VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

ČTEČKA NOVINEK VE FORMÁTU ATOM A RSS S PODPOROU TLS

NEWS READER IN ATOM AND RSS FORMAT WITH TLS SUPPORT

TECHNICKÁ DOKUMENTACE
TECHNICAL DOCUMENTATION

AUTOR PRÁCE — **ADAM JANDA**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. LIBOR POLČÁK, Ph.D.**
SUPERVISOR

BRNO 2022

Obsah

Kapitola 1

Úvod

Tento dokument popisuje riešenie automatického získavania a spracovania noviniek zo vzdialeného serveru s podporou pre zabezpečenú komunikáciu.

Nasledujúcich kapitolách sa priblížia základné a zjednodušené fungovanie jednotlivých častí implementácie. Najprv sa uzrejmia základné informácie, odlišnosti od zadania, obmedzenia, spôsob kompilácie celého projektu/aplikácie a testy aplikácie. Potom sa spomenie stronová štruktúra súborov a priečinkov, a lokalizácia. Nakoniec sa v stručnosti vysvetlí implementácia, chybové ošetrenia a príklady použitia.

Kapitola 2

Základné informácie

Naimplementovaná aplikácia sa pripája k požadovanému zdroju, stiahne ATOM alebo RSS 2.0 dáta, spracuje ich a vypíše užívateľovi na štandardný výstup. Pri HTTPS komunikácii, aplikácia podporuje SSL/TLS pre overenie certifikátu zdroju.

Požadovaný zdroj môže byť vložený pomocou jednej URL alebo pomocou súboru s viacerými URL pri spustení aplikácie. Vždy sa vypíše názov zdroju a názvy jednotlivých príspevkov. Užívateľ si môže nechať vypísať dátum publikovania, URI na konkrétny príspevok a autora príspevku.

Rozšírenie

- Rôzne návratové hodnoty pri chybách, definovaných v `../src/error.h`.
- Lokalizácia správ aplikácie do čestiny a angličtiny. Prepínač je v `../src/error.h`.

Obmedzenia

- Kompilácia zdrojového kódu sa musí spraviť pomocou GNU Make.
- Aplikácia je vyvinutá pre UNIX OS.

Kompilácia

Predstavované riešenie obsahuje aj základné testy vstupných parametrov. Kompilácia testov nie je súčasť predvoleného kompilovania zdrojových súborov. Je ich možné pridať pridaním parametru `test` k GNU make príkazu. Taktiež je možné testy skompilovať spolu s ostatnými zdrojovými súbormi aplikácie pomocou parametru `all`. Príklady použitia sú v kapitole 5. Po kompilácii sa vytvoria spustiteľné súbory `feedreader` a `test` v priečinku s Makefile. Záleží či kompiluje len aplikácia, alebo len testy, alebo všetko.

Testy

Implementácia obsahuje aj unit testy, ktoré testujú základné spracovanie vstupných parametrov.

Kapitola 3

Návrh aplikácie

Stromová štruktúra

```
  root
├── bin
├── src
│   ├── atom.c
│   ├── atom.h
│   └── ...
├── tests
│   ├── examples
│   │   ├── FIT-cert
│   │   └── ...
│   └── ...
├── Makefile
├── manual.pdf
└── README.md
```

Význam priečinkov aplikácie:

- priečinok `bin` bude po kompilácii aplikácie obsahovať objektové súbory zdrojových súborov
- priečinok `src` obsahuje zdrojové súbory implementácie aplikácie
- priečinok `tests` obsahuje zdrojové súbory testov aplikácie a vzorové príklady

Lokalizácia

Nad rámec zadania bola naimplementovaná dvojjazyčná mutácia aplikácie. Lokalizácia poskytuje českú a anglickú jazykovú mutáciu.

Predvoleným jazykom je čestina a zmena je možná prenastavením hodnoty **LANG** v `../src/error.h` na 0 (nula). Toto nastavenie ovplyvňuje preklad častí výstupu pre užívateľa, ako napr. *Autor:* na *Author:* pri výpise autora článku. Ako ďalší vplyv tohto nastavenia je preklad všetkých chybových hláškok, viac v kapitole 4.

Kapitola 4

Implementácia

Naimplementovaná čítačka noviniek je napísaná v programovacom jazyku C. Pri vývoji aplikácie boli použité štandardné knižnice jazyka C (*stdio, stdlib, string, getopt, time...*), sieťové knižnice(`sys/*, netdb`), knižnica openssl pre TLS/SSL a knižnica libxml pre spracovanie XML odpovede.

Vstupné parametre

Spracovanie užívateľského vstupu má na starosti `../src/parameters.c`. Kontroluje správnosť kombinácii vstupných parametrov a ich požadovaného správania, podľa zadania projektu.

Možnosti

- URL so schématom http alebo https
- -f *cesta-k-suboru-s-urls*
- -c *cesta-k-suboru-pertifikatu*
- -C *cesta-k-priečinku-pertifikatov*
- prepínače zobrazenia doplňujúcich dát článku: -a, -u, -T

Kombinácie

Aplikácia **nepodporuje** kombinácie:

- URL a -f *cesta-k-suboru-s-urls*
- -c *cesta-k-suboru-pertifikatu* a -C *cesta-k-priečinku-pertifikatov*
- mnohonásobné použitie prepínačov alebo URL

Validné príklady použitia sú k nahliadnutiu v kapitole 5.

Spracovanie URL

URL môže byť užívateľom vložená do aplikácie dvomi spôsobmi:

- pri spustení cez terminál

- vložený do súboru, ktorý je predaný aplikácii s parametrom `-f`

O spracovanie URL z príkazového riadku a súboru sa starajú funkcie súboru `../src/host.c`, konkrétne funkcie *parse_url()* a *read_urls()*.

Spracovanie URL prebieha pomocou regulárneho výrazu[1], ktorý bol navrhnutý tak, aby pri použití na URL vyhľadal a jednotlivé časti (skupiny) URL. A to:

- http:// alebo https://
- www.
- *domenovy-nazov-serveru.domenu-najvyssej-urovne*
- *:port*
- *cesta-k-suboru*

Súbor pre parameter `-f` môže obsahovať jednu URL na každom riadku. Ak má na začiatku znak '#', tento riadok je braný ako komentár. Každý nekomentový riadok je spracovaný funkciou *parse_url()*. Pri chybnom formáte URL, je na štandardný chybový výstup vypísaná chyba a aplikácia pokračuje ďalej vo vykonávaní implementácie.

TCP komunikácia

Implementácia TCP časti sa nachádza v `../src/tcp_communication.c`, ktorého podstata vychádza z prednášky a jej príkladov predmetu ISA, konkrétne prednášku docenta Matoušku, Pokrocilé programování sítí TCP/IP[3].

HTTP a HTTPS

HTTP a HTTPS komunikácia sú implementované v zdrojový súboroch `../src/http_communication.c` a `../src/https_communication.c`. Funkcie (`send_http_request()` a `send_https_request()`) oboch týchto komunikácií sú volané, po prevedí predošlých úkonov, napr. naviazenie TLS spojenia pri HTTPS. Zasielajú HTTP 1.0 dotaz na cieľovú destináciu. Inšpiráciou pre HTTP dotaz, po menšej úprave, bola časť kódu z referečnej knihy predmetu ISA[5].

Prijímanie a uloženie dát

Prijímanie a uloženie je taktiež naimplementované v súboroch `../src/http_communication.c` a `../src/https_communication.c`. Výsledným produktom, pri úspešnej odpovedi cieľového serveru a prijatí všetkých dát, je dynamicky alokovaný reťazec, ktorý sa ďalej spracováva.

Ako úspešnú odpoveď, implementácia považuje odpoveď so statusom `200`. Všetky ostatné[2] sú brané ako neúspešná odpoveď a spracovanie dát sa ukončuje. Užívateľ si môže v chybovej hláške nájsť.

Inšpiráciou pre prijímanie odpovedí z sielového serveru boli už spomínaná prednáška docenta Matoušku[3] a refrečná kniha predmetu ISA[5].

Ukážky kódu z refrečnej knihy predmetu ISA[5], po úpravách a rozšíreniach, boli taktiež použité, ako inšpirácia pre pre TLS komunikáciu a prácu s certifikátmi vo funkciách súboru `../src/https_communication.c`.

Spracovanie XML

Hlavné spracovanie XML odpovede sa deje v `../src/feed.c`, kde sa reťazec konvertuje do XML objektu pomocou funkcie knižnice *libxml*[4]. Po úspešnom prekonvertovaní a nájdení koreňa XML, nasleduje spracovanie podľa typu štruktúry XML. Pri spracovaní sa vypíše názov zdroja a jednotlivé názvy článkov.

Ak užívateľ zadal príslušné paramatri a odpoveď obsahuje také údaje, aplikácia vypíše informácie o autorovi, dátume vydania a URI článku.

V prípade, že boli použité dodatočné parametre, články sú oddelené jedným prázdnym riadkom. Informácie o článku sú v poradí:

1. názov článku
2. meno autora článku
3. URI článku
4. dátum publikovania článku

Ak nejaká dodatočná informácia chýba, alebo nebola vybraná, preskočí sa a vypíše sa ďalšia v poradí.

Atom a RSS

Spracovanie je implementované v `../src/atom.c` a v `../src/rss.c`.

Oštrenie chýb

Každá chybová hláška má svoju českú a anglickú verziu. V zdrojových súboroch je volaná funkcia `error_msg()` s anglickou hláškou.

Pred výpisom sa prejde matica dvojic. Jedna dvojica zodpovedá jednej z chybových možností. A teda pri prechode maticou sa nájde riadok vyskytnutej chyby a prepínačom `LANG` sa určí stĺpec. Takto sa vyberá chybová hláška k výpisu pre užívateľa.

V matici dvojic sa nenachádzajú dva prípady chýb. Prvá je vypršanie času pri prijímaní dát v `../src/http_communication.c` a `../src/https_communication.c`. Tieto chyby si generujú správu samé a nastavujú globálne premenné návratovej hodnoty.

Druhou je zlá odpoveď na http dotaz rovnako v `../src/http_communication.c` a `../src/https_communication.c`. Na rozdiel od predošlej varianty, sa volá funkcia `error_msg()`, ktorá iným spôsobom vyhodnocuje návratovú hodnotu aplikácie.

Ku matici dvojíc existuje matica návratových hodnôt, ktorá má na rovnakých pozíciach hodnotu, ktorá zodpovedá dvojici. Preto keď sa nájde riadok dvojice, ktorej chyba sa prejavila, tak vie aplikácia nájsť správnu návratovú hodnotu.

Úspešný chod aplikácie vráti hodnotu 0. Pri chybách, aplikácia vráti príslušnú chybovú hodnotu, definovanú v `../src/error.h`.

Výpis návratových hodnôt:

- 0 - úspech
- 1 - všeobecná chyba
- 2 - chyba pri kompilácii regulárného výrazu

- 10 - žiadne vstupné parametre
- 11 - neznámy parameter
- 12 - niekoľkonásobné použitie rovnakého parametru
- 13 - nepovolená kombinácia parametrov
- 14 - viaceré zdroje cieľových destinácii
- 15 - zlý formát URL
- 16 - chýbajúca cesta k súboru na cieľovej destinácii
- 17 - nezadaný zdroj cieľovej destinácie
- 18 - chýbajúci/neznámy súbor s certifikátom
- 19 - chýbajúci/neznámy priečinok s certifikátmi
- 20 - nepodarilo sa preložiť URL na IP adresu
- 21 - neplatná cieľová destinácia
- 22 - vypršal čas na odpoveď
- 23 - vytváranie schránky zlyhalo
- 24 - nepodarilo sa naviazať spojenie s cieľovou destináciou
- 30 - chyba pri čítaní súboru
- 31 - súbor je prázdny alebo neobsahuje valídne URL
- 40 - neúspešná odpoveď od cieľového serveru, http status > 200
- 50 - chyba pri vytváraní CTX kontextu
- 51 - chyba pri vytváraní SSL objketu
- 52 - chyba pri naväzovaní SSL komunikácie
- 53 - chyba pri identifikovaní serveru
- 54 - nepodarilo sa získať certifikát cieľovej destinácie
- 60 - XML odpoveď je prázdna
- 61 - spracovanie XML stromu zlyhalo
- 62 - chyba pri konvertovaní XML na XML objekt
- 70 - chyba pri načítaní súboru certifikátu
- 71 - chyba pri načítaní priečinku certifikátov
- 72 - chyba pri nastavovaní predvoleného priečinku certifikátov
- 73 - overenie certifikátu zlyhalo

Kapitola 5

Príklady použitia

Kompilácia

Pri predvolenom GNU make.

- make – kompilácia zdrojových súborov
- make test – kompilácia zdrojových súborov testov
- make all – kombilácia všetkých zdrojových súborov projektu

Spustenie aplikácie

- `./feedreader https://www.fit.vut.cz/fit/news-rss/`
- `./feedreader -T https://www.fit.vut.cz/fit/news-rss/ -a`
- `./feedreader https://www.fit.vut.cz/fit/news-rss/ -T -a -u`
- `./feedreader -f test/examples/hosts.txt -T -a -u`
- `./feedreader -u -c test/examples/FIT-cert https://www.fit.vut.cz/fit/news-rss/`
- `./feedreader -u -c test/examples/FIT-cert -f test/examples/hosts.txt`

Literatúra

[1] *Regex(3) — Linux manual page* [online]. [cit. 2022-10-02]. Dostupné z: https://man7.org/linux/man-pages/man3/regexec.3.html.

[2] IBM. *Status codes and reason phrases* [online]. [cit. 2022-10-24]. Dostupné z: https://www.ibm.com/docs/en/cics-ts/5.2?topic=concepts-status-codes-reason-phrases#dfhtl_httpstatus.

[3] Matoušek, P. *Pokrocilé programování sítí TCP/IP* [přednáška]. 2022 [cit. 2022-10-31]. Dostupné z: https://moodle.vut.cz/pluginfile.php/502879/mod_resource/content/2/isa-sockets.pdf.

[4] Veillard, D. *Libxml2 Reference Manual* [online]. [cit. 2022-10-27]. Dostupné z: https://gnome.pages.gitlab.gnome.org/libxml2/devhelp/libxml2-parser.html#xmlParseDoc.

[5] Winkle, L. V. *Hands-On Network Programming with C : Learn Socket Programming in C and Write Secure and Optimized Network Code*. 1. vyd. Packt Publishing, Limited, 2019. ISBN 9781789344080.