Kyrys Dokumentace

Martin Brakl, Martin Knotek

May 10, 2017

1 Úvod

Tento dokument se zabývá popisem implementace projektu Kyrys, zobrazením základních příkazů a použití. Projekt Kyrys je komunikační program se zaměřením na bezpečný přenos zašifrovaných zpráv mezi uživateli. Popíšeme implementaci, zmíníme základní bezpečnostní analýzu a předvedeme postup pro použití a přeložení programu.

2 Popis implementace

2.1 Použité nástroje

Projekt byl napsán v jazyce C++, s použitím knihoven Qt, MbedTLS a Catch. Překlad byl vyzkoušen s použitím g++5.3 a g++6.2. Verze Qt byla použita 5.8. Projekt je vystavěn pomocí CMake, který umožňuje přenositelnost a dobrou údržbu. Primárně byl spouštěn a testován na systémech Win 7 a Win 8. Zkompilování bylo otestováno i v IDE Clion a přes standartní příkazový řádek. Pro správný chod programu je důležité, aby knihovna Qt obsahovala pluginy SQL, které jsou použity jako databázový systém.

2.2 Struktura

Projekt je i na úrovní zdrojových kódů rozělen na dvě části - server a klient. Následující kapitoly obsahují popis jednotlivých částí.

2.3 Server

Server plní funkci přijímání příchozích připojení od klientů a plnění jejich požadavků. Každé příchozí připojení od klienta je obsluhováno v jednom vlákně, server je tedy konkurentní a schopen obsloužit více klientů najednou.

Vlákna jsou vytvářena pomocí QThread a tato vytvořená vlákna jsou serverem udržována v paměti tak, aby byl server schopen přeposlat zprávu od jednoho klienta druhému.

Požadavky od klienta jsou posílány ve formátu JSON, ve kterém se specifikuje jakou funkci má server vykonat a s jakými parametry. Tyto JSON zprávy jsou posílány přes QSslSocket. Kyrys tedy používá ssl spojení a to tak, že klient i server jsou distribuovány s certifikáty, které ověřují totožnost serveru. V případě nesouhlasu certifikátu na straně klienta a certifikátu poslaného serverem, je spojení odmítnuto.

Zpracování JSON zpráv je vykonáno konečným automatem (v projektu možno najít pod názvem *Resolver*). Tento automat mění svůj vnitřní stav podle příchozích zpráv a poté vykoná akci danou jeho vnitřním stavem.

Jak bylo řečeno výše, server zpracovává požadavky od klientů; mezi takovéto požadavky patří např. registrace nového uživatele, přihlášení uživatele do systému nebo zahájení chatování. Aby se uživatel mohl přihlásit, musí se nejdříve zaregistrovat.

Pro uchování dat uživatelů byla použita SQL databáze, ke které přistupujeme pomocí knihovny Qt. Pro přeložení projektu je potřeba mít Qt distribuováno s touto knihovnou a potřebnými pluginy. V databázi se hesla uchovávají v zahashované podobě a to pomocí algoritmu SHA3_512.

2.4 Klient

Kyrys klient je samostatný program, který spouští uživatel a zadává do něj příkazy. Příkazy od uživatele jsou poté zvalidovány a pokud jsou zadány správně (zadán známý příkaz, při registraci správně zadáno heslo, atp.) jsou tyto příkazy přepsány do formátu JSON a odeslány serveru. Server zprávu zpracuje a podle jejího obsahu vytvoří odpovídající odpověď.

I v klientovi je použito QThread, a sice pro zajištění možnosti zároveň psát i přijímat zprávy. Přijímání zpráv je zajištění pomocí zaregistrování callbacků, které se volají když klient nějakou zprávu přijme. V dalším vláknu potom běží rozhraní, které slouží pro zadávání klientsých příkazů.

Pokud klient posílá zprávu jinému klientovi, je tato zpráva zašifrována pomocí algoritmu AES. Klíče pro dešifrování jsou napevno zadány ve zdrojovém kódu, viz. známé nedostatky

3 Použití Kyrys

Tato kapitola popisuje postup jak nainstalovat a zkompilovat Kyrys a jak jej používat.

3.1 Kompilace

Pro zkompilování potřebujete:

- g++5.3
- Qt 5.8 + qslq pluginy
- CMake 3.0+
- MbedTLS
- Catch
- mingw, popř. jiný nástroj make

Archiv s projektem rozbalte a v kořenové složce projektu v souboru CMakeLists.txt upravte lokaci Qt (řádky obsahující '*QT_INCLUDE_DIR*' a '*CMAKE_PREFIX_PATH*'). Tímto upravíte CMakeLists.txt podle vaší instalace Qt. Cesty, které zadáte musí ukazovat ke stejným souborům jaké jsou již zadány.

V kořenovém adresáři projektu spusťtě příkaz '*cmake . -G "MinGW Makefiles"*'. Tímto nakonfigurujete cmake a vytvoříte makefily pro váš stroj.

Makefily máte vytvořené, můžete se pustit do kompilace: '*mingw32-make <-j N>*', kde '-j N' je nepovinný přepínač, který spustí kompilaci na N jádrech a tím ji zrychlí.

Pokud kompilace proběhla úspěšně, měla by být vytvořená složka bin, která obsahuje certifikáty, dvě dll knihovny a dva spustitelé soubory - klient a server aplikace.

V jednom příkazovém řádku spusťte kyrys Server, měli byste vidět hlášku že server poslouchá na portu.

V jiném příkaz. řádku spusťte klienta a uvidíte uživatelské rozhraní.

3.2 Registrace, login, chat

Pro zaregistrování nového uživatele zadejte příkaz 'register' a postupujte podle pokynů na obrazovce. Pro přihlášení zadejte příkaz 'login' a postupujte podle pokynů.

Pokud se vám povedlo příhlásit, pomocí příkazu 'chat' vstoupíte do chatovacího okna, kde pomocí '#sendTo' můžete poslat zprávu nějakému uživateli, jehož ID musíte znát.

4 Známé nedostatky

Z důvodu časového vytížení jsme několik následujících věcí nestihli dodělat do konce a nachází se proto v rozpracovaném nebo vynechaném stavu:

- Hesla se ukládají v hashované podobě, server nepřidává žádný salt,
- Odeslání zprávy probíha ne zcela dokončeným rozhraním, zejména pomocí funkce *sendTo*, která nemusí být zrovna příjemná pro použití, zejména kvůli nutnosti znát ID příjemce,

- Server i klient obsahují certifikát, který je navržen pro použití na localhostu, nicméně k použití i ve větších sítí, popř. internetu je Kyrys připraven právě díky konceptu přeposílání zpráv přes server, ovšem takové použití by vyžadovalo spuštení Kyrys serveru na stroji s veřejnou IP a vygenerování certifikátu, který obsahuje správné údaje i o tomto stroji a předání takového certifikátu klientům,
- Zprávy, které si vyměňuje klient se serverem neobsahují číslování,
- Pokud komunikuje klient s jiným klientem, jejich zprávy jsou přeposílány serverem. Tyto zprávy jsou zašifrovány, bohužel klíč, který jsme použili jsme museli napevno zapsat do kódu. Tento nedostatek lze vyřešit použitím nějaké obměny algoritmu Diffie-Hellman, při kterém by se oba klienti podíleli na tvorbě šifrovacího klíče. Takto předpokládáme, že klienti (skuteční uživatelé) si klíč zvolí sami a sdělí si jej jinou formou komunikace.

O těchto nedostatcích víme a jejich dokončení je otázka následující práce ve volném čase.

5 Bezpečnostní analýza

V této části popíšeme několik potenciálních útoků na Kyrys a pokusíme se poukázat na řešení v projektu.

5.1 Útok na spojení mezi účastníky

Ztráta zprávy při přenosu sítí Přenos zprávy zo stanice A do B je zajištěn protokolem TCP, který garantuje doručení zpráv.

Zablokování zprávy útočníkem, popř. Man in the middle útok Útočník odchytí zprávu a nepošle ji dál, popř. se vydává za někoho jiného.

Obsah zprávy je šifrovaný, útočník tedy není schopen odchycenou zprávu přečíst, viz dočasná limitace v známé nedostatky.

Spojení mezi klientem a serverem je šifrováno pomocí SSL, což zajišťuje jistou úroveň bezpečnosti, navíc klient je dodán s certifikátem serveru.

DoS útok Možná zranitelnost, není implementována žádna forma obrany?

Replay, reflection útok Možná zranitelnost, není implementována žádna forma obrany?

5.2 Útok na klienta

Snaha získat heslo klienta V tomto případě je nutné aby klient zajistil svou vlastní bezpečnost, bezpečně ukládal a zadával heslo, jeho PC neobsahoval viry atd.

Snaha vydávat se za jiného klienta Klient se při přihlašování prokazuje už. jménem a heslem. Server může přeposlat zprávu pouze na přihlášené uživatele.

5.3 Útok na server

Pokus o převzítí kontroly nad serverem Server běží na nějakém PC, je otázka jak dobře je toto PC zabezpečeno, jak je přístupné, kteří lidé k němu mají přístup a od toho se odvíjí i bezpečnost serveru.

Pokus o ukradnití databáze hesel Hesla jsou uložena v hashovaném formátu. Dočasná limitace viz. známé nedostatky.

Úplná nedostupnost serveru Server může být odstaven DoS útokem.