VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Síťové aplikace a správa sítí – projekt

Čtečka novinek ve formátu Atom a RSS s podporou TLS

14. novembra 2022

Juraj Dedič

Obsah

1 Novinové zdroje

Úlohou bolo vytvoriť program feedreader, ktorý bude sťahovať a zobrazovať informácie v zdrojoch (feed) vo formáte RSS 2.0 alebo Atom[2]. Program po spustení stiahne zadané zdroje z URL alebo tvz. *feedfile* a vypíše dotazované informácie na štandardný výstup.

2 Implementácia

Zvolený implmentačný jazyk je C++, s využitím knižníc openssl a libxml2. Minimálna verzia knižnice openssl, na ktorej bol program testovaný je `1.0.2k-fips` z 26. januára 2017, nachádzajúca sa na serveri merlin. V súbore `makefile` je používaný GNU C++ prekladač, výsledný binárny súbor má názov `feedreader`. Program je rozdelený do niekoľkých zdrojových súborov:

- `feedreader.{hpp|cpp}` – obsahuje hlavné časti programu akými sú napríklad načítanie argumentov príkazového riadku, čítanie súboru `feedfile` alebo volanie funkcií ostatných súčastí programu,
- `downloader.{hpp|cpp}` – zabezpečuje načítanie a uloženie zdrojov,
- `parser.{hpp|cpp}` – prechádza zdrojový XML súbor a vypisuje potrebné informácie na výstup.

2.1 Spracovanie argumentov

Spracovanie argumentov je riešené pomocou funkcie `getopt_long`. Použitím tejto funkcie program zachytí prepínače pre zobrazenie autora, času a URL adresy článku.

Pre zobrazenie zvoleného zdroja je možné zadať URL zdroja ako argument programu, čo je zachytené druhým cyklom. Program akceptuje aj možnosť zadania umiestení zdrojov do tzv. `feedfile`, kde sa nachádza množina adries zdrojov oddelených riadkami. Umiestenie `feedfile` je špecifikované prepínačom `-f`.

Na poradí argumentov programu nezáleží, s vínimkou prepínača `-f`, za ktorým musí nasledovať umiestnenie súboru `feedfile`.

2.2 Príprava

Po spracovaní argumentov program pripraví dočasný adresár `./temp/` pre sťahovanie zdrojov.

Následne program spustí sťahovanie a výpis nad URL zadanou v argumentoch. V prípade špecifikovania súboru `feedfile`, je využitý cyklus ktorý prechádza tento súbor po riadkoch. Pokiaľ je riadok neprázdny a zároveň nie je komentárom, program spustí sťahovanie a výpis nad URL adresou na tomto riadku. Cyklus takto ďalej pokračuje až pokým sa nedostane na koniec `feedfile`.

Pred sťahovaním súboru s novinkami je potrebné získať informácie z URL adresy. Toto je riešené pomocou funkcie `parse_url`, ktorá vracia štruktúru obsahujúcu hostname, port, umiestnenie zdroja a informáciu, či zdroj využíva HTTPS.

2.3 Sťahovanie

Spustená funkcia `read_from_url` z hlavnej časti programu zavolá príslušnú funkciu z `downloader.hpp` podľa toho, či sa jedná o zdroj využívajúci HTTP alebo HTTPS.

V prípade, že sa jedná o HTTPS, program zavolá inicializačnú funkciu, ktorá je vyžadovaná pre kompatibilitu s verziou openssl nižšou ako `1.1`. Program následne pomocou makra z dokumentácie openssl zvolí TLS metódu, kde pri starších verziách knižnice bude použitá metóda TLS `1.2`. Program nepodporuje verzie TLS/SSL nižšie ako `TLS 1.2` pokiaľ je verzia openssl `1.0.x` a nižšia, to hlavne z dôvodu že ich podpora je podstatne nižšia ako podpora vyššie spomínanej verzie[5]. Nový `SSL_CTX` objekt umožňujúci TLS spojenie bude potom využívať túto metódu.

Pokiaľ užívateľ špecifikoval cestu k adresáru alebo súboru s certifikátmi, bude použitá funkcia pre načítanie týchto certifikátov. Pri absencii týchto parametrov, bude využité úložisko certifikátov poskytnuté funkciou `SSL_CTX_set_default_verify_paths`. Na základe týchto certifikátov budú verifikované certifikáty poskytnuté serverom.

Ďalej sú nastavené parametre TLS spojenia a šifrovacie algoritmy. V tomto špecifikujeme zákaz používania niektorých dnes nedostatočných technológií a naopak povolíme šifrovacie sady spadajúce pod kritérium HIGH popísané v dokumentácii openssl[3]. Program pokračuje nadviazaním spojenia a dokončením handshaku. Po tomto je získaný a verifikovaný certifikát serveru.

U HTTP bez TLS je postup pripojenia zjednodušený vynechaním funkcií TLS. Pri HTTP je iniciovaný Basic Input/Output (BIO) pre komunikáciu medzi klientom a serverom pomocou funkcie `BIO_new_connect`. Volanie `BIO_do_connect` zaistí zahájenie spojenia.

Po zahájení spojenia aplikácia pošle pomocou HTTP metódy GET požiadavku pre získanie zdroja v zadanom umiestnení a špecifikuje využitie HTTP/1.0[1] a ďalšie potrebné prvky. Po prijatí odpovede je telo (obsahúce XML) oddelené od HTTP hlavičky. Následne je tento obsah zapísaný do súboru `./temp/temp.xml` a uvoľnia sa alokované prostriedky.

V prípade zlyhania niektorej z častí sťahovania alebo niektorej z volaných funkcií, program vypíše chybovú hlášku na štandardný chybový výstup a bude pokračovať sťahovaním prípadného ďalšieho súboru.

2.4 Parsovanie a výpis

Po stiahnutí súboru je získaný koreňový element. Ak je názov koreňového elementu *rss*, vieme že súbor používa tento formát a preto nastavíme za nový root element s názvom *channel*. V prípade, že názov je *feed*, budeme s ním pracovať ako s formátom Atom. Ak ani jeden z týchto názvov nezodpovedá, bude tento súbor preskočený.

Následne program iteruje cez elementy kanálu, kde zvyčajne prvým elementom je nadpis, ktorý je vypísaný. Pre každý element *entry*, ktorý predstavuje jeden článok je volaná funkcia `parse_item`. Táto funkcia postupuje taktiež iteratívne. Prvý cyklus hľadá a vypíše nadpis. Druhý vnútri entry hľadá a vypisuje informácie z nasledujúcich elementov:

- *title* – obsahuje v tele nadpis článku,
- *link* – v prípade štandardu RSS 2.0 obsahuje URL článku v parametre *href*. U formátu Atom je táto informácia uložená priamo v tele prvku,
- *published* (*pubDate* v RSS) – obsahuje dátum publikovania,
- *author* – obsahuje informácie o autorovi (meno a email) v dielčích elementoch.

Názvy elementov sú medzi RSS a Atom často odlišné[4]. Pri porovnávaní názvov elementov sú zadávané názvy vo formáte Atom. Z tohoto dôvodu je pri spracovaní formátu RSS pred porovnaním zavolaná funkcia `atom_to_rss` ktorá prevedie meno do jeho ekvivalentu v RSS 2.0 pre správnu funkčnosť. Príklad výpisu jedného článku:

```
Electrofishing for Whales
Autor:  xkcd
Autor (email): whatif@xkcd.com
URL: https://what-if.xkcd.com/156/
Aktualizace: 2017-03-09T00:00:00Z
```

[1] Táto verzia bola zvolená z dôvodu, že pri použití verzie 1.1 obsahovalo telo odpovede nechcené artefakty. Toto je zrejme spôsobené využitím chunked transfer encoding u verzie 1.1.

3 Použitie programu

Kompiláciu programu je možné vykonať príkazom `make`. Kompilácia prebieha pomocou GNU C++ prekladača. Program potrebuje knižnice *libxml2* a *openssl* od verzie `1.0.2k`. Čítačku noviniek je možné spustiť nasledovne:

```
feedreader <URL | -f <feedfile» [-c <certfile>] [-C <certaddr>] [-T] [-a]
[-u]
```

Význam argumentov:

- `-T` - program zobrazí publikovania článku,
- `-a` - program vypíše informácie o autorovi,
- `-u` - program zobrazí URL adresu článku,
- `-f feedfile` - zadanie súboru zdrojov, (each feed channel is at a separate line),
- `-c certfile` - špecifikovanie súboru s certifikátom, ktorý bude použitý pre overenie platnosti TLS certifikátov poskytnutých serverom,
- `-C certaddr` - špecifikovanie adresáru s certifikátmi,
- `URL` - adresa zdroja.

Pre spustenie testovacieho skriptu je možné použiť príkaz `make test`. Testovací skript vyžaduje Node.js V12 a vyššie.

4 Testovanie

Pre počiatočný debugging boli používané zdroje noviniek zo stránok theregister.com, what-if-xkcd.com a wikipedia.org. Pre testovanie rôznych zmien je ale nepraktické znovu zadávať príkazy a porovnávať správnosť výstupu, ale je vhodnejšie použiť testovací skript.

4.1 Testovací skript

Problémom testovacieho skriptu je skutočnosť, že zdroje sa na webových stránkach menia. Preto bola využitá možnosť hostingu vlastných zdrojov, ktoré zostanú súčasťou testovacieho adresára `./tests/`. Zvolené implementačné prostredie testovacieho skriptu je Node.js (testované sú verzie Node.js od v12 a vyššie, dostupné na serveroch Merlin a Eva), najmä pre relatívne kompaktný kód potrebný na hostovanie testovacích XML súborov.

Testovací skript po spustení vytvorí lokálny server na náhodne zvolenom voľnom porte, ktorý umožňuje prenos súborov z testovacieho adresára. Časť kódu ktorá zabezpečuje vytvorenie lokálneho serveru je prevzatá z Mozilla Developer Network [1] a je označená v zdrojovom súbore skriptu.

Po vytvorení serveru skript prehľadá súbory *testovacie súbory* s prípojkou `.json`. Názov každého z týchto súborov je braný ako názov konkrétneho testu. Každý testovací súbor obsahuje parameter s argumentami, ktoré budú použité pre spustenie čítačky noviniek. V tomto súbore sa ďalej nachádzajú adresy pre `feedfile`. Prípadne je tu jedna adresa, ktorá bude zadaná do argumentov pred spustením testovacieho behu programu. Tieto adresy musia byť len mená testovacích XML súborov, ktoré sú umiestené v `./tests/`. Skript všetky tieto adresy pred spustením programu konvertuje na URL adresy pre stiahnutie z lokálneho serveru. V prípade `feedfile` je tento súbor vygenerovaný vo formáte `<nazov-testu>.feed`[2] v testovacom adresári. Zároveň je potrebné zadať cestu k tomuto súboru (vzhľadom na feedreader) do argumentov v testovacom súbore. V testovacom súbore je taktiež definovaný predpokladaný návratový kód testu.

V prípade, že je očakávaný návratový kód 0, testovací skript načíta súbor s uloženým predpokladaným správnym výstupom: `./tests/<nazov-testu>.out`. Následne je pomocou volania funkcie `exec` spustený program s príslušnými argumentami. Skript ďalej porovnáva zhodnosť návratových kódov. V prípade, že očakávaný ako aj skutočný kód je rovný 0, skript porovnáva výstup *stdout* s referenčným súborom. Pri tomto porovnaní sa ignorujú medzery kvôli zamedzeniu falošnej detekcie chybového výstupu.

Na konci testovacieho skriptu sú vypísané výsledky jednotlivých testov. Pre výpis podrobných informácií o výsledkoch je možné skript spustiť s parametrom `-verbose`. Ak nie je skript spustený s prepínačom `-no-clean`, vymaže vygenerované feed súbory. Pokiaľ nebol zadaný parameter `-wait`, skript ukončí lokálny server, v opačnom prípade musí byť ukončený manuálne.

[2] `<nazov-testu>` je potrebné nahradiť názvom testu, ktorým je meno príslušného JSON súboru bez prípony

Literatura

[1] MDN: Node.js server without a framework. Dostupné z: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Node_server_without_framework

[2] Nottingham, M.; Sayre, R.: The Atom Syndication Format. RFC 4287, Prosinec 2005, doi:10.17487/RFC4287. Dostupné z: https://www.rfc-editor.org/info/rfc4287

[3] OpenSSL: ciphers. Dostupné z: https://www.openssl.org/docs/man1.1.1/man1/openssl-ciphers.html

[4] Ruby, S.: Rss20AndAtom10Compared - atom wiki - intertwingly. Sep 2008. Dostupné z: https://www.intertwingly.net/wiki/pie/Rss20AndAtom10Compared

[5] Warburton, D.: The 2021 TLS telemetry report. Oct 2021. Dostupné z: https://www.f5.com/labs/articles/threat-intelligence/the-2021-tls-telemetry-report