Síťové aplikace a správa sítí

Monitorovanie DHCP komunikácie

Matej Kezniki - xkezni01

6.10.2023

Obsah

1 Úvod do problematiky

1.1 DHCP

Dynamic Host Configuration Protocol (DHCP) je protokol, ktorý umožňuje automatické pridelenie IP adries a ďalších sieťových konfigurácií zariadeniam v počítačovej sieti. Jeho hlavným účelom je zjednodušiť správu IP adries a zabezpečiť, aby každé zariadenie v sieti malo správnu konfiguráciu pre komunikáciu v sieti [1].

1.2 Prefix

Prefix v kontexte DHCP sa vzťahuje na časť IP adresy, ktorá je spoločná pre všetky adresy v danej sieti. To zahŕňa prvé bity IP adresy, ktoré určujú sieťovú adresu. Napríklad, ak máte prefix /24, znamená to, že prvých 24 bitov adresy tvorí sieťová časť a zostávajúce bity predstavujú individuálne zariadenia v tejto sieti.

1.3 Z akého dôvodu je nutné analyzovať využitie prefixu?

Dynamickým prideľovaním adries DHCP môže dôjsť k vyčerpaniu dostupných adries v danom prefixe alebo rozsahu. Keďže adresa je prideľovaná dynamicky a môže byť uvoľnená, monitorovanie obsadenosti umožňuje sledovať dostupné adresy a predchádzať problémom s nedostatkom adries. To je obzvlášť dôležité v rozsiahlych sieťach, kde môže dochádzať k častejšiemu pripájaniu a odpojovaniu zariadení. Pre administrátorov je to kritické pretože to. zabezpečuje správne fungovanie siete a minimalizuje riziko konfliktov IP adries. Taktiež to umožňuje plánovať a prípadne rozšíriť adresné rozsahy alebo rozsahy, aby bola zabezpečená dostatočná kapacita pre súčasnú sieťovú prevádzku. Efektívne riadenie dostupných adries je tak kľúčovým aspektom pre **stabilnú** a **bezproblémovú** prevádzku siete.

2 Návrh a implementácia aplikácie

Aplikácia bola implementovaná v jazyku C++, revízia C++20 (ISO/IEC 14882:2020).
Z dôvodu, že sa jedná o rozsiahlejší projekt – má okolo 1000 riadkov, bolo nutné rozdeliť činnosti do funkcií, tried alebo modulov pre lepšiu čitateľnosť a kvalitu kódu.

2.1 Štruktúra

Zdrojový kód je rozdelený do viacerých modulov, pričom súčasťou každého modulu sú *.cpp a *.hpp súbory. Hlavným modulom je *dhcp-stats*, ktorý tvorí jadro programu a medzi ďalšie moduly patria:

- *Arguments*
  - spracovanie a overenie správnosti argumentov príkazovej riadky a práca s vektorom prefixov
- *IP_prefix*
  - operácie s vektorom Clients a združovanie informácií o jednotlivých prefixoch
- *Clients*
  - predstavuje klienta s pridelenou IP adresou
- *DHCP*
  - správa a overenie informácií získaných z DHCP packetov a overenie žiadateľov o pridelenie IP adresy
- *Negotiator*
  - predstavuje žiadateľa o pridelenie IP adresy
- *Utils*
  - funkcie využívané na viacerých miestach v rámci projektu
- *Error*
  - modul obsahujúci funkcie potrebné na vyriešenie chýb zo vstupu alebo v programe

2.2 Spracovanie a uloženie argumentov príkazovej riadky

Spracovanie argumentov má na práci trieda Arguments. V triede sú uložené všetky argumenty, ich stavy a taktiež vektor prefixov zadaných od užívateľa. V rámci spracovania argumentov trieda taktiež overuje, či sa jedná o korektne zadaný prefix. Tieto kontroly majú na starosti inštančné metódy triedy Arguments - *check_overlap()*, *check_ip_format()* a *check_mask()*. V prípade, ak je prefix správny, vytvorí sa inštancia triedy IP_prefix, kde sa uložia všetky potrebné informácie o prefixe a následne sa inštancia vloží do vektora prefixov.

2.3 Zachytávanie paketov

Pokiaľ prebehlo všetko v poriadku a do príkazového riadku nebol zadaný nejaký nezmysel alebo nebola vypísaná *help* správa, program sa presunie do tejto časti.

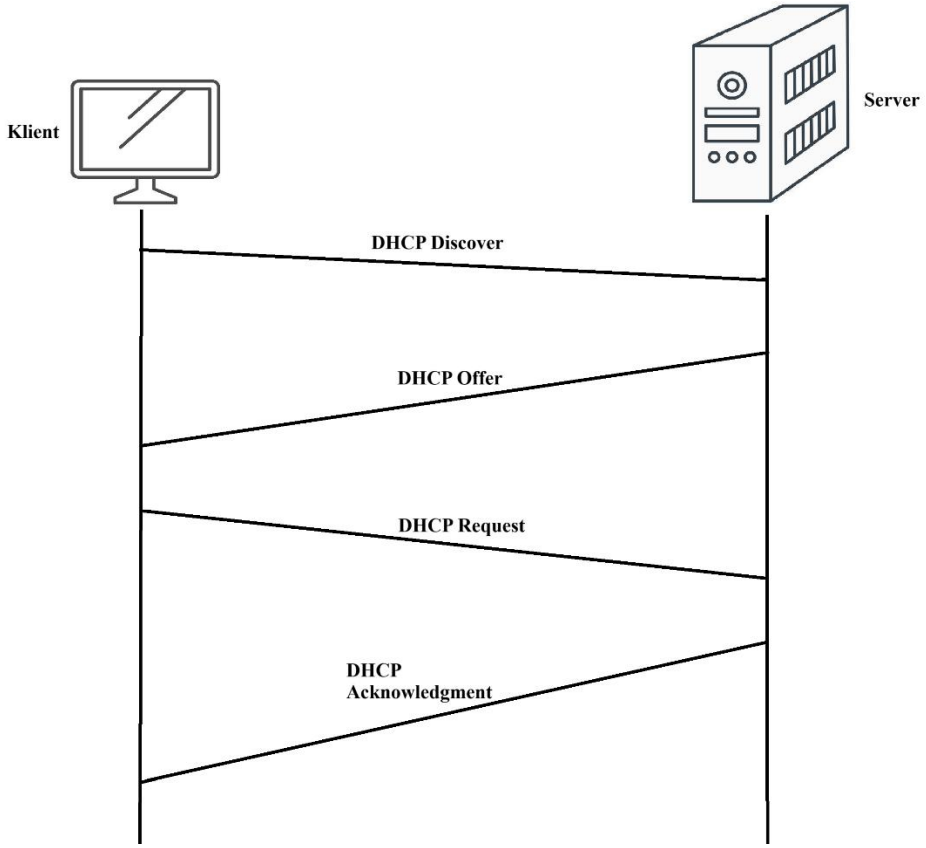
Podľa vstupu od užívateľa sa otvorí buď rozhranie, na ktorom bude program zachytávať pakety alebo sa otvorí .pcap súbor. Aplikuje sa a nainštaluje zachytávací filter (capture filter), ktorý je *(udp port 67 or udp port 68) and ip* pretože protokol DHCP pracuje iba v rámci protokolov IPv4 (Internet Protocol version 4) a UDP (User Datagram Protocol) na portoch 67 a 68 [1].

V prípade akéhokoľvek neúspechu v týchto činnostiach je program ukončený s chybovou hláškou oznamujúcou užívateľa o dôvode chyby.

Následne začne funkcia *pcap_loop()* zachytávať pakety vyhovujúcemu zachytávajúcemu filtru. O ich ďalšie spracovanie sa stará funkcia *packet_handler().*

2.4 Spracovanie paketov

Spracovanie zachytených DHCP paketov má na starosti funkcia *packet_handler() a trieda DHCP*. Zachytený paket je ako parameter predaný funkcii *verify_dhcp_negotiation()*, ktorá vytvorí inštanciu triedy *Negotiator* do ktorej uloží potrebné informácie o žiadateľovi o IP adresu od DHCP servera a inštanciu vloží do vektora *clients_without_ip*. Overuje tiež poradie správ a úplnosť komunikácie medzi serverom a klientom DHCP tak ako je možné vidieť na Obrázku č.1.



Obrázok č.1

Ak prebehne komunikácia medzi serverom a klientom DHCP korektne, overí sa identita žiadateľa s informáciami z *DHCP Acknowledgement* správy funkciou *check_MAC_IP_pair(),* aby sa vylúčila chyba prideľovania adries DHCP servera. Jedná sa o chybu veľmi málo pravdepodobnú chybu, ale je nutné aby MAC a IP adresa získaná z komunikácie sedela s informáciami z *DHCP Acknowledgement* správy.

V prípade korektnosti komunikácie medzi klientom a serverom DHCP bude vytvorená inštancia triedy *Client*, kde budú uložené informácie o klientovi. Následne bude inštancia uložená do daných prefixov, do ktorých IP adresa klienta patrí a aktualizujú sa riadky aplikácie obsahujúce prefixy do ktorých adresa klienta patrí.

Aplikácia taktiež podporuje uvoľňovanie IP adries z prefixu v prípade poslania správy *DHCP RELEASE* klientom, pričom sa jedná o rozšírenie projektu nad rámec zadania. Uvoľnenie zabezpečuje metóda *release()*. Avšak klient nemá žiadnu povinnosť správu *DHCP RELEASE* serveru poslať.

Z tohoto dôvodu sú informácie, ktoré poskytuje aplikácia mierne skreslené a pre použitie v praxi by bolo potrebné taktiež implementovať mechanizmus, ktorý by overoval, že daný klient má stále pridelenú IP adresu – lease time.

V prípade, ak je prefix zaplnený aspoň na 50%, vypíše sa na štandardný výstup hláška *" prefix x.x.x.x/y exceeded 50% of allocations."* a identická hláška sa zapíše aj do logu pomocou mechanizmu syslog. Ďalšie hlášky budú vypisované v prípade 60,70,80 a 90 % využitia prefixu.

2.5 Ukončenie

Ukončenie aplikácie je možné pomocou stlačenia CTRL+C. Stlačenie vygeneruje signál SIGINT, ktorý je zachytený pomocou funkcie *signal_handler()*, pričom sú taktiež uvoľnené všetky prostriedky, ktoré boli potrebné pre chod programu.

3 Implementačné detaily a chovanie

Implementačné detaily a popis chovania:

- Podporovaný protokol
  - Aktuálne je podporovaný iba DHCPv4 protokol.

- Chyby za behu
  - Vypisujú sa tesne pred ukončením programu.
  - Popis chyby reflektuje stav, do ktorého sa program dostal.

4 Testovanie

Testovanie DHCP analyzátora je kritický proces, ktorý overuje jeho správne fungovanie a spoľahlivú analýzu DHCP komunikácie v sieti, preto bol naň kladený veľký dôraz.

4.1 Testovanie pomocou skriptu

Testovanie prebiehalo pomocou skriptu test.py, ktorý je implementovaný v jazyku Python. Na testovanie bola využívaná knižnica scapy.py.

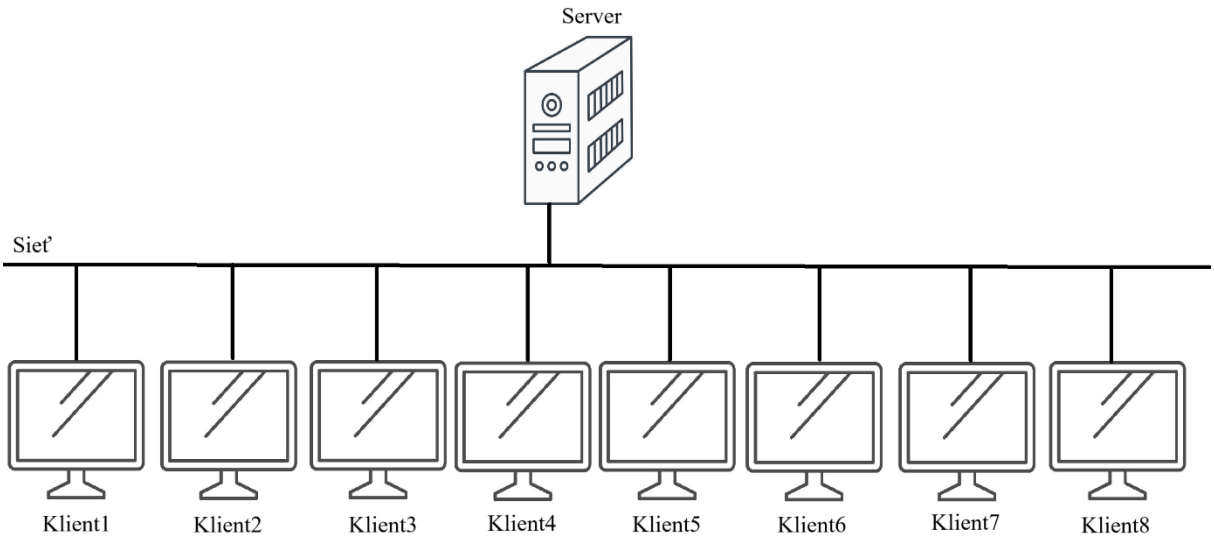
4.2 Testovanie vo virtuálnej sieti

Pre testovanie vo virtuálnej sieti bolo nutné si vytvoriť virtuálne stroje, ktoré predstavovali:

- DHCP klientov – 8 virtuálnych strojov s operačným systémom Lubuntu, ktorý vyžaduje veľmi málo prostriedkov
- DHCP server – 1 virtuálny stroj s operačným systémom Ubuntu 22.04, ktorý vyžaduje okolo 4GB RAM

Pre testovanie veľkých rozsahov by počet klientov bol v normálnom prípade nedostačujúci, ale kvôli veľkosti pamäte na testovacom stroji bolo nutné testovať iba s daným počtom klientov.

Topológia siete pri testovaní:



Obrázok č.2

Testovanie analyzátora dhcp-stats prebiehalo v topológii uvedenej na Obrázku č.2, pričom sa menili nastavenia DHCP servera tak, aby bolo možné otestovať viac prefixov, medzi ktoré patria:

- 10.0.0.0/8
  - 10.0.0.0/16
  - 10.0.0.0/24
  - 10.0.0.0/28
- 192.168.0.0/16
  - 192.168.128.0/17
  - 192.168.128.0/18
  - 192.168.128.0/24
- A mnohé ďalšie

4.3 Testovanie pomocou nástroja tcpreplay

Testovanie prebiehalo pomocou nástroja tcpreplay. Na testovanie boli využívané .pcap a .pcapng súbory zo zložky PCAPs.

Príklad spustenia nástroja tcpreplay:

tcpreplay -i <interface> -t -K --loop <count> <pcap_file>

kde:

- <interface> je meno rozhranie na ktore chceme posielať pakety
- -t špecifikuje to, že chceme počítať časovanie (timing)
- -K špecifikuje nastavenie – zuršenie validácie checksum
- --loop <count> špecifikuje koľko iterácií je nutné vykonať cez .pcap súbor, 0 pre nekonečný počet iterácií
- <pcap_file> je cesta k .pcap súboru, ktorého obsah chceme znova poslať

5 Návod na použitie

Po rozbalení archívu program skompilujete príkazom make a potom spustíte takto:

./dhcp-stats [-r <filename>] [-i <interface-name>] [--ext] <ip-prefix> [<ip-prefix> [...]

V prípade, ak si nie ste istý syntaxou, spustite program s parametrom -h alebo –help, ktorý podrobne popíše syntax spustenia.

Pre úspešný preklad je nutné mať na stroji prekladač g++ podporujúci verziu C++20.

Makefile taktiež podporuje príkazy (spustenie *make <príkaz>):*

- clean – zmaže pracovné súbory
- remake – kratka pre make clean a make
- manpage – zobrazí manuálovú stránku pre dhcp-stats
- pack – zabalí obsah adresára

Bibliografia

[1] Kurose, J. F.; Ross, K. W. (2012). "4.3 The Internet Protocol (IP): IPv4, Addressing, IPv6, and More" In *Computer Networking: A Top-Down Approach Featuring the Internet* (6th edition), str. 388. Addison-Wesley. ISBN 0-321-17644-8.