави во употреба. Речиси без исклучок поголемиот дел преддефинирани правила не би можеле да се применат на одредена мрежа поради ограничениот број на сервиси кои работат не неа. Непотребното споредување на пакети со шеми дефинирани со правила кои не би се искористиле може да доведе до прекумерно користење на уредите и бавност на мрежата, голем број на лажно позитивни резултати кои би му ја отежнале работата на администраторот, непотребно отфрлање на пакети особено во случајот на системи за превенирање напад, или нецелосна анализа на сите пакети кои поминуваат. Во оваа дипломска работа анализиравме општи случаи на различни мрежи и покажавме дека за голем дел од апликациите и услугите кои би можеле да функционираат на мрежата може да се направи прелиминарна слика. Според таа слика може да се предвиди протокот на мрежните пакети и соодветно да се напишат правилата на системот за откривање напади што би водело кон значително подобрување во неговата функционалност. Во дипломската работа се наведени три општи множества правила кои би можеле да служат како модели за понатамошно развивање на правила кои би ги задоволиле потребите на специфична мрежа.

Иако инсталирање систем за откривање напади во cloud, како додатно ниво на безбедност во моментов е малку истражувана идеја, покажано е дека стандардните правила работат и во ваква околина. Специфично пишување правила кои би биле оптимизирани за таква околина е тема која е спомната, но малку истражена. Би можеле да хипотетизираме дека правецот во кој би се движеле IDS правилата во cloud би вклучувал генерација на нови правила со креирање нови корисници, како и со промени на нивните привилегии така што системот за откривање напади би функионирал слично на сервер за утврдување идентитет кој постои на многу вакви платформи, но независен од апликациите, и апликативната безбедносна логика на cloud платформата.

# 7. Користена литература

[1] Glen E. Clark, CompTIA Security+ Study Guide, Third Edition, McGraw-Hill Education, 2018

[2] Sean-Philip Oriyano, CEH v9 - Certified Ethical Hacker Version 9 Study Guide – 3E, Third Edition, Sybex, 2016

[3]  Shahid Anwar, Jasni Mohamad Zain , Mohamad Fadli Zolkipli , Zakira Inayat, Suleman Khan, Bokolo Anthony and Victor Chang, From Intrusion Detection to an Intrusion Response System: Fundamentals, Requirements, and Future Directions, Multidisciplinary Digital Publishing Institute, 2016

[4] Peter Kim, The Hacker Playbook 3 Practical Guide To Penetration Testing, Independently published, 2018

[5] James F. Kurose, Keith W. Ross, Computer Networking: A Top-Down Approach, Sixth Edition, Pearson, 2012

[6] Gordon Fyodor Lyon, Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning, Nmap Project, 2009