

# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

## FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

ISA - Síťové aplikace a správa sítí

dokumentácia k 2.variante projektu

dns-export

# Obsah

<b>1</b>	<b>Úvod</b>	<b>1</b>
<b>2</b>	<b>Návrh aplikácie</b>	<b>1</b>
2.1	Objektový návrh . . . . .	1
2.2	Spracovanie paketov . . . . .	1
2.3	Ukladanie štatistík . . . . .	1
2.4	Odosielanie a výpis štatistík . . . . .	1
<b>3</b>	<b>Implementácia</b>	<b>2</b>
3.1	DnsExport.cpp . . . . .	2
3.2	DnsPacket.cpp . . . . .	2
3.3	DnsRR.cpp . . . . .	3
3.4	Helpers.cpp . . . . .	3
<b>4</b>	<b>Zaujímavé časti kódu</b>	<b>4</b>
4.1	LINUX_SLL datalink . . . . .	4
4.2	Odosielanie na Syslog . . . . .	5
4.3	Pole "hostname" v Syslog správe . . . . .	5
4.4	Získanie aktuálneho timestampu . . . . .	6
4.5	Vytvorenie správ na odoslanie . . . . .	7
<b>5</b>	<b>Použitie</b>	<b>8</b>
5.1	Čítanie pcap súboru . . . . .	8
5.2	Odchytávanie na rozhraní . . . . .	8
5.3	Zaslanie signálu SIGUSR1 . . . . .	8
<b>6</b>	<b>Prehľad naštudovanej literatúry</b>	<b>9</b>
<b>7</b>	<b>Literatúra</b>	<b>9</b>

# 1 Úvod

**dns-export** je aplikácia na spracovávanie dát protokolu *DNS* (*Domain Name System*). Aplikácia teda musí vedieť identifikovať a spracovať *paket* daného protokolu. Jednotlivé pakety môže buď čerpať z *pcap* súboru alebo odchytať v reálnom čase zo sieťového rozhrania.

Zo spracovaných dát vytvára štatistiky, ktoré v časovom intervale posiela protokolom *Syslog* na centrálny logovací server alebo jednoducho vypisuje na štandardný výstup. Zbierané štatistiky sú: `domain-name` `rr-type` `rr-answer` a `count`.

## 2 Návrh aplikácie

### 2.1 Objektový návrh

Svoju aplikáciu som sa rozhodol naprogramovať objektovo. Dodržiaval som štruktúru jedna *trieda* na jeden zdrojový súbor. Každá trieda má svoju významom vyhradenú funkciu.

### 2.2 Spracovanie paketov

Aplikácia spracuje vstupné argumenty a po ich validácii sa buď spustí spracovanie *pcap* súboru, alebo odchytyvanie paketov zo sieťového rozhrania. Pre každý spracovávaný paket sa vytvorí objekt držiaci dáta paketu.

V tomto objekte sa najprv spracujú jednotlivé hlavičky paketu. Ak sa zistí, že paket obsahuje odpovede na DNS otázky, Vytvorí sa pre každú odpoveď objekt, ktorý tieto odpovede spracováva.

### 2.3 Ukladanie štatistík

Po úspešnom spracovaní odpovedí sa každá odpoveď pridá do objektu, ktorý obsahuje kolekciu všetkých nazbieraných štatistík. Pomocou metódy sa zistí, či sa daná štatistická informácia v objekte už nachádza a v tom prípade sa len inkrementuje počítadlo.

### 2.4 Odosielanie a výpis štatistík

Po prečítaní z *pcap* súboru sa štatistiky uložené v objekte odošlú na server ak je zadán alebo vypíšu na štandardný výstup. V prípade živého odchytyvania sa pred jeho spustením vytvorí separátne vlákno, ktoré bude v zadanom časovom intervale odosielať všetky doposiaľ nazbierané štatistiky. Okrem toho bude aplikácia pri odchytení signálu **SIGUSR1** volať funkciu na výpis štatistík na štandardný výstup.

## 3 Implementácia

### 3.1 DnsExport.cpp

Po spustení aplikácie je vytvorený objekt tejto triedy a zavolaná funkcia **Main()**. Nastaví sa odchyťovanie signálov a na základne vstupných argumentov načítaných pomocou vstavanej funkcie **getopt()**<sup>1</sup> sa rozhodne, či sa spustí funkcia **sniffInterface()** alebo **sniffFile()**.

V prvom prípade sa pred započatím odchyťovania spustí funkcia **sendingLoop()**, ktorá sa v separátnom vlákne stará o pravidelné odosielanie na Syslog server. Pre odoslanie volá funkcia **sendStats()**, ktorá si volá pomocné funkcie na vytvorenie správ ako napr.: **getMessages()** alebo **getFormattedTime()** a následne **UDP** protokolom odosiela správy. Správa nie je dlhšia ako 1 kB a jednotlivé štatistiky sú oddelené znakom |.

V druhom prípade ak nebol zadaný Syslog server, je po prečítaní súboru zavolaná funkcia **getMessages()**, ktorá štatistiky vypíše na štandardný výstup a aplikácia končí. Rovnaká funkcia je volaná aj pri prijatí signálu **SIGUSR1**.

O všetky akcie ohľadom odchyťovania paketov sa stará knižnica **libpcap**<sup>2</sup>. Pre každý paket je volaná *callback* funkcia **pcapHandler()**. Tam je vytvorený objekt triedy **DnsPacket** a zavolaná funkcia **Parse()**.

Ak boli v pakete nájdené validné DNS odpovede - *answers*, je zavolaná funkcia **addRecords()**, ktorá do globálnej premennej **recordList** pridá nový objekt triedy **DnsRecord**, ak sa daná štatistika v liste ešte nenachádza, alebo len zvýši počítadlo u nájdenej štatistiky.

### 3.2 DnsPacket.cpp

Vo funkcii **Parse()** je najprv zavolaná funkcia **getTransProt()**. Tá parsuje *ethernet* hlavičku (podporované typy sú IPv4 a IPv6) a následne zistí transportný protokol (podporované typy sú UDP a TCP) a skočí na pozíciu, kde by sa mala nachádzať DNS hlavička.

Parsovanie DNS hlavičky začína vo funkcii **dnsParse()**. Tam prebehnú validácie, či ide skutočne o DNS paket. *query* pakety sú ignorované, aplikácia pracuje len s *answers*. Ak sú nájdené *answers*, je zavolaná funkcia **parseRRSet()**, kde je hneď na začiatku zavolaná funkcia **skipQuestion()**, ktorá skočí na pozíciu prvej *answer*.

Pre každú *answer* je vytvorený objekt triedy **DnsRR**, ktorý spracuje dáta a nastaví pozíciu na ďalšiu *answer*. Ak boli získané nejaké dáta, objekt je uložený do listu **Answers**.

---

<sup>1</sup><http://man7.org/linux/man-pages/man3/getopt.3.html>

<sup>2</sup><https://www.tcpdump.org/pcap.html>

### 3.3 DnsRR.cpp

Každá answer začína *domain-name*, ktorý je získaný funkciou `readDomainName()`. Ak sa podarí získať *domain-name*, nasleduje získanie *rr-type* a dĺžky dát. Vo funkcii `applyParser()` je `switch`, v ktorom sa na základe typu záznamu rozhodne ktorý z parserov<sup>3</sup> pre dáta použiť. Okrem toho sa typ uloží ako *string* hodnota pre jednoduchšie ukladanie štatistík. Ak sa podarilo získať dáta, je vrátená pozícia za odpoveďou a aplikácia pokračuje parsovaním ďalšej answer.

### 3.4 Helpers.cpp

Táto trieda obsahuje makrá pre logovanie **LOGGING** a ukončovanie aplikácie ak nastane problém **ERR\_RET**.

Funkcia `ToHex()` prevedie netlačiteľný znak do hexadecimálneho tvaru, napr.: 0x01. Je používaná ak sa napríklad v *domain-name* alebo *TXT* zázname objaví netlačiteľný znak.

Funkcia `Base64Encode()` je využívaná parseroch pre záznamy typu *DNSKEY*, *DS* a *RRSIG*. Kóduje bajtové pole ako base64 string, ktorý sa zobrazí v štatistikách. Funkcia je dostupná na githube<sup>4</sup> pod *Zlib* licenciou.

---

<sup>3</sup>Konkrétnejší popis jednotlivých parserov je v komentároch v zdrojovom kóde. Veľmi špecifický je záznam typu *SPF*. V súčasnosti sa posielajú iba záznamy typu *TXT* ktoré sa dajú na základe určitých pravidiel označiť ako *SPF*. Toto aplikácia nezisťuje a každý *TXT* záznam označí typom *TXT*. Keby náhodou prišiel záznam typu *SPF*, je parsovaný rovnako ako klasický *TXT* záznam.

<sup>4</sup><https://github.com/ReneNyffenegger/cpp-base64/blob/master/base64.cpp>

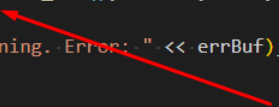
## 4 Zaujímavé časti kódu

### 4.1 LINUX\_SLL datalink

Pri odchyťávaní na všetkých rozhraniach (interface=any), nemajú pakety ethernet hlavičku ale tzv. "linux-cooked header". To sa dá zistiť pomocou funkcie `pcap_datalink()`. Ak ide o `LINUX_SLL_DATALINK`, hlavička je parsovaná iným spôsobom ako ethernet:

```
// Open device for sniffing. TO_MS = 100 ms is value used by Wireshark
if ((myPcap = pcap_open_live(parameters.interface.c_str(), BUFSIZ, true, TO_MS, this->errBuf)) == NULL)
{
    ... ERR_RET("Unable to open interface for listening. Error: " << errBuf);
}

datalink = pcap_datalink(myPcap);
```



môže byť "any"

```
if (this->datalink == LINUX_SLL_DATALINK)
{
    ... ethType = ntohs(*(uint16_t*)(packet + 14));
    ... this->position += 16;
}
else if (this->datalink == ETHERNET_DATALINK)
{
    ... // Read the ethernet header
    ... eptr = (struct ether_header*)packet;
    ... ethType = ntohs(eptr->ether_type);
    ... this->position += sizeof(struct ether_header);
}
else
{
    ... return false; ... // Unsupported
}

// Check ethernet type
switch (ethType)
```

## 4.2 Odosielanie na Syslog

Pri spustení vlákna na odosielanie sa najprv vyčká 20 ms a až potom sa spustí cyklus na odosielanie. Pretože aplikácia využíva parameter `to_ms` vo funkcii `pcap_open_live()`, chceme aby sa stihli vytvoriť všetky štatistiky pred odoslaním. Kód je inšpirovaný vláknom na stránke [cplusplus.com](http://www.cplusplus.com/forum/beginner/91449/)<sup>5</sup>:

```
// Thread sending stats to syslog server
static void* sendingLoop(void* time)
{
    ....// Initial sleep -- waiting for pcap to start and parse something
    ....chrono::milliseconds openLiveT0(20); // By given parameter
    ....this_thread::sleep_for(openLiveT0);

    ....// http://www.cplusplus.com/forum/beginner/91449/
    ....chrono::seconds interval*((int*)time); // By given parameter
    ....while (true)
    ....{
    ....    this_thread::sleep_for(interval);
    ....    sendStats();
    ....}
}
```

## 4.3 Pole "hostname" v Syslog správe

RFC 5424<sup>6</sup> špecifikuje prioritu. Aplikácia nepredpokladá existenciu FQDN ani statickú IP adresu, preto sa najprv pokúsi získať hostname a až v prípade neúspechu ip adresu:

```
// Get hostname
string hostname;
char tmp [1024] = "";
if (!gethostname(tmp, sizeof tmp))
{
    ....hostname = tmp;
}
else
{
    ....// Return value != 0 signalize error. Try to get ip address
    ....struct sockaddr_in ipAddress;
    ....socklen_t length = sizeof(ipAddress);

    ....if (!getsockname(sock, (struct sockaddr*)&ipAddress, &length))
    ....{
    ....    hostname = ipAddress.sin_addr.s_addr;
    ....}
}
```

<sup>5</sup><http://www.cplusplus.com/forum/beginner/91449/>

<sup>6</sup><https://tools.ietf.org/html/rfc5424>

## 4.4 Získanie aktuálneho timestampu

Syslog správa potrebuje timestamp v špecifickom tvare. Kód je inšpirovaný vláknom na stránke [codereview.stackexchange.com](https://codereview.stackexchange.com)<sup>7</sup>.

```
// Returns the local date/time formatted as 2014-03-19 11:11:52
// https://stackoverflow.com/questions/7411301/how-to-introduce-date-and-time-in-log-file
static string getFormattedTime()
{
    timeval curTime;
    gettimeofday(&curTime, NULL);
    int milli = curTime.tv_usec / 1000;

    char buffer[80];
    strftime(buffer, 80, "%Y-%m-%dT%H:%M:%S", localtime(&curTime.tv_sec));

    char currentTime[84] = "";
    sprintf(currentTime, "%s.%03dZ", buffer, milli);

    return currentTime;
}
```

---

<sup>7</sup><https://codereview.stackexchange.com/questions/11921/getting-current-time-with-milliseconds>



## 4.5 Vytvorenie správ na odoslanie

Aplikácia vytvára správy s dĺžkou nepresahujúcou 1KB. Na Syslog server je v jednej správe posielaných viac štatistík oddelených znakom | .:

```
//Return statistics formatted as syslog messages
static list<string> getMessages(string hostname)
{
    list<string> messages;
    string message;

    //Create list of all messages
    list<DnsRecord>::iterator it;

    //Create copy (safer multithreading)
    list<DnsRecord> listToSend = recordList;

    for(it = listToSend.begin(); it != listToSend.end(); it++)
    {
        string record = it->GetString();

        //We can add another record to one message if limit wasn't exceeded
        if ((record + message).size() > MESSAGE_SIZE)
        {
            message.erase(0, 3);
            message = "<134>1 " + getFormattedTime() + " " + hostname + " dns-export " + "-----" + message;
            messages.push_back(message);
            message = "";
        }

        record = " | " + record;
        message += record;
    }

    message.erase(0, 3);

    //At the end add last message to the list
    message = "<134>1 " + getFormattedTime() + " " + hostname + " dns-export " + "-----" + message;
    messages.push_back(message);

    return messages;
}
```

## 5 Použitie

### 5.1 Čítanie pcap súboru

Aplikácia sa spúšťa s argumentmi **-r** a **-s**:

```
xnerec00@d01-206b:/mnt/c/Users/tomn/Documents/git/FIT/5.sem/isa/src$ ./dns-export -r sample/cname.pcap -s 192.168.139.3
```

V tomto prípade sa na **stdout** nič nevypíše, štatistiky sú odoslané na Syslog server, napr.:

```
Nov 16 22:52:35 d01-206b dns-export solid.preyproject.com CNAME control.preyproject.com 2 i control.preyproject.com A 107.178.244.155 1 i control.preyproject.com AAAA 2600:1901:0:9af5:0:0:0:0 1
```

V prípade, že nie je zadán Syslog server sa štatistiky vypíšu na **stdout**:

```
xnerec00@d01-206b:/mnt/c/Users/tomn/Documents/git/FIT/5.sem/isa/src$ ./dns-export -r sample/cname.pcap
solid.preyproject.com CNAME control.preyproject.com 2
control.preyproject.com A 107.178.244.155 1
control.preyproject.com AAAA 2600:1901:0:9af5:0:0:0:0 1
```

### 5.2 Odchyťovanie na rozhraní

Aplikácia sa spúšťa s argumentmi **-i**, **-s** a **-t**:

```
xnerec00@d01-206b:/mnt/c/Users/tomn/Documents/git/FIT/5.sem/isa/src$ ./dns-export -i enp0s8 -s 192.168.139.3 -t 45
```

V tomto prípade budú štatistiky odosielané na Syslog server každých 45 sekúnd. Ak nie je zadán argument **-t**, štatistiky sú odosielané každých 60 sekúnd. Ak nie je zadán argument **-s**, štatistiky sa neodosielajú.

### 5.3 Zaslanie signálu SIGUSR1

Pri odchyťovaní na rozhraní sa štatistiky vypíšu na **stdout** pri zaslaní signálu SIGUSR1:

```
root      1869  0.0  0.0 18712   776 tty1    R+   23:28   0:00 ./dns-export -i any
root      1874  0.0  0.1 155296  1748 pts/0   R+   23:28   0:00 ps aux
[root@isa ~]# kill -10 1869
```

```
[root@isa src]# ./dns-export -i any
google.com A 172.217.23.206 1
```

## 6 Prehľad naštudovanej literatúry

Práca s knižnicou libpcap a parsovanie ethernetovej hlavičky a transportných protokolov - príklady k predmetu ISA (`sniff.c`, `read-pcap.c`) a rôzne voľne dostupné zdroje na internete.

Syslog protokol - prednáška `isa-logovani.pdf`

Parsovanie DNS hlavičky a jednotlivých typov záznamov - RFC 1035

Parsovanie záznamov pre DNSSEC - RFC 4034

Parsovanie hlavičky pri LINKTYPE\_LINUX\_SLL - `tcpdump.org`

Špecifickosť SPF záznamu - `support.dnssimple.com/`

## 7 Literatúra

<https://tools.ietf.org/html/rfc1035>

<https://www.ietf.org/rfc/rfc4034.txt>

[http://www.tcpdump.org/linktypes/LINKTYPE\\_LINUX\\_SLL.html](http://www.tcpdump.org/linktypes/LINKTYPE_LINUX_SLL.html)

<https://support.dnssimple.com/articles/spf-record/>

[https://en.wikipedia.org/wiki/List\\_of\\_DNS\\_record\\_types](https://en.wikipedia.org/wiki/List_of_DNS_record_types)

<https://solarianprogrammer.com/2011/12/16/cpp-11-thread-tutorial/>

<https://www.rhyous.com/2011/11/13/how-to-read-a-pcap-file-from-wireshark-with-c/>

<https://www.tcpdump.org/pcap.html>

[https://www.tcpdump.org/manpages/pcap\\_open\\_live.3pcap.html](https://www.tcpdump.org/manpages/pcap_open_live.3pcap.html)