

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

MONITOROVÁNÍ DNS KOMUNIKACE

ADAM PASTIERIK

Obsah

| | |
|------------------------------------------------------------------------|----|
| 1. Uvedenie do problematiky | 3 |
| 2. Popis implementácie | 3 |
| 2.1 Modul spracovania argumentov príkazového riadku | 3 |
| 2.2 Modul na zachytávanie a spracovanie DNS paketov | 3 |
| 2.2.1 Zachytávanie DNS paketov | 3 |
| 2.2.2 Spracovanie DNS správ | 4 |
| 2.3 Výpis DNS paketov | 4 |
| 2.4 Hľadanie doménových mien a ich preklad na IPv4 a IPv6 adresy | 5 |
| 3. Návod na použitie | 5 |
| 3.1 Príkazové argumenty | 5 |
| 3.2 Spustenie programu | 5 |
| 4. Testovanie | 6 |
| 4.1 Vykonané testy | 7 |
| 4.1.1 Základný test | 7 |
| 4.1.2 Verbose mode | 7 |
| 4.1.3 MX Záznam | 8 |
| 4.1.4 SOA Záznam | 8 |
| 4.1.5 IPv6 Packet | 9 |
| 5. Bibliografia | 10 |

1. Uvedenie do problematiky

Základom sieťovej komunikácie tvoria rôzne protokoly, ktoré zabezpečujú spoľahlivý prenos dát medzi počítačmi a servermi. Jedným z najvýznamnejších protokolov je DNS (Domain Name System). Jeho hlavnou úlohou je preklad doménových mien na IP adresy, ktoré sú potrebné na komunikáciu v sieti.

V tomto projekte sa zameriame na monitorovanie DNS prevádzky pomocou nástroja na analýzu paketov, akým je napríklad `libpcap`, a vytvoríme aplikáciu, ktorá bude zaznamenávať DNS odpovede a ukladať preklady doménových mien na IP adresy do súboru.

2. Popis implementácie

Implementácia aplikácie je rozdelená do viacerých modulov, pričom každý z nich má na starosti určitú časť funkčnosti. Tieto moduly spoločne zabezpečujú zachytávanie DNS komunikácie, spracovanie paketov a výpis zistených informácií. Implementácia využíva knižnicu `libpcap` na zachytávanie sieťovej komunikácie.

2.1 Modul spracovania argumentov príkazového riadku

Tento modul je implementovaný v triede `ArgParser`, ktorá slúži na spracovanie a validáciu argumentov príkazového riadku. Používa funkciu `getopt` na jednoduché parsovanie argumentov, pričom podporuje nasledovné parametre:

- **-i:** určuje sieťové rozhranie, z ktorého sa budú zachytávať DNS pakety.
- **-p:** určuje pcap súbor, ktorý sa bude analyzovať, ak sa nepoužíva sieťové rozhranie.
- **-v:** nastaví mód s podrobným výstupom (verbose mode).
- **-d:** určuje cestu k súboru, kde sa budú ukladať preložené domény.
- **-t:** určuje cestu k súboru s prekladmi domén na IP adresy.

Ak používateľ nešpecifikuje rozhranie ani pcap súbor, aplikácia vypíše chybu a skončí. Tento modul tiež zabezpečuje, že nie je možné súčasne špecifikovať rozhranie aj pcap súbor.

2.2 Modul na zachytávanie a spracovanie DNS paketov

Hlavný modul aplikácie je implementovaný v triede `DnsMonitor`, ktorá spracováva zachytené DNS pakety. Tento modul vykonáva nasledujúce funkcie:

2.2.1 Zachytávanie DNS paketov

V metóde `process_packets` používa knižnicu `libpcap` na zachytávanie paketov zo zvoleného sieťového rozhrania alebo načítava pakety z pcap súboru. Aplikácia používa filter, aby zachytávala iba DNS pakety. Z knižnice využíva funkciu `pcap_loop` kde v smyčke prechádza jednotlivými paketmi a spracováva ich.

2.2.2 Spracovanie DNS správ

Spracovanie DNS správ v aplikácii začína analýzou zachytených paketov na úrovni Ethernetovej hlavičky, kde sa v metóde `get_ip_version` zistí, či sa jedná o IPv4 alebo IPv6 paket. Pred samotným rozlíšením IP verzie sa však vykonáva kontrola, či ide o tzv. "Linux cooked" packet, čo indikuje špecifický formát zachytených paketov v Linuxe.

Toto zisťovanie prebieha prostredníctvom metódy `pcap_dataLink`, ktorá vracia typ linkovej vrstvy zachyteného paketu. Ak je výsledkom tejto metódy hodnota `DLT_LINUX_SLL`, znamená to, že paket je Linux cooked (s formátom hlavičky sll), a teda hlavička je dlhá 16 bajtov. V takomto prípade sa typ vyššej vrstvy (IP alebo IPv6) určuje z poľa `sll_proto` v hlavičke. Naopak, ak je typom linkovej vrstvy `DLT_EN10MB`, paket používa štandardnú Ethernetovú hlavičku s dĺžkou 14 bajtov, pričom typ vyššej vrstvy sa získa z poľa `ether_type` tejto hlavičky.

Po určení IP verzie sa z IP hlavičky zistia zdrojové a cieľové IP adresy a nasleduje spracovanie UDP hlavičky. UDP hlavička obsahuje zdrojový a cieľový port, ktoré sa analyzujú. Za UDP hlavičkou sa nachádza DNS paket, ktorý obsahuje informácie o DNS dotazoch a odpovediach.

2.3 Výpis DNS paketov

Výpis paketov je realizovaný pomocou samostatnej metódy `print_dns_packet`. V prípade štandardného režimu (bez verbose) aplikácia vypisuje len základné informácie o DNS pakete, ako je timestamp, zdrojová a cieľová IP adresa, a počet záznamov v jednotlivých sekciách DNS správy. Ak je povolený verbose režim, aplikácia vypíše podrobné informácie o DNS pakete. Zobrazuje sa čas prijatia paketu, zdrojová a cieľová IP adresa, zdrojový a cieľový port, ID DNS správy, príznaky DNS správy a záznamy sekcie Question, prípadne Answer, Authority. Additional. Podporované typy záznamov sú A, AAAA, NS, MX, SOA, CNAME a SRV. Ak je v sekcii aspoň jeden záznam (aj keď je ignorovaný), vypíše sa názov sekcie. Naopak, ak je sekcia úplne prázdna, jej názov sa nevypíše.

Metóda `print_dns_question` začína inicializáciou ukazovateľa `currentPtr`, ktorý ukazuje na aktuálnu pozíciu v DNS pakete (na začiatok sekcie Question). Vytvárajú sa dve mapy (`unordered_map`), ktoré mapujú číselné hodnoty QTYPE (typ dotazu) a QCLASS (trieda dotazu) na ich reťazcové reprezentácie. Cyklus prechádza cez všetky otázky v sekcii podľa hodnoty `qdCount`. Pre každú otázku vytvára novú inštanciu triedy `Section`, ktorá spracuje doménové meno, typ a triedu dotazu. Ukazovateľ `currentPtr` sa posunie na nasledujúcu časť DNS paketu po spracovaní aktuálnej otázky.

Pre výpis ďalších sekcií sa využíva metóda `print_section`, ktorá prechádza každým záznamom sekcie a vytvára inštanciu triedy `Section`, ktorá obsahuje doménové meno, typ záznamu, triedu a TTL. Tieto údaje sa potom odovzdajú funkcii `print_record`, ktorá spracuje záznam na základe jeho typu. Metóda vracia ukazovateľ na ďalší záznam sekcie alebo ďalšiu sekciu.

2.4 Hľadanie doménových mien a ich preklad na IPv4 a IPv6 adresy

Trieda `DnsMonitor` obsahuje atribúty typu `List domainNames` a `translations` na ukladanie doménových mien a ich prekladov. Pri každom vytvorení inštancie triedy `Section` sa zavolá metóda `add_to_domain_list`, ktorá skontroluje, či sa dané doménové meno už vyskytuje v liste, a ak nie, pridá ho tam.

Na podobnom princípe funguje aj metóda `add_to_translations`, ktorá sa však volá len v prípade A a AAAA záznamov a pridáva preklad do listu.

Ak užívateľ špecifikoval pri spustení parametre `-d` alebo `-t`, tak sa do týchto súborov potom vypíšu doménové mená a ich preklady.

3. Návod na použitie

3.1 Príkazové argumenty

Aplikácia akceptuje nasledujúce príkazové argumenty:

- `-i <interface>`: názov rozhrania, na ktorom bude program počúvať
- `-p <pcapfile >`: cesta k PCAP súboru na spracovanie predtým zachytených paketov.
- `-v`: režim „verbose“ – povolí kompletný výpis detailov o DNS pakete
- `-d <domainsfile>`: súbor na uloženie doménových mien.
- `-t <translationsfile>`: súbor na uloženie DNS prekladov na IP adresy.

3.2 Spustenie programu

Pred spustením programu je nutné ho najskôr skompilovať pomocou príkazu `make`, ktorý vytvorí spustiteľný súbor zo zdrojových kódov. Potom je možné program spustiť nasledovne:

- Zachytávanie paketov na sieťovom rozhraní

```
• ./dns_monitor -i eth0
```

- Spracovanie paketov z pcap súboru

```
• ./dns_monitor -p pcapfile.pcap
```

- Spustenie v režime „verbose“

```
• ./dns_monitor -p pcapfile.pcap -v
```

- Uloženie doménových mien do súboru

```
• ./dns_monitor -p pcapfile.pcap -d domains.txt
```

4. Testovanie

Na overenie správnosti funkčnosti aplikácie som vykonal testovanie pomocou nástroja Wireshark. Testoval som počúvanie na rozhraní ale aj použitím pcap súborov. Na základe výstupov z aplikácie som skontroloval, či boli správne extrahované doménové mená, preklady na IP adresy a ďalšie údaje porovnaním výsledkov z aplikácie Wireshark.

4.1 Vykonané testy

4.1.1 Základný test

Výstup programu

```
admp@DESKTOP-JM83620:~/Programming/isa_project$ ./dns_monitor -p packets/basic.pcap
2024-10-13 16:35:27 192.168.88.1 -> 192.168.88.154 (R 1/1/0/0)
```

Wireshark

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|----------|--------------|----------------|----------|--------|------------------------------------------------------------|
| 1 | 0.000000 | 192.168.88.1 | 192.168.88.154 | DNS | 86 | Standard query response 0xa4f3 A www.vut.cz A 147.229.2.90 |

| | |
|-------------------------------------------------------------------------------------------|---|
| < | > |
| > Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface eth0 | |
| > Ethernet II, Src: Routerboardc_b4:5b:c1 (c4:ad:34:b4:5b:c1), Dst: LCFC00:00:00:00:00:00 | |
| > Internet Protocol Version 4, Src: 192.168.88.1, Dst: 192.168.88.154 | |
| > User Datagram Protocol, Src Port: 53, Dst Port: 59231 | |
| ▼ Domain Name System (response) | |
| Transaction ID: 0xa4f3 | |
| > Flags: 0x8180 Standard query response, No error | |
| Questions: 1 | |
| Answer RRs: 1 | |
| Authority RRs: 0 | |
| Additional RRs: 0 | |

| | | | |
|------|-------------------------------------------------|------|-----|
| 0000 | 88 a4 c2 69 34 e3 c4 ad 34 b4 5b c1 08 00 45 00 | ... | i |
| 0010 | 00 48 44 0d 00 00 40 11 04 ac c0 a8 58 01 c0 a8 | ... | HD |
| 0020 | 58 9a 00 35 e7 5f 00 34 f7 d1 a4 f3 81 80 00 01 | X... | 5 |
| 0030 | 00 01 00 00 00 00 03 77 77 77 03 76 75 74 02 63 | ... | ... |
| 0040 | 7a 00 00 01 00 01 c0 0c 00 01 00 01 00 00 01 2c | z... | ... |
| 0050 | 00 04 93 e5 02 5a | ... | ... |

4.1.2 Verbose mode

Výstup programu

```
admp@DESKTOP-JM83620:~/Programming/isa_project$ ./dns_monitor -p packets/basic.pcap -v
Timestamp: 2024-10-13 16:35:27
SrcIP: 192.168.88.1
DstIP: 192.168.88.154
SrcPort: UDP/53
DstPort: UDP/59231
Identifier: 0xa4f3
Flags: QR=1, OPCODE=0, AA=0, TC=0, RD=1, RA=1, Z=0, RCODE=0

[Question Section]
www.vut.cz. IN A

[Answer Section]
www.vut.cz. 300 IN A 147.229.2.90
=====
```

Wireshark

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|----------|--------------|----------------|----------|--------|------------------------------------------------------------|
| 1 | 0.000000 | 192.168.88.1 | 192.168.88.154 | DNS | 86 | Standard query response 0xa4f3 A www.vut.cz A 147.229.2.90 |

| | |
|-------------------------------------------------------------------------------------------|---|
| < | > |
| > Ethernet II, Src: Routerboardc_b4:5b:c1 (c4:ad:34:b4:5b:c1), Dst: LCFC00:00:00:00:00:00 | |
| > Internet Protocol Version 4, Src: 192.168.88.1, Dst: 192.168.88.154 | |
| > User Datagram Protocol, Src Port: 53, Dst Port: 59231 | |
| ▼ Domain Name System (response) | |
| Transaction ID: 0xa4f3 | |
| > Flags: 0x8180 Standard query response, No error | |
| Questions: 1 | |
| Answer RRs: 1 | |
| Authority RRs: 0 | |
| Additional RRs: 0 | |
| ▼ Queries | |
| > www.vut.cz: type A, class IN | |
| ▼ Answers | |
| ▼ www.vut.cz: type A, class IN, addr 147.229.2.90 | |
| Name: www.vut.cz | |
| Type: A (1) (Host Address) | |
| Class: IN (0x0001) | |
| Time to live: 300 (5 minutes) | |
| Data length: 4 | |
| Address: 147.229.2.90 | |

| | | | |
|------|-------------------------------------------------|------|-----|
| 0000 | 88 a4 c2 69 34 e3 c4 ad 34 b4 5b c1 08 00 45 00 | ... | i |
| 0010 | 00 48 44 0d 00 00 40 11 04 ac c0 a8 58 01 c0 a8 | ... | HD |
| 0020 | 58 9a 00 35 e7 5f 00 34 f7 d1 a4 f3 81 80 00 01 | X... | 5 |
| 0030 | 00 01 00 00 00 00 03 77 77 77 03 76 75 74 02 63 | ... | ... |
| 0040 | 7a 00 00 01 00 01 c0 0c 00 01 00 01 00 00 01 2c | z... | ... |
| 0050 | 00 04 93 e5 02 5a | ... | ... |

4.1.3 MX Záznam

Výstup program

```
admp@DESKTOP-JM83620:~/Programming/isa_project$ ./dns_monitor -p packets/mx.pcap -v
Timestamp: 2024-10-04 14:52:49
SrcIP: 192.168.88.1
DstIP: 192.168.88.156
SrcPort: UDP/53
DstPort: UDP/49248
Identifier: 0x766c
Flags: QR=1, OPCODE=0, AA=0, TC=0, RD=1, RA=1, Z=0, RCODE=0

[Question Section]
example.com. IN MX

[Answer Section]
example.com. 86400 IN MX 0 <Root>
=====
```

Wireshark

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|----------|--------------|----------------|----------|--------|-----------------------------------------------------------|
| 1 | 0.000000 | 192.168.88.1 | 192.168.88.156 | DNS | 86 | Standard query response 0x766c MX example.com MX 0 <Root> |

| | | | | | | | | | | | | | | | | | |
|--------------------------------------------------------------------------|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|---|
| < | | | | | | | | | | | | | | | | | > |
| > Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) | | | | | | | | | | | | | | | | | |
| > Ethernet II, Src: Routerboardc_b4:5b:c1 (c4:ad:34:b4:5b:c1), Dst: LCFC | | | | | | | | | | | | | | | | | |
| > Internet Protocol Version 4, Src: 192.168.88.1, Dst: 192.168.88.156 | | | | | | | | | | | | | | | | | |
| > User Datagram Protocol, Src Port: 53, Dst Port: 49248 | | | | | | | | | | | | | | | | | |
| ▼ Domain Name System (response) | | | | | | | | | | | | | | | | | |
| Transaction ID: 0x766c | | | | | | | | | | | | | | | | | |
| > Flags: 0x8180 Standard query response, No error | | | | | | | | | | | | | | | | | |
| Questions: 1 | | | | | | | | | | | | | | | | | |
| Answer RRs: 1 | | | | | | | | | | | | | | | | | |
| Authority RRs: 0 | | | | | | | | | | | | | | | | | |
| Additional RRs: 0 | | | | | | | | | | | | | | | | | |
| ▼ Queries | | | | | | | | | | | | | | | | | |
| > example.com: type MX, class IN | | | | | | | | | | | | | | | | | |
| ▼ Answers | | | | | | | | | | | | | | | | | |
| > example.com: type MX, class IN, preference 0, mx <Root> | | | | | | | | | | | | | | | | | |

| | | | | | | | | | | | | | | | | | |
|------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|
| 0000 | 88 | a4 | c2 | 69 | 34 | e3 | c4 | ad | 34 | b4 | 5b | c1 | 08 | 00 | 45 | 00 | ... |
| 0010 | 00 | 48 | 1b | 3a | 00 | 00 | 40 | 11 | 2d | 7d | c0 | a8 | 58 | 01 | c0 | a8 | ... |
| 0020 | 58 | 9c | 00 | 35 | c0 | 60 | 00 | 34 | 97 | 94 | 76 | 6c | 81 | 80 | 00 | 01 | ... |
| 0030 | 00 | 01 | 00 | 00 | 00 | 00 | 07 | 65 | 78 | 61 | 6d | 70 | 6c | 65 | 03 | 63 | ... |
| 0040 | 6f | 6d | 00 | 00 | 0f | 00 | 01 | c0 | 0c | 00 | 0f | 00 | 01 | 00 | 01 | 51 | ... |
| 0050 | 80 | 00 | 03 | 00 | 00 | 00 | | | | | | | | | | | ... |

4.1.4 SOA Záznam

Výstup program

```
admp@DESKTOP-JM83620:~/Programming/isa_project$ ./dns_monitor -p packets/soa.pcap -v
Timestamp: 2024-10-04 14:55:46
SrcIP: 192.168.88.1
DstIP: 192.168.88.156
SrcPort: UDP/53
DstPort: UDP/49670
Identifier: 0x5a46
Flags: QR=1, OPCODE=0, AA=0, TC=0, RD=1, RA=1, Z=0, RCODE=0

[Question Section]
example.com. IN SOA

[Answer Section]
example.com. 3600 IN SOA ns.icann.org. noc.dns.icann.org. (
    2024081432 ; Serial
    7200 ; Refresh
    3600 ; Retry
    1209600 ; Expire
    3600 ; Minimum TTL
)
=====
```


Wireshark

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|----------|--------------|----------------|----------|--------|-----------------------------------------------------------------|
| 1 | 0.000000 | 192.168.88.1 | 192.168.88.156 | DNS | 127 | Standard query response 0x5a46 SOA example.com SOA ns.icann.org |

| | |
|-------------------------------------------------------|-------------------------------------------------------------|
| < | > |
| > Flags: 0x8180 Standard query response, No error | 0000 88 a4 c2 69 34 e3 c4 ad 34 b4 5b c1 08 00 45 00 ...i |
| Questions: 1 | 0010 00 71 1b 51 00 00 40 11 2d 3d c0 a8 58 01 c0 a8 ...q·Q |
| Answer RRs: 1 | 0020 58 9c 00 35 c2 06 00 5d 6a ae 5a 46 81 80 00 01 X...5 |
| Authority RRs: 0 | 0030 00 01 00 00 00 00 07 65 78 61 6d 70 6c 65 03 63 |
| Additional RRs: 0 | 0040 6f 6d 00 00 06 00 01 c0 0c 00 06 00 01 00 00 0e om.. |
| Queries | 0050 10 00 2c 02 6e 73 05 69 63 61 6e 6e 03 6f 72 67 ..,i |
| Answers | 0060 00 03 6e 6f 63 03 64 6e 73 c0 2c 78 a5 08 18 00 ...nor |
| ▼ example.com: type SOA, class IN, mname ns.icann.org | 0070 00 1c 20 00 00 0e 10 00 12 75 00 00 00 0e 10 ... |
| Name: example.com | |
| Type: SOA (6) (Start Of a zone of Authority) | |
| Class: IN (0x0001) | |
| Time to live: 3600 (1 hour) | |
| Data length: 44 | |
| Primary name server: ns.icann.org | |
| Responsible authority's mailbox: noc.dns.icann.org | |
| Serial Number: 2024081432 | |
| Refresh Interval: 7200 (2 hours) | |
| Retry Interval: 3600 (1 hour) | |
| Expire limit: 1209600 (14 days) | |
| Minimum TTL: 3600 (1 hour) | |
| [Unsolicited: True] | |

4.1.5 IPv6 Packet

Výstup programu

```
admp@DESKTOP-JM83620:~/Programming/isa_project$ ./dns_monitor -p packets/ipv6.pcap -v
Timestamp: 1999-03-11 14:45:02
SrcIP: 3ffe:501:4819::42
DstIP: 3ffe:507:0:1:200:86ff:fe05:80da
SrcPort: UDP/53
DstPort: UDP/2396
Identifier: 0x6
Flags: QR=1, OPCODE=0, AA=1, TC=0, RD=1, RA=1, Z=0, RCODE=0

[Question Section]

[Answer Section]
itojun.org. 3600 IN NS coconut.itojun.org.
itojun.org. 3600 IN NS tiger.hiroo.oshokuji.org.
itojun.org. 3600 IN MX 10 coconut.itojun.org.
itojun.org. 3600 IN MX 20 kiwi.itojun.org.
itojun.org. 3600 IN A 210.160.95.97
itojun.org. 3600 IN SOA itojun.org. root.itojun.org. (
    199903080 ; Serial
    3600 ; Refresh
    300 ; Retry
    3600000 ; Expire
    3600 ; Minimum TTL
)

[Authority Section]
itojun.org. 3600 IN NS coconut.itojun.org.
itojun.org. 3600 IN NS tiger.hiroo.oshokuji.org.

[Additional Section]
coconut.itojun.org. 3600 IN A 210.160.95.97
tiger.hiroo.oshokuji.org. 3600 IN A 210.145.33.242
kiwi.itojun.org. 3600 IN AAAA 3ffe:501:410:0:2c0:dfff:fe47:33e
kiwi.itojun.org. 3600 IN AAAA 3ffe:501:410:100:5254:ff:feda:48bf
kiwi.itojun.org. 3600 IN A 210.160.95.99
=====
```

Wireshark

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|----------|-------------------|------------------------|----------|--------|-----------------------------------------------------------|
| 1 | 0.000000 | 3ffe:501:4819::42 | 3ffe:507:0:1:200:86... | DNS | 510 | Standard query response 0x0006 ANY itojun.org NS coconut. |

<

>

> Ethernet II, Src: 3Com_07:69:ea (00:60:97:07:69:ea), Dst: Megahertz_05:80:da (00:00:86:00:00:01) [0.000000]

0000 00 00 86 05 80 da 00 60 97 07 69 ea

> Internet Protocol Version 6, Src: 3ffe:501:4819::42, Dst: 3ffe:507:0:1:200:86ff:fe05:80 [0.000000]

0010 00 00 01 c8 11 e6 3f fe 05 01 48 19

> User Datagram Protocol, Src Port: 53, Dst Port: 2396 [0.000000]

0020 00 00 00 00 00 42 3f fe 05 07 00 00

> Domain Name System (response) [0.000000]

0030 86 ff fe 05 80 da 00 35 09 5c 01 c8

Transaction ID: 0x0006 [0.000000]

0040 85 80 00 01 00 06 00 02 00 05 06 69

> Flags: 0x8580 Standard query response, No error [0.000000]

0050 6e 03 6f 72 67 00 00 ff 00 01 c0 0c

Questions: 1 [0.000000]

0060 00 00 0e 10 00 14 07 63 6f 63 6f 6e

Answer RRs: 6 [0.000000]

0070 74 6f 6a 75 6e 03 6f 72 67 00 c0 0c

Authority RRs: 2 [0.000000]

0080 00 00 0e 10 00 1a 05 74 69 67 65 72

Additional RRs: 5 [0.000000]

0090 6f 6f 08 6f 73 68 6f 6b 75 6a 69 03

> Queries [0.000000]

00a0 c0 0c 00 0f 00 01 00 00 0e 10 00 16

> itojun.org: type ANY, class IN [0.000000]

00b0 6f 63 6f 6e 75 74 06 69 74 6f 6a 75

> Answers [0.000000]

00c0 67 00 c0 0c 00 0f 00 01 00 00 0e 10

> itojun.org: type NS, class IN, ns coconut.itojun.org [0.000000]

00d0 04 6b 69 77 69 06 69 74 6f 6a 75 6e

> itojun.org: type NS, class IN, ns tiger.hiroo.oshokuji.org [0.000000]

00e0 00 c0 0c 00 01 00 01 00 00 0e 10 00

> itojun.org: type MX, class IN, preference 10, mx coconut.itojun.org [0.000000]

00f0 61 c0 0c 00 06 00 01 00 00 0e 10 00

> itojun.org: type MX, class IN, preference 20, mx kiwi.itojun.org [0.000000]

0100 6f 6a 75 6e 03 6f 72 67 00 04 72 6f

> itojun.org: type A, class IN, addr 210.160.95.97 [0.000000]

0110 74 6f 6a 75 6e 03 6f 72 67 00 0b ea

> itojun.org: type SOA, class IN, mname itojun.org [0.000000]

0120 0e 10 00 00 01 2c 00 36 ee 80 00 00

> Authoritative nameservers [0.000000]

0130 00 02 00 01 00 00 0e 10 00 14 07 63

> itojun.org: type NS, class IN, ns coconut.itojun.org [0.000000]

0140 75 74 06 69 74 6f 6a 75 6e 03 6f 72

> itojun.org: type NS, class IN, ns tiger.hiroo.oshokuji.org [0.000000]

0150 00 02 00 01 00 00 0e 10 00 1a 05 74

> Additional records [0.000000]

0160 05 68 69 72 6f 6f 08 6f 73 68 6f 6b

> coconut.itojun.org: type A, class IN, addr 210.160.95.97 [0.000000]

0170 6f 72 67 00 07 63 6f 63 6f 6e 75 74

> tiger.hiroo.oshokuji.org: type A, class IN, addr 210.145.33.242 [0.000000]

0180 00 01 00 00 0e 10 00 04 d2 a0 5f 61

> kiwi.itojun.org: type AAAA, class IN, addr 3ffe:501:410:0:2c0:dfff:fe47:33e [0.000000]

0190 65 72 05 68 69 72 6f 6f 08 6f 73 68

> kiwi.itojun.org: type AAAA, class IN, addr 3ffe:501:410:100:5254:ff:feda:48bf [0.000000]

01a0 69 c0 13 00 01 00 01 00 00 0e 10 00

> kiwi.itojun.org: type A, class IN, addr 210.160.95.99 [0.000000]

01b0 f2 04 6b 69 77 69 c0 0c 00 1c 00 01

[Unsolicited: True] [0.000000]

01c0 00 10 3f fe 05 01 04 10 00 00 02 c0

[0.000000]

01d0 03 3e c1 73 00 1c 00 01 00 00 0e 10

[0.000000]

01e0 05 01 04 10 01 00 52 54 00 ff fe da

[0.000000]

01f0 00 01 00 01 00 00 0e 10 00 04 d2 a0

5. Bibliografia

- <https://datatracker.ietf.org/doc/html/rfc1035>
- <https://datatracker.ietf.org/doc/html/rfc2065>
- <https://datatracker.ietf.org/doc/html/rfc3596>
- <https://www.tcpdump.org/pcap.html>
- <https://wiki.wireshark.org/SLL>
- <https://stackoverflow.com/questions/51358018/linux-cooked-capture-in-packets>