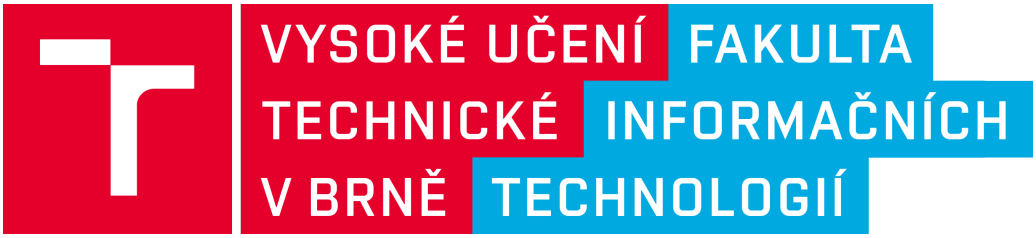

Projekt ISA 2022

Čítačka noviniek vo formáte Atom a RSS s podporou TLS

10. novembra 2022 Denis Horil (xhoril01)

Obsah

1 Úvod

Cieľom projektu bolo vytvoriť program `feedreader` v C/C++, ktorý bude vypisovať požadované informácie z tzv. `feedov`. Informácie sú uvedené v XML súboroch vo formáte RSS 2.0 [3] alebo Atom [4].

2 Popis vstupného súboru

Ako bolo spomenuté v 1, program stiahne informácie z webovej adresy (angl. `feed`) a vypíše ich.

Internetový *feed* je formát dokumentu, ktorý dovoľuje publikáciu zoznamov s informáciami alebo konkrétnymi prepojeniami [1]. Tento dokument obsahuje informácie o *feede* samotnom (tzv. *metadata*) - nadpis daného *feedu*, jeho autora, dátum publikácie a kde ho vieme nájsť na internete. Za tým nasleduje zoznam **prvkov**(tzv. *item*) a **vstupov**(tzv. *entry*).

Bežný prvok obsahuje *link*, *nadpis* a *popis*. Množstvo prvkov obsahuje aj iné *metadata* napr. dátum publikácie alebo meno či email autora.

Najčastejšie používané formáty feedov sú **RSS2.0** alebo **Atom**.

2.1 RSS

RSS je publikovací formát obsahu na internete. Skratka *RSS* je akronymom pre "Really Simple Syndication"(*Naozaj jednoduchá publikácia*). Tento formát je "dialektom"XML formátu a preto musí odpovedať špecifikácii XML 1.0[2]. Formát *RSS* má niekoľko verzií [3]:

- RSS 0.90
- RSS 0.91
- RSS 0.92
- RSS 1.0
- RSS 2.0

Napriek spoločnému názvu RSS sa jednotlivé verzie líšia, a to najmä v prvkoch (*elementoch*).

2.1.1 RSS 1.0

Verzia 1.0 bola založená na verzii 0.90 a aby si ju osvojili tvorcovia obsahu, bola zabezpečená aj spätná kompatibilita týchto 2 verzií [4].

Verzia RSS 1.0 obsahuje koreňový prvok `<rdf:RDF>`, v ktorom je povinný prvok `<title>`. Spoločný s ostatnými verziami je však prvok `<channel>`, ktorý sa vyskytuje aj v tejto verzii len raz. Narozdiel od verzie RSS 2.0 sa prvky `<item>` nachádzajú na rovnakej úrovni ako prvok `<channel>`. Prvok `<item>` musí obsahovať 2 prvky:

[1] https://www.mnot.net/rss/tutorial/
[2] https://www.w3.org/TR/REC-xml/
[3] https://www.rssboard.org/rss-history
[4] https://validator.w3.org/feed/docs/rss1.html#s7

- `<title>`
- `<link>`
- `<description>`

2.1.2 RSS 2.0

Verzia 2.0 bola založená na verzii 0.91, takže aj tu je zabezpečená spätná kompatibilita, no medzi verziami 1.0 a 2.0 spätná kompatibilita **NIE JE** zabezpečená[3].

Verzia RSS 2.0 (a teda aj RSS 0.91 a 0.92) obsahuje koreňový prvok `<rss>`, kde je povinne uvedená daná verzia. V tomto koreňovom prvku sa nachádza jediný prvok `<channel>`, v ktorom je povinný prvok `<title>`. Všetky prvky `<item>` sú podprvkami prvku `<channel>`. Povinné prvky `<item>` sú:

- `<title>`
- `<link>`
- `<description>`

Vo verzii RSS 0.91 nie sú informácie o autorovi alebo dátum úpravy podporované, vo verzii RSS 0.92 boli pridané niektoré nepovinné prvky a vo verzii RSS 2.0 sú podporované nepovinné prvky:

- `<author>`
- `<pubDate>`

2.2 Atom

Atom je publikovací formát založený na báze XML, ktorý popisuje zoznamy súvisiacich informácii - `feedov`. `Feedy` pozostávajú z viacerých prvkov tzv. *vstupov* (angl. `entry`). Každý takýto prvok obsahuje rozšíriteľnú množinu pripojených metadát napr. každý `entry` má nadpis (`title`)[5].

Koreňovým elementom je prvok `<feed>`. Ten musí obsahovať 3 povinné prvky:

- `<id>` - identifikuje *feed* unikátnou a permanentnou URI [2]
- `<title>` - titulok/názov *feedu*
- `<updated>` - posledná úprava *feedu*

Po predchadzájúcich metadátach nasleduje ľubovoľný počet prvkov `<entry>`, ktoré reprezentujú jednotlivé položky zdroja.

Každý prvok `<entry>`, rovnako ako `<feed>`, musí obsahovať 3 povinné prvky `<id>`, `<title>` a `<updated>`. Ďalej sú odporúčané aj prvky:

- `<author>` - uvádza jedného autora *feedu*
- `<content>` - obsahuje kompletný obsah prvku `<entry>`
- `<link>` - identifikuje asociovanú URL
- `<summary>` - poskytuje krátke zhrnutie, abstrakt alebo úryvok zo záznamu

[5] https://datatracker.ietf.org/doc/html/rfc4287

Pri prvku `<author>` je to ale trochu zložitejšie. Každý prvok `<entry>` môže obsahovať viacerých autorov alebo aj žiadneho. Prvok `<author>` obsahuje povinne prvok `<name>` (meno autora) a nepovinne `<email>` (email autora) alebo `<uri>`(domovská stránka autora). Pokiaľ nie je uvedený autor, použije sa autor z prvku `<source>`. Ak ani ten nie je uvedený, použije sa autor uvedený v prvku `<feed>`.

3 Implementácia

Program `feedreader` je čítačkou noviniek vo formáte Atom 2.2 a RSS 2.1 s podporou TLS. Program po spustení stiahne zadané zdroje a vypíše na štandardný výstup informácie požadované úžívateľom (napr. názvy článkov).

Program je implementovaný v C++, a teda je využitý objektovo orientovaný prístup. Celý program je implementovaný v 2 súboroch: `feedreader.cpp` a `classes.cpp`.

3.1 `feedreader.cpp`

Súbor `feedreader.cpp` je hlavným súborom, ktorý spúšťa celú čítačku. V tomto súbore sú definované 2 funkcie: `prog_interrupt` a `main`.
Funkcia `prog_interrupt` zabezpečuje uvoľnenie pamäte pri nečakanom ukončení programu (CTRL+C).

Funkcia `main` vytvorí inštancie tried *Args* 3.2.4 a *Process* 3.2.5. Táto funkcia vyvolá postupne funkcie na spracovanie vstupných argumentov a stiahnutie *feedov* zo zadaných zdrojov.

3.2 `classes.cpp`

V súbore `feedreader.cpp` sú implementované triedy a všetky pomocné funkcie a štruktúry. Takisto sú tu definované makrá pre výpis chybových stavov a uvoľnenie pamäte po vyvolaní chyby.

3.2.1 Makrá

V celom programe sú použité 3 makrá - `ERR_STAT`, `ERR_STAT_ARG` a `IM_FREEEEE`.
`ERR_STAT` je makro pre výpis chybovej hlášky na štandardný chybový výstup bez parametra.
`ERR_STAT_ARG` je podobne ako `ERR_STAT` makro pre výpis chybovej hlášky na štandardný chybový výstup, ale s pridaným parametrom, ktorý sa má vypísať.
`IM_FREEEEE` je makro, ktoré uvoľní pamäť alokovanú pri vytváraní pripojenia na zadanú webovú adresu.

3.2.2 Pomocné funkcie

Ďalej sú definované funkcie `schemeCheck` a `checkURL`, ktoré slúžia na kontrolu správnosti zadanej URL.
Funkcia `schemeCheck` skontroluje, či je správny protokol pre sťahovanie zdrojov. Povolené protokoly sú *HTTP* a *HTTPS*..
Funkcia `checkURL` skontroluje správnosť zadanej URL, rozloží URL na jednotlivé zložky a uloží tieto zložky do štruktúry `parsedURL` 3.2.3.1. Pokiaľ vrámci URL nebolo špecifikované číslo portu, implicitne sa použije číslo portu pre daný protokol tj. 80 pre protokol *HTTP* a 443 pre *HTTPS*

3.2.3 Pomocné štruktúry

Štruktúry uvedené nižšie slúžia zväčša na uloženie infromácií pre lepšiu manipuláciu s danými dátami.

3.2.3.1 `parsedURL`

V tejto štruktúre je uložená URL rozložená na jednotlivé časti. URL sa skladá z:

- Protokol (*scheme*) - protokol *HTTP* alebo *HTTPS*
- Doména (*host*) - napr. `www.vutbr.cy`
- Číslo portu (*port*) - ak nie je zaané, implicitne sa nastaví podľa protokolu
- Cesta (*path*) - cesta k danému *feedu* napr. s príponou `.atom`
- Autorita (*authority*) - kombinácia v tvare `host:port`

3.2.3.2 `Feed`

Táto štruktúra slúži na uloženie cesty k užívateľom zadanému `feedfile` a na uloženie zoznamu validných URL zdrojov, ktoré boli zadané v `feedfile`.

3.2.3.3 `Author`

Štruktúra `Author` uchováva v sebe informácie o autorovi aktuálneho *feedu* - jeho meno resp. jeho email.

3.2.3.4 `XMLContent`

V štruktúre `XMLContent` sú uložené všetky informácie o danom *feede*. Slúži na výpis užívateľom požadované informácie. Štruktúra si uchováva názov aktuálneho *feedu*, zoznam jeho autorov, zoznam asociovaných URL a taktiež dátum poslednej úpravy.

3.2.4 Trieda *Args*

V triede *Args* sú implementované všetky funkcie potrebné pre spracovanie argumentov zadané užívateľom na príkazovom riadku.

Hlavnou funkciou je funkcia `argsProcesser()`, ktorá pomocou funkcie `getopt_long()`[6] identifikuje vstupné argumenty zadané na príkazovom riadku 4. Podľa zadaných argumentov sa uložia jedntolivé parametry do premenných.

Po úspešnom uložení parametrov a argumentov sa skontroluje funkciou `reqArgsCheck()` či bol zadaný aspoň jeden z povinných parametrov.

Ďalej je tu implementovaná funkcia `printHelp()`, ktorá vypíše nápovedu na štandardný výstup a takisto funkcie štýlu *"getters"* - tieto funkcie vrátia zadaný argument či parameter (napr. funckia `getURL()` vráti zadanú URL adresu alebo funkcia `isTime()` vráti, či bol alebo nebol zadaný parameter *-T*).

3.2.5 Trieda *Process*

V triede *Process* sú implementované všetky funkcie potrebné pre získanie zadaných informácií. V závislosti na zadaných parametroch sú 2 spúšťacie "metódy":

- 1. - bola zadaná URL a teda rovno je možné spustiť hlavnú funkciu `connect()` 3.2.5.1

[6] https://www.gnu.org/software/libc/manual/html_node/Getopt-Long-Option-Example.html

- 2. - bol zadaný argument `-f <feedfile>` a treba najskôr získať URL z *feedfile*

V prípade 2. metódy bude spustená funkcia `feedfile2List()`. Táto funkcia zo zadaného súboru *feedfile* vytiahne všetky URL adresy a uloží do zoznamu v štruktúre `Feed` 3.2.3.2. Ďalej bude vyvolaná funkcia `loopConnect()`, ktorá iteruje cez uloženy zoznam URL adries, kontroluje ich a posiela do funkcie `connect()` 3.2.5.1.

3.2.5.1 Funkcia `connect()`

Funkcia `connect()` zabezpečuje pripojenie (či už zabezpečené alebo nie) na zadanú URL adresu a sťahuje z nej *feedy*. Pripojenie sa uskutočňuje pomocou knižnice `OpenSSL`, ktorá využíva pripojenie cez *BIO* sokety[1].

Po úspešnom pripojení na server sa do premennej `response` načíta celá odpoveď zo serveru a spracuje sa vo funkcii `responseProcess()` 3.2.5.2. Po úspešnom spracovaní odpovede nastáva *"parsovanie"* obsahu vráteného funkciou `responseProcess()` vo funkcii `parseXML()` 3.2.5.3. V tejto funkcii sa priamo volá aj funkcia pre výpis požadovaných informácii.

3.2.5.2 Funkcia `responseProcess()`

Funkcia `responseProcess()` oddelí hlavičku a obsah*HTTP* odpovede a obsah v tvare popísanom v sekcii 2 vráti do funkcie `connect()`.

3.2.5.3 Funkcia `parseXML()`

Táto funkcia zodpovedá za parsovanie *feedu* v tvare XML s pomocou knižnice `libxml2`. S pomocou funkcie `xmlParseDoc()` sa sparsuje zadaný *feed* a pomocou funkcie `xmlDocGetRootElement()` sa získa koreňový prvok.

Na základe koreňového prvku sa rozhodne, akým štýlom sa daný *feed* bude parsovať - Atom alebo RSS2.0 . (v kóde je implementovaná aj funkcia na parsovanie RDF/RSS1.0 *feedu*, ale je zakomentovaná z dôvodu že tento typ nie je presne určený v zadaní projektu).

Po úspešnom parsovaní funkcia `printInfo()` vypíše požadované informácie na štandardný výstup podľa formátu určeného v zadaní 6.

4 Spustenie programu

Program sa spúšťa cez príkazový riadok na platforme Linux. Možnosti spustenia preloženého programu:
`feedreader <URL | -f <feedfile>> [-c <certfile>] [-C <certaddr>] [-T] [-a] [-u]`
kde:

- `URL` - URL adresa daného *feedu*
- `-f <feedfile>` - cesta k súboru s URL adresami zdrojov 4.1
- `-c <certfile>` - cesta k súboru s certifikátmi
- `-C <certaddr>` - cesta k adresáru s certifikátmi

- `-T` - vypíšu sa aj informácie o čase poslednej zmeny daného záznamu
- `-a` - vypíše sa autor daného záznamu (ak bol zadaný) alebo jeho email (ak bol zadaný)
- `-u` - vypíšu sa asociované URL s daným záznamom

Pri spustení na poradí parmetrov nezáleží, no musí byť uvedený jeden z parametrov `URL` alebo `-f <feedfile>`, nie však oba. Ak nebude zadaný ani jeden z parametrov `-c <certfile>` alebo `-C <certaddr>`, na získanie certifikátov bude použitá funkcia `SSL_CTX_set_default_verify_paths()`. Pre výpis nápovedy stačí použiť parameter `-h`, `--help` alebo `make help`. Po výpise nápovedy sa program ukončí.

4.1 Popis súboru `feedfile`

Feedfile je textový súbor, ktorý obsahuje URL adresy záznamov, ktoré má `feedreader` stiahnuť. Tieto zdroje sú oddelené novým riadkom. Ignorujú sa prázdne riadky, biele znaky aj komentáre (komentár začína znakom '#').

```
 1 https://www.theregister.com/headlines.atom
 2 #commentary
 3
 4
 5 ftp://www.theregister.co.uk/data_centre/headlines.atom
 6 https://www.theregister.com/software/headlines.atom
 7 https://www.theregister.com/hardware/headlines.atom
 8 http://feeds.rssboard.org/rssboard
 9 https://www.rssboard.org/files/sample-rss-091.xml
10 https://www.rssboard.org/files/sample-rss-092.xml
11 https://www.rssboard.org/files/sample-rss-2.xml
```

Obr. 1: Ukážka *feedfile.txt*

5 Preklad progrmau

Preklad programu je možný 2 spôsobmi:

5.1 Preklad cez príkazový riadok

Pre preklad stačí na príkazový riadok napísať príkaz:

```
g++ -Wall -Wextra -Werror -pedantic -I /usr/include/libxml2/ -o feedreader ./src/feedreader
-lcrypto -lssl -lxml2
```

Program je prekladaný ekvivalentom k prekladaču `gcc - g++`. Pri preklade sú použité všetky možnosti zachytávania chýb.

5.2 Použitie `Makefile`

Pre preklad pomocou `Makefile` stačí na príkazový riadok napísať príkaz `make` alebo `make all`.

6 Výstup programu

Na štandardný výstup sa vypíše názov zdroja uvedený „***“ a ukončený „***“, napr. **** Príklad ****. Na ďálších riadkoch budú uvedené názvy jedntolivých záznamov, ktoré budú oddelené prázdnym riadkom. V prípade zadania parametrov `-T`, `-a` alebo `-u`, budú pod názvom záznamu uvedené dodatočné informácie uvedené reťazcom „Aktualizace:“, „Autor:“ resp. „URL:“. Ak názov záznamu nebol uvedený vypíše sa reťazec `<No Title>`.

```
*** Liftoff News ***
Star City
Updated: Tue, 03 Jun 2003 09:39:21 GMT
URL: http://liftoff.msfc.nasa.gov/news/2003/news-starcity.asp

<No Title>
Updated: Fri, 30 May 2003 11:06:42 GMT

The Engine That Does More
Updated: Tue, 27 May 2003 08:37:32 GMT
URL: http://liftoff.msfc.nasa.gov/news/2003/news-VASIMR.asp

Astronauts' Dirty Laundry
Updated: Tue, 20 May 2003 08:56:02 GMT
URL: http://liftoff.msfc.nasa.gov/news/2003/news-laundry.asp
```

Obr. 2: Príklad výstupu

7 Testovanie

Pre účely testovania som vytvoril skript `test.sh`, uložený v priečinku `test/`, ktorý rekurzívne vyhľadá všetky súbory `feedfile` v zadanom adresári. Výsledky testov uloží do priečinka `test/test_outputs` do súborov s príponami `.err` a `.out`. Tieto výstupné súbory sú nazvané podľa originálneho názvu `feedfile`, napr. ak má `feedfile` názov `abc.txt`, tak v súbore `test/test_outputs` vzniknú 3 súbory:

- `abc_c-Input.out` - v tomto súbore budú uložené výstupy všetkých záznamov, pri ktorých nenastala chyba
- `abc_w-Input.out` - v tomto súbore budú uložené chybové hlásenia z rôznych nevalidných vstupov
- `abc_c-Input.err` - v tomto súbore budú uložené chybové hlásenia záznamov, pri ktorých nastala chyba aj napriek validnému vstupu

Testovací skript je možné spustiť príkazom `make test` alebo `./test.sh [--nodir] [-d <addr>]`, kde `--nodir` zabezpečí vymazanie obsahu celého priečinka `test_outputs` a `<addr>` je cesta k adresáru so súbormi `feedfile`. Ak nebude zadaný parameter `-d`, tak bude implicitne použitý adresár `test/files/`.

Pre vypísanie nápovedy testovacieho skriptu stačí použiť príkaz `make test_help` alebo spustiť skript s parametrom `-h` alebo `--help`.

8 Záver

V projekte sa mi podarilo implementovať všetky časti aplikácie, ktoré boli v zadaní projektu. Testovanie nebolo veľmi rozsiahle, ale myslím že testy pokryli väčšinu validných i nevalidných vstupov.

Samotný projekt nebol veľmi časovo náročný (približne 40 - 50 hodín), avšak mne osobne najviac času zabralo naštudovanie potrebných informácii napr. ako funguje knižnica `OpenSSL`, knižnica `libxml2` alebo formáty `Atom` a `RSS2.0`, čo mi zabralo ešte nejakých 5 - 10 hodín navyše.

V každom prípade som si pri práci zdokonalil svoje vedomosti v oblasti zabezpečenej a nezabezpečenej internetovej komunikácii, ako funguje TLS a overovanie certifikátov alebo aké sú možnosti vytvárania *feedov*. Ale hlavne som sa zdokonalil v programovaní v C++. Vyskúšal som prvýkrát OO prístup v C++ a využíval som množstvo regulárnych výrazov vďaka knižnici `<regex.h>`. Takisto som si rozšíril vedomosti o nové druhy súborov na štýl XML - `Atom` a `RSS2.0`. V neposlednom rade som sa naučil základy vytvárania dokumentu v programe LaTeX, čo určite využijem pri vytváraní BP.

Zdroje

[1] Kenneth Ballard. Secure programming with the openssl api. [online], pub. 21.júla 2004. [vid. 19.októbra 2022].

[2] Tim Berners-Lee, Roy Fielding, and Larry Masinter. Uniform resource identifier (uri): Generic syntax. Technical report, 2005. [vid. 19.októbra 2022].

[3] RSS Advisory Board. Rss 2.0 specification. [online]. [vid. 19.októbra 2022].

[4] AtomPub Working Group. Introduction to atom. [online]. [vid. 19.októbra 2022].