**<Net Rogue>**

**Programmas dokumentācija**

**Kursā Programmēšana**

**Versija 0.0.5**

|  |  |  |  |
| --- | --- | --- | --- |
| **Datums** | **Versija** | **Apraksts** | **Autors** |
| 2024.11.07. | 0.0.1 | Programmas idejas apraksts | Rihards Bukovskis |
| 2025.02.13. | 0.0.2 | Shēmas un to apraksts | Rihards Bukovskis |
| 2025.04.01. | 0.0.3 | Programmas testēšanas daļas iesniegšana | Rihards Bukovskis |
| 2025.04.06. | 0.0.4 | Pabeigtas programmas augšupielāde github repozitorijā | Rihards Bukovskis |
| 2025.04.07. | 0.0.5. | Pabeigtas dokumentācijas augšupielāde github repozitorijā | Rihards Bukovskis |

**Darba autori:**

**Rihards Bukovskis**

**Saturs**

[1. Problēmas izpēte un analīze (15 punkti) 3](#_Toc197548648)

[2. Gitlab vai github repozitorijs – uzaicināts pasniedzējs (15 punkti) 4](#_Toc197548649)

[3. Shēmas un to apraksti 5](#_Toc197548650)

[4. Izstrāde 9](#_Toc197548651)

[5. Testēšanas pārskats 12](#_Toc197548652)

[6. Izmantotie resursi 13](#_Toc197548653)

# Problēmas izpēte un analīze (15 punkti)

Šis projekts būs dažādu WIFi ievainojamību apkopojums vienā rīkā. Lietotājam būs iespēja izvēlēties kādu no implementētajām ievainojamībām un to palaist, izmantojot savu tīkla karti. Tomēr programmai būs divi ierobežojumi. Tā var tikt palaista tikai uz linux sistēmas, jo uz windows ir daudz grūtāk darboties tīkla zemajā līmenī, kā arī lietotāja tīkla kartei ir jāatbalsta “Monitor Mode”. Ja tīkla kartei nav iespējams uzlikt Monitor Mode, tas nozīme, ka ar tīkla karti nav iespējams noklausīties visu satiksmi, kas to padara neizmantojamu WiFi uzbrukumiem.

Sākumā programmā būs iespējams palaist 3+1 ievainojamību:

1. Programmā tiks iekļauti OWE uzbrukumi, kas tika izstrādāti mana ZPD laikā. Tiks izveidoti visi 3 uzbrukumi un ir sagaidāms, ka, tos izstrādājot C valodā, tie būs efektīvāki nekā iegūts ZPD darbā.
2. Tiks iekļauts Twin AP uzbrukums, kurā tiek viltots piekļuves punkts. Šo uzbrukumu, visticamāk, ka būs sarežģītāk implementēt nekā citus, daudzo mainīgo dēļ. Piemēram, atkarībā no drošības protokola ir vairāki varianti, tad vēl ir piekļuves punkta specifiskie parametri, kā piemēram, datus pārraides ātrumi.
3. Pēdējā ievainojamība, ko būs iespējams testēt ir ievainojamība, kur, izmantojot noklausīto PMKID no WPA2 tīkla ir iespējams uzlauzt tīkla paroli. Programma pievienosies pie piekļuves punkta līdz 4-pušu rokasspiediena pirmajai ziņai – tajā ir minēts PMKID. To izmantojot, ir iespējams lokāli iegūt paroli.

Šis projekts ir domāts, manas C valodas pielietojuma prasmju uzlabošanai un labākai WiFi ievainojamību izprašanai. Ir viegli lasīt par ievainojamībām un domāt, ka viss ir saprasts, tomēr reāli mēģinot tās implementēt, ir iespējams tās ļoti labi izprast, kā arī implementēšanas procesā var rasties arī jaunu ievainojamību idejas. Šis projekts ir arī domāts manu C valodas prasmju uzlabošanai. Iepriekš es esmu izstrādājis dažas ievainojamības Python programmēšanas valodā. Lai gan ir viegli uztaisīt ātrus ievainojamību testus, tie ne vienmēr izpilda nepieciešamās prasības, piemēram, tie nebūs pietiekami ātri, ņemot vērā, ka Python ir augsta līmeņa programmēšanas valoda. Tāpēc es esmu izvēlējies šo projektu izveidot, izmantojot C valodu, kas sniedz zema līmeņa piekļuvi tīklam un paketēm, kā arī C valodai ir “zema aprēķinu slodze” (no ang. Val. “Low computational overhead”), kas nozīmē, ka būs iespējams izveidot daudz ātrākas programmas, kas dažos gadījumos var uzlabot uzbrukuma izdošanās efektivitāti. Labs piemērs šim ir viens no maniem ZPD uzbrukumiem, kur tika viltota klienta asociācijas pieprasījuma pakete. Ar Python šim uzbrukumam bija tikai 10% efektivitāte, jo pakete bija jāizsūta ļoti īsā laika sprīdī. Mana teorija ir, ka, ja šo pašu uzbrukumu implementētu C valodā, tad uzbrukuma efektivitāte ievērojami uzlabotos. Kā arī ir daži uzbrukumi, kuriem ir nepieciešams ļoti liels ātrums, lai vispār tie izdotos, kā piemēram, “reactive jamming” uzbrukums, kur tikai konkrētas paketes tiek bloķētas. Tas notiek reāllaikā analizējot paketes, kas tiek sūtītas, un, ja konkrētā paketē atbilst kritērijiem, tad atlikusī paketes daļa tiek bloķēta. Tāpēc pēc iespējams implementējamo uzbrukumu analīzes, esmu izvēlējies šo projektu izstrādāt C valodā.

Šīs programmas galvenie lietotāji būtu WiFi entuziasti, kas interesējas par drošību. Ar šīs programmas palīdzību lietotāji būs spējīgi paši izmēģināt uzbrukumus, lai, piemēram, pārbaudītu vai viņu tīkla iekārtas nav ievainojamas pret dotajiem uzbrukumiem.

# Gitlab vai github repozitorijs – uzaicināts pasniedzējs (15 punkti)

Izveidots repozitorijs, piekļuve grupas biedriem un pasniedzējam (5p) Linku ielikt šeit dokumentā

<https://github.com/rihrds/NetRogue>

**Ierobežojumi**

Programma strādās tikai uz linux ierīcēm, jo linux piedāvā visērtāko pieeju zemajam tīkla līmenim, kamēr ar windows to izdarīt ir daudz sarežģītāk, un tikai ar tīkla kartēm, kuras ir iespējams ielikt Monitor mode, jo tikai tā var noklausīt visu tīkla satiksmi.

Owe uzbrukumus var palaist, tad, ja ir vismaz 1 tīkla karte, bet PMKID un Twin AP uzbrukumiem ir nepieciešamas 2 tīklas kartes, bet tikai 1 ir jāatbalsta Monitor režīms

# 3. Shēmas un to apraksti

A diagram of a program

Description automatically generated

Šajā diagrammā aprakstīts, kā notiks Twin AP uzbrukuma izpilde. No sākuma klients palaiž programmu un tai dod argumentu, ka vēlas izpildīt uzbrukumu. Klients var dot papildus argumentus, piemēr, konkrētu kanālu uz kura vajadzētu darboties viltus Ap, ar kādu SSID tam darboties.. Tālāk programma pārbauda, vai lokāli jau eksistē AP ar doto nosaukumu, ja tas eksistē, tad programma nokopē AP parametrus no tā beacon paketēm, lai būtu pēc iespējas līdzīgāks īstajam AP. Ja neatrod AP ar doto SSID, tad pārējos klienta nepiedāvātos parametrus tas izvēlas pats, piemēram, kanālu tas izvēlas visneaizņemtāko. Tālāk tas sāk pārraidīt savas Beacon paketes, mēģinot piesaistīt klientus, lai tie pievienotos viltus tīklam. Kad kāds lietotājs, mēģina pievienoties, programma to veiksmīgi asociē un paziņo lietotājam, ka ir kāds pievienojies pie viltus AP. Tālāk programma veic upura pakešu maršrutēšanu caur kādu tīklu, pie kura tas ir pievienojies, lai upuris tik un tā spētu piekļūt internetam, bet satiksmi būtu iespējams noklausīt.

A diagram of a program

Description automatically generated

Šis uzbrukums ir paņemts no mana ZPD darba, kur programma viltos Piekļuves punkta asociācijas atbildes paketi. Šis uzbrukums balstās uz principa, ka klients pirmo apstrādās viltoto asociācijas atbildes paketi un ignorēs īsto paketi.

Programma vispirms savienosies ar īsto piekļuves punktu, lai iegūtu pēc iespējas pareizāku asociācijas atbildi. Ja tas mēģinātu uztaisīt pats savu asociācijas atbildi, tad iespējams tas tiktu neprecīzi uztaisīts, kas nozīmē, ka klients to neapstrādātu un uzbrukums būtu neizdevies. Tātad programma no sākuma pievienosies pie īstā piekļuves punkta, iegūs tā asociācijas atbildi, to saglabās un atvienosies. Tālāk tas sāks klausīties, kad kāds klients sāks pievienoties pie īstā piekļuves punkta (šo programmu ir iespējams arī mērķēt uz konkrētiem klientiem, norādot to mac adresi). Kad tiks uztverts, ka piekļuves punkts izsūta autentifikācijas atbildi (atkarībā no programmas ātrumu, programma varētu arī reaģēt uz asociācijas pieprasījumu), programma izsūtīs viltoto asociācijas atbildi. Vienīgais veids, kā programma spēs noteikt, vai uzbrukums ir izdevies, ir, klausoties, vai netiek nosūtīta 4-pušu rokasspiediena 3. Ziņa. (Par veiksmes gadījumu vēl tiks domāts tālāk, jo kā ZPD rāda, dažām ierīcēm vajag 15 sekundes, lai padotos uz savienošanos ar piekļuves punktu)

A diagram of a program

Description automatically generated with medium confidence

Šajā uzbrukumā programma mēģinās pieslēgties pie visiem klienta minētajiem piekļuves punktiem, lai iegūtu to PMKID no to EAPOL ziņas, kas ir 4-pušu rokasspiediena 1. Ziņā atrodams. Klients var norādīt vienu, vairākus, vai visus lokālos AP, kuriem uzbrukt. Tālāk programma pēc kārtas, pievienosies pie šiem AP, ja tie eksistē un iegūs to PMKID. Pēc PMKID iegūšanas programma paziņos, ka ir veiksmīgi iegūts konkrētā piekļuves punkta PMKID, un tiks sākta PMKID uzlaušana ar Bruteforce metodi, kur tiks mēģinātas miljoniem iespējamo paroļu pēc kārtas, ja, izejot cauri visām populārākajām parolēm cauri, neizdodas uzlauzt PMKID, tad PMKID tiks aizsūtīts lietotājam ar ziņu, ka nav programmai neizdevās to uzlauzt.

Šis uzbrukums balstās uz hashcat galvenā izstrādātāja atklājuma 2016. Gadā. Šī metode ir labāka par iepriekš plaši izmantoto metodi, kur, lai varētu mēģināt uzlauzt tīkla paroli bija jānoklausa pareizi visas 4-pušu rokasspiediena ziņas. Šim uzbrukumam vajag noklausīt tikai 1. Ziņu. Šis uzbrukums nav tik efektīvs pret WPA3 tīkliem, jo to šifrēšanas metodes ir advancētākas, tomēr WPA2 vēl ar vien tiek plaši izmantoti.

A diagram of a program

Description automatically generated

# Izstrāde

Projekts, kā jau tika vairākas reizes iepriekš minēts, tika izstrādāts C valodā, tās zemā tīkla līmeņa piekļuves un augstās izpildes ātruma dēļ. Projekts tika sadalīts pa vairākiem failiem, lai tos būtu vieglāk pārskatīt, ātrāk atrast un vieglāk izmainīt. Katram c failam, izņemot galveno netrogue.c failu ir atbilstoši .h *header* jeb galvenes faili. Tie satur Makro definīcijas, kas pārliecinās, ka *Preprocessor* galvenajā saliktajā programmā neimportētu to pašu failu vairākas reizes, kā arī galvenes failos atrodas dažādu struktūru definīcijas un funkciju prototipi, kas nepieciešami, lai *Assembler* varētu pārliecināties, kas visas funkcijas tiek pareizi izmantotas, kad tas liek kopā gala programmu.

Kopumā ir 7 dažādi C faili, īsumā par katru no tiem:

* netrogue.c – galvenais programmas fails, kas satur kodu, kas apstrādā programmas palaišanas ievadītos parametrus, kā arī iestata interfeisu Monitor režīmā un arī atrod piekļuves punkta Mac adresi un kanālu, uz kura tas darbojas un attiecīgi interfeisu nomaina uz to kanālu. Šī programma attiecīgi pēc programmas palaišanas parametriem sāk attiecīgā uzbrukuma izpildi.
* helper.c - šajā failā atrodas dažādas kopējas funkcijas, ko izmanto dažādi faili šajā programmā. Tā, kā nebūtu loģiski, ja katram failam būtu sava vienādas funkcijas definīcija, tad tika nolemts, ka būs šāds helper fails.
* owe\_attacks.c – Šī programma satur visu 3 owe uzbrukumu palaišanas funkcijas, kā arī uzbrukumiem nepieciešamo filtru funkcijas.
* packet.c – Šajā failā atrodas dažādas funkcijas, kuras izveido paketes, kuras tiek izmantotas owe uzbrukumos
* pmkid\_attack.c – Šī programma satur pmkid uzbrukuma funkcijas kodu.
* sniff.c – Šajā failā ir funkcija, ko izmanto owe uzbrukumos, lai noklausītos tīklu un atrastu specifisku paketi, kas nepieciešama uzbrukuma izpildei. Pareizo pakešu filtra funkcijas tiek pasniegtas kā funkcijas *pointer* argumentā.
* twin\_ap\_attack.c – Šajā failā atrodas Twin AP uzbrukuma palaišanas kods.

Šajā projektā tika izmantotas dažādas tehnoloģijas dažādiem uzbrukumiem, jo nebija tādas vienas tehnoloģijas, kas varētu izpildīt dažādu uzbrukumu prasības, kā tika atklāts izstrādes procesā. Owe uzbrukumiem un piekļuves punkta atrašanai tiek izmantota *Raw Socket* tehnoloģijas, kas atver *socket*, kas ļauj piekļūt noklausītajām paketēm pirms tās tiek apstrādātas, kas ļauj tiešu saziņu ar tīklu un maksimāli kontrolētu pakešu izveidi. Tieši OWE uzbrukumiem šī tehnoloģija ir svarīga, jo owe uzbrukumos ir nepieciešams izveidot specifiskas paketes un tās ļoti kontrolēti izsūtīt. Piekļuves punktu meklēšana arī notiek izmantojot *raw socket*, jo visos uzbrukumos, kur ir nepieciešams atrast piekļuves punktus, tajos jau tiek izveidoti *raw socketi* un to ir ērti izmantot, lai pārbaudītu dažādus kanālus.

PMKID uzbrukumi gan tiek veikti nedaudz citādāk. Monitor interfeisa uzstādīšanai tiek izmantoti *raw socket*, bet savienojumu ar interfeisu veic cits interfeisu. Šajā gadījumā visērtākais variants bija ļaut *wpa\_supplicant* aplikācijai vadīt savienošanās procesu, jo nekādas specifiskas paketes nav nepieciešams izsūtīt. *Wpa\_supplicant* programmatūra ir tā, kas pamatā regulē savienošanās procesus tīkliem. Lai no *pcap* failiem iegūtu *pmkid* *hash*, tika izmantota programma, kas nenāk pamatā ar linux un ko ir nepieciešams atsevišķi instalēt caur github: *hcxpcapngtool*. Šī aplikācija no *pcap* faila, kas satur EAPOL ziņas, kuras iegūtas savienošanās procesā ar WPA2 tīklu, spēj iegūt *hash*, ko tālāk var mēģināt uzlauzt. Netrogue programmatūra pati neuzlauž šos *hash*, bet sniedz lietotājam komandu, ar kuru to ir iespējams izdarīt. *Hash* uzlaušana tiek veikta ar *hashcat* programmas palīdzību. Tīkla noklausīšanai tika izmantota vēl viena programma, kas lielākajā daļā gadījumu nāk kopā ar linux sistēmu: *tcpdump****.***

Twin AP uzbrukumā tiek izmantotas vairākas pamata linux tīkla programmas, jo nav nepieciešams no jauna izveidot to, kas jau pastāv. Tālāk īsumā aprakstīts, kam katra no tām tiek izmantota

* hostapd – tiek izmantots, lai izveidotu un pārvaldītu piekļuves punktu
* dnsmasq – piedāvā DHCP pievienojušajiem klientiem
* iptables – tiek izmantots, lai uzstādītu NAT, lai piedāvātu interneta savienojumu klientiem
* ip route – tiek izmantots lai pārvaldītu maršrutēšanas tabulu
* nmcli - tiek izmantots, lai izslēgtu NetworkManager kontroli pār specifisku interfeisu
* ifconfig – tiek izmantots, lai uzstādītu statisku ip adresi
* /proc/sys/net/ipv4/ip\_forward – tiek izmantots lai izslēgtu/izslēgtu IP pāradresāciju

Projektā izmantotās “bibliotēkas”: *stdio.h, stdlib.h, unistd.h, string.h, signal.h, sys/types.h, sys/wait.h, fcntl.h, errno.h, stdbool.h, sys/socket.h, net/ethernet.h, netinet/in.h, linux/if.h, linux/if\_packet.h, linux/wireless.h, sys/ioctl.h, time.h, stdint.h*

Visas minētās bibliotēkas ir pieejamas bez atsevišķas lejupielādes uz linux sistēmām.

**Projekta izstrādes gaita**

A screen shot of a computer

AI-generated content may be incorrect.Projekts tapa pa soļiem. Vispirms tika izveidots owe\_asso\_resp\_replay uzbrukums bez jebkādiem programmatūras ievades vai piekļuves punkta meklēšanas funkcijām, ar iestatītām vērtībām, lai pārliecinātos, ka raw socket strādā. Tālāk jau tika izveidoti pārējie 2 owe uzbrukumi pēc līdzīgas metodikas. Kad tika tapuši visi 3 owe uzbrukumi, tad tika izveidots arī tas, ka lietotājs var norādīt, kuru uzbrukumu vēlas veikt un norāda arī attiecīgi interfeisu, ar kuru tas vēlas to veikt. Tālāk tika izveidots tas, ka programma pati spēj iestatīt interfeisu tā, lai tas būtu izmantojams uzbrukumos, tādejādi, klientam pašam interfeiss nav jāieliek Monitor režīmā. Ap šo brīdi arī sākās testēšana.

**1.attēls. Programmas interfeiss ap testēšanas laiku**

Nākamais uzbrukums, kas tika implementēts bija PMKID uzbrukums, jo tas pēc uzstādīšanas principa ir līdzīgs OWE uzbrukumiem. Tas no sākuma tika izveidots, kā individuāla programma, un, kad tika pārbaudīts, ka tā strādā, tad tā tika integrēta oriģinālajā netrogue programmatūrā. Līdzīgs process bija ar Twin AP uzbrukumu. Līdz ar to tika arī atjaunināts programmas sākšanas interfeiss (atjaunināts norādījums par to, kurus uzbrukums ir iespējams palaist un kādi parametri tiem nepieciešami) Pēdējā lieta, kas bija jāimplementē bija piekļuves punkta atrašana pēc SSID. Šo bija diezgan vienkārši implementēt ar iegūtajām A screen shot of a computer

AI-generated content may be incorrect.zināšanām un programmas esošo pamatu.

**2.attēls. Programmas interfeiss izstrādes beigās.**

***Piezīme:***

*Tā kā programma tiek palaista un darbojas komandrindas saskarnē, tad man šajā sadaļā nebija daudz ekrānšāviņu, ko pievienot, lai parādītu programmas progresu izveides procesā.*

# Testēšanas pārskats

Testus veica: Georgiy Furanets

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Akcepttesta apraksts | Ievade/darbība | Sagaidāms rezultāts | Rezultāts | |
| owe\_asso\_resp\_replay uzbrukumam jābūt spējīgam neļaut katrai no piedavātām ierīcēm pieslēgties pie tīkla | Uzbruk rūterim, izmantojot owe\_asso\_resp\_replay algoritmu, tad pieslēdzas ar katru ierīci un novero pieslēgšanas rezultātus | Uzbrukums ir veiksmīgs | Tālrunis (Samsung) | Izdevās |
| Tālrunis (iPhone) | Izdevās |
| Dators (Linux) | Izdevās |
| Dators (Windows) | ⅖ Izdevās |
| owe\_asso\_resp\_code uzbrukumam jābūt spējīgam neļaut katrai no piedavātām ierīcēm pieslēgties pie tīkla | Uzbruk rūterim, izmantojot owe\_asso\_resp\_code algoritmu, tad pieslēdzas ar katru ierīci un novero pieslēgšanas rezultātus | Uzbrukums ir veiksmīgs | Tālrunis (Samsung) | Izdevās |
| Tālrunis (iPhone) | Izdevās |
| Dators (Linux) | Izdevās |
| Dators (Windows) | ¾ Izdevās |
| owe\_asso\_req uzbrukumam jābūt spējīgam neļaut katrai no piedavātām ierīcēm pieslēgties pie tīkla | Uzbruk rūterim, izmantojot owe\_asso\_req algoritmu, tad pieslēdzas ar katru ierīci un novero pieslēgšanas rezultātus | Uzbrukums ir veiksmīgs | Tālrunis (Samsung) | Izdevās |
| Tālrunis (iPhone) | Izdevās |
| Dators (Linux) | ⅔ Izdevās |
| Dators (Windows) | Neizdevās |
| Uzbrukuma uzstādīšanas laiks nepārsniedz 500 ms (0,5 s) | Palaiž katru no esošajiem uzbrukumiem, pārbaudot to uzstādīšanas laikus | Algoritma izpildes laiks <500 ms | owe\_asso\_resp\_replay | 25,6 ms |
| owe\_asso\_resp\_code | 28,9 ms |
| owe\_asso\_req | 118,4 ms |

# Izmantotie resursi

Projekta izstrādē pārsvarā tika izmantoti blogi, kur tika aprakstīti implementētie uzbrukumi, kā arī citas ar uzbrukumu implementāciju saistītas lietas, piemēram, par socketiem. Projekts izstrādē netika izmantoti zinātniskie pētījumi vai citu autoru rezultāti. Tālāk var redzēt saišu uzskaitījumu, kas tika izmantotas, lai implementētu katru uzbrukumu:

**OWE uzbrukumi:**

* <https://www.man7.org/linux/man-pages/man7/netdevice.7.html>
* <https://mrncciew.com/2014/10/10/802-11-mgmt-authentication-frame/>
* <https://mrncciew.com/2014/10/28/802-11-mgmt-association-reqresponse/>
* <https://beej.us/guide/bgnet/pdf/bgnet_a4_c_1.pdf>
* <https://www.opensourceforu.com/2015/03/a-guide-to-using-raw-sockets/>
* <https://www.baeldung.com/cs/raw-sockets>
* <https://datatracker.ietf.org/doc/html/rfc8110>

**Twin AP uzbrukums:**

* <https://samhilliardblog.com/building-your-own-evil-twin-access-point-from-scratch/>
* <https://samhilliardblog.com/evil-twin-part-2-spying-on-victims-with-a-mitm-attack/>
* <https://stackoverflow.com/questions/5237482/how-do-i-execute-an-external-program-within-c-code-in-linux-with-arguments>

**PMKID uzbrukums:**

* <https://null-byte.wonderhowto.com/how-to/hack-wi-fi-cracking-wpa2-passwords-using-new-pmkid-hashcat-attack-0189379/>
* Saite uz *hcxpcapngtool* github repozitoriju ar norādēm kā instalēt: <https://github.com/ZerBea/hcxtools>
* <https://www.tcpdump.org/>

Projekta izstrādē arī tika izmantots mākslīgais intelekts. Tika izmantots claude.ai, nevis chatgpt, jo claude sniedz daudz pilnvērtīgākas un sakarīgākas atbildes. Ar mākslīgo intelektu notika konsultēšanās par neskaidriem jautājumiem projekta izstrādes laikā, kā arī ar programmu nesaistītām lietām, piemēram, interfeisu restartēšanu pēc uzbrukumiem.

Visas projektā izmantotās bibliotēkas, kas minētas izstrādes sadaļā, ir jau pieejamas ar linux sistēmu, tāpēc nav īsti saišu, kuru te pievienot.

**Saite uz projekta github repozitoriju:** <https://github.com/rihrds/NetRogue/tree/main>