

ŠOLSKI CENTER ZA POŠTO, EKONOMIJO IN TELEKOMUNIKACIJE
LJUBLJANA

VIŠJA STROKOVNA ŠOLA

DIPLOMSKA NALOGA

MATEVŽ GROZNIK

Ljubljana, februar 2025



ŠOLSKI CENTER ZA POŠTO, EKONOMIJO IN TELEKOMUNIKACIJE
LJUBLJANA
VIŠJA STROKOVNA ŠOLA

DIPLOMSKA NALOGA

Ustvarjanje varnega domačega omrežja z uporabo Raspberry Pi

Diplomant: Matevž Groznik

Študijski program: Telekomunikacije

Mentor: mag. Alojz Hudobivnik

Lektorica: Manja Belina, mag. prof. slov. in mag. prof. špan.

Vpisna številka: 12130084435

Ljubljana, februar 2025

ZAHVALA

Zahvaljujem se mag. Alojzu Hudobivniku za sprejem mentorstva, strokovno pomoč pri pisanju naloge in usmerjanje pri celotnem procesu.

IZVLEČEK

Zaradi vedno večjega števila nevarnosti v svetovnem spletu potrebujemo nove rešitve za varno uporabo interneta. V diplomski nalogi bom predstavil, kako bi to rešil v domačem omrežju. Predstavil bom podrobno namestitev WireGuarda, Pi-hole in svoj program DDoS. Pi-hole deluje kot DNS strežnik, ki izboljša uporabniško izkušnjo in ščiti pred neželeno vsebino in potencialnimi grožnjami. WireGuard nam ponuja varen in šifriran dostop do omrežja tudi, ko nas ni doma. DDoS zaščito bom naredil v programskem jeziku Python, ki bo zagotavljal osnovno zaščito pred DDoS napadi. Poleg njihove namestitve bom z različnimi testi prikazal, kako aplikacije delujejo in ali so učinkovite. Aplikacije bom implementiral na nizkocenovno platformo Raspberry Pi, ki je znana po nizki porabi energije in široki dostopnosti.

Ključne besede: Pi-hole, DNS, DDoS, Raspberry Pi, VPN, WireGuard

ABSTRACT

Due to the increasing number of dangers in the Internet world, we need new solutions for safe use of the Internet. In my diploma thesis, I will present how I would solve this in the home network. I will present a detailed installation of WireGuard, Pi-hole and my DDoS program. Pi-hole acts as a DNS server that improves the user experience and protects against unwanted content and potential threats. WireGuard offers us secure and encrypted access to the network even when we are not at home. I will do DDoS protection in the Python programming language, which will provide basic protection against DDoS attacks. In addition to their installation, I will show how the applications work and whether they are effective with various tests. I will implement the applications on the low-cost Raspberry Pi platform, which is known for its low power consumption and wide availability.

Key words: Pi-hole, DNS, DDos, Raspberry Pi, VPN, Wireguard

KAZALO VSEBINE

1	UVOD.....	8
2	KIBERNETSKA VARNOST.....	9
2.1	NEVARNOSTI JAVNEGA WI-FI.....	9
2.2	ZAŠČITA VPN V JAVNEM OMREŽJU WI-FI.....	10
2.3	ZAŠČITA PROTI OGLASOM	11
3	RASPBERRY PI.....	12
3.1	UPORABA NAPRAVE RASPBERRY PI.....	13
3.2	OPERACIJSKI SISTEM.....	14
3.3	PROTOKOL SECURE SHELL (SSH).....	14
4	POŽARNI ZID	15
5	DOCKER	15
6	PI-HOLE	17
6.1.1	Blokiranje oglasov na ravni brskalnika	19
6.2	SISTEM DOMENSKIH IMEN (DNS).....	19
6.3	DELOVNAJE DNS	20
6.4	NAMESTITEV PI-HOLE	20
7	WIREGUARD VPN.....	28
7.1	PROTOKOL BREZ POVEZAVE.....	28
7.2	KRIPTOGRAFIJA.....	28
7.2.1	IZMENJAVA KLJUČEV	29
7.3	NAMESTITEV WIREGUARD VPN	30
7.3.1	PODOMREŽJA.....	39
7.4	TUNEL	40
7.4.1	WIRESHARK.....	40

8	ZAŠČITA DDOS.....	41
8.1	VRSTE NAPADOV DDOS.....	41
8.2	PROGRAM DDOS.....	42
8.3	TEST PROGRAMA.....	45
8.4	IZ SKRIPTA V PROGRAM.....	48
9	ZAKLJUČEK.....	50
10	VIRI IN LITERATURA.....	52

KAZALO SLIK

Slika 1:	Oglas Google.....	11
Slika 2:	Dostop prek SSH.....	14
Slika 3:	Dostop do Raspberry Pi.....	15
Slika 4:	Namestitev Dockerja.....	16
Slika 5:	Tekstovna datoteka .yaml.....	17
Slika 6:	Delovanje aplikacije Pi-hole.....	18
Slika 7:	Tekstovna datoteka Pi-hole .yaml.....	20
Slika 8:	Ustvarjanje tekstovne datoteke .yaml.....	21
Slika 9:	Preverjanje delovanja Pi-hole v Dockerju.....	21
Slika 10:	Izvedba izjeme v požarnem zidu.....	21
Slika 11:	Nastavitev Raspberry Pi kot primarni strežnik DNS.....	22
Slika 12:	Dostop do aplikacije Pi-hole.....	22
Slika 13:	Prva stran Pi-hole.....	22
Slika 14:	Spreminjanje gesla za Pi-hole.....	23
Slika 15:	Nadzorna plošča Pi-hole.....	23
Slika 16:	Pregled nad napravami Pi-hole.....	24
Slika 17:	Pregled Pi-hole nad povpraševanji DNS.....	25
Slika 18:	Dodajanje Pi-hole na črno listo.....	25
Slika 19:	Spletna stran za adliste.....	26
Slika 20:	Test za oglase pred Pi-hole.....	27

Slika 21: Test za oglase, ko Pi-hole deluje	27
Slika 22: Vmesnik WireGuard	30
Slika 23: Datoteka WireGuard .yaml	31
Slika 24: Koda QR za dostop do strežnika WireGuard VPN	31
Slika 25: Definiranje podomrežja .yaml	32
Slika 26: Podomrežja v omrežju	32
Slika 27: Dostop omrežja WireGuard VPN prek požarnega zida.....	33
Slika 28: Preusmerjanje priključkov v glavnem modemu	33
Slika 29: Preusmerjanje priključkov v drugem modemu	34
Slika 30: Prva stran WireGuard VPN.....	34
Slika 31: Stran WireGuard VPN za dodajanje uporabnikov.....	35
Slika 32: Dodajanje uporabnika WireGuard VPN	35
Slika 33: Pregled uporabnikov WireGuard VPN	35
Slika 34: Koda QR na spletni strani Wireguard VPN	36
Slika 35: Dodajanje uporabnika prek kode QR v aplikaciji.....	37
Slika 36: Dodajanje uporabnika v aplikaciji Wireguard VPN.....	37
Slika 37: Imenovanje tunela	37
Slika 38: Pregled povezave.....	37
Slika 39: Delovanje uporabnika	38
Slika 40: Tunel VPN.....	40
Slika 41: Kriptacija tunela VPN	40
Slika 42: Koda programa DDoS.....	43
Slika 43: Napad SYN Flood.....	45
Slika 44: Aplikacija Metasploit.....	46
Slika 45: Delovanje napada DDoS.....	46
Slika 46: Normalno delovanje programa za preprečevanje napada DDoS	47
Slika 47: Delovanje programa DDoS pod napadom DDoS.....	47
Slika 48: Iz skripte v program	48
Slika 49: Datoteka .service	48
Slika 50: Pregled delovanja v ozadju	49

SEZNAM UPORABLJENIH KRATIC

AES	Advance Encryption Standard (napredni standard šifriranja)
AVX	Advance Vector Extensions (napredne vektorske razširitve)
CPE	Osrednja centralno procesna enota
DDoS	Distributed Denial of Service (distribuirani napad z zavrnitvijo storitve)
DNS	Domain Name System (sistem domenskih imen)
GUI	Graphical User interface (grafični uporabniški vmesnik)
IP	Internetni protokol
OS	Operating System (operacijski sistem)
SSH	Secure Shell (varna lupina)
SYN	Synchronize (sinhroniziran)
TCP	Transmission Control Protocol (protokol za nadzor prenosa)
UDP	User Datagram Protocol (protokol uporabniškega datagrama)
VPN	Virtual Private Network (navidezno zasebno omrežje)

1 UVOD

S širjenjem digitalnega okolja so se tudi nevarnosti na internetu povečale, zato je varnost domačega omrežja postala ključnega pomena. V tej diplomski nalogi bom raziskal, kako z uporabo platforme Raspberry Pi ustvariti varno domače okolje. Za učinkovito izvajanje in ločevanje funkcij bom uporabil kontejnerje Docker. Na Raspberry Pi bom vzpostavil Pi-hole, program za filtriranje oglasov, in lastno rešitev za preprečevanje distribuiranih napadov z zavrnitvijo storitve (angl. Distributed Denial of Service, DDoS). Poleg tega bom implementiral navidezno zasebno omrežje (angl. Virtual Private Network, VPN), ki mi bo omogočalo varno povezovanje v domače omrežje tudi, ko sem zunaj doma. Z združevanjem teh elementov v kontejnerje Docker bom ustvaril celovito rešitev za varno in zanesljivo domače omrežje, ki bo omogočalo varno uporabo tudi, ko sem na poti.

2 KIBERNETSKA VARNOST

Kibernetska varnost je zaščita sistemov, omrežij in programov pred digitalnimi napadi, ki lahko ciljajo na občutljive informacije, izsiljujejo denar ali motijo poslovne procese. Izvajanje učinkovitih ukrepov kibernetske varnosti je zahtevno zaradi vse večjega števila naprav in inovativnih napadalcev. Uspešen pristop vključuje večplastno zaščito v računalnikih, omrežjih, programih ali podatkih. V organizacijah morajo ljudje, procesi in tehnologija sodelovati, da ustvarijo učinkovito obrambo pred kibernetskimi napadi. Enoten sistem za upravljanje groženj lahko avtomatizira integracije med varnostnimi izdelki in pospeši ključne operativne funkcije, kot so odkrivanje, preiskava in sanacija (povz. po: <https://www.kaspersky.com/resource-center/definitions/what-is-cyber-security/>).

2.1 NEVARNOSTI JAVNEGA WI-FI

Ko hekerji izkoristijo brezplačen javni Wi-Fi, sledijo našim osebnim podatkom. Glavni primeri dragocenih podatkov, ki bi lahko bili izpostavljeni z uporabo brezplačnega omrežja Wi-Fi, so:

- e-poštni podatki za prijavo,
- bančni podatki,
- osebne fotografije in videi,
- domači naslov.

Hekerji lahko izkoristijo javni Wi-Fi na različne načine:

- Napad Evil Twin: usmerjevalnike se da zlahka preimenujati v karkoli. Kibernetski kriminalci bodo ustvarili lažna omrežja Wi-Fi, da bi nas preslepili, da se povežemo. Verjetno bi se v hipu povezali z javnim omrežjem Wi-Fi, imenovanim »Mcdonalds_Wifi_Free«, ne da bi vedeli, da ga morda izvaja heker.
- Vbrizgavanje zlonamerne programske opreme: nezanesljiva povezava lahko povzroči, da zlonamerna programska oprema, ki jo je težko zaznati, zdrsne

v naš računalnik. Lahko zmanjša našo pasovno širino, poškoduje sistem in hekerjem zagotovi stranska vrata do vseh naših osebnih datotek.

- Človek v sredini (MITM): kibernetški kriminallec svojo napravo postavi med povezavo z našo napravo in javno dostopno točko Wi-Fi. To mu omogoča, da diskretno spremlja našo dejavnost in celo nadzira naš promet, kar nas lahko preusmeri na spletno mesto, ki so ga ustvarili, in nas zavede, da vnesemo svoje poverilnice ali bančne podatke.
- Vohanje Wi-Fi: to je bolj pasivno dejanje kot aktivni napad, kot je MITM. Z uporabo osnovne in legalne programske opreme lahko heker spremlja in beleži vse podatkovne pakete, ki gredo skozi omrežje Wi-Fi.

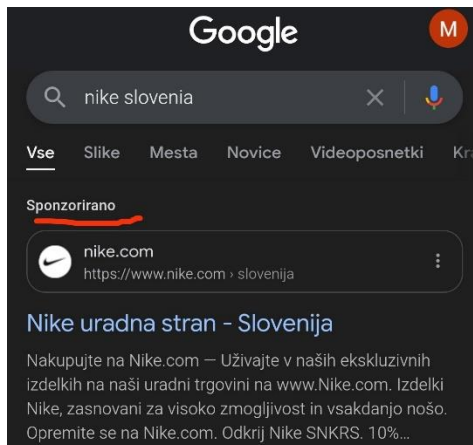
Z izboljšanjem javnega omrežja Wi-Fi z varno povezavo VPN bo večina teh napadov neuporabnih. Postopek šifriranja učinkovito ščiti vse naše podatke pred radovednimi očmi. Čeprav bo heker lahko videl, da smo povezani z internetom prek javnega omrežja Wi-Fi, ne bo mogel videti podrobnosti o prenesenem prometu, zato se priporoča uporaba VPN za Wi-Fi (povz. po: <https://www.aura.com/learn/dangers-of-public-wi-fi/>).

2.2 ZAŠČITA VPN V JAVNEM OMREŽJU WI-FI

VPN je orodje, ki varuje našo spletno dejavnost tako, da preusmeri našo internetno povezavo prek zasebnega strežnika, skrije naš pravi naslov internetnega protokola (IP) in zakrije našo dejavnost. Njegovi šifrirni protokoli zagotavljajo, da kdorkoli poskuša vohuniti za nami, ne vidi mimo VPN. Kljub temu, da ne uporabljajo varne povezave Wi-Fi, VPN zagotavljajo povezavo povsod in odpravljajo potrebo po zunanji zaščiti. Nevarna omrežja Wi-Fi zunaj domov so pogosto slabo zaščitena in kriminalci zlahka dostopajo do njih. Slabi akterji lahko tudi nastavijo svoje usmerjevalnike in razkrijejo zasebne podatke, ko se potnik nevede poveže z omrežjem (povz. po: <https://nordvpn.com/blog/securing-public-wi-fi/>).

2.3 ZAŠČITA PROTI OGLASOM

Eden večjih varnostnih problemov, ki sem jih opazil, so oglasi Google Search Engine. Google ponuja možnost plačila oglasa, ki se bo pokazal čisto na vrhu iskanja kot prvi rezultat. Označen bo z »ad«, vendar je to napisano z malimi črkami, da se z lahkoto spregleda.



Slika 1: Oglas Google

V Sloveniji se je tak problem pojavil s trgovino Nike, ki ima samo eno uradno spletno stran s spletnim naslovom nike.com. V oglasu se je kot prvi rezultat pojavil spletni naslov nike.si. Stran je bila identična kot originalna, le da so ponujali nekaj dodatnih popustov. V »vabo« je ugriznilo kar nekaj Slovencev, saj so jim ukradli osebne podatke in podatke bančne kartice.

Še bolj nevaren primer se je zgodil s programom za snemanje videov OBS. V tem primeru so si uporabniki program namestili na svoj računalnik in s tem predali dostop do svojega računalnika »slabim igralcem«. Na koncu so imeli uporabniki, ki so namestili ta program na svoj računalnik, »srečo v nesreči«, saj je šlo samo za »bitcoin miner«.

3 RASPBERRY PI

Raspberry Pi je majhen, cenovno dostopen računalniški sistem, ki ga je razvila fundacija Raspberry Pi Foundation v Veliki Britaniji. Gre za majhen računalnik velikosti kreditne kartice, ki ga je mogoče uporabiti za različne projekte, od učenja programiranja do avtomatizacije domačega okolja. Raspberry Pi je zelo prilagodljiv in je primeren tako za začetnike kot za izkušene uporabnike, zaradi česar je postal priljubljena platforma za številne aplikacije v domačem okolju, izobraževanju in industriji.

Prednosti uporabe Raspberry Pi in običajnega računalnika/strežnika za gostovanje aplikacij so (povz. po: <https://opensource.com/resources/raspberry-pi/>):

- Cenovna ugodnost: Raspberry Pi je cenovno dostopen računalniški sistem v primerjavi z običajnimi strežniki/računalniki. Za razliko od dražjih možnosti je Raspberry Pi cenovno dostopen tako za nakup strojne opreme kot za vzdrževanje.
- Nizka poraba energije: Raspberry Pi porabi zelo malo energije v primerjavi z običajnimi strežniki. Zaradi nizke porabe energije je Raspberry Pi energetsko učinkovit, kar zmanjšuje stroške obratovanja.
- Enostavnost uporabe in vzdrževanja: Raspberry Pi je majhen in prenosen, kar omogoča enostavno postavitve in namestitve v različna okolja. Poleg tega je upravljanje in vzdrževanje Raspberry Pi preprosto in ne zahteva obsežnega tehničnega znanja, kar je še posebej pomembno za manj izkušene uporabnike.
- Prilagodljivost: Raspberry Pi je odprtokodna platforma, ki omogoča prilagajanje različnim potrebam in zahtevam. S široko paleto operacijskih sistemov in aplikacij je Raspberry Pi primeren za gostovanje različnih vrst aplikacij, vključno s spletnimi stranmi, strežniki za igre, medijskimi centri in še več.
- Skupnost in podpora: Raspberry Pi ima veliko skupnost uporabnikov in razvijalcev, ki neprestano izboljšujejo in razvijajo nove projekte in aplikacije.

Zaradi tega je Raspberry Pi odlična izbira za tiste, ki iščejo podporo in skupnost za svoje projekte gostovanja aplikacij.

3.1 UPORABA NAPRAVE RASPBERRY PI

Verjetno ima vsak doma kakšen star računalnik, na katerega bi lahko samo naložil vse programe Linux, hkrati pa bi bil tudi močnejši kot Raspberry Pi. Glavni razlog, zakaj sem se odločil za uporabo Raspberry Pi, je poraba električne energije. Star računalnik ima še vedno približno 100 W porabe, medtem ko ima Raspberry Pi porabo zgolj 5 W.

Mesečna poraba z uporabo Raspberry Pi 5 W, če je kWh 0,21 €, tako znaša (<https://www.stat.si/StatWeb/Field/Index/5/30/>):

$$0,005 \text{ kW} \times 24 \text{ ur na dan} \times 30 \text{ dni} = 3,6 \text{ kWh}$$

Mesečni strošek:

$$3,6 \text{ kWh} \times 0,21 \text{ € na kWh} = 0,756 \text{ €}$$

Mesečna poraba z uporabo računalnika 100 W, če je kWh 0,21 €, tako znaša:

$$0,1 \text{ kWh} \times 24 \text{ ur na dan} \times 30 \text{ dni} = 72 \text{ kWh}$$

Mesečni strošek:

$$72 \text{ kWh} \times 0,21 \text{ € na kWh} = 15,12 \text{ €}$$

Razlika je torej za 14,36 €, kar je velik znesek, še posebej, če upoštevamo celo leto delovanja.

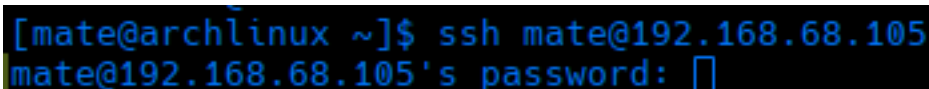
3.2 OPERACIJSKI SISTEM

Samoumevno je, da za operacijski sistem uporabljamo neko distribucijo Linuxa. Lahko bi uporabljali tudi Windowse, vendar je Linux stroškovno učinkovitejši in odprtokodni operacijski sistem zaradi svojih brezplačnih distribucij, visokih možnosti prilagajanja, varnosti, zmogljivosti, upravljanja programske opreme in paketov, združljivosti s programskimi jeziki, podpore skupnosti, nadzora zasebnosti, sistemskih posodobitev in idealnega primera uporabe. Prav tako je bolj varen, deluje bolje na starejši strojni opremi in omogoča preproste posodobitve sistema za razliko od sistema Windows, ki zahteva ponovni zagon sistema. Naložil sem Raspbian OS, ker je bil narejen za Raspberry Pi, lahko pa bi naložil tudi druge distribucije, kot so Ubuntu, Arch, Fedora itd.

3.3 PROTOKOL SECURE SHELL (SSH)

Protokol varna lupina (angl. Secure Shell, SSH) ščiti računalniške ukaze prek nezavarovanih omrežij s kriptografijo, omogoča tuneliranje in posredovanje vrat, zaradi česar je uporaben za nadzor oddaljenega strežnika, upravljanje infrastrukture in prenos datotek. Vse, kar sem izvajal na Raspberry Pi, sem delal prek SSH, ker se mi je zdelo veliko lažje delovati na svojem osebнем računalniku, ki je veliko hitrejši, zraven pa lahko iščem rešitve na internetu, če se pojavi kakšen problem. Povezava do Raspberry Pi s SSH je zelo enostavna, na svojem računalniku odpremo terminal in napišemo »ssh ime@ip_naslov«.

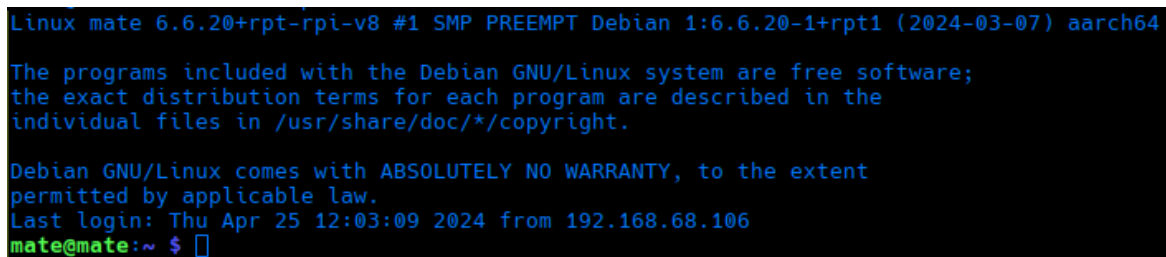
(povz. po: <https://www.cloudflare.com/learning/access-managment/what-is-ssh/>).



```
[mate@archlinux ~]$ ssh mate@192.168.68.105
mate@192.168.68.105's password: █
```

Slika 2: Dostop prek SSH

Ko vpišemo geslo, se izpišejo osnovni podatki o operacijskem sistemu, na katerega dostopamo. Ime v terminalu se spremeni, kot bi bili v terminalu na računalniku, do katerega dostopamo.



```
Linux mate 6.6.20+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.6.20-1+rpt1 (2024-03-07) aarch64
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Apr 25 12:03:09 2024 from 192.168.68.106
mate@mate:~ $
```

Slika 3: Dostop do Raspberry Pi

4 POŽARNI ZID

Linux je znan po svoji močni varnosti, saj ponuja orodja za namizno in strežniško uporabo. V preteklosti se je za varnost uporabljal zmogljiv sistem iptables, vendar je lahko za nove uporabnike zapleten. Uncomplicated Firewall (UFW) ponuja enostavnejši sprednji del za iptables, zagotavlja uporabniku prijazen okvir za upravljanje netfilterja in vmesnik ukazne vrstice za delo s požarnim zidom. UFW ponuja tudi orodja grafičnih uporabniških vmesnikov (angl. graphical user interface, GUI) za poenostavljeno upravljanje sistema.

(povz. po: <https://www.linux.com/training-tutorials/introduction-uncomplicated-firewall-ufw/>).

5 DOCKER

Aplikacije običajno izvajam v kontejnerjih Docker, ker sem imel že večkrat slabe izkušnje, ko sem različne programe izvajal neposredno na operacijskem sistemu. Programa se lahko motita med seboj ali pa že pri inštalaciji potrebujeta isti paket, ampak različne verzije za svoje delovanje. Docker uporabljam, da jih lahko namestim kot na mini sistemu, ki uporablja tiste pakete, ki jih potrebuje, ima pa tudi druge prednosti (povz. po: <https://docs.docker.com/get-started/overview/>):

- Izolacija: aplikacije v vsebniku so bolj izolirane od okolice. Tako je veliko preprosteje izvesti »odstranitev« in začeti znova.
- Različice aplikacij: namestitev aplikacij je neposredno odvisna od upravitelja paketov, ki ga uporabljamo. Pogosto ne moremo dobiti najnovejše različice paketa, ker še ni bil zapakiran za našo distribucijo, ali pa je različica, ki je na voljo, preveč nova ali pa ni različica, ki jo želimo. Če ima projekt na drugi strani slike Docker, je preprosto potegniti točno tisto različico, ki jo želimo uporabiti in ki ni odvisna od distribucije, ki jo izvajamo, tudi če je bistveno starejša in ni na voljo v upravitelju paketov.
- Selitev: ko imamo aplikacije, ki se izvajajo v vsebnikih, je veliko lažje premakniti stvari na novega gostitelja, kot da bi obstoječega gostitelja obdržali tako dolgo, da moramo izvajati nadgradnje na mestu. Običajno je hitreje ustvariti novega gostitelja in nato preseliti aplikacijo in podatke.
- Testiranje: Docker omogoča zelo enostavno izvajanje več kopij iste stvari na istem računalniku ali celo različnih različic iste stvari, kar je izjemno uporabno za testiranje.
- Skupna raba virov: posamezne aplikacije na splošno same ne uporabljajo veliko virov. Tradicionalni pristop izvajanja ločenih virtualnih strojev za ločene aplikacije zagotavlja izolacijo, ki jo želimo, vendar pogosto na račun izgubljenih virov. Namesto tega si vsebniki, če jih izvajamo več na enem gostitelju, delijo vire gostitelja in potencialno lahko omogočijo izvajanje več stvari iz iste količine centralne procesorske enote (angl. central processing unit, CPU) oziroma pomnilnika.

Namestitev Dockerja je zelo enostavna.

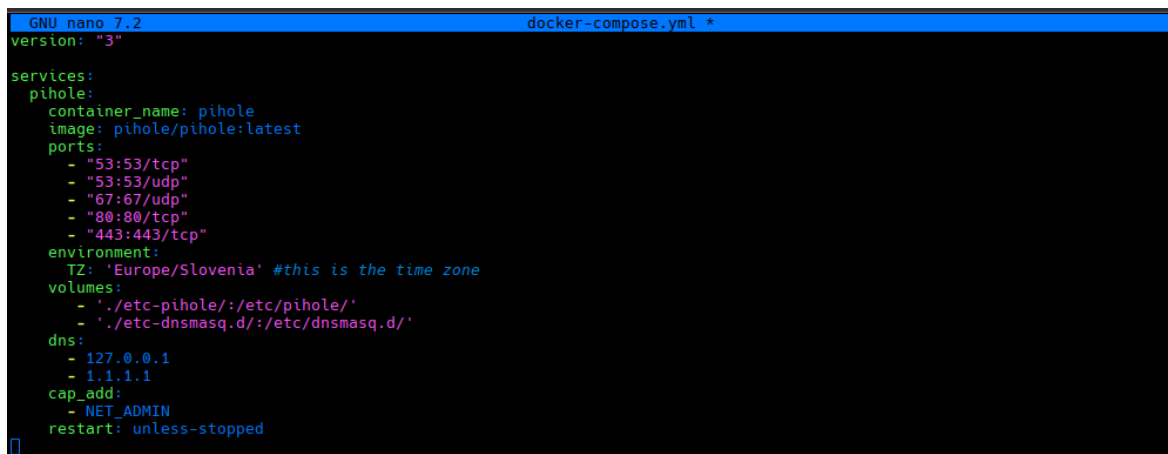


```
mate@mate:~$ sudo apt-get install docker docker-compose
```

Slika 4: Namestitev Dockerja

Na spletni strani GitHub posamezniki ali skupine ljudi izdelujejo projekte, pri katerih je mogoče kopirati nekaj linij kode, nato pa Docker namesti vse, kar je treba, zato

aplikacije najraje nameščam preko GitHub. Prednost tega je, da so vsi projekti odprtokodni in da lahko z lahkoto prilagajmo kodo svojim željam in potrebam. Za namestitev aplikacije je treba ustvariti tekstovno datoteko .yaml in nato pognati celo kodo z enostavnim ukazom DOCKER COMPOSE.



```
GNU nano 7.2 docker-compose.yml *
version: "3"

services:
  pihole:
    container_name: pihole
    image: pihole/pihole:latest
    ports:
      - "53:53/tcp"
      - "53:53/udp"
      - "67:67/udp"
      - "80:80/tcp"
      - "443:443/tcp"
    environment:
      TZ: 'Europe/Slovenia' #this is the time zone
    volumes:
      - './etc-pihole:/etc/pihole/'
      - './etc-dnsmasq.d:/etc/dnsmasq.d/'
    dns:
      - 127.0.0.1
      - 1.1.1.1
    cap_add:
      - NET_ADMIN
    restart: unless-stopped
```

Slika 5: Tekstovna datoteka .yaml

V datoteki je najbolj pomembno, da Dockerju povemo, katero aplikacijo bomo namestili iz njegove baze podatkov pod IMAGE. Nato poljubno izberemo svoje nastavitve ali pa aplikacijo namestimo s privzetimi nastavitvami.

6 PI-HOLE

Pi-hole je programska oprema za filtriranje oglasov in zaščito omrežja pred zlonamerno programsko opremo, ki jo je mogoče namestiti na majhne računalniške sisteme, kot je Raspberry Pi. Deluje kot strežnik sistema domenskih imen (angl. Domain Name System, DNS) in blokira dostop do domen, ki so znane po prikazovanju oglasov, sledenju ali vsebujejo zlonamerno programsko opremo.

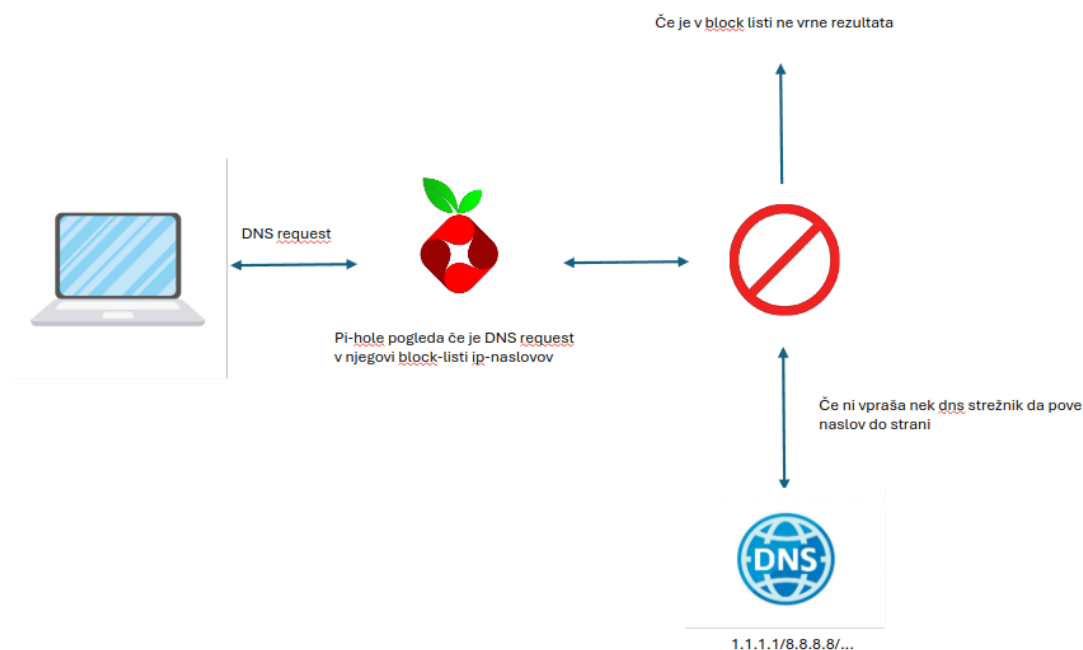
Glavne značilnosti Pi-hole vključujejo:

- Blokiranje oglasov: Pi-hole blokira oglaševalske domene, kar omogoča bolj prijetno brskanje po spletu brez nadležnih oglasov.
- Zaščita pred zlonamerno programsko opremo: Pi-hole lahko blokira dostop do domen, ki vsebujejo zlonamerno programsko opremo ali so povezane s sumljivimi dejavnostmi, kar povečuje varnost omrežja.

- Statistika in nadzor: Pi-hole ponuja orodja za spremljanje statistike o blokiranih domenah, omogoča beleženje dejavnosti omrežja in zagotavlja nadzor nad tem, kateri spletni viri so blokirani ali dovoljeni.
- Preprosta namestitvev: Pi-hole je enostavno namestiti in konfigurirati na Raspberry Pi ali drugih podobnih računalniških sistemih. Nudi tudi različne možnosti prilagajanja in konfiguracije po meri.
- Odprtokodnost: Pi-hole je odprtokodni projekt, kar pomeni, da je na voljo brezplačno in da lahko uporabniki prispevajo k razvoju in izboljšavam.

Pi-hole je torej uporabno orodje, ki omogoča boljšo varnost in učinkovitost omrežja ter izboljša uporabniško izkušnjo z manj oglasi in manjšo verjetnostjo izpostavljenosti zlonamerni programski opreми.

(povz. po. <https://en.wikipedia.org/wiki/Pi-hole/>).



Slika 6: Delovanje aplikacije Pi-hole

6.1.1 Blokiranje oglasov na ravni brskalnika

Pi-hole in aplikacije, ki blokirajo oglase na ravni brskalnika, kot sta Adblock in uBlock, so si podobne, ampak delujejo čisto drugače. Pi-hole ne ustavi ničesar, samo posreduje napačne naslove za domene, ki so znane kot oglaševanje in so na črni listi. Pridobi sezname domen, ki domnevno gostijo oglaševanje ali vsebino z zlonamerno programsko opremo. Kadar koli naprava v omrežju zaprosi za eno od takih domen, Pi-hole zavede napravo, da gre namesto tega na napačen naslov. Postopek poteka na omrežni ravni in nima nikakršnega pregleda vsebine. Aplikacije, ki blokirajo oglase na ravni brskalnika, so tiste, ki se ukvarjajo z vsebino spletne strani, ki naj bi jo nalagali, in je ne bodo pokazali po lastni izbiri. To so npr. oglasi, ki jih sicer Pi-hole ne more blokirati, ker pridejo z istim naslovom IP kot stran. Adblocker pa lahko znotraj strani išče nezaželene vsebine, skripte, slike itd. Te aplikacije tudi vzdržujejo in redno posodablajo ponudnika in ponujajo mnogo več stvari, ki jih posredovalnik DNS ni sposoben narediti.

6.2 SISTEM DOMENSKIH IMEN (DNS)

DNS je telefonski imenik interneta. Ljudje dostopajo do informacij na spletu prek imen domen, kot sta nytimes.com ali espn.com. Spletni brskalniki komunicirajo prek naslovov IP. DNS prevede imena domen v naslove IP, tako da lahko brskalniki naložijo internetne vire.

Vsaka naprava, povezana z internetom, ima edinstven naslov IP, ki ga drugi stroji uporabljajo za iskanje naprave. Strežniki DNS odpravljajo potrebo po tem, da bi si morali ljudje zapomniti naslove IP, kot je 192.168.1.1 (v IPv4), ali bolj zapletene novejša alfanumerične naslove IP, kot je 2400:cb00:2048:1::c629:d7a2 (v IPv6). (povz. po. <https://www.cloudflare.com/learning/dns/what-is-dns/>).

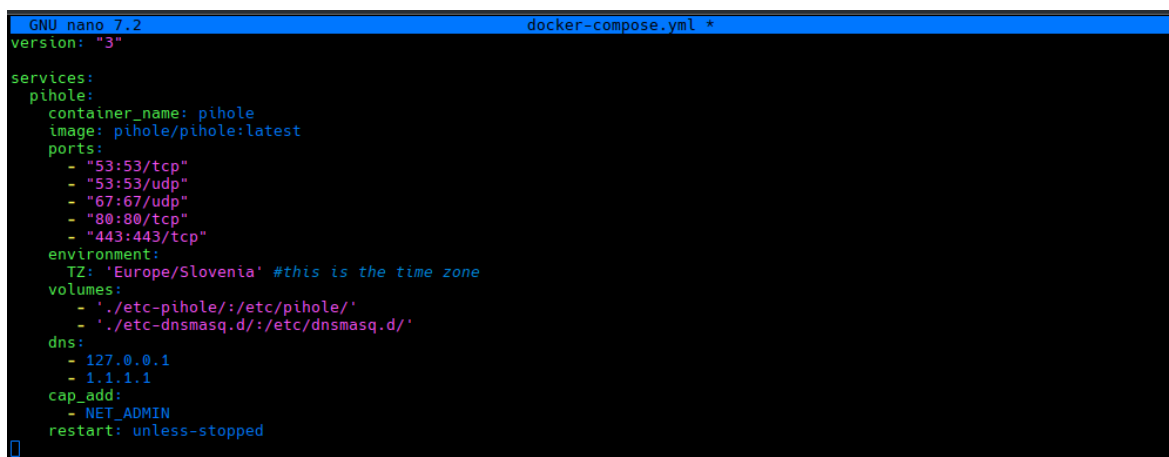
6.3 DELOVNAJE DNS

Postopek razreševanja DNS vključuje pretvorbo imena gostitelja (kot je `www.example.com`) v računalniku prijazen naslov IP (kot je `192.168.1.1`). Vsaki napravi v internetu je dodeljen naslov IP in ta naslov je potreben za iskanje ustrezne internetne naprave – tako kot se naslov ulice uporablja za iskanje določenega doma. Ko uporabnik želi naložiti spletno stran, mora priti do prevoda med tem, kar uporabnik vnese v svoj spletni brskalnik (`example.com`), in strojno prijaznim naslovom, ki je potreben za iskanje spletne strani `example.com`.

Da bi razumeli postopek za razreševanje DNS, je pomembno, da se seznanimo z različnimi komponentami strojne opreme, med katerimi mora biti poizvedba DNS. Pri spletnem brskalniku se iskanje DNS izvaja »v zakulisju« in ne zahteva nobene interakcije z uporabnikovim računalnikom, razen začetne zahteve.
(povz. po. <https://www.cloudflare.com/learning/dns/what-is-dns/>).

6.4 NAMESTITEV PI-HOLE

Namestitev Pi-hole je dokaj enostavna. Ustvarjalci te aplikacije imajo na spletni strani GitHub dobro razložen postopek. Najprej sem naredil novo mapo v tekstovni datoteki `.yaml`.



```
GNU nano 7.2                                docker-compose.yml *
version: "3"

services:
  pihole:
    container_name: pihole
    image: pihole/pihole:latest
    ports:
      - "53:53/tcp"
      - "53:53/udp"
      - "67:67/udp"
      - "80:80/tcp"
      - "443:443/tcp"
    environment:
      TZ: 'Europe/Slovenia' #this is the time zone
    volumes:
      - './etc-pihole:/etc/pihole/'
      - './etc-dnsmasq.d:/etc/dnsmasq.d/'
    dns:
      - 127.0.0.1
      - 1.1.1.1
    cap_add:
      - NET_ADMIN
    restart: unless-stopped
```

Slika 7: Tekstovna datoteka Pi-hole `.yaml`

```
mate@mate:~/semi $ touch docker-compose.yml
mate@mate:~/semi $ sudo nano docker-compose.yml
mate@mate:~/semi $ docker-compose up -d
```

Slika 8: Ustvarjanje tekstovne datoteke .yaml

Pri Dockerju je to res enostavno. Nato sem vtiskal ukaz »docker-compose up -d«. Docker prenese datoteke, ki jih potrebuje, in jih namesti v kontejner. »-d« pomeni, da se to nato izvaja v ozadju. Če bi bilo kaj narobe, bi se to izpisalo v konzoli. V nasprotnem primeru pa ponudi samo novo vrstico, da lahko naprej pišemo ukaze. Lahko tudi pogledamo, ali je vse v redu z ukazom »docker ps«.

```
mate@mate:~$ sudo docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
7fc6f62c476b	pihole/pihole:latest	"/s6-init"	Up 7 seconds (health: starting)	Up 7 seconds (health: starting)	0.0.0.0:53->53/udp, :::53->53/udp, 0.0.0.0:53->53/tcp, :::53->53/tcp, 0.0.0.0:80->80/tcp, 0.0.0.0:67->67/udp, :::80->80/tcp, :::67->67/udp	pihole

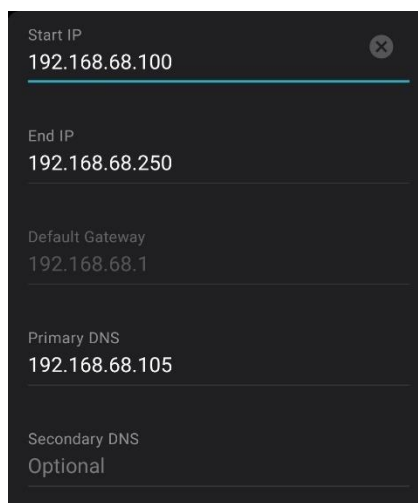
Slika 9: Preverjanje delovanja Pi-hole v Dockerju

To nam pove, da se aplikacija izvaja. Nato je treba požarnemu zidu sporočiti, da lahko čez določene porte spusti promet – tj. port 53 za promet tcp in udp, ta port je standardni port za strežnik DNS strežnik, ter port 80 za promet tcp, ta port se standardno uporablja za spletne strani HTTP.

```
mate@mate:~$ sudo ufw allow 53/tcp
sudo ufw allow 53/udp
sudo ufw allow 80/tcp
```

Slika 10: Izvedba izjeme v požarnem zidu

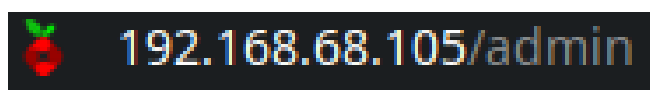
Zdaj bo šel cel promet čez strežnik DNS. V nastavitvah modema je treba označiti, da je to zdaj njegov privzeti strežnik DNS.



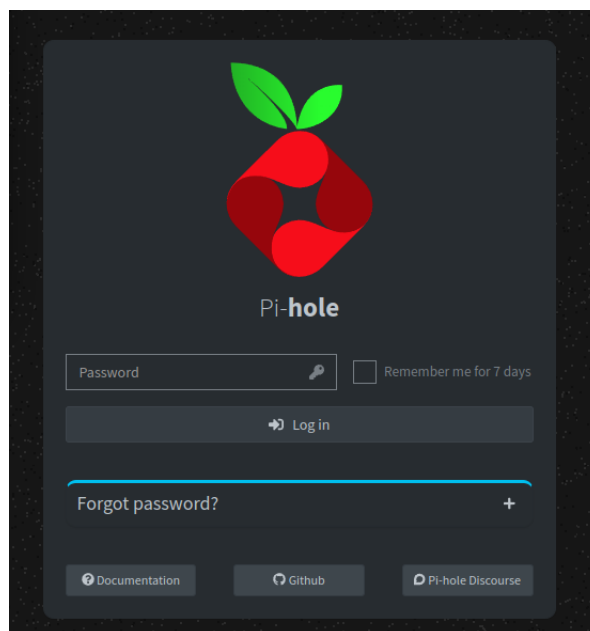
A screenshot of the Raspberry Pi configuration tool's 'DNS' settings screen. The screen has a dark background with white text. It contains several input fields with their respective labels: 'Start IP' with the value '192.168.68.100', 'End IP' with '192.168.68.250', 'Default Gateway' with '192.168.68.1', 'Primary DNS' with '192.168.68.105', and 'Secondary DNS' with 'Optional'. A small 'X' icon is in the top right corner of the window.

Slika 11: Nastavitev Raspberry Pi kot primarni strežnik DNS

Pi-hole ima tudi svoj vmesnik GUI. Do njega dostopamo tako, da napišemo »ip_naslov/admin« v spletni brskalnik.



Slika 12: Dostop do aplikacije Pi-hole



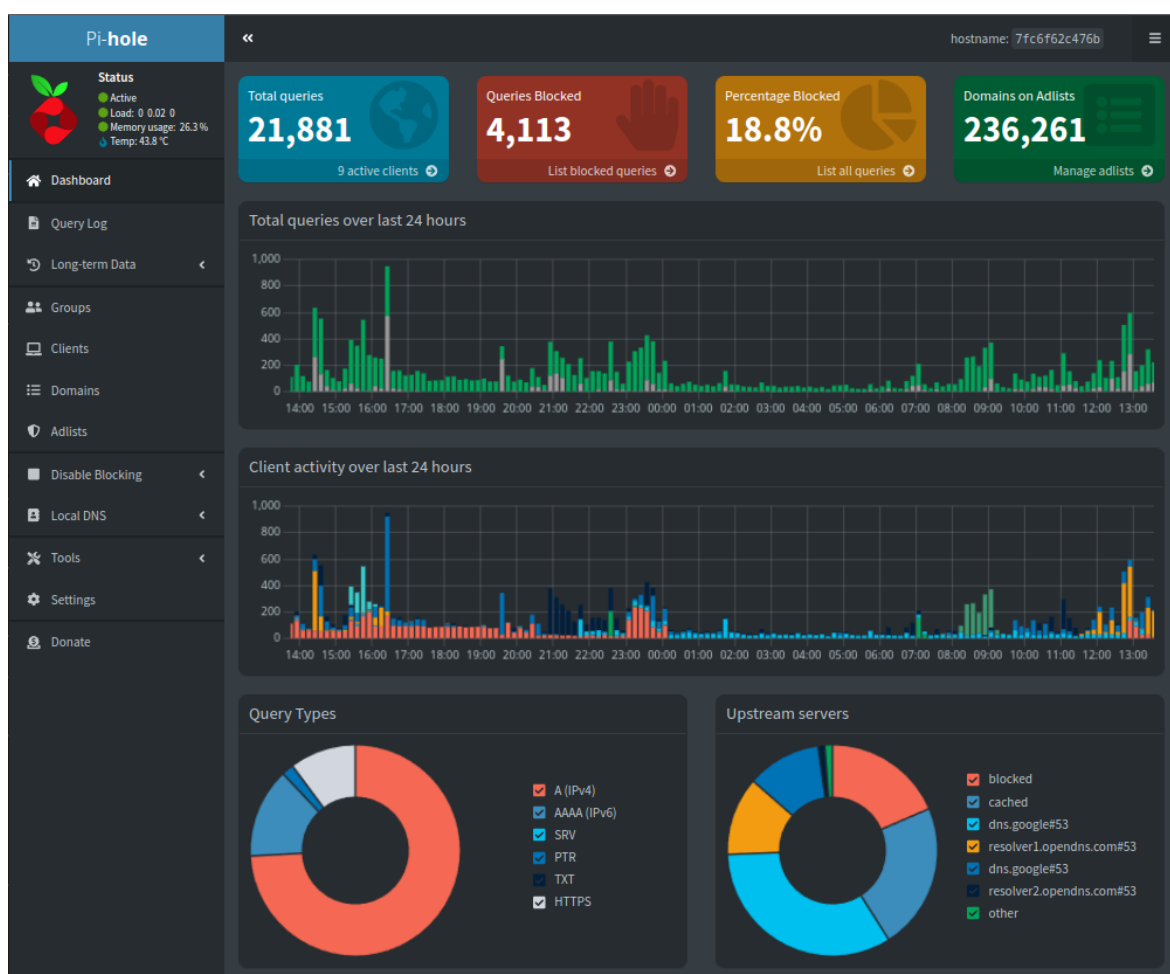
Slika 13: Prva stran Pi-hole

Odpre nam stran, kjer nas vpraša za geslo, čeprav v tekstovni datoteki nisem nastavil gesla. V navodilih piše, da je to precej enostavno popraviti. Treba je izvesti ukaz znotraj kontejnerja Docker za zamenjavo gesla.

```
mate@mate:~$ sudo docker exec -it 7fc6f62c476b pi-hole -a -p
Enter New Password (Blank for no password):
```

Slika 14: Spreminjanje gesla za Pi-hole

Ko enkrat vpišemo geslo, nas pelje na to stran.



Slika 15: Nadzorna plošča Pi-hole

Na prvi strani je prikazano, koliko je bilo izvedenih poizvedb DNS, koliko jih je bilo blokiranih in koliko domen je na adlisti. Spodaj je to prikazano tudi v grafih. Te podatke lahko tudi podrobneje pogledamo, če želimo. Če pritisnemo na »total

queries«, nam pokaže, iz katerega naslova IP je prišla katera poizvedba in ali jo je strežnik spustil čez ali jo je blokiral.

Network overview

Search:

Show 10 entries

IP address	Hardware address	Interface	Hostname	First seen	Last Query	Number of queries	Uses Pi-hole	Action
192.168.68.106	N/A	eth0		2024-02-10 16:21:00	2024-04-26 13:38:45	781,901	✓	
192.168.68.101	N/A	eth0		2024-02-11 14:03:00	2024-04-26 13:38:08	14,995	✓	
192.168.68.103	N/A	eth0		2024-02-11 16:06:00	2024-04-26 13:36:40	10,791	✓	
192.168.68.100	N/A	eth0		2024-02-10 18:30:00	2024-04-26 13:35:59	57,676	✓	
192.168.68.104	N/A	eth0		2024-02-26 09:31:00	2024-04-26 13:19:19	11,278	✓	
127.0.0.1	00:00:00:00:00:00 virtual interface	lo	localhost	2024-02-10 16:10:00	2024-04-26 13:00:01	12,550	✓	
192.168.68.107	N/A	eth0		2024-02-26 09:35:00	2024-04-26 09:11:34	14,229	✓	
192.168.68.102	N/A	eth0		2024-02-11 15:57:00	2024-04-26 07:35:23	2,597	✓	
172.19.0.1	02:42:55:34:1a:f5	eth0		2024-04-24 19:22:00	2024-04-25 20:00:00	1,202	✓	
192.168.68.105								
	02:42:38:57:9e:7b	eth0	unknown	2024-04-24 18:55:00	2024-04-24 19:00:20	151	?	

Showing 1 to 10 of 34 entries

Slika 16: Pregled nad napravami Pi-hole

Lahko pogledamo, katere poizvedbe so bile blokirane in iz katerega naslova IP prihajajo.

Recent Queries (showing queries blocked by Pi-hole)

Search: Type / Domain / Client

Show 10 entries

Previous 1 2 3 4 5 ... 412 Next

Time	Type	Domain	Client	Status	Reply	Action
2024-04-26 13:39:49	A	www.googletagmanager.com	192.168.6.8.101	Blocked (gravity)	IP (0.5ms)	✓ Whitelist
2024-04-26 13:38:08	A	api.device.xiaomi.net	192.168.6.8.101	Blocked (gravity)	IP (0.3ms)	✓ Whitelist
2024-04-26 13:37:59	A	api.device.xiaomi.net	192.168.6.8.101	Blocked (gravity)	IP (0.2ms)	✓ Whitelist
2024-04-26 13:37:51	A	api.device.xiaomi.net	192.168.6.8.101	Blocked (gravity)	IP (0.3ms)	✓ Whitelist
2024-04-26 13:36:10	A	api.ad.intl.xiaomi.com	192.168.6.8.101	Blocked (gravity)	IP (0.0ms)	✓ Whitelist
2024-04-26 13:36:10	A	api.ad.intl.xiaomi.com	192.168.6.8.101	Blocked (gravity)	IP (0.1ms)	✓ Whitelist
2024-04-26 13:35:59	A	sdkconfig.ad.intl.xiaomi.com	192.168.6.8.100	Blocked (gravity)	IP (0.2ms)	✓ Whitelist
2024-04-26 13:35:16	A	sdkconfig.ad.intl.xiaomi.com	192.168.6.8.100	Blocked (gravity)	IP (0.2ms)	✓ Whitelist
2024-04-26 13:35:10	A	firebase logging.googleapis.com	192.168.6.8.100	Blocked (gravity)	IP (0.1ms)	✓ Whitelist
2024-04-26 13:35:02	A	app-measurement.com	192.168.6.8.100	Blocked (gravity)	IP (0.0ms)	✓ Whitelist
Time	Type	Domain	Client	Status	Reply	Action

Showing 1 to 10 of 4,117 entries

Previous 1 2 3 4 5 ... 412 Next

Slika 17: Pregled Pi-hole nad povpraševanji DNS

Lahko tudi dodajamo ali odstranjujemo domene na adlisto ali dodamo adliste.

Add a new adlist

Address: URL or space-separated URLs

Comment: Adlist description (optional)

Hints:

- Please run `pihole -g` or update your gravity list [online](#) after modifying your adlists.
- Multiple adlists can be added by separating each *unique* URL with a space
- Click on the icon in the first column to get additional information about your lists. The icons correspond to the health of the list.

Add

List of adlists

Show 10 entries

Search:

Previous 1 Next

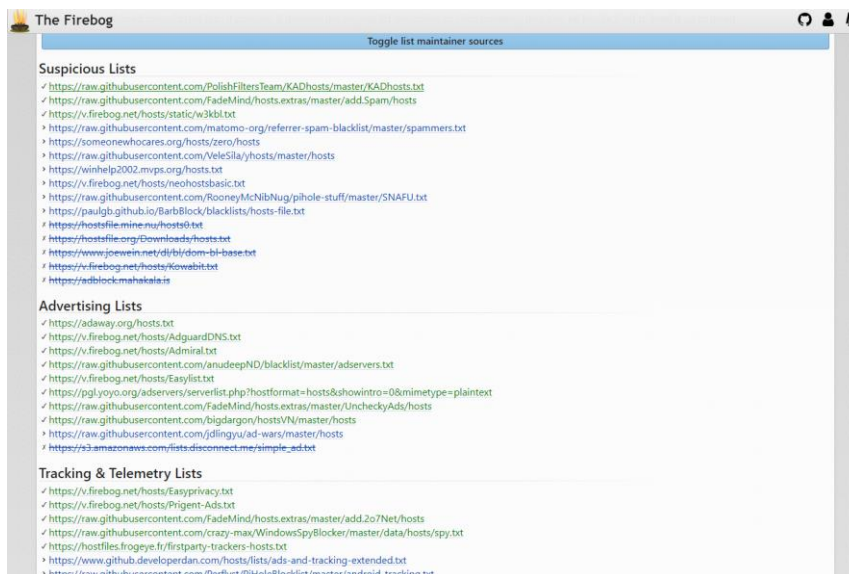
	Address	Status	Comment	Group assignment	
<input type="checkbox"/>	✓ https://raw.githubusercontent.com/StevenBlack/hosts/master/hosts	Enabled	Migrated from /etc/pihole/	Default	
<input type="checkbox"/>	🟡 https://adaway.org/hosts.txt	Enabled		Default	
<input type="checkbox"/>	🟡 https://y.firebog.net/hosts/AdguardDNS.txt	Enabled		Default	
<input type="checkbox"/>	🟡 https://raw.githubusercontent.com/anudeepND/blacklist/master/adservers.txt	Enabled		Default	
<input type="checkbox"/>	🟡 https://raw.githubusercontent.com/bigdargon/hostsVN/master/hosts	Enabled		Default	
<input type="checkbox"/>	🟡 https://raw.githubusercontent.com/FadeMind/hosts.extras/master/ad.Scam/hosts	Enabled		Default	

Showing 1 to 6 of 6 entries

Previous 1 Next

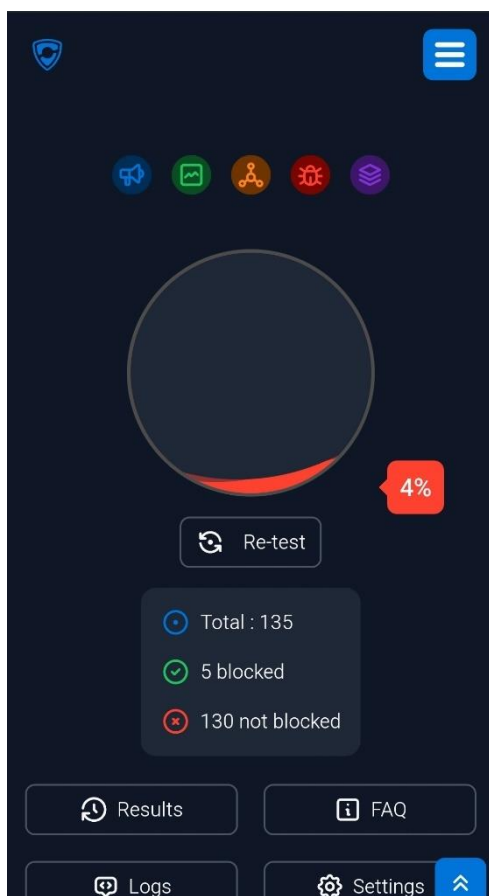
Slika 18: Dodajanje Pi-hole na črno listo

Obstajajo tudi strani, ki so narejene zato, da dodajajo nove adliste, ki jih lahko prenesemo in dodamo v svoj sistem. Ena od teh strani je firebog.net.

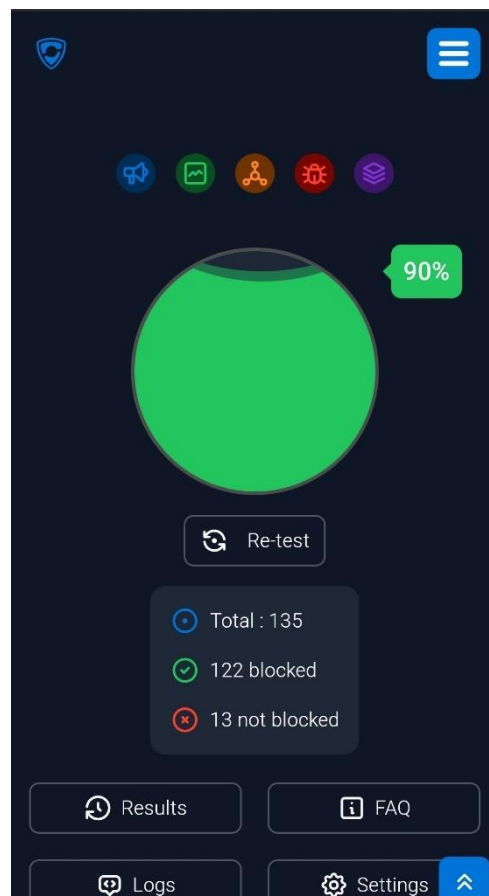


Slika 19: Spletna stran za adliste

Sedaj je treba narediti samo še test, ki ga bom izvedel na strani <https://d3ward.github.io/toolz/adblock>. Ta stran preveri 135 najpogostejših domen za oglase, sledilnike itd. Test bom izvedel na privzetem brskalniku na telefonu Android.



Slika 20: Test za oglase pred Pi-hole



Slika 21: Test za oglase, ko Pi-hole deluje

Razlika je precej občutna. Pi-hole pomaga blokirati dodatnih 86 % oglasov in drugih strani.

7 WIREGUARD VPN

WireGuard je sodoben, preprost VPN, ki uporablja napredno kriptografijo, da je hitrejši, preprostejši in uporabnejši od IPsec. Zasnovan je za vgrajene vmesnike in super računalnike in je zdaj medplatformski. Sprva je bil izdan za jedro Linuxa, zdaj pa ga je mogoče široko uporabljati in velja za najbolj varno, enostavno za uporabo in najpreprostejšo rešitev VPN v panogi.

7.1 PROTOKOL BREZ POVEZAVE

Vsak varen protokol zahteva ohranitev nekega stanja, zato obstaja začetno zelo preprosto rokovanje, ki vzpostavi simetrične ključe, ki se uporabljajo za prenos podatkov. To rokovanje se zgodi vsakih nekaj minut, da se zagotovijo vrtljivi ključi za popolno tajnost v nadaljevanju. Izvaja se na podlagi časa in ne na podlagi vsebine prejšnjih paketov, ker je zasnovan tako, da elegantno obravnava izgubo paketov. Obstaja pameten impulzni mehanizem, ki zagotavlja, da so najnovejši ključi in rokovanja posodobljeni, po potrebi pa se znova pogaja s samodejnim zaznavanjem, kdaj so rokovanja zastarela. Uporablja ločeno čakalno vrsto paketov na gostitelja, tako da lahko zmanjša izgubo paketov med rokovanjem, hkrati pa zagotavlja enakomerno delovanje za vse odjemalce. Z drugimi besedami, napravo dvignemo in vse ostalo se samodejno uredi namesto nas. Ni nam treba skrbeti, da bi zahtevali ponovno povezavo ali prekinitev povezave ali ponovno inicializacijo ali kaj podobnega. (<https://www.wireguard.com/protocol>).

7.2 KRIPTOGRAFIJA

Med kriptografskimi algoritmi za aplikacije VPN sta najbolj priljubljena standarda AES-256 in ChaCha20. Razvijalci WireGuarda so se odločili za ChaCha, ker:

- je AES hiter, vendar ga ne podpirajo vsi procesorji. Večina mobilnih in vgrajenih procesorjev, ki so na voljo, ne podpirajo protokola;

- CPE za splošne namene imajo velike težave pri izvajanju programske opreme AES na varen način. Tehnologija je ranljiva za napad na časovni predpolnilnik. Nasprotno je ChaCha20 veliko enostavnejši algoritem;
- ChaCha20 je hiter za splošne namene, njegova uporaba pa tudi ni težavna. Leta 2015 je bil trikrat hitrejši od AES na strojni opremi, ki ni AES-NI. Optimiziran je za učinkovito uporabo vektorskih registrov(SSE/AVX);
- ker vektorske razširitve, kot je AVX, postajajo širše in hitrejša, se ChaCha znatno pospeši.

Chacha20 je šifriran tok. Njegov vhod vključuje 256-bitni ključ, 32-bitni števec, 96-bitni nonce in golo besedilo. Njegovo začetno stanje je matrika 4×4 32-bitnih besed. Prva vrstica je konstanten niz »expand 32-byte k«, ki je razrezan na 4×32 -bitne besede. Drugi in tretji sta zapolnjena z 256-bitnim ključem. Prva beseda v zadnji vrstici je 32-bitni števec, druge pa 96-bitni nonce. V vsaki ponovitvi ustvari 512-bitni tok ključev za šifriranje 512-bitnega bloka navadnega besedila. Ko je preostalo golo besedilo manjše od 512 bitov po večkratnem šifriranju, v zadnjih vhodnih podatkih zapolnimo levo z 0 s (MSB) in odstranimo iste bitne neuporabne podatke iz zadnjih izhodnih podatkov. Njegovo šifriranje in dešifriranje sta enaka, dokler so pri vnosu enaki začetni ključ, števec in nonce.

(https://xilinx.github.io/Vitis_Libraries/security/2019.2/guide_L1/internals/chacha20.html).

7.2.1 IZMENJAVA KLJUČEV

WireGuard vključuje asimetrično šifrirno izmenjavo ključev, ki sledi naslednjim konceptom:

- Javni in zasebni ključ: vsaka naprava ima zasebni ključ, ki se uporablja za šifriranje in podpisovanje. Javni ključ se deli z drugimi napravami za šifrirno komunikacijo.
- Algoritem za izmenjavo ključa: uporablja Curve25519 za izmenjavo ključev. Je sodoben in učinkovit kriptografski algoritem ki izračuna skupni tajni ključ, ki ga odjemalec in strežnik uporabljata za šifriranje komunikacij.

- Generiranje ključev: naprave ustvarijo par javnih in zasebnih ključev. Javni ključi se varno porazdelijo med naprave prek konfiguracijskih datotek ali strogo zaupnih kanalov.
- Izmenjava ključa: obe napravi uporabljata svoj zasebni ključ in javni ključ druge naprave za izvedbo izračuna. Rezultat izračuna je skupni skrivni ključ.
- Generiranje ključa seje: s spremenjenimi ključi se generirajo efemerni ključi, ki se uporabljajo za šifriranje dejanske komunikacije.
- Protokol 1-RTT: WireGuard izvaja zelo hiter protokol 1-RTT, kar pomeni, da je potrebna le ena izmenjava povratnega sporočila, preden se vzpostavi varna povezava.

(povz. po. <https://www.wireguard.com/protocol>)

```
mate@mate:/opt/wireguard-server $ sudo docker exec -it wireguard wg
interface: wg0
  public key: McIktZ5PjY/ekURfV7jZta0+n6IwZbG+08XAcbfwx0A=
  private key: (hidden)
  listening port: 51820

peer: A+q5GeVDMvCppfrAbKX7UadaKWQxzuxr/9lfHtW3/XY=
  preshared key: (hidden)
  allowed ips: 10.13.13.2/32

peer: oVjQeSsMzbiDZR6MX5nPRE3su52HsM0A6IlKpNm9LLM=
  preshared key: (hidden)
  allowed ips: 10.13.13.3/32

peer: VyLMqQNPgacv+/R69tZa5CQga360epxdqVNzHn/EHBI=
  preshared key: (hidden)
  allowed ips: 10.13.13.4/32
```

Slika 22: Vmesnik WireGuard

7.3 NAMESTITEV WIREGUARD VPN

Ekipa WireGuard ne ponuja uradne slike Docker, zato sem po raziskovanju našel sliko, ki jo je razvila ekipa LinuxServer. To so tudi najbolj priporočali, ampak sam naletel na nekaj nevarnosti, saj sem moral že v sami datoteki .yaml definirati, koliko odjemalcev bo imel moj VPN.

```

GNU nano 7.2                                docker-compose.yaml
services:
  wireguard:
    image: linuxserver/wireguard
    container_name: wireguard
    cap_add:
      - NET_ADMIN
      - SYS_MODULE
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Europe/Ljubljana
      - SERVERURL=auto
      - SERVERPORT=51820
      - PEERS=3
      - PEERDNS=auto
      - INTERNAL_SUBNET=10.13.13.0
    volumes:
      - /opt/wireguard-server/config:/config
      - /lib/modules:/lib/modules
    ports:
      - 51820:51820/udp
    sysctls:
      - net.ipv4.conf.all.src_valid_mark=1
    restart: unless-stopped

```

Slika 23: Datoteka WireGuard .yaml

To ni tako velik problem, če pa bi jih definiral 100, bi mapa postala prenapolnjena. Prav tako je DNS delal samo, če je bil nastavljen na avtomatski način, posledično ni izbral mojega strežnika DNS, ampak ga je nastavil na 1.1.1.1. Motilo me je tudi to, da sem moral za vsakega odjemalca napisati ukaz, da mi je pokazal kodo QR, če sem se hotel povezati s telefonom na strežnik VPN.



Slika 24: Koda QR za dostop do strežnika WireGuard VPN

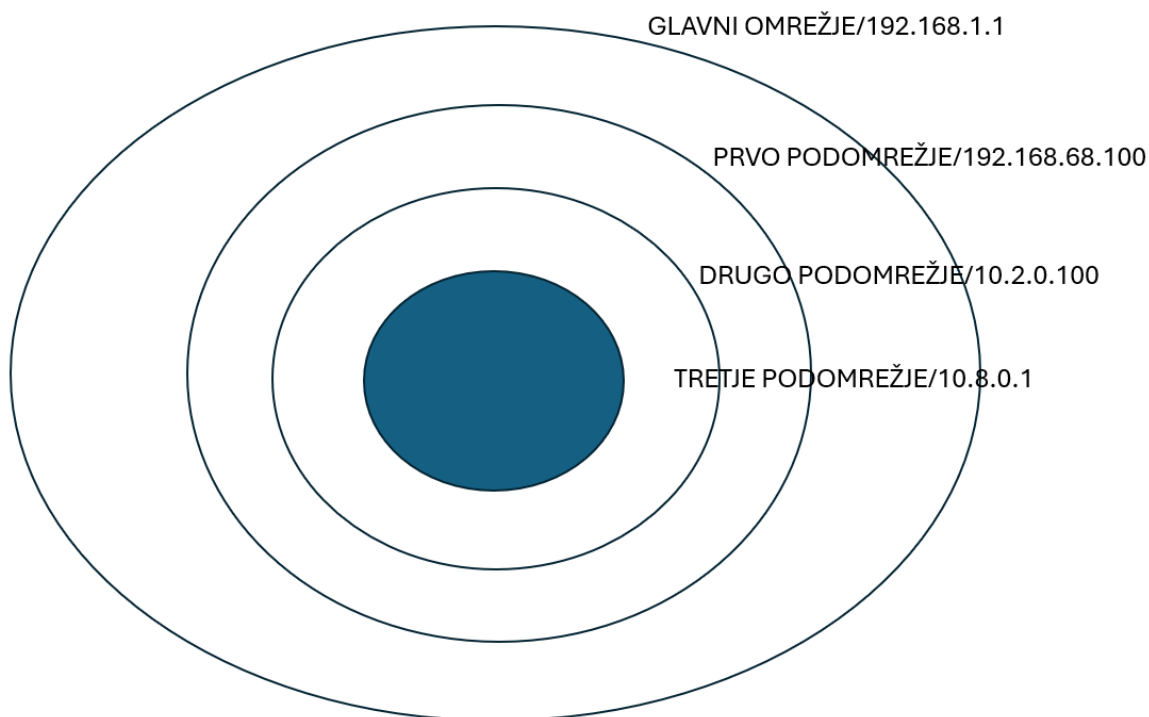
To ne bi bilo praktično, sploh če bi bil na telefonu in bi hotel komu dati dostop do mojega strežnika VPN. Na telefon bi moral naložiti aplikacijo, ki bi mi dala dostop do konzole. Po raziskovanju na internetu sem za vse navedene probleme našel

rešitev. Da je lahko začel strežnik VPN delati s strežnikom DNS, sem ju moral dati v svoje podomrežje. To sem naredil v datoteki .yaml z odstavkom, ki ju je dal v novo podomrežje.

```
networks:  
  private_network:  
    ipv4_address: 10.2.0.200
```

Slika 25: Definiranje podomrežja .yaml

Kar naenkrat sem imel tri podomrežja, ampak je bil to edini način, da lahko vse deluje tako, kot mora. Prednost teh podomrežij je dodatna varnost. Če bi kdo slučajno vdrl v eno od podomrežij, bi tam tudi ostal.



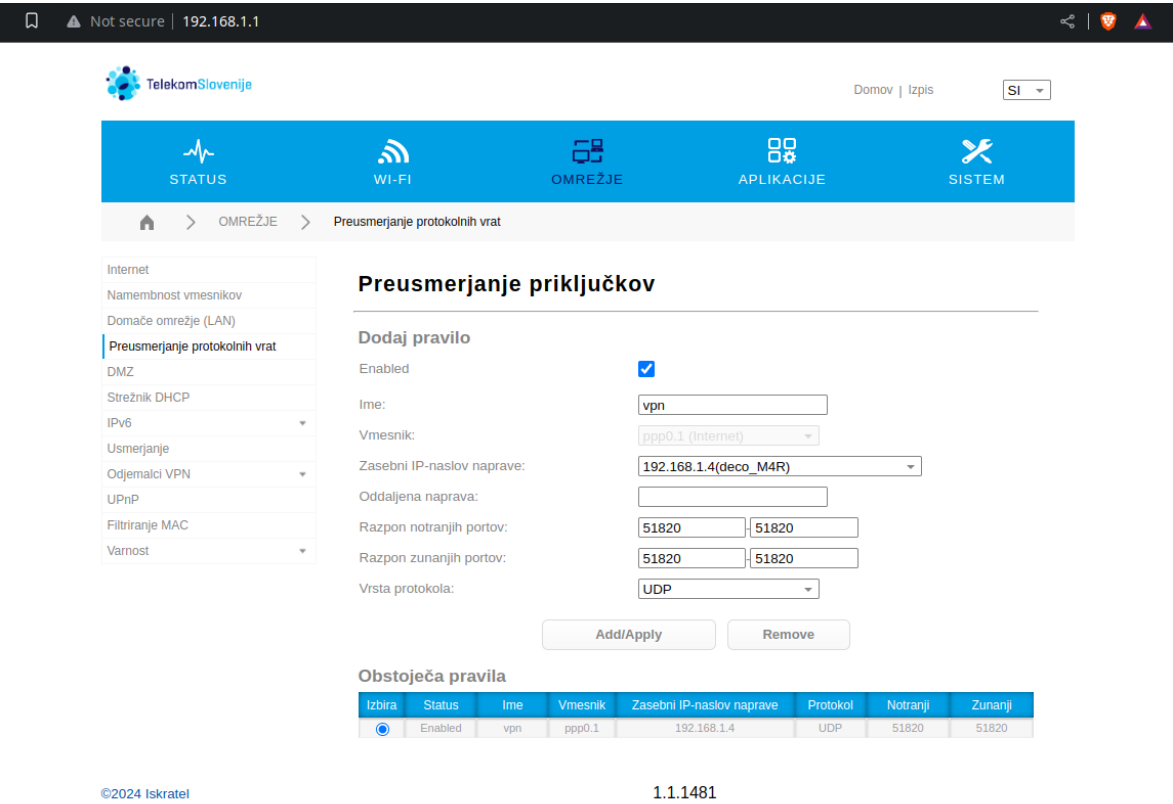
Slika 26: Podomrežja v omrežju

Našel sem tudi vmesnik GUI za dodajanje in brisanje uporabnikov v strežnik VPN, zato tudi vse izgleda veliko bolj pregledno. Na požarnem zidu sem moral vključiti še port 51820/udp, ki bo spuščal promet VPN, in port 51820/tcp, ki se bo uporabljal za dostop do vmesnika GUI preko internetnega brskalnika.

```
mate@mate:~$ sudo ufw allow 51820/udp 51820/tcp
```

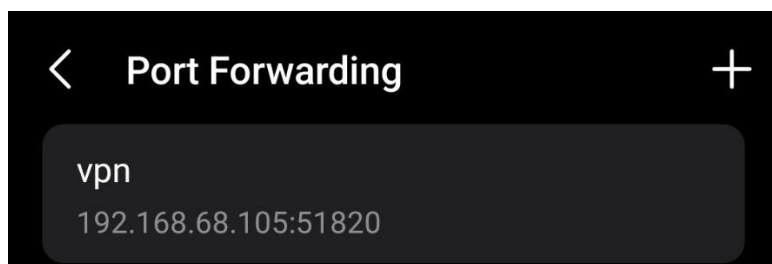
Slika 27: Dostop omrežja WireGuard VPN prek požarnega zida

Preden preverimo, ali deluje, moramo še pri glavnem in svojem modemu narediti »port forwarding«. V brskalnik zato pri glavnem modemu vpišemo 192.168.1.1 in se premaknemo pod odstavek »preusmerjanje protokolnih vrat«.



Slika 28: Preusmerjanje priključkov v glavnem modemu

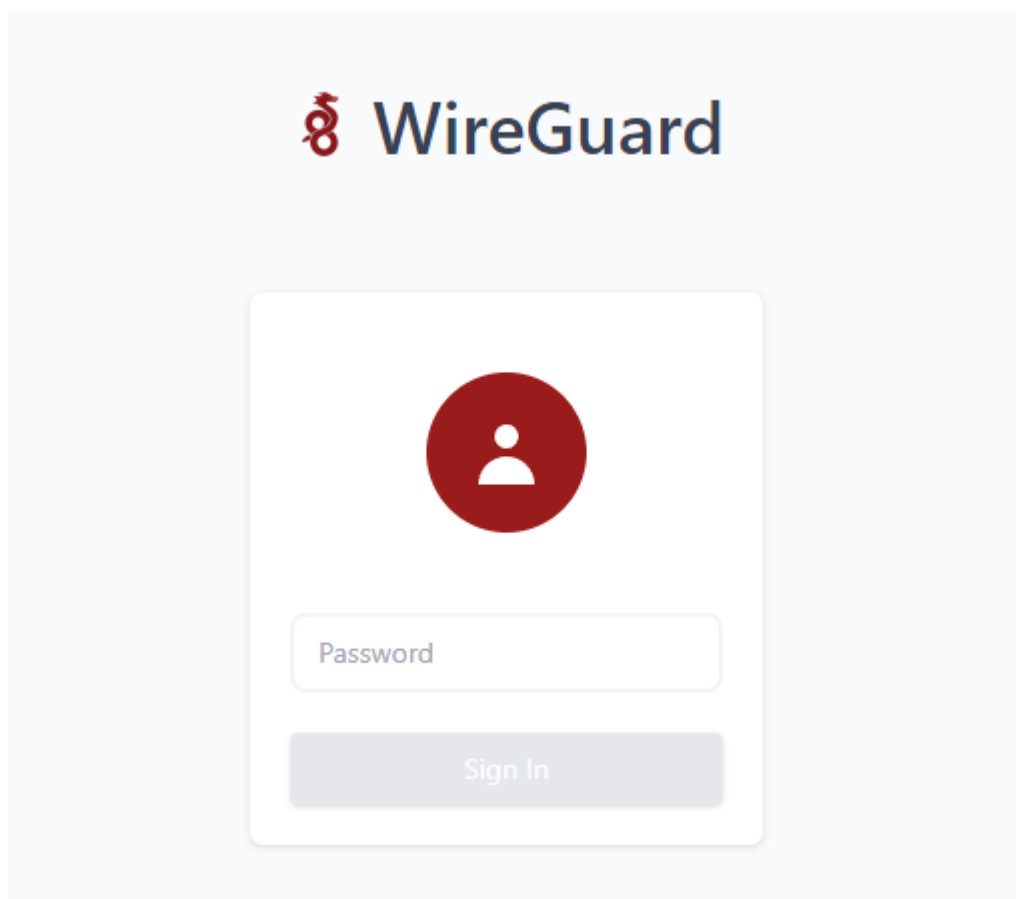
V kolikor prihaja promet udp skozi port 51820, ga posreduje naprej na moj modem, ki je priključen na fizični port 4. Nato isto naročim še svojemu modemu.



Slika 29: Preusmerjanje priključkov v drugem modemu

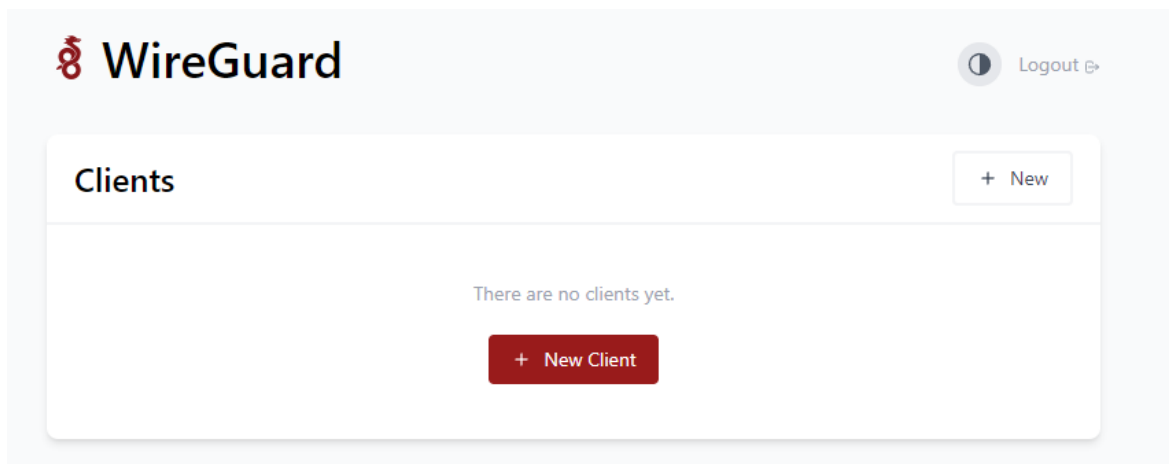
V kolikor prihaja promet udp skozi port 51820, ga posreduje naprej na napravo, ki ima naslov IP 192.168.68.105.

Za dostopanje do spletne strani WireGuard je treba v spletni brskalnik vpisati »ip_naslov:51281«, torej 192.168.68.105:21281. Najprej nas preusmeri na stran »login«.



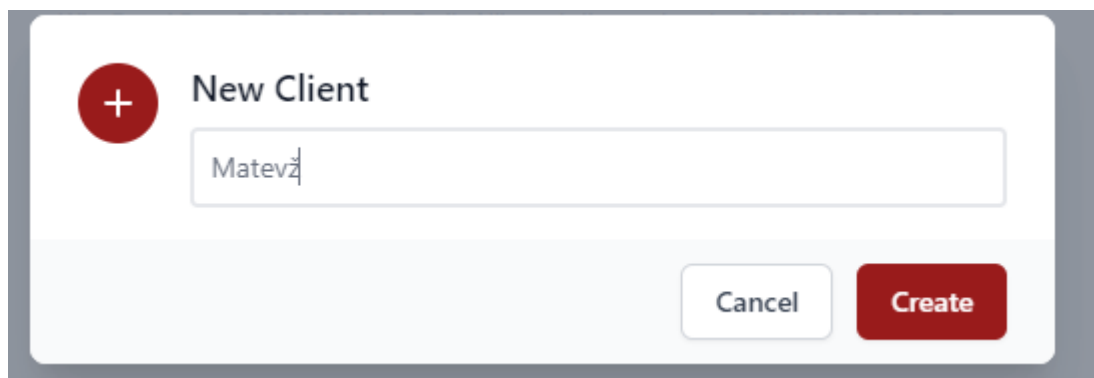
Slika 30: Prva stran WireGuard VPN

Ko vpišem geslo, me nato spusti na to stran.

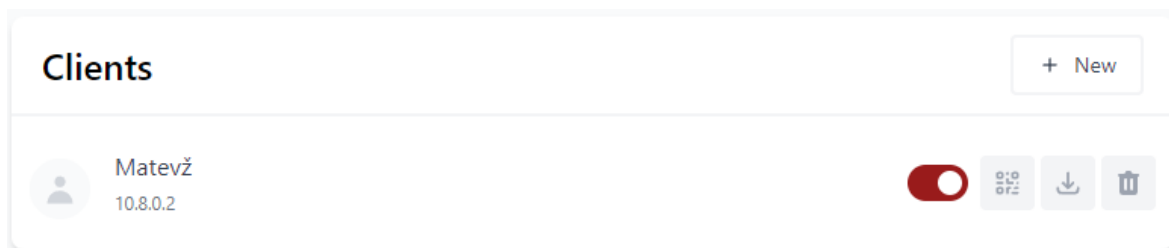


Slika 31: Stran WireGuard VPN za dodajanje uporabnikov

To je zdaj veliko bolj pregledno in tudi za uporabo bolj enostavno. Za dodajanje uporabnikov samo pritisnemo gumb »New client« in vpišemo, kako bomo imenovali uporabnika. Vse ostalo naredi WireGuard sam.



Slika 32: Dodajanje uporabnika WireGuard VPN



Slika 33: Pregled uporabnikov WireGuard VPN

Poleg imena je tudi nekaj dodatnih informacij in gumbov, ki kažejo, kakšen naslov IP bo imel uporabnik, gumb za vklop in izklop uporabnika, gumb za odstranitev uporabnika, gumb za prenos konfiguracije ter gumb, kjer pokaže kodo QR za enostavno povezavo.



Slika 34: Koda QR na spletni strani Wireguard VPN

S tem je povezava zelo enostavna. Na telefon naložim aplikacijo WireGuard, ki je na voljo v trgovini Google in Apple. Takoj ko odprem aplikacijo, se prikaže gumb za dodajanje strežnika in možnost skeniranja kode QR.



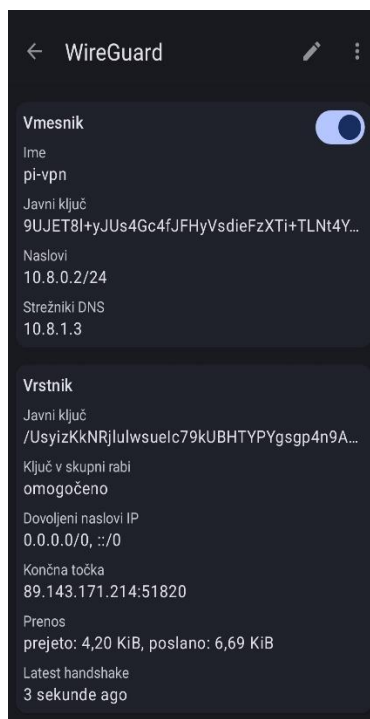
Slika 36: Dodajanje uporabnika v aplikaciji Wireguard VPN



Slika 35: Dodajanje uporabnika prek kode QR v aplikaciji

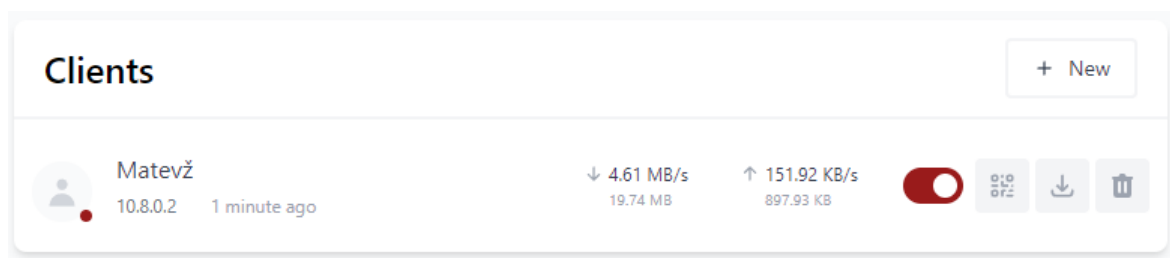


Slika 37: Imenovanje tunela



Slika 38: Pregled povezave

Ko poimenujem strežnik in se povežem, čisto spodaj piše »leatest handshake«, kar je znak, da lahko obiščem brskalnik Google in da zadeva deluje v aplikaciji. To je tudi znak, da moj telefon in strežnik komunicirata. Na spletni strani strežnika tudi kaže, da je uporabnik povezan, poleg tega kaže tudi hitrost prenosa in nalaganja.



Slika 39: Delovanje uporabnika

7.3.1 PODOMREŽJA

Podomrežje je metoda, ki se uporablja pri načrtovanju omrežja za izboljšanje zmogljivosti, zmanjšanje zastojev, izboljšanje varnosti, upravljanje rasti in olajšanje administracije. Tukaj je kratek pregled njegovih glavnih prednosti:

- Izboljšanje delovanja omrežja: povezovanje v podomrežja zmanjša število oddajnih paketov, ki jih prejme vsaka naprava, kar vodi k boljši splošni zmogljivosti omrežja z nadzorom pretoka prometa.
- Zmanjšanje prezasedenosti omrežja: podomrežje z ohranjanjem prometa znotraj lastnega podomrežja zmanjša prezasedenost. Brez podomrežij vse naprave prejmejo nepotreben promet, kar upočasni omrežje.
- Povečanje varnosti omrežja: podomrežje omogoča bolj obvladljiv nadzor prometa, kar olajša prepoznavanje in obravnavanje morebitnih groženj, hkrati pa omejuje dostop do določenih delov omrežja.
- Nadzor rasti omrežja: podomrežje pomaga načrtovati prihodnjo rast z določitvijo prave velikosti za vsako podomrežje na podlagi pričakovanega števila naprav, kar zagotavlja učinkovito uporabo naslovov IP.
- Enostavno upravljanje: povezovanje v podomrežja poenostavlja upravljanje omrežja, tako da skrbnikom omogoča učinkovito spremljanje in odpravljanje težav z napravami v manjših segmentih omrežja.

(povz. po. <https://www.networkcomputing.com/ip-subnetting/5-subnetting-benefits>)

7.4 TUNEL

Če bi v omrežje gledali s programom Wireshark, bi pokazalo, da naprava pošilja podatke na naslov IP strani, na katero je povezan. Res je, da že moderni protokoli, kot je HTTPS, vse podatke kriptirajo.

No.	Time	Source	Destination	Protocol	Length	Info
3498	670.433951	192.168.68.110	178.79.227.76	HTTP	234	HEAD /pr/492350f6-3a01-4f97-b9c0-c7c6ddf67d60/Office/Data/v32_16.0.17628.20144.cab HTTP/1.1
3500	670.445559	178.79.227.76	192.168.68.110	HTTP	659	HTTP/1.1 200 OK
3509	670.937188	192.168.68.110	192.168.68.1	HTTP	251	GET /rootDesc.xml HTTP/1.1
3512	670.963343	192.168.68.1	192.168.68.110	HTTP/XL	1288	HTTP/1.1 200 OK
3531	675.173410	192.168.68.110	178.79.227.76	HTTP	234	HEAD /pr/492350f6-3a01-4f97-b9c0-c7c6ddf67d60/Office/Data/v32_16.0.17628.20144.cab HTTP/1.1
3533	675.193653	178.79.227.76	192.168.68.110	HTTP	659	HTTP/1.1 200 OK
3538	675.251933	192.168.68.110	178.79.227.76	HTTP	276	HEAD /pr/492350f6-3a01-4f97-b9c0-c7c6ddf67d60/Office/Data/v32_16.0.17628.20144.cab HTTP/1.1
3540	675.263858	178.79.227.76	192.168.68.110	HTTP	659	HTTP/1.1 200 OK
3541	675.301435	192.168.68.110	178.79.227.76	HTTP	345	GET /pr/492350f6-3a01-4f97-b9c0-c7c6ddf67d60/Office/Data/v32_16.0.17628.20144.cab HTTP/1.1
3543	675.330883	178.79.227.76	192.168.68.110	HTTP	701	HTTP/1.1 206 Partial Content
3544	675.364708	192.168.68.110	178.79.227.76	HTTP	276	HEAD /pr/492350f6-3a01-4f97-b9c0-c7c6ddf67d60/Office/Data/v32_16.0.17628.20144.cab HTTP/1.1
3546	675.385836	178.79.227.76	192.168.68.110	HTTP	659	HTTP/1.1 200 OK
3547	675.409903	192.168.68.110	178.79.227.76	HTTP	327	GET /pr/492350f6-3a01-4f97-b9c0-c7c6ddf67d60/Office/Data/v32_16.0.17628.20144.cab HTTP/1.1
3555	675.424868	178.79.227.76	192.168.68.110	HTTP	1234	HTTP/1.1 200 OK
5639	1896.405806	192.168.68.110	193.77.14.153	HTTP	165	GET /connecttest.txt HTTP/1.1
5641	1896.412519	193.77.14.153	192.168.68.110	HTTP	241	HTTP/1.1 200 OK (text/plain)
5651	1896.725600	192.168.68.110	193.77.14.153	HTTP	165	GET /connecttest.txt HTTP/1.1
5653	1896.733383	193.77.14.153	192.168.68.110	HTTP	241	HTTP/1.1 200 OK (text/plain)
5883	3362.659932	192.168.68.110	193.77.14.153	HTTP	165	GET /connecttest.txt HTTP/1.1
5886	3362.673503	193.77.14.153	192.168.68.110	HTTP	241	HTTP/1.1 200 OK (text/plain)
6151	3369.261879	192.168.68.110	2.23.27.37	HTTP	267	GET /en-G8/livetile/preinstall?region=SI&appid=C98EA580842DBB94058BF071E1DA76512D21FE36&FORM=T
6155	3369.281573	2.23.27.37	192.168.68.110	HTTP/XL	245	HTTP/1.1 200 OK

> Frame 3538: 276 bytes on wire (2208 bits), 276 bytes captured (2208 bits) on interface 0	0000	5c e9 31 04 2a 98 40 a3 cc 13 65 50 08 00 45 00	\.1.*@...eP...E-
> Ethernet II, Src: IntelCor_13:65:50 (40:a3:cc:13:65:50), Dst: TPLink_04:2a:98 (5c:00:10:00:00:00)	0010	01 06 cf 81 40 00 00 06 8f bd c0 a8 44 6e b2 4f	...@...Dn...O
> Internet Protocol Version 4, Src: 192.168.68.110, Dst: 178.79.227.76	0020	e3 4c dd f1 00 50 00 6f 25 b4 d5 41 11 ac 50 18	...L...P:o %:A...P
> Transmission Control Protocol, Src Port: 56817, Dst Port: 80, Seq: 1, Len: 276	0030	02 01 ee b6 00 00 48 45 41 44 20 ef 70 72 2f 34HE AD//pr/4
> Hypertext Transfer Protocol	0040	39 32 33 35 30 60 36 2d 33 61 30 31 2d 34 66 39	92350f6- 3a01-4f9
	0050	37 2d 62 39 63 30 2d 63 37 63 36 64 66 36 37	7-b9c0-c 7c6ddf67

Slika 40: Tunel VPN

Ko je vzpostavljena povezava s strežnikom VPN, poskrbimo, da gredo vse povezave do istega naslova IP. Vse poteka preko protokola UDP, saj je hitrejši kot protokol TCP.

3618	681.563105	192.168.68.110	142.250.180.170	UDP	71	59162 → 443 Len=29
3619	681.582300	142.250.180.170	192.168.68.110	UDP	67	443 → 59162 Len=25
3677	688.446257	142.250.180.170	192.168.68.110	UDP	120	443 → 59162 Len=78
3678	688.452145	192.168.68.110	142.250.180.170	UDP	75	59162 → 443 Len=33
3679	688.655311	192.168.68.110	142.250.180.170	UDP	71	59162 → 443 Len=29
3680	688.676782	142.250.180.170	192.168.68.110	UDP	67	443 → 59162 Len=25

Frame 3618: 71 bytes on wire (568 bits), 71 bytes captured (568 bits) on interface 0	0000	5c e9 31 04 2a 98 40 a3 cc 13 65 50 08 00 45 00	\.1.*@...eP...E-
Ethernet II, Src: IntelCor_13:65:50 (40:a3:cc:13:65:50), Dst: TPLink_04:2a:98 (5c:00:10:00:00:00)	0010	00 39 6c 2a 40 00 80 11 45 ce c0 a8 44 6e 8e fa	..91*...E...Dn...
Internet Protocol Version 4, Src: 192.168.68.110, Dst: 142.250.180.170	0020	b4 aa e7 1a 01 bb 00 25 62 01 55 ff f9 3f 83 30% b-U...?..0
User Datagram Protocol, Src Port: 59162, Dst Port: 443	0030	02 12 14 b8 d2 74 4d 3d 60 65 f7 25 a4 8c 2c 9ctm...e %:..,
Data (29 bytes)	0040	9e 11 7f 99 69 cb b31..

Slika 41: Kriptacija tunela VPN

7.4.1 WIRESHARK

Wireshark je najpomembnejši analizator omrežnih protokolov na svetu. Omogoča nam, da vidimo, kaj se dogaja v našem omrežju na mikroskopski ravni. To je de facto (in pogosto de jure) standard v mnogih panogah in izobraževalnih ustanovah (<https://www.wireshark.org/about.html>).

8 ZAŠČITA DDOS

Orodje za zaščito pred napadi DDOS optimizira omrežni promet tako, da blokira nelegitimne podatkovne pakete in tako poveča pasovno širino za uporabnike. To tudi poveča pretok zakonitih sočasnih dostopov. Idealno bi bilo, če bi naložil na Raspberry Pi zaščito IDS/IPS, kot je suricata, ampak tega moj Raspberry Pi žal ne premore, zato sem naredil svoj program proti napadom DDoS. Zaščita mora biti nameščena, saj bosta naslov IP strežnika VPN in posledično naslov IP mojega omrežja vidna na zunanjih omrežjih.

8.1 VRSTE NAPADOV DDoS

Volumetrični napadi DDoS so namenjeni preobremenitvi strežnika, omrežja ali baze podatkov s prekomernim prometom. Ti napadi lahko motijo normalno delovanje spletnih mest in spletnih virov, tako da zasičijo njihovo pasovno širino in izčrpajo njihove zmogljivosti.

- Vrste volumetričnih napadov DDoS: napadi UDP Flood Attacks – ti napadi poplavijo strežnik s pošiljanjem številnih paketov UDP, ki jih mora strežnik obdelati in pri tem izčrpati svoje vire. CharGEN Flood – izkorišča prednosti protokola CharGEN tako, da preobremeni strežnike z zahtevami, ki ustvarjajo nepotreben promet. ICMP Flood – preplavi tarčo z zahtevami za odmev (ping), ki porabljajo pasovno širino in procesorsko moč.
- Različice poplavnih napadov: UDP Fragmentation Flood – uporablja fragmentirane pakete, da zmede ciljni strežnik. Napadi z razširitvijo – napadalci pošiljajo zahteve večjim strežnikom s ponarejenim naslovom IP, kar ima za posledico ogromno odzivov tarči.
- Napadi protokol DDoS: napadalci izkoriščajo posebne protokole za porabo strežniških virov, kot so SYN Floods in ACK Floods, ki manipulirajo s postopkom povezave TCP.

- Napadi Session in Slowloris: ti napadi ohranjajo odprte povezave s pošiljanjem nepopolnih zahtev, pri čemer porabljajo strežniške vire brez znatne uporabe pasovne širine.
- Podedovani napadi: napadi Ping of Death in Smurf se opirajo na manipuliranje z omejitvami protokola, kar povzroča zrušitve ali prevelike odzive na oddajanje pingov.

Volumetrični napadi DDoS so resna grožnja spletnim storitvam, saj uporabljajo različne tehnike za izkoriščanje sistemskih ranljivosti in preobremenitev virov. Razumevanje teh napadov pomaga pri pripravi boljše obrambe in varovanju omrežne infrastrukture. (povz. po. <https://www.esecurityplanet.com/networks/types-of-ddos-attacks/>).

8.2 PROGRAM DDOS

Program bom naredil v programskem jeziku Python s pomočjo knjižnice Scapy. Scapy je zmogljiv interaktivni program in knjižnica za manipulacijo paketov, ki temelji na Pythonu. Sposoben je ponarediti ali dekodirati pakete širokega števila protokolov, jih poslati po žici, jih zajeti, shraniti ali prebrati z uporabo datotek pcap, ujemati zahteve in odgovore in še veliko več. Zasnovan je tako, da omogoča hitro izdelavo prototipov paketov z uporabo privzetih vrednosti, ki delujejo. Cilj programa bo, da bo spremljal tako promet tcp kot udp, štel bo, v koliko paketov pride iz vsakega naslova IP v nekem časovnem času. Programa ni bilo težko narediti, ker ima Scapy dobro dokumentacijo in ker je pred mano že veliko ljudi delalo podobne projekte. Objavljali so jih na spletnih straneh, kot sta Stack Overflow in GitHub.

```

GNU nano 7.2 dos_protection.py *
from scapy.all import *

packet_counts = {}

MAX_PACKETS_PER_SECOND = 200

WHITELIST = ["192.168.68.105", "192.168.68.106"]

def analyze_packet(packet):
    global packet_counts

    if IP in packet:
        src_ip = packet[IP].src

        if src_ip in WHITELIST:
            #print(f"Whitelisted IP address {src_ip}. Packet allowed.")
            #print(packet.summary())
            return

        packet_counts[src_ip] = packet_counts.get(src_ip, 0) + 1

        if packet_counts[src_ip] > MAX_PACKETS_PER_SECOND:
            print(f"Suspected DDoS attack from {src_ip}. Dropping packet.")
            return

    print(packet.summary())

def main():
    sniff(prn=analyze_packet, store=0)

if __name__ == "__main__":
    main()

```

Slika 42: Koda programa DDoS

Razlaga kode:

- packet_counts: ta »dictionary« se uporablja za shranjevanje števila paketov na izvorni naslov IP.
- Max_packets_per_second: ta spremenljivka določa največje število paketov v 1 sekundi.
- Whitelist: to je seznam z naslovi IP, ki jih ne želimo pregledovati.
- Funkcija (analyze_packet) se pokliče za vsak paket, ki ga zajame Scapy.
- Najprej preveri ali paket vsebuje IP. Če ne, preskoči.
- Izvleče izvorni naslov IP iz paketa (src_ip).
- Preveri, ali je naslov IP na seznamu whitelist. Če je, paketa ne obravnavamo.
- Če naslova IP ni na whitelisti, se število paketov za ta IP naslov začne šteti.
- Če število paketov preseže maksimalno število paketov na sekundo (max_packets_per_second), natisne sporočilo, ki nakazuje domnevni napad DOS in odvrže paket.
- Na koncu napiše povzetek paketa.
- Funkcija main (def main()) sproži pregledovanje paketov z uporabo Scapy.
- Posreduje funkcijo analyze_packet kot parameter prn (povratni klic), tako da vsak zajeti paket obdela funkcija analyze_packet.

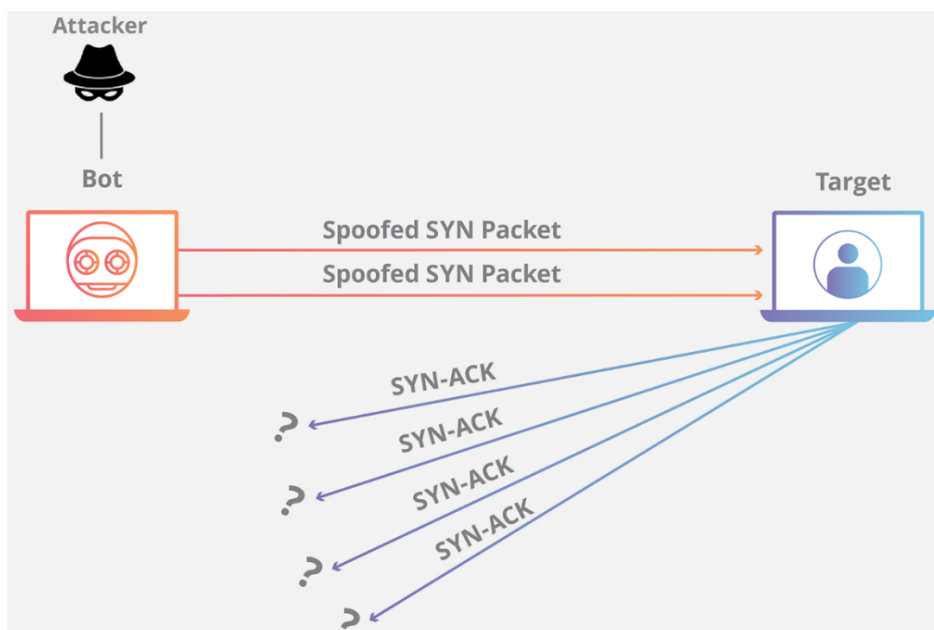
- Parameter store=0 zagotavlja, da zajeti paketi niso shranjeni v pomnilniku, da se prepreči prepolnitev pomnilnika.

8.3 TEST PROGRAMA

Za testiranje programa bom uporabil program Metasploit, projekt računalniške varnosti, ki zagotavlja informacije o varnostnih ranljivostih in pomaga pri testiranju prodora in razvoju podpisa IDS. Je v lasti bostonskega varnostnega podjetja Rapid7 s sedežem v Massachusettsu. V programu bom sprožil TCP SYN FLOOD. Poplava SYN je napad DDoS, ki porabi vsa razpoložljiva strežniška sredstva z večkratnim pošiljanjem paketov začetne zahteve za povezavo (SYN), zaradi česar se ciljni strežniški stroj počasi ali sploh ne odziva na zakonit promet.

1. Najprej odjemalec pošlje strežniku paket SYN, da vzpostavi povezavo.
 2. Strežnik se nato na ta začetni paket odzove s paketom SYN/ACK, da potrdi komunikacijo.
 3. Nazadnje odjemalec vrne paket ACK, da potrdi prejem paketa s strežnika.
- Po zaključku tega zaporedja pošiljanja in prejemanja paketov je povezava TCP odprta in lahko pošilja in prejema podatke.

(povz. po: <https://www.cloudflare.com/learning/ddos/syn-flood-ddos-attack/>)



Slika 43: Napad SYN Flood

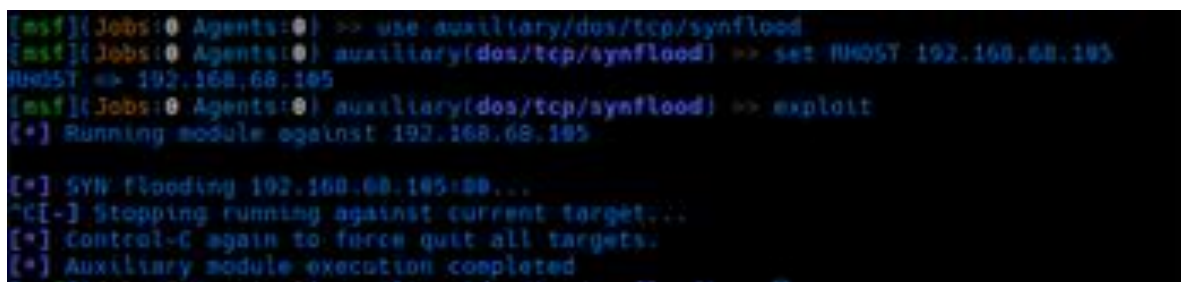
(<https://www.cloudflare.com/img/learning/ddos/syn-flood-ddos-attack/syn-flood-attack-ddos-attack-diagram-2.png/>)

Na drugem računalniku, kjer imam naložen Linux Parrot OS, kjer je program že naložen, ga enostavno odprem v konzoli z ukazom »msfconsole«.



Slika 44: Aplikacija Metasploit

Znotraj konzole Metasploit izberem napad SYN FLOOD z ukazom »use auxiliary/dos/tcp/synflood« in nastavim RHOST 192.168.68.105 ter nato exploit, da se napad začne izvajati.



Slika 45: Delovanje napada DDoS

```
Ether / Dot1Q / IP / UDP / BOOTP / DHCP Ack
Ether / Dot1Q / IP / UDP / BOOTP / DHCP Ack
Ether / IP / UDP 192.168.68.104:56069 > 239.255.255.250:1900 / Raw
Ether / IP / UDP 192.168.68.104:56069 > 239.255.255.250:1900 / Raw
Ether / ARP who has 192.168.68.250 says 192.168.68.1 / Padding
5c:e9:31:04:29:74 > 01:80:c2:00:00:13 (0x893a) / Raw
Suspected DDoS attack from 8.8.4.4. Dropping packet.
Ether / Dot1Q / 10.77.192.1 > 224.0.0.1 igmp / Raw / Padding
Ether / ARP who has 192.168.68.1 says 192.168.68.107 / Padding
Ether / IP / UDP / DNS Qry "b'192.168.68.107.in-addr.arpa.'"
Ether / IP / UDP / DNS Qry "b'192.168.68.107.in-addr.arpa.'"
Suspected DDoS attack from 8.8.4.4. Dropping packet.
5c:e9:31:04:29:74 > 01:80:c2:00:00:13 (0x893a) / Raw
Suspected DDoS attack from 8.8.4.4. Dropping packet.
Ether / ARP who has 192.168.68.102 says 192.168.68.1 / Padding
Ether / IP / UDP / BOOTP / DHCP Inform
5c:e9:31:04:29:74 > 01:80:c2:00:00:13 (0x893a) / Raw
Suspected DDoS attack from 8.8.4.4. Dropping packet.
Ether / ARP who has 192.168.68.1 says 192.168.68.105
Ether / ARP is at 5c:e9:31:04:2a:98 says 192.168.68.1 / Padding
5c:e9:31:04:29:74 > 01:80:c2:00:00:13 (0x893a) / Raw
Suspected DDoS attack from 8.8.4.4. Dropping packet.
Ether / IPV6 / UDP fe80::e9a6:a2a2:2558:54d6:dhcpv6_client > ff02::1:2:dhcpv6_server / DHCP6_Solicit / DHCP6OptElapsedTime / DHCP6OptClientId / DHCP6OptIA_NA / DHCP6OptClientIdFQDN / DHCP6OptVendorClass / DHCP6OptOptReq
Ether / ARP who has 192.168.68.249 says 192.168.68.1 / Padding
Suspected DDoS attack from 8.8.4.4. Dropping packet.
5c:e9:31:04:29:74 > 01:80:c2:00:00:13 (0x893a) / Raw
Ether / IP / UDP 192.168.1.9:39938 > 239.255.255.250:1900 / Raw
Ether / IP / UDP 192.168.1.9:39938 > 239.255.255.250:1900 / Raw
Ether / IP / UDP 192.168.1.9:39938 > 239.255.255.250:1900 / Raw
Suspected DDoS attack from 8.8.4.4. Dropping packet.
Ether / IP / UDP 192.168.1.9:39938 > 239.255.255.250:1900 / Raw
Ether / IP / UDP 192.168.1.9:39938 > 239.255.255.250:1900 / Raw
Ether / IP / UDP 192.168.1.9:39938 > 239.255.255.250:1900 / Raw
```

Slika 46: Normalno delovanje programa za preprečevanje napada DDoS

Ko iz nekega naslova IP prihaja več paketov, kot je dovoljeno v dani sekundi, ga označi kot napad DDoS in začne njegove pakete metati stran.

[illegible]

Slika 47: Delovanje programa DDoS pod napadom DDoS

8.4 IZ SKRIPTE V PROGRAM

Sedaj moram skripto dejansko narediti v programu, ki se bo cel čas izvajal v ozadju. Postopek za to je precej enostaven, najprej bom iz skripte naredil program, ki se bo izvajal, ga dodal v sistem Linux pod storitve in ga nato vključil.

```
mate@mate:~/Desktop/dos_protect $ ls
dos_protection.py
mate@mate:~/Desktop/dos_protect $ sudo chmod +x /home/mate/Desktop/dos_protect/dos_protection.py
mate@mate:~/Desktop/dos_protect $ ls
dos_protection.py
mate@mate:~/Desktop/dos_protect $ sudo nano /etc/systemd/system/dos_protection.service
mate@mate:~/Desktop/dos_protect $ sudo nano /etc/systemd/system/dos_protection.service
mate@mate:~/Desktop/dos_protect $ sudo systemctl daemon-reload
Unknown command verb daemon-reload.
mate@mate:~/Desktop/dos_protect $ sudo systemctl daemon-reload
mate@mate:~/Desktop/dos_protect $ sudo systemctl start dos_protection
mate@mate:~/Desktop/dos_protect $ sudo systemctl enable dos_protection
```

Slika 48: Iz skripte v program

```
GNU nano 7.2 /etc/systemd/system/dos_protection.service
[Unit]
Description=DDOS program
After=network.target

[Service]
ExecStart=/usr/bin/python3 /home/mate/Desktop/dos_protect/dos_protection.py
Restart=on-failure
User=mate
Group=nogroup

[Install]
WantedBy=multi-user.target
```

Slika 49: Datoteka .service

V mapo system, ki se nahaja na /etc/systemd, to je mapa za uporabnikove storitve, moram dodati novo storitev. To naredim v tekstovni datoteki .service. V datoteki definiram opis storitve in kdaj se izvaja v mojem primeru, ko je vzpostavljena internetna povezava. Določiti moram, kateremu uporabniku je to namenjeno. Da preverim, ali ta program dejansko deluje, v konzolo napišem systemctl status dos_protection. Napisati mora, da je program ACTIVE in ENABLED.

```
mate@mate:~/Desktop/dos_protect $ sudo systemctl status dos_protection
● dos_protection.service - DDOS program
   Loaded: loaded (/etc/systemd/system/dos_protection.service; enabled; preset: enabled)
   Active: active (running) since Mon 2024-06-03 20:29:14 CEST; 988ms ago
     Main PID: 360907 (python3)
        Tasks: 1 (limit: 1576)
           CPU: 979ms
    CGroup: /system.slice/dos_protection.service
            └─360907 /usr/bin/python3 /home/mate/Desktop/dos_protect/dos_protection.py
```

Slika 50: Pregled delovanja v ozadju

9 ZAKLJUČEK

V diplomski nalogi sem predstavil, kako bi ustvaril svoje varno omrežje, na katere probleme, nevšečnosti lahko pri tem naletim in kako lahko te probleme rešim.

Pri delu sem si zastavil tri hipoteze, prva hipoteza je narediti svoj program DDoS. Mislim, da sem v nalogi dobro pokazal, kako bi se tega lotil. Na internetu je na voljo ogromno že narejenih knjižnic, s katerimi si je mogoče pomagati, poleg tega obstajajo forumi, kjer so ta problem posamezniki že reševali, nekateri pa so tudi pripravljeni pomagati pri reševanju tega problema. Kar se v nalogi ne pokaže, je, da potrebujemo vsaj osnovno znanje programiranja, delovanja omrežja in osnovno znanje v programskem jeziku, v katerem bi ta program naredili.

Druga hipoteza je bila implementacija popolnoma neprebojnega sistema za zaščito omrežja na Raspberry Pi, ki bo zagotovila absolutno varnost pred vsemi vrstami kibernetских napadov. Ta hipoteza bi bila verjetno preveč absolutna in nedosegljiva, saj je v kibernetickem svetu težko zagotoviti absolutno varnost. Varnostne rešitve so tako lahko izpostavljene ranljivosti in napredni napadalci lahko najdejo načine, kako prebiti tudi najboljše varnostne mehanizme. Zato bi bila ta hipoteza verjetno preveč idealistična in nedosegljiva v praksi. Vse, kar lahko naredimo, je, da imamo vedno posodobljene programe, operacijske sisteme, posodobljene na najnovejše različice, in da posodabljammo gesla.

Tretja hipoteza je bila, da učinkovita uporaba filtriranja DNS lahko zmanjša število nezaželenih poizvedb in poveča varnost omrežja. Dobro sem pokazal, kako je lahko to učinkovito narejeno in tudi kakšne razlike so brez filtriranja in s filtriranjem. Za vsako stran, ki jo kliknemo na spletnem brskalniku, je za njo naslov IP in za vsako poizvedbo se povpraša strežnik DNS za cilj. Če že strežniku povemo, kateri naslovi IP niso varni, se tako lahko izognemo obisku te strani.

Zaključek:

V sklopu te naloge sem uspešno raziskal in implementiral različne varnostne ukrepe za ustvarjanje varnega domačega omrežja z uporabo platforme Raspberry Pi in kontejnerjev Docker. Z vzpostavitvijo Pi-hole za filtriranje oglasov, lastne rešitve za preprečevanje napadov DDoS in implementacijo VPN sem ustvaril celovito rešitev za varno in zanesljivo omrežje. Pregledal sem pomen filtriranja DNS, ki lahko zmanjša nezaželene poizvedbe in poveča varnost omrežja. Poleg tega sem se lotil tudi zaščite pred napadi DDoS, kar je ključnega pomena za ohranjanje stabilnosti omrežja v primeru napadov. Z uporabo kontejnerjev Docker sem omogočil učinkovito izvajanje in ločevanje funkcij ter zagotovil enostavno upravljanje aplikacij v omrežju. S tem sem ustvaril varno okolje, ki omogoča varno povezovanje v omrežje tudi izven doma. V prihodnosti bi lahko nadgradil varnostne ukrepe z implementacijo naprednejših orodij za analizo prometa, rednim testiranjem varnostnih rešitev in uvedbo večplastne varnostne strategije. S tem bi še dodatno okrepil varnost omrežja in se bolje zaščitil pred morebitnimi kibernetскими grožnjami. Ustvaril sem varno in zanesljivo domače omrežje, ki omogoča varno uporabo tudi v profesionalnem okolju. Varnost omrežja je ključnega pomena v digitalnem svetu, zato je nenehno izboljševanje in prilagajanje varnostnih ukrepov ključnega pomena za ohranjanje integritete in zaupanja v omrežje.

10 VIRI IN LITERATURA

1. *An introduction to uncomplicated firewall (UFW)* [online]. 2015. Dostopno na spletnem naslovu: <<https://www.linux.com/training-tutorials/introduction-uncomplicated-firewall-ufw/>> [Citirano 30. maj. 2024; 19.00].
2. *Cene energentov* [online]. 2024. Dostopno na spletnem naslovu: <<https://www.stat.si/StatWeb/Field/Index/5/30/>> [Citirano 29.maj. 2024; 12.00].
3. *Complete guide to the types of DdoS Attacks* [online]. 2022. Dostopno na spletnem naslovu: <<https://www.esecurityplanet.com/networks/types-of-ddos-attacks/>> [Citirano 29.maj. 2024; 12.00].
4. *Dangers of public Wi-fi* [online]. 2023. Dostopno na spletnem naslovu: <<https://www.aura.com/learn/dangers-of-public-wi-fi>> [Citirano 22. maj. 2024; 18.45].
5. *Docker* [online]. 2024. Dostopno na spletnem naslovu: <<https://docs.docker.com/get-started/overview/>> [Citirano 28. maj. 2024; 21.30].
6. *Pi-hole* [online]. 2024. Dostopno na spletnem naslovu: <<https://en.wikipedia.org/wiki/Pi-hole>> [Citirano 4. maj. 2024; 19.45].
7. *Protocol* [online]. 2024. Dostopno na spletnem naslovu: <<https://www.wireguard.com/protocol/>> [Citirano 29. maj. 2024; 18.01].
8. *Raspberry-pi* [online]. 2024. Dostopno na spletnem naslovu: <<https://opensource.com/resources/raspberry-pi>> [Citirano 15. maj. 2024; 22.00].
9. *Safe public Wi-Fi with a VPN* [online]. 2023. Dostopno na spletnem naslovu: <<https://nordvpn.com/blog/securing-public-wi-fi/>> [Citirano 29.maj. 2024; 13.00].
10. *Subneting benefits* [online]. 2017. Dostopno na spletnem naslovu: <<https://www.networkcomputing.com/ip-subnetting/5-subnetting-benefits>> [Citirano 29.maj. 2024; 13.09].

11. *Syn flood ddos attack* [online]. 2024. Dostopno na spletnem naslovu: <<https://www.cloudflare.com/learning/ddos/syn-flood-ddos-attack/>> [Citirano 29. maj. 2024; 21.10].
12. *Wireguard* [online]. 2024. Dostopno na spletnem naslovu: <<https://www.wireguard.com/>> [Citirano 28. maj. 2024; 21.00].
13. *Wireshark* [online]. 2024. Dostopno na spletnem naslovu: <<https://www.wireshark.org/about.html>> [Citirano 28. maj. 2024; 11.02].
14. *What is cybersecurity?* [online]. 2024. Dostopno na spletnem naslovu: <<https://www.kaspersky.com/resource-center/definitions/what-is-cyber-security/>> [Citirano 29.maj. 2024; 16.00].
15. *What is DNS?* [online]. 2024. Dostopno na spletnem naslovu: <<https://www.cloudflare.com/learning/dns/what-is-dns/>> [Citirano 20. maj. 2024; 18.00].
16. *What is ssh* [online]. 2024. Dostopno na spletnem naslovu: <<https://www.cloudflare.com/learning/access-management/what-is-ssh/>> [Citirano 7. maj. 2024; 20.05].