VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

ZABEZPEČENÍ KOMUNIKACE A OCHRANA DAT V INTERNETU VĚCÍ
SECURE COMMUNICATION AND DATA PROTECTION IN THE INTERNET OF THINGS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Pavel Chadim**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Lukáš Malina, Ph.D.**
SUPERVISOR

BRNO 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Telekomunikační a informační technika**
Ústav telekomunikací

Student: Bc. Pavel Chadim ***ID:*** 146836
Ročník: 2 ***Akademický rok:*** 2017/18

NÁZEV TÉMATU:

Zabezpečení komunikace a ochrana dat v Internetu věcí

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s kryptografickými metodami a knihovnami, které jsou vhodné pro zabezpečení komunikace v Internetu věcí (IoT), kde jsou využité výpočetně a paměťově omezená zařízení. Zhodnoťte vybrané kryptografické knihovny (např. OpenSSL, wolfSSL, micro-ecc). Dále změřte a porovnejte jejich efektivitu a zhodnoťte jejich praktickou aplikovatelnost do systémů IoT. Navrhněte vhodné metody a knihovny pro zabezpečený komunikační scénář v IoT, který bude poskytovat nejen autentizaci entit v IoT a ustanovení šifrovacích klíčů, ale i důvěrnost, autentičnost a integritu přenášených dat mezi prvky IoT. Proveďte konfiguraci a implementaci řešení a určete výpočetní a paměťovou náročnost.

DOPORUČENÁ LITERATURA:

[1] STALLINGS, William. Cryptography and Network Security. 4th edition. [s.l.] : [s.n.], 2006. 592 s. ISBN 0131873164.

[2] MENEZES, Alfred, VAN OORSCHOT, Paul, VANSTONE, Scott. Handbook of applied cryptography. Boca Raton : CRC Press, 1997. 780 s. ISBN 0849385237.

Termín zadání: 5.2.2018 ***Termín odevzdání:*** 21.5.2018

Vedoucí práce: Ing. Lukáš Malina, Ph.D.
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

Abstrakt

Tato Diplomová práce „Zabezpečení komunikace a ochrana dat v Internetu věcí“ se zabývá kryptografií a kryptografickými knihovnami, které jsou mezi sebou porovnány podle podporovaného algoritmu a standardu. K porovnání byly použity následující knihovny: openSSL, wolfSSL, nanoSSL a matrixSSL. Praktická část práce je zaměřena na testování výkonnosti jednotlivých šifer a protokolů z knihoven openssl a wolfssl na zařízení Raspberry Pi 2. Dále se práce zabývá návrhem komunikačního scénáře klient-server v Internetu věcí (IoT). Jednoduchý autentizační protokol klient-server byl implementován a simulován na zařízení Raspberry Pi 2.

Klíčová slova

kryptografie, kryptografické knihovny, Internet věcí, openSSL, wolfSSL, nanoSSL, matrixSSL, TLS, SSL, CoAP, Raspberry Pi 2, klient-server

Abstract

This Master's thesis „Secure communication and data protection in the internet of things“ is dealing with crypthografy and crypthographic libraries, which are compared with eachother according to supporting algorithm and standard. For comparing therewere used following libraries: openSSL, wolfSSL, nanoSSL and matrixSSL. Practical part of the thesis is focused on testing the productivity of each ciphers and protocols of openSSL and wolfSSL libraries on RaspberryPi 2 device. Further, the thesis shows the design of communication scenario client-server in the Internet of Things (IoT). Simple authentication protocol client-server was implemented and simulated on RaspberryPi 2 device.

Keywords

cryptography, cryptographic libraries, Internet of Things, openSSL, wolfSSL, nanoSSL, matrixSSL, TLS, SSL, CoAP, Raspberry Pi 2, client-server

CHADIM, P. Zabezpečení komunikace a ochrana dat v Internetu věcí. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2018. 53 s. Vedoucí diplomové práce Ing. Lukáš Malina, Ph.D..

Prohlášení

Prohlašuji, že svou diplomovou práci na téma „Zabezpečení komunikace a ochrana dat v Internetu věcí“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následku porušení ustanovení § 11 a následujících autorského zákona c.121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne .....

.....

podpis autora

Poděkování

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Lukášovi Malinovi Ph.D. za odborné vedení, konzultace, trpělivost a praktické rady.

V Brně dne .....

.....
podpis autora

SIX research centre
sensor, information and communication systems

Faculty of Electrical Engineering
and Communication

Brno University of Technology
Technicka 12, CZ-61600 Brno, Czechia

http://www.six.feec.vutbr.cz

Výzkum popsaný v této diplomové práci byl realizovaný v laboratořích podpořených projektem Centrum senzorických, informačních a komunikačních systémů (SIX); registrační číslo CZ.1.05/2.1.00/03.0072, operačního programu Výzkum a vývoj pro inovace.

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

2007–13
OP Výzkum a vývoj pro inovace

Tato práce vznikla jako součást klíčové aktivity KA6 - Individuální výuka a zapojení studentů bakalářských a magisterských studijních programů do výzkumu v rámci projektu OP VVV Vytvoření double-degree doktorského studijního programu Elektronika a informační technologie a vytvoření doktorského studijního programu Informační bezpečnost, reg. č. CZ.02.2.69/0.0/0.0/16_018/0002575.

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Projekt je spolufinancován Evropskou unií.

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

Úvod

Kryptografie se v dnešní době používá prakticky ve všech oblastech, které používají počítač nebo internetové připojení. Například platba platební kartou přes internet, internetové bankovnictví, zabezpečený přístup na webové stránky či šifrování souborů. Na internet jsou často zasílána a ukládána velmi důležitá data, která musí být chráněna, aby jich nebylo zneužito. Proto pro úspěšný rozvoj těchto technologií je zapotřebí nezapomínat i na bezpečnost a zabezpečení služeb, které jsou uživateli prostřednictvím internetu poskytovány. U některých služeb (bankovnictví, zdravotnictví) je vyžadován vysoký stupeň zabezpečení a u jiných služeb (osvětlení budovy, termostat) není kladen na zabezpečení takový důraz.

První kapitola se zabývá v dnešní době zmiňovaným pojmem Internet věcí, v anglické literatuře označeném jako Internet of Things (IoT). Termín Internet věcí představuje propojení celé řady zařízení z různých odvětví, která zpracovávají data v lokální síti, ale pro splnění podstaty IoT je data nutné odeslat na centrální server, který je umístěn v internetu. Tato zařízení jsou dnes používána v domácnostech, v průmyslu, ve zdravotnictví, nebo se dají využít i v dalších odvětvích, jako třeba např. chytré křižovatky, které ovlivňují plynulost provozu, nebo pro chytré parkování, kde senzory hlídají volná parkovací místa, využití je v dnešní době spousta. Důvodem, proč se zvyšuje počet těchto zařízení je díky komfortu, který kvůli možné vzdálené správě nabízejí. Dále se kapitola věnuje komunikačním protokolům a to MQTT (Message Queuing Telemetry Transport), CoAP (Constrained Application Protocol), TLS (Transport Layer Security) a DTLS (Transport Layer Security), které jsou v IoT využívány.

V dalších kapitolách jsou popsány jednotlivé kryptografické knihovny a jejich funkce. Porovnání kryptografických knihoven podle podporovaného algoritmu a standardu, například blokové a proudové šifry, hashovací algoritmy a podpora jednotlivých protokolů.

Dále je v práci provedeno experimentální měření kryptografických knihoven na zařízení Raspberry Pi 2, za pomocí nástrojů, které obsahují knihovny od vývojářů. Na těchto knihovnách je změřena výkonnost jednotlivých proudových a blokových šifer a i asymetrických protokolů. Dále je provedeno vlastní měření knihovny wolfSSL pomocí podpory crypto algoritmů.

Poslední kapitola je věnována návrhu komunikačního scénáře v IoT, kde tento scénář je implementován a simulován na zařízení Raspberry Pi 2, kde také proběhne jeho měření. V poslední části práce proběhne zamyšlení, kde by se náš navržený protokol dal použít v IoT prostředí.

1 INTERNET VĚCÍ

Jelikož termín věc vyznívá velmi obecně, je nezbytné jej nejprve vymezit v rámci oblasti IoT stejně jako jiné související pojmy a na jejich základě pak popsat historický kontext, vývoj a aktuální stav této velmi současné oblasti, a rovněž nastínit její perspektivy do budoucna.

1.1 Vymezení pojmu a historický kontext IoT

Internet věcí – Internet of Things (IoT) je pojem, který se začal používat již v roce 1999, kdy ho poprvé použil britský inovátor Evin Ashton [1] v názvu prezentace pro firmu Procter & Gamble [2], v níž se zabýval technologiemi propojení objektů běžného života do Internetové sítě pomocí senzorů [3]. Nutno dodat, že nezávisle na Ashtonovi shodou okolností v témže roce se podobné problematice věnoval také profesor Neil A. Gershenfeld ze známého MIT (Massachusetts Institute of Technology) ve své knize When Things Start to Think [4].

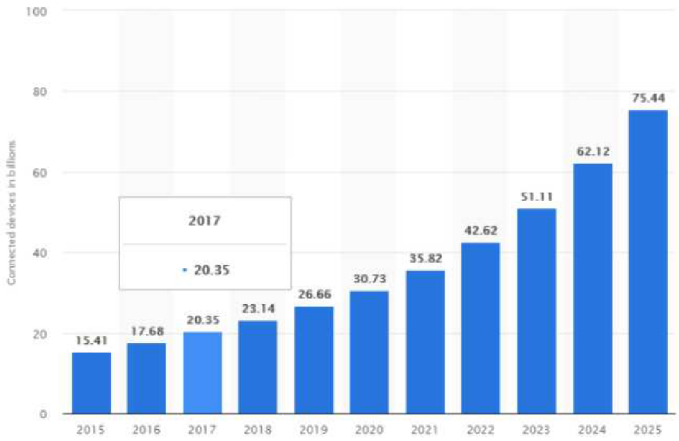
Myšlenka datové komunikace objektů původně nepočítačového světa, jejíž prvotní rozvojové fáze zaznamenaly významnější pokrok spolu s rozvojem bezdrátových komunikačních technologií v 90. letech 20. století, začíná u principu M2M (Machine to Machine), který je možné již považovat za jednoduchou verzi IoT, resp. viděno současnou optikou za jeden z pilířů IoT [5]. Dynamický rozvoj bezdrátové komunikace umožnil vzájemnou komunikaci jednotlivých zařízení na větší vzdálenosti bez nutnosti instalace kabeláže, což také kromě vzdálenosti umožnilo variabilitu umístění a později také mobilitu, což společně velmi razantně rozšířilo možnosti použití jednotlivých zařízení. V tomto období se také započalo využívat IP protokolu ke komunikaci se zařízeními k tomu primárně neurčenými [6]. Objevily se např. první kuchyňské spotřebiče vybavené síťovým rozhraním a identifikací IP, jež mohly být ovládány (zapínány či vypínány) přes Internet.

V rámci jedné z používaných definic lze věci vnímané jako prvky světa IoT definovat jako zařízení, jež využívají inteligentní datová rozhraní, pomocí kterých jsou začleněna do informační sítě, nejčastěji Internetu, přičemž zde vystupují pod vlastní identitou [7]. Takový způsob výkladu tedy vidí IoT jako síťovou infrastrukturu, přičemž klade důraz na její dynamičnost. Jiná definice IoT, kterou používá např. renomovaná americká analytická IT společnost Gartner, ponechává pojem Internet zcela stranou a IoT vidí jako síť fyzických prvků obecného významu s integrovanou komunikační technologií, a tedy schopností poskytovat informace o svém stavu svému síťovému okolí [8]. Zdůrazňuje tak zejména schopnost komunikace objektů IoT. Z výše uvedeného tak vyplývá, že účast věcí v oblasti IoT se zakládá jednak na schopnosti znát a uložit svůj stav a především pak dávat tyto informace k dispozici, a jednak na schopnosti vykonávat rozhodnutí na základě interakce s dalšími věcmi [7]. Pojem síť zde hraje roli komunikačního prostředku v podobě komunikační sítě nebo

brány spojující několik věcí do tzv. cloudu, tedy infrastruktury shromažďující a ukládající získaná data (např. vzdálené servery v datových centrech). Není tomu vždy, v některých případech data končí na lokálním serveru, nebo se data vyměňují mezi uzly pro jejich lepší funkčnost. Pojem IoT se také používá pro označení vestavěného (angl. embedded) zařízení připojeného k Internetu pomocí kabelového, bezdrátového připojení WLAN (Wireless Local Area Network), WPAN (Wireless Personal Area Network), WMAN (Wireless Metropolitan Area Network), WAN (Wide Area Network) nebo datového připojení mobilní sítě typu GSM (Global System for Mobile communication) apod. Specifickou vlastností těchto zařízení je schopnost prostřednictvím Internetu sdílet data, ať již jednosměrným způsobem ve smyslu pouze předávání informace z koncového zařízení (např. senzoru) do řídicího systému, nebo vzájemnou interakcí mezi oběma uzly [7].

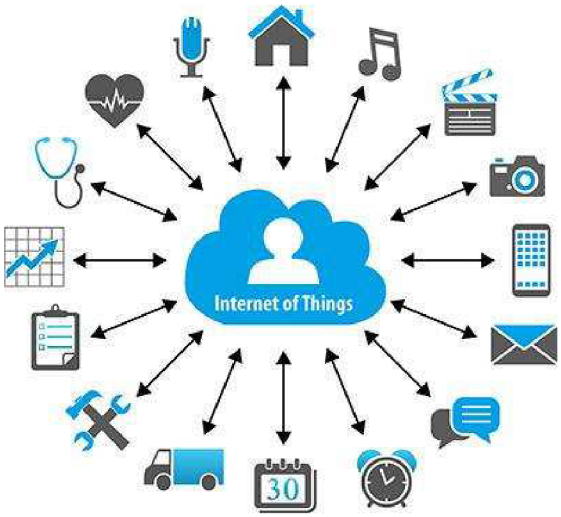
1.2 Současnost, vývoj a využití IoT

Počet fyzických objektů připojených k Internetu vzrůstá v dnešní době nevídaným tempem. Studie z roku 2011 týkající se sektoru IoT vydaná renomovanou IT společností Cisco názorně ilustruje dřívější předpoklady dynamického rozvoje a rozšíření technologií IoT když uvádí, že zařízení spadající do sektoru IoT početně přesáhla lidskou populaci na planetě již v roce 2008 a po roce 2010 již byl počet zařízení IoT vzhledem k počtu obyvatel na Zemi bezmála dvojnásobný (v roce 2010 na jednoho obyvatele Země připadalo v průměru 1,84 zařízení) [9]. Předpověď do roku 2020 v rámci této studie hovořila až o čísle 50 miliard jednotek a Intel byl v roce 2015, tedy před necelými třemi lety, ve svých odhadech dokonce ještě optimističtější když uváděl, že rok 2020 bude pro svět IoT znamenat až 200 miliard zařízení. Dle současných čísel se tento předpoklad jeví jako poněkud přehnaný, neboť údaje z konce loňského roku hovoří o počtu přes 20 miliard připojených zařízení, v roce 2020 pak má toto číslo mírně přesáhnout 30 miliard kusů a hranice 50 miliard zařízení v sektoru IoT má být překonána během roku 2023 [10]. Nicméně prudce rostoucí počet IoT zařízení s dokonce přibližně exponenciální růstovou tendencí je dobře patrný z grafu vývoje na Obr. 1.1[10].



Obr. 1.1: Statistika skutečného stavu do roku 2016 s předpokladem do roku 2025 [10]

Systémy HVAC (Heating, Ventilating and Air Conditioning; česky vytápění, ventilace a klimatizace) obecně sloužící k monitorování a role systémů v chytrých domácnostech jsou typickým příkladem rostoucího využití IoT v běžných podmínkách. Další oblasti, kde lze IoT využít jsou např. doprava, zdravotnictví, průmyslová automatizace či nouzové reakce na přírodní nebo člověkem způsobené katastrofy, kdy je složité situaci vyhodnotit lidským faktorem [11].



Obr. 1.2: Možnosti IoT převzato z [11]

Na obrázku 1.2 je názorně ilustrováno, jak široké spektrum oblastí v současné době fenomén IoT postihuje. Člověk, který vlastní komunikační zařízení (chytrý telefon, tablet, laptop), je schopen pomocí příslušné aplikace a prostřednictvím sítě Internet monitorovat, komunikovat či ovládat (zapnout či vypnout) dříve naprosto samostatně a odděleně fungující zařízení v domě, jako je např. vytápění a osvětlení [11].

Lze nalézt mnoho dalších současných příkladů využití IoT a lze očekávat, že aplikací bude do budoucna rapidně přibývat zejména co do aktivního propojení domácností se sektorem obchodu a služeb. V souvislosti s IoT je často uváděn koncept tzv. inteligentních domácností a v širším pojetí pak dokonce i inteligentních měst. Jako jednoduchý příklad z prostředí domácnosti může posloužit chytré záplavové čidlo poblíž pračky či myčky, které při detekci úniku vody předá informaci řídicímu systému, jež přes připojené aktuátory zajistí zavření hlavního přívodu vody a případně ještě předá informaci majiteli objektu a kontaktuje servisní firmu. Lze si představit také lednici vybavenou systémem senzorů v rámci IoT zapojenou do Internetu, jež registruje jednotlivé položky svého obsahu. Nejenže může upozornit na chybějící oblíbené potraviny nebo blížící se konec záruční doby jednotlivých položek, ale může také s ohledem na často používané zboží upozornit na aktuální akční slevy v daném obchodním řetězci, může sama provádět objednávky zboží, online platby a koordinovat dodávky zboží tak, aby sortiment uživatelem definovaných potravin byl neustále k dispozici. Podobně může lednice sama vzdáleně upozornit na nárůst teploty a možnou

závadu a sama si zavolat servis. Princip predikce závad a předobjednání servisu před výpadkem se uplatní zejména ve výrobě, kde znamená každý výpadek finanční ztrátu. Stejně tak na jednoduchém principu lednice může fungovat zajištění skladových zásob a s tím spojené logistiky. Příkladů na efektivní využití autonomních jednotek propojených přes Internet do spolupracujícího komplexu lze vymyslet nepřeberné množství [11]. IoT lze využít také ke vzdálenému monitorování zdravotního stavu sledovaného pacienta, k zabezpečení objektů, a mnoho dalších způsobů využití. Další z mnoha příkladů týkající se současného problému opakujícího se sucha může být zařízení pro měření objemu srážek, které na dálku předává informace o množství srážek v dané oblasti a dle změřené skutečnosti řídí závlahové režimy.

Lze říci, že hlavním faktorům, jež umožňují současnou prudkou akceleraci rozmachu IoT, bezpochyby patří velmi dobrá konektivita, tedy snadný přístup k vysokorychlostnímu Internetu. Také rozvoj technologických procesů miniaturizace elektronických součástek a obvodů, zejména mikroprocesorů, umožňuje vývoj a produkci velmi malých, energeticky nenáročných a cenově dostupných zařízení, která se právě tak často využívají v aplikacích oblasti IoT [12]. Rovněž vzestup cloud computingu v posledních letech představuje důležitý faktor pro efektivní implementaci IoT [13]. Cloudová architektura totiž poskytuje pro IoT nanejvýš vhodnou platformu pro spravování dat, jelikož agregace a zpracování dat na jednom místě je velmi důležitým aspektem IoT. Agregaci umožňuje také rapidní nárůst kapacity a nová efektivní řešení datových úložišť a současně také významný pokrok v analýze dat.

1.3 Důvody pro řešení zabezpečení IoT

Prudký rozvoj technologií způsobující velmi dynamický průnik fenoménu IoT do prakticky všech oblastí lidského života, které zmiňuje a na příkladech ilustruje předchozí kapitola, znamená také ruku v ruce s pokrokem jdoucí vyšší bezpečnostní riziko. Více zařízení online znamená také více možných bran pro neoprávněný průnik do internetového světa a potažmo vzdáleně jeho prostřednictvím také penetraci do jednotlivých komponent v něm propojených. Znamená to také nárůst nároků na důslednost zabezpečení. Zvláště pak nabývá tato myšlenka na důrazu v případě IoT, kdy se často jedná o zařízení jednoduchá s nepříliš valnou dispozicí systémových prostředků (výkon procesoru, kapacita pamětí apod.), představující jednu z nativních vlastností prvků IoT [13]. Je zřejmé, že nevelká kapacita také neposkytuje v rámci šetrného nakládání s těmito prostředky dostatečný převis pro implementaci náročných bezpečnostních procedur, což může tato zařízení učinit zranitelnými.

Tento efekt je dále umocněn ještě tím, že zvyšování či minimálně udržování a aktualizace úrovně zabezpečení, které se v případě běžných počítačů a chytrých telefonů odehrává v intervalech řádu měsíců (aktualizace) či let (výměna za nový model) se v případě IoT často

neuplatňuje, neboť senzorické technologie, vybavení chytrých domácností apod. je obvykle kalkulováno na mnohem delší dobu provozu a aktualizace často neprobíhají vůbec. Celkový poměr nezabezpečených zařízení se do budoucna zřejmě sníží jen s obtížemi, či vůbec [13].

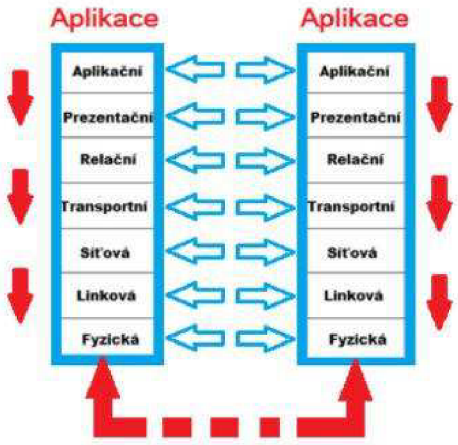
Internet of Things může pro potenciálního útočníka (hackera) znamenat způsob, jak snadno získat osobní údaje či jiná důvěrná data (obvyklý denní režim a zvyklosti osob či celých společností apod.) využitelná pro podporu další kriminální činnosti nebo pouze ke zdánlivě nevinnému efektivnímu cílení komerčních aktivit. Mnohem horší verzí hrozby průniku útočníka do prvku IoT je ve srovnání s přístupem k datům další úroveň, a tou je získání kontroly nad zařízením. U těch nejjednodušších zařízení na bázi senzorů to může znamenat snaha ovlivnění vyšších systémů zfalšováním a podstrčením dat (např. systém zavlažování se spustí, pokud senzory falešně tvrdí, že je sucho), ale u složitějších systémů IoT vybavených aktivními i řídicími úlohami hrozí i jejich zneužití prostřednictvím získání kontroly nad nimi (průnikem a ovládnutím řídicí jednotky dopravního prostředku může hacker zapříčinit dysfunkci brzdového systému či automatické navigace). Lze shrnout, že v oblasti neoprávněné manipulace s daty mohou útočníci s daty v zásadě provést tři základní úkony, a sice odcizit je, měnit je nebo zabránit oprávněnému vlastníkovi v přístupu k nim. Právě poslední dva jmenované způsoby se s příchodem IoT mohou podle mnoha odborníků stát extrémně účinné. Ještě závažnější potenciální hrozbou je pak fakt, že dobrá a stále se zlepšující konektivita současně posiluje schopnost entit IoT stát se pro útočníka zadní bránou (odborně označovanou anglicky jako backdoor) pro průnik do rozsáhlejších sítí a skutečně významných uzlů globální sítě Internet. Uváženy do důsledků jsou tedy systémy, které jsou vzájemně propojené, více či méně ohrožené v principu všechny současně a útok proti jednomu segmentu systému ohrožuje všechny další části, přičemž s narůstajícím počtem zařízení toto riziko z podstaty věci narůstá. To je posíleno již zmíněným faktem, že vysokou dynamikou nárůstu se vyznačuje právě segment zařízení IoT s omezeným stupněm zabezpečení. Účelné a také systémové řešení zabezpečení sektoru IoT je k udržení a zvyšování globální bezpečnosti prostředí Internetu naprostou nezbytností [13].

1.4 Vrstvová struktura IoT

Pro účely znázornění fungování počítačové sítě a zejména jejích hardwarových komponent, a také pro demonstraci a rozdělení úloh jednotlivých segmentů od operačních systémů a aplikací až po samotný datový přenos v kabeláži či v rádiovém prostředí se s výhodou používají tzv. vrstvové modely sítě.

1.4.1 Referenční model ISO/OSI

Základní vrstvový model sítě, známý zejména jako Referenční model ISO/OSI, zahrnuje veškeré úpravy digitálních dat až po fyzické propojení konkrétních harwarových prvků [14]. Celý název tohoto standardu zní Reference Model of Open Systems Interconnection a v praxi se obvykle označuje zkratkou názvu jako RM OSI nebo jen ISO/OSI [15]. Model zahrnuje sedm vrstev a základní myšlenka spočívá v tom, že každá z vrstev zahrnuje sadu svých protokolů, jejichž prostřednictvím komunikuje s vrstvou stejné úrovně jiného uzlu, což představuje jakousi formu horizontální komunikace. V rámci vertikální mezivrstvové komunikace v jednom zařízení pak daná vrstva komunikuje s vrstvou buď sousední nižší, nebo sousední vyšší, přičemž nižší vrstvy vždy poskytují služby bezprostředně vyšší vrstvě a tato vyšší vrstva využívá dle potřeby služeb vrstvy nižší. Průběh komunikace v síti na základě vrstvového modelu je patrný z Obr. 1.3.

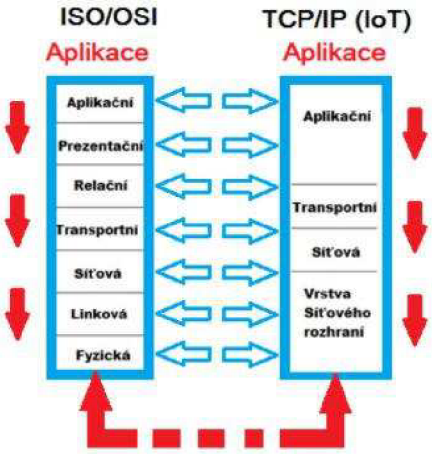


Obr. 1.3: Struktura a princip komunikace dle referenčního modelu ISO/OSI

Data z aplikace na jedné straně komunikace prostupují do aplikační vrstvy, ta je zpracuje v souladu s některým z protokolů a předá data dál nejbližší nižší vrstvě. Takto se data propadají vrstvami až k vrstvě fyzické, která představuje skutečné přenosové médium (kabel, rádiový signál atd.) a dojde k přenosu k druhé straně komunikačního řetězce. V cíli pak data prostupují vrstvami směrem vzhůru a každá z vrstev z příchozího datového balíčku rozpoznává a zpracovává informace, které zapadají do některého z jejích protokolů, a zbylá data předává dále vyšší vrstvě až k aplikačnímu softwaru. Právě toto přebírání informací v rámci příslušných vrstev přestavuje pomyslnou horizontální komunikaci, která je na Obr. 1.3 znázorněna modrými šipkami, zatímco skutečná komunikace je vyjádřena šipkami červenými. Nutno doplnit, že norma referenčního modelu ISO/OSI nespecifikuje žádné konkrétní protokoly, ani neobsahuje žádné definice přesného rozhraní mezi jednotlivými vrstvami modelu [14]. Model pouze obecně specifikuje jednotlivé vrstvy a rámcově shrnuje soubor úkolů, které těmto vrstvám přísluší řešit. Protokoly a služby pro každou z vrstev vznikají až dodatečně jako samostatné standardy [14].

1.4.2 Vrstvový model IoT

Pro účely prostředí sítí používajících protokolové sady TCP/IP, což je také případ prostředí Internetu a tedy i oblasti IoT, se obecný model ISO/OSI modifikoval a zjednodušil na míru komunikačního schématu na pouhé čtyři vrstvy (Obr. 1.4) [14].



Obr. 1.4: Vrstvový model TCP/IP vedle klasického modelu ISO/OSI

V rámci modelu TCP/IP tři nejvyšší vrstvy ISO/OSI splynuly v jednu vrstvu aplikační, transportní a síťová vrstva zůstaly osamostatněny a zbývající dvě nejnižší vrstvy se proměnily v jednu vrstvu síťového rozhraní. Takto zjednodušený model lépe odpovídá principům fungujícím v rámci rodiny TCP/IP, tedy třeba klasické webové služby, ale také velmi dobře koresponduje s komunikačními scénáři zařízení IoT [14].

Aplikační vrstva

Vrstva obsahuje protokoly nejpoužívanějších služeb v rámci rodiny TCP/IP, tedy klasické internetové protokoly HTTP (Hyper Text Transfer Protocol), FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol) a další, ale také celá řada dalších protokolů, které byly vytvořeny speciálně na míru pro konkrétní oblast [16]. Výjimkou není ani sektor IoT, který využívá celou řadu speciálních protokolů, jimž je samostatně věnována kap. 1.5.

Transportní vrstva

Vrstva má za úkol vytváření logického spojení na úrovni dvou komunikujících bodů a adresování datového toku na úrovni aplikací [14]. Spadá sem především protokol TCP (Transmission Control Protocol), který reprezentuje spojově orientovanou neboli spolehlivou, a tedy potvrzovanou službu. Navazuje spojení pouze za podmínky dostupnosti celé přenosové trasy a strana příjemce vždy kontroluje a potvrzuje straně odesilatele korektní přijetí datových

segmentů. Na úrovni protokolu TCP se používá termín soket (tedy zásuvka, konektor), který je v zásadě tvořen IP adresou koncového uzlu a číslem portu. Protokol TCP navíc umožňuje řízení přenosové rychlosti dle kapacity odesilatele a adresáta. Potvrzované spojení TCP je sice náročnější na realizaci a také je často díky potvrzování apod. pomalejší, je však nepostradatelné např. při přenosu souborů, kde je nutností bezchybný přenos celého datového bloku. Druhý z významných protokolů transportní vrstvy UDP (User Datagram Protocol) transportuje datové bloky bez potvrzení z opačné strany, bez ověřování přenosové trasy a bez řízení toku dat. Díky nízkým režijním nárokům a obecně vyšší rychlosti se uplatňuje např. přenosu streamovaného videa, internetové IP telefonie a obecně tam, kde případná ztráta malé části informace nehraje významnou roli. Použití spolehlivé, či nespolehlivé služby, tedy TCP či UDP, určuje především charakter aplikace, resp. služby, která tyto protokoly pro svou práci využívá. Nejsou výjimkou případy, kdy jedna softwarová aplikace poskytující několik různých služeb může využít dle typu požadavku TCP, nebo UDP. Protokoly aplikační vrstvy používané v oblasti IoT využívají ve větší míře nepotvrzovanou službu UDP, ovšem TCP je u některých z nich využito také (viz kap. 1.5), opět samozřejmě v závislosti na typu služby [14].

Síťová vrstva

Síťová vrstva se v rámci TCP/IP obsahuje především protokol IP (Internet Protocol) [14]. Protokol obstarává zejména směrování datových paketů na úrovni sítě, tedy na základě IP adresy v hlavičce každého z rámců zajišťuje procházení dílčími uzly sítě směrem k adresátovi. V současnosti stále nejvíce používané IP adresy verze 4 (Ipv4) mají délku 32 bitů (čtyři osmice obvykle číselně dekadicky vyjádřené čtveřicí trojciferných čísel oddělených tečkou). Nicméně množství kombinací těchto čísel, potažmo takto vytvořených adres, již současnému množství zařízení v prostředí IP nedostačuje, proto vzniklo řešení adres o čtyřnásobné délce, tedy 128 bitů. Tato verze nese označení Ipv6 a další adresní prostor takto získaný čítá $3{,}4 \times 10^{38}$ unikátních adres, a je tedy dostatečně velký, aby pokryl poptávku po unikátní identifikaci v rámci IP na velmi dlouho dopředu [14].

Vrstva síťového rozhraní

Tato vrstva postihuje skutečnou fyzickou podobu přenosové cesty a její konkrétní podoba tedy závisí na typu použité sítě. Paleta možností je velmi pestrá a dle použité technologie může mít tato vrstva u dvou zařízení zcela odlišnou fyzickou konfiguraci [6]. Základní a obecně v rámci TCP/IP stále nejpoužívanější je kabelová síť Ethernet, ale díky již výše zmíněnému dlouhodobě, dynamicky a trvale se rozvíjejícímu oboru rychlých bezdrátových sítí se velmi výrazně prosazují sítě WiFi (Wireless Fidelity), vedle klasického GSM (Global System for Mobile Communications, původně Groupe Special Mobile) také další mobilní technologie, jako CDMA (Carrier Division Multiple Access), LTE (Long Term Evolution),

WiMAX (Worldwide Interoperability for Microwave Access), ale i další bezdrátové technologie přímo na míru určené pro sektor IoT [6]. Pro použití v oblasti IoT se pochopitelně hodí v drtivé většině případů bezdrátové verze fyzického rozhraní, které umožňují snadnou a rychlou instalaci (odpadá nutnost instalace kabelů) a částečnou mobilitu v závislosti na konkrétním bezdrátovém řešení. Ovšem zásadní úskalí plynoucí ze samotného principu radiové komunikace je fakt, že přenos probíhá volným prostorem obecně přístupným komukoli, kdo se vyskytne v rádiovém dosahu. Především právě z tohoto důvodu je nutné důkladně řešit zabezpečení přenosu proti neoprávněnému přístupu [6].

Nicméně u zařízení, u nichž se z nějakého důvodu počítá s umístěním déle na jednom místě (silové napájení, velké rozměry či hmotnost, nutnost speciální stavební připravenosti atp.), jako jsou např. výrobní linky, pokladní či bránové systémy, z domácnosti např. kotle, pračky nebo myčky nádobí, lze v zásadě bez následných komplikací využít k připojení do sítě klasického kabelu. Nicméně v případě použití bezdrátové technologie při případném přemístění zařízení minimálně tento problém odpadá. Je také třeba myslet i na možnost změny konfigurace přenosové sítě jako takové. Z těchto důvodů se bezdrátové technologie často využívají i u stacionárních zařízení. U malých, přenosných či dokonce mobilních zařízení nezávislých na kabelovém napájení a dalších pevných technologiích je pak pochopitelně bezdrátová technologie naprostou nezbytností [17].

K tradičním řešením vrstvy síťového rozhraní používané pro IoT se řadí především ta, jež původně sloužila k připojení počítačů a jejich příslušenství. Patří sem zejména technologie WiFi a Bluetooth (pro IoT byla upravena speciální verze s nízkou energetickou náročností BLE - Bluetooth Low Energy). Pro připojení ve smyslu IoT se také běžně používá technologie GSM, která prapůvodně sloužila pouze pro účely mobilní telefonie, ale s rozvojem datových přenosů v této mobilní síti a nových technologických generací GSM (3G a 4G) se tyto mobilní technologie staly součástí také IoT [17]. Speciálně pro oblast IoT vedle toho vznikly i další bezdrátové protokoly, které se úzce profilují směrem k charakteru této kategorie zařízení. Sem patří mj. technologie Zigbee, Z-Wave, LoRa či Sigfox.

Zigbee – představuje průmyslový standard pro bezdrátové sítě pro pásmo 2,4 GHz určený pro aplikace, jimž postačují datové přenosy s nízkou četností a malým objemem přenášených dat (maximálně desítek kbit/s) v rámci dosahu do 100 m, např. uvnitř budov [18]. Výhody ZigBee spočívají především v jednoduchosti protokolu a v minimálních nárocích na hardware i energetickou spotřebu. Hlavní použití nalézá v oblasti senzorů, monitorovacích systémů nebo regulátorů v průmyslu. V rámci domácností a koncových zákazníků je to např. v sektoru spotřební elektroniky. Vzhledem k charakteru přenosového protokolu zde není z hlediska bezpečnosti takovou hrozbou fenomén backdoor, nýbrž spíše riziko zneužití či zfalšování dat jdoucích od senzoru k řídicí jednotce [18].

Z-Wave – pracuje ve vyhrazeném frekvenčním pásmu 900 MHz, v otevřeném prostoru vykazuje dosah až 100 m, v budově pak zhruba 30 až 50 m [19]. Rychlost přenosu je maximálně 100 kbit/s. Standard slouží především pro potřeby domácích zařízení a domácí automatizace (např. chytré termostaty a obecně chytré domy). Jeho hlavní výhodou je možnost snadného vzájemného propojení zařízení různých výrobců a také fakt, že na rozdíl od WiFi či Bluetooth vykazuje díky použitému frekvenčnímu pásmu vysokou odolnost vůči elektromagnetickému rušení [19].

Sigfox – jde v tuzemsku o velmi významnou síť francouzské provenience patřící do kategorie LPWAN (Low-Power Wide Area Network), která v sobě snoubí mobilní přenos dat na vzdálenosti v řádech km a energeticky úsporný provoz [20]. Sigfox využívá úzké frekvenční pásmo z bezlicenční části spektra, konkrétně v Evropě jde o pásmo 868 MHz a v USA je to pásmo 915 MHz [21]. Síť zakládá na hvězdicové topologii, kdy mezi dvěma stanicemi existuje vždy jen jediný spoj. Pro svůj provoz a koordinaci síť vyžaduje mobilního operátora. Základní koncepci nízké spotřeby při dlouhém dosahu je přizpůsobeno celé fungování sítě, zejména koncová zařízení. Ta vysílají maximálně 12 bytů v rámci jedné zprávy, a to ještě maximálně 140 cyklů za den. Úzké pásmo a vhodné kódování umožňují přenos až na vzdálenost 50 km [21]. Také příjem dat ze sítě obvykle probíhá specifickým způsobem: zařízení nejsou připojena do sítě trvale, ale pouze v určitých časových intervalech, např. po odeslání zprávy, přičemž mohou přijmout od přístupových bodů sítě denně maximálně 4 zprávy velikosti 8 bajtů. Hvězdicová topologie je založená na buňkovém principu, kde základnové stanice pokrývají svým dosahem plochu buňky [21]. V současné době síť Sigfoxu již funguje, nebo se buduje v mnoha zemích světa, přičemž fungující sítě lze nalézt převážně v Evropě. V rámci ČR síť nalézá využití např. v dálkovém odečtu spotřeby energií, zařízení pro chytrou zemědělskou výrobu, jako jsou jednotky pro monitoring teploty, vlhkosti a srážek [22]. K dalším aplikacím mohou patřit kouřová čidla, detektory pohybu či polohy, senzory určující obsazení či neobsazení parkovacího místa na exponovaném parkovišti apod. Zabezpečení přenášených dat přes rádiové rozhraní mezi koncovými zařízeními a přístupovými body zabezpečuje unikátní klíč, kterým zařízení signuje odeslaná data, čímž je možné detekovat úmyslně falšovanou či nesprávně přijatou zprávu.

LoRa – další v tuzemsku velmi zajímavou technologií pro IoT je LoRa, přesněji LoRaWAN [23]. Jedná se o přímou konkurenci sítě Sigfox, a tak nepřekvapí, že mezi stěžejními vlastnostmi lze najít nejednu podobnost. Kromě komunikace na velké vzdálenosti podporuje LoRaWAN také oboustrannou komunikaci (potvrzení zpráv, příkazy, aktualizace firmwaru či nastavení apod.). V rámci kategorie IoT nechybí ani velmi nízká spotřeba energie díky technice probouzení zařízení jen pro vysílání zprávy apod. Síť LoRaWAN využívá pro své fungování dokonce i stejné kmitočtové pásmo, jako technologie Sigfox, tedy v Evropě

868 MHz, v USA pak 915 MHz. V Evropě je v rámci pásma vytvořeno 8 přenosových kanálů [23]. Pro identifikaci zařízení v rámci sítě LoRa se používá unikátní identifikační číslo spolu s dvojicí digitálních klíčů, které lze pevně přidělit při výrobě, nebo pomocí aktivace na dálku, kdy centrální řídicí server sítě tyto informace zařízení přidělí [23]. Klíče o délce 128 bitů se pak využívají pro zabezpečení rádiové datové komunikace, jež je zajištěna šifrou AES. Jeden z klíčů slouží pro šifrování přenosu mezi zařízením a přístupovým bodem, druhý je určen pro zabezpečení komunikace mezi sítí a uživatelskou aplikací. Data získaná z koncových zařízení se shromažďují na cloudovém úložišti provozovatele sítě LoRa, k němuž pak mohou přistupovat jednotliví uživatelé. Scénáře nasazení se také v zásadě shodují se sítí Sigfox, tedy vzdálený odečet spotřeby vody či plynu, sledování polohy movitého majetku atd.

1.5 Protokoly aplikační vrstvy IoT

Jak již bylo zmíněno v předchozích kapitolách IoT zahrnuje širokou škálu možných protokolů aplikační vrstvy sloužících pro přenos dat od jednotlivých entit k cílovému prvku. Tyto protokoly většinou nebyly primárně vyvíjeny pro Internet věcí, ale jsou odvozeny od protokolů telekomunikačních již dříve vytvořených pro klasické TCP/IP struktury. Jejich hlavním znakem charakteristickým právě pro IoT je přenos malého objemu dat (řádově desítek až stovek bitů), a to navíc v nepravidelném režimu (často také pouze na požádání) z důvodu maximálního snížení energetických nároků. Také je třeba zde často počítat s nižší kapacitou paměťových jednotek a samozřejmě oproti standardním zařízením nižšímu výpočetnímu výkonu mikroprocesorů. Ačkoli samozřejmě nelze v rámci IoT vyloučit použití klasických aplikačních protokolů, je jejich implementace u zařízení IoT převážně nepříliš obratným řešením, protokoly jsou často příliš robustní či režijně i energeticky neúsporné. Proto se jeví jako nanejvýš vhodné vytvořit protokoly na míru přizpůsobené aplikacím typickým pro IoT, a také těchto protokolů vznikla postupně celá řada. Několik vybraných z těchto protokolů představují následující kapitoly [17].

1.5.1 MQTT

V případě MQTT (Message Queuing Telemetry Transport) se jedná o komunikační protokol vyvinutý společností IBM v roce 1999 právě pro méně výkonná zařízení s omezenou výpočetní kapacitou pracující v sítích s omezenou propustností a vyšší latencí. Je tedy přímo určený pro provozování rozlehlých sítí s množstvím malých zařízení, kterých provoz je řízen serverem v síti Internet. Pro co protokol naopak určený a vhodný není, je přímé propojení jednotlivých zařízení bez centrálního řízení [24]. Hlavními výhodami protokolu MQTT jsou jednoduchost a snadná implementace. Používá se např. v lékařské technice (komunikace s

kardiostimulátory). Není bez zajímavosti, že se s tímto protokolem lze nepřímo setkat také v mobilní aplikaci Facebook Messenger. V dnešní době patří tento protokol mezi dva nejpoužívanější protokoly (spolu s CoAP) v sektoru Internetu věcí [25].

1.5.2 CoAP

Protokol CoAP (Constrained Application Protocol) představuje jedno z nejnovějších z protokolů aplikační vrstvy na míru určených pro provoz na zařízeních z oblasti IoT, tedy vyžadujících implementaci režimu nízké spotřeby energie a poskytujících pouze omezené výpočetních a komunikačních kapacity. Protokol vytvořený pod hlavičkou společnosti IETF (Internet Engineering Task Force) vychází z protokolu aplikační vrstvy HTTP, který na úrovni transportní vrstvy ovšem využívá nespolehlivý transportní protokol UDP (na rozdíl od klasického HTTP, které využívá spolehlivý protokol TCP). CoAP koncepčně staví na architektuře REST (Representational State Transfer), která umožňuje klientům a serverům vystavit a používat také webové služby s využitím REST interakcí (metody GET, POST, PUT a DELETE) [11], [26].

CoAP tedy umožňuje obecně jednoduchým zařízením z oblasti IoT komunikaci prostřednictvím sítě Internet jako páteřní infrastruktury, ovšem také na rozdíl od předchozího protokolu MQTT je přímo použitelný pro vytvoření infraktruktury bez centrálního řídicího prvku na páteřní síti, tedy pro přímé propojení dvou a více zařízení mezi sebou. Své použití spatřuje CoAP v zařízeních typu různých senzorů, přepínačů či ventilů, u nichž nastává nutnost dálkového monitoringu a řízení. Pro zabezpečení provozu v rámci CoAP protokolu se využívá DTLS (Datagram Transport Layer Security), který kromě jiného řeší problematiku správného řazení zpráv příchozích v nesprávném pořadí či ztrátovost či chybný přenos paketu.

1.5.3 TLS

Protokol TLS (Transport Layer Security) spolu se svým předchůdcem SSL (Secure Sockets Layer) patří do skupiny kryptografických protokolů. V rámci vrstvového modelu TCP/IP nepatří čistě do aplikační vrstvy, nýbrž se pohybuje na pomezí mezi protokolem TCP transportní vrstvy a aplikační vrstvy. TLS představuje účinnou obranu proti falšování či odposlouchávání zpráv, zajišťuje vzájemnou autentizaci koncovým bodům, která vychází z certifikátů (server, klient), dále je povinná autentizace serveru a klient má možnost volby. Protokol se zabývá důvěrností a integritou dat. Symetrické šifrování, které předchází dohodě uživatelů na podporovaných formátech algoritmů je založeno na veřejném klíči, jenž uživatelé získají výměnou pomocí svých vlastních klíčů [27].

1.5.4 DTLS

Protokol DTLS (Datagram Transport Layer Security) byl specifikován společností IETF a má, stejně jako TLS, dvě podvrstvy (Record Protocol a Handshake Protocol), které zastávají stejné funkce, jako v případě TLS [28]. Nicméně na rozdíl od TLS není protokol DTLS spojově orientovaný, neboť pro přenos využívá protokol UDP. Odpadá tedy nutnost navazování spojení end-to-end a tím odpadá i potřeba vysílání zpráv pro sestavení tohoto spojení, čímž nevzniká nadbytečná režie a z ní plynoucí spotřeba energie. DTLS je tedy vhodnější pro sítě s požadavkem na nízkou spotřebu, kde se toleruje určitá ztrátovost. Jako asymetrický šifrovací algoritmus se ve standardu používá RSA [28].

Nutno doplnit, že kromě několika výše zmíněných datových protokolů aplikační vrstvy se běžně používá celá řada dalších specifických protokolů, mezi jinými např. XMPP (Extensible Messaging and Presence Protocol), AMQP (Advanced Message Queuing Protocol), DDS (Data-Distribution Service for Real-Time Systems), LWM2M (Lightweight M2M) nebo SSI (Simple Sensor Interface) [29].

2 KRYPTOGRAFICKÉ KNIHOVNY

Knihovna je soubor funkcí, procedur, datových typů a zdrojů, které se využívají při programování. Tato práce se zaměřuje na oblast kryptografie a věci s ní spojené. S knihovnami pracujeme pomocí jejich API, které nám reprezentuje názvy funkcí, předávané parametry a návratové hodnoty spolu s jejich popisem.

Vybrali jsme tyto knihovny, které jsou popsané níže. Vybrali jsme je podle toho, že podporují námi testované kryptografické šifry a funkce spolu s jejich snadnou a podporovanou implementací. Nakonec jsou všechny tyto knihovny stále vyvíjeny a aktualizovány vydavateli nebo odbornou komunitou.

2.1 OpenSSL

OpenSSL je svobodná, multiplatformní implementace protokolů SSL a TLS a běžné kryptografické funkcionality. Na knihovnu OpenSSL se stejně jako na GnuTLS nevztahují omezení exportu, jedná se o Evropský (původně Australský) projekt. Knihovna používá duální licencování, je dostupná pod licencemi OpenSSL license a SSLeay license[30].

Tato knihovna původně vznikla v roce 1995 jako projekt pánů Erica A. Younga a Tima J. Hudsona s názvem SSLeay. Knihovna byla šířena pod licencí typu Apache / BSD a obsahovala obsáhlou dokumentaci. Její vývoj skončil v roce 1998, kdy tyto dva pánové odešli pracovat do firmy RSA Security. Následně se knihovna přejmenovala na OpenSSL ale typ licence zůstal nezměněn, proto je nyní projekt spravovaný celosvětovou komunitou lidí. Nevýhodou této situace je to, že se knihovna rozvíjí pomalu a její dokumentace spolu s API není obsáhlá, místy až nulová [30].

Jak již název vypovídá, primární funkce knihovny je implementace protokolu SSL (vrstva bezpečnostních soketů - Secure Sockets Layer) a TLS (zabezpečení transportní vrstvy - Transport Layer Security) ale obsahuje i další kryptografické funkce. Jako základ využívá jazyk C, proto se nejčastěji používá v systémech Unix, Windows a MacOS, ale jde ji využít i v jazyce JAVA. Podporuje všechny standardní kryptografické algoritmy, od hašovací funkcí MD5 a SHA přes symetrické šifry typu DES a AES až po asymetrické zastupující nejčastěji používané RSA [30].

Funkce

- SSL 3.0, TLS 1.0, 1.1 a 1.2
- DTLS 1.0 a 1.2
- Hashovací algoritmy
    - MD5, SHA-1, SHA-2, SHA-256, SHA-384, RIPEMD-160, MDC-2, BLAKE2
- Symetrické šifry
    - AES, DES, Blowfish, Camellia, SEED, CAST-128, IDEA, RC2, RC4, RC5
- Asymetrické protokoly
    - RSA, ECDH, ECDSA

2.2 WolfSSL

WolfSSL je odlehčená SSL/TLS knihovna psaná v ANSI standardu C a cílená pro vložení do prostředí RTOS, především z důvodů své malé velikosti, rychlosti a sady funkcí. Knihovna wolfSSL se používá v běžných provozních podmínkách, protože je díky své volné šiřitelnosti bez licenčních poplatků. WolfSSL podporuje průmyslové standardy až do současné TLS 1.2, je až 20krát menší než OpenSSL, a nabízí progresivní šifry, jako je ChaCha20, Curve25519, NTRU a Blake2b. WolfSSL je postavena pro maximální přenosnost a je obecně velmi snadno sestavitelná na nových platformách. Knihovna podporuje programovací jazyk C jako primární rozhraní, dále také podporuje několik dalších hostitelských jazyků, včetně Java, C#, PHP, Perl a Python [31].

Funkce

- SSL 3.0 ve výchozím nastavení zakázán, TLS 1.0, 1.1 a 1.2
- DTLS 1.0 a 1.2
- Hashovací algoritmy
    - MD5, SHA-1, SHA-2, SHA-256, SHA-384, SHA-512, BLAKE2b
- Symetrické šifry
    - AES, Camellia, DES, 3DES, ARC4, RABBIT, HC-128, ChaCha20, IDEA
- Asymetrické protokoly
    - RSA, DSS DH, ECDH, NTRU

- PKCS #7
- PKCS #10
- Podpora IPv4 a IPv6

2.3 NanoSSL

NanoSSL je součástí softwarové platformy Mocana. Malá SSL/TLS knihovna je speciálně navržená pro efektivitu a vysoký výkon tak, aby urychlila vývoj produktů, a zároveň poskytovala nejlepší vlastnosti ve své třídě zabezpečení [32].

Funkce

- TLS 1.0, 1.1 a 1.2
- Hashovací algoritmy
    - MD2, MD4, MD5, SHA-1, SHA-224, SHA-256, SHA-384, SHA-512
- Symetrické šifry
    - AES, DES, 3DES
- Asymetrické protokoly
    - RSA, Diffie-Hellman, ECDH
- PKCS #7
- PKCS #10
- PKCS #12

2.4 MatrixSSL

MatrixSSL je otevřená embedded SSL/TLS knihovna určená pro malé nároky na kapacitu aplikací a zařízení, které umožní bezpečnou správu vzdálených zařízení s operačním systémem FreeRTOS, VxWorks, eCos, pSos, Linux a další embedded operační systémy [33].

Funkce

- SSL 3.0 ve výchozím nastavení vypnuto při kompilaci, TLS 1.0, 1.1 a 1.2
- DTLS 1.0 a 1.2
- Hashovací algoritmy
    - SHA-256, SHA-384, SHA-512, Poly1305
- Symetrické šifry
    - AES, ChaCha20, 3DES ve výchozím nastavení zakázán
- Asymetrické protokoly
    - RSA, DH, DHE, EDCH, PSK
- PKCS #8
- PKCS #12

3 SROVNÁNÍ KRYPTOGRAFICKÝCH KNIHOVEN

Kryptografické knihovny jsou srovnány podle následujících metrik podporovaného algoritmu a standardu, například symetrické a asymetrické šifry, hashovací algoritmy a TLS. Výsledky ze srovnání jsou uvedeny v tabulkách, ve kterých je vyznačena podpora/nepodpora konkrétních algoritmů. Pod jednotlivými tabulkami se pak nachází shrnutí poznatků, které lze z dané tabulky vyčíst.

3.1 Podporované algoritmy a standardy

Zjištění podpory následujících algoritmů a standardů v kryptografických knihovnách bylo provedeno pomocí datasheetů dostupných na stránkách knihoven, nebo v případě nedostupnosti ve zdrojových kódech daných knihoven.

Blokové šifry

Tab. 3.1: Blokové šifry

Šifra	OpenSSL	wolfSSL	NanoSSL	MatrixSSL
DES	ANO	ANO	ANO	NE
3DES	ANO	ANO	ANO	Ve výchozím nastavení zakázán
AES	ANO	ANO	ANO	ANO
Camellia	ANO	ANO	NE	NE
Idea	ANO	ANO	NE	NE

Z tabulky 3.1 můžeme vidět podporu blokových šifer. Knihovny OpenSSL a wolfSSL podporují všechny blokové šifry. Šifra AES je podporovaná všemi kryptografickými knihovnami.

Proudové šifry

Tab. 3.2: Proudové šifry

Šifra	OpenSSL	wolfSSL	NanoSSL	MatrixSSL
HC-128	NE	ANO	NE	NE
ChaCha20	ANO	ANO	NE	NE
Rabbit	NE	ANO	NE	NE
ARC4	ANO	ANO	NE	ANO

Z tabulky 3.2 můžeme vidět podporu proudových šifer. Knihovna OpenSSL nepodporuje nejnovější proudové šifry HC-128 a Rabbit z důvodu dřívějšího vývoje na rozdíl od knihovny wolfSSL, která podporuje všechny proudové šifry.

Asymetrické protokoly

Tab. 3.3: Asymetrické protokoly

Šifra	OpenSSL	wolfSSL	NanoSSL	MatrixSSL
RSA	ANO	ANO	ANO	ANO
DSA	ANO	ANO	ANO	ANO
ECDH	ANO	ANO	ANO	ANO
ECDSA	ANO	ANO	ANO	ANO

Z tabulky 3.3 můžeme vidět podporu asymetrických protokolů, které podporují všechny srovnávané kryptografické knihovny.

Hashovací algoritmy

Tab. 3.4: Hashovací algoritmy

Hash	OpenSSL	wolfSSL	NanoSSL	MatrixSSL
MD5	ANO	ANO	ANO	NE
SHA1	ANO	ANO	ANO	NE
SHA256	ANO	ANO	ANO	ANO
SHA384	ANO	ANO	ANO	ANO
SHA512	ANO	ANO	ANO	ANO

Z tabulky 3.4 můžeme vidět podporu hashovacích algoritmů. Kryptografické knihovny OpenSSL, wolfSSL a NanoSSL podporují všechny hashe, které byly vybrány k porovnání. Kryptografická knihovna MatrixSSL nepodporuje starší hashovaní algoritmy MD5 a SHA1. Hashovací algoritmus SHA1 byl začátkem roku 2017 prolomen společností GOOGLE a už není požadován za bezpečný.

TLS

Z tabulky 3.5 můžeme vidět podporu jednotlivých protokolů, ve srovnání s kryptografickými knihovnami. Protokoly TLS 1.0, TLS 1.1 a TLS 1.2 podporují všechny kryptografické knihovny. Protokoly DTLS 1.0 a DTLS 1.2 nepodporuje knihovna NanoSSL, u ostatních knihoven jsou protokoly podporovány.

Tab. 3.5: Protokoly

Protokol	OpenSSL	wolfSSL	NanoSSL	MatrixSSL
SSL 3.0	Ve výchozím nastavení povolen	Ve výchozím nastavení zakázán	NE	Ve výchozím nastavení vypnuto při kompilaci
SSL 2.0	NE	NE	ANO	NE
TLS 1.0	ANO	ANO	ANO	ANO
TLS 1.1	ANO	ANO	ANO	ANO
TLS 1.2	ANO	ANO	ANO	ANO
DTLS 1.0	ANO	ANO	NE	ANO
DTLS 1.2	ANO	ANO	NE	ANO

4 EXPERIMENTÁLNÍ MĚŘENÍ VÝKONNOSTI KRYPTOGRAFICKÝCH KNIHOVEN V IoT PROSTŘEDÍ

Experimentální měření výkonnosti kryptografických knihoven bylo prováděno na zařízení Raspberry Pi 2 Model B v1.1. Jedná se o malý, jednočipový a cenově dostupný počítač, se jmenovitým výkonem 4W. Toto zařízení můžeme vidět na obrázku 4.1.

Obr. 4.1: Raspberry Pi 2 Model B v1.1

Specifikace zařízení

- Procesor Broadcom BCM2836 ARM Cortex-A7 Quad Core(4 jádrový) 900MHz s grafickým jádrem VideoCore IV @ 250MHz
- Paměť 1GB RAM
- Operační systém RASPBIAN
- USB 2.0 4x
- Video výstup: HDMI, Kompozitní RCA; Audio výstup: 4-pólový jack pro připojení stereo audio výstupu a kompozitního video výstupu
- Micro SD slot pro kartu s operačním systémem
- 10/100mbs Ethernet Adapter

Pro experimentální měření byly vybrány tyto kryptografické knihovny wolfSSL a OpenSSL, protože jsou volně dostupné pro vývojáře, kdežto MatrixSSL a NanoSSL jsou uzavřené pro vývojáře a navíc knihovna NanoSSL je za poplatek.

Verze vybraných kryptografických knihoven jsou následující:

- wolfSSL verze 3.14.0 3. Února 2018.
- OpenSSL verze 1.1.0g 2. Listopadu 2017.

4.1 Test knihoven OpenSSL a wolfSSL pomocí nástrojů knihoven

Test výkonnosti kryptografické knihovny OpenSSL byl prováděn na zařízení Raspberry Pi 2, na kterém v době testu byl spuštěn pouze terminálu, ve kterém se zadávaly jednotlivé příkazy. Zadáním příkazu `openssl` si spustíme knihovnu OpenSSL, se kterou můžeme dále pracovat. Zadáním příkazu `speed` se spustí test výkonnosti knihovny. Pokud chceme testovat jednotlivé šifry zvlášť, musíme zadat příkaz např. `speed -evp aes-128-ctr` tímto se nám spustí test blokové šifry AES-CTR, který trvá 3s pro různé velikosti bloků dat, tento výpis v terminálu můžeme vidět níže. Takto se pokračovalo pro ostatní šifry a asymetrické protokoly, kde se zadal daný příkaz pro jednotlivé šifry za příkazem `speed`. Jednotlivé příkazy jsou vypsané níže.

```
openssl speed -evp aes-128-ctr
Doing aes-128-ctr for 3s on 16 size blocks: 2150378 aes-128-ctr's in 3.00s
Doing aes-128-ctr for 3s on 64 size blocks: 665818 aes-128-ctr's in 2.96s
Doing aes-128-ctr for 3s on 256 size blocks: 174804 aes-128-ctr's in 2.96s
Doing aes-128-ctr for 3s on 1024 size blocks: 47981 aes-128-ctr's in 3.00s
Doing aes-128-ctr for 3s on 8192 size blocks: 6155 aes-128-ctr's in 2.96s
Doing aes-128-ctr for 3s on 16384 size blocks: 3154 aes-128-ctr's in 2.99s
OpenSSL 1.1.0g  2 Nov 2017
```

Příklady příkazů pro testování ostatních šifer a asymetrických protokolů:

```
openssl speed -evp aes-256-cbc
openssl speed -evp aes-256-gcm
openssl speed -evp camellia
openssl speed -evp idea
openssl speed rsa
openssl speed ecdsa
```

Ověření, že asymetrický protokol ECDSA obsahuje knihovna OpenSSL.

```
pi@raspberrypi:~ $ openssl speed ecdsa
Doing 160 bit sign ecdsa's for 10s: 10095 160 bit ECDSA signs in 9.93s
Doing 160 bit verify ecdsa's for 10s: 2959 160 bit ECDSA verify in 10.00s
Doing 192 bit sign ecdsa's for 10s: 8123 192 bit ECDSA signs in 10.00s.....
```

Test výkonnosti kryptografické knihovny wolfSSL byl prováděn na zařízení Raspberry Pi 2, na kterém v době testu byl spuštěn pouze terminálu, na kterém se prováděl test za pomocí zadávání jednotlivých příkazů, které jsou popsané dále v textu. V první řadě je zapotřebí vstoupit do adresáře, kde máme umístěnou knihovnu wolfSSL. Aplikace pro test je spuštěna zadáním následujícího příkazu z kořenového adresáře knihovny `./wolfcrypt/benchmark/benchmark`. Některé šifry a asymetrické protokoly nejsou v základním nastavení povoleny, bylo tedy zapotřebí je povolit. Povolení probíhalo zadáním příkazu `./configure --enable-aesctr` v kořenovém adresáři knihovny. Dále je zapotřebí zadat příkaz `make`, aby došlo k překladu. Typický výstup z testu výkonnosti knihovny pomocí benchmarků bude vypadat podobně, jako výstup níže (kde můžeme vidět propustnost jednotlivých šifer v MB/s).

```
wolfCrypt Benchmark (block bytes 1048576, min 1.0 sec each)
AES-128-CBC-enc     10 MB took 1.171 seconds,    8.541 MB/s
AES-128-CBC-dec     10 MB took 1.324 seconds,    7.554 MB/s
AES-256-CBC-enc     10 MB took 1.543 seconds,    6.481 MB/s
AES-256-CBC-dec     10 MB took 1.757 seconds,    5.690 MB/s
AES-128-GCM-enc      5 MB took 2.583 seconds,    1.936 MB/s
AES-128-GCM-dec      5 MB took 2.571 seconds,    1.945 MB/s
AES-256-GCM-enc      5 MB took 2.816 seconds,    1.776 MB/s
AES-256-GCM-dec      5 MB took 2.804 seconds,    1.783 MB/s
CHACHA              15 MB took 1.085 seconds,   13.829 MB/s
```

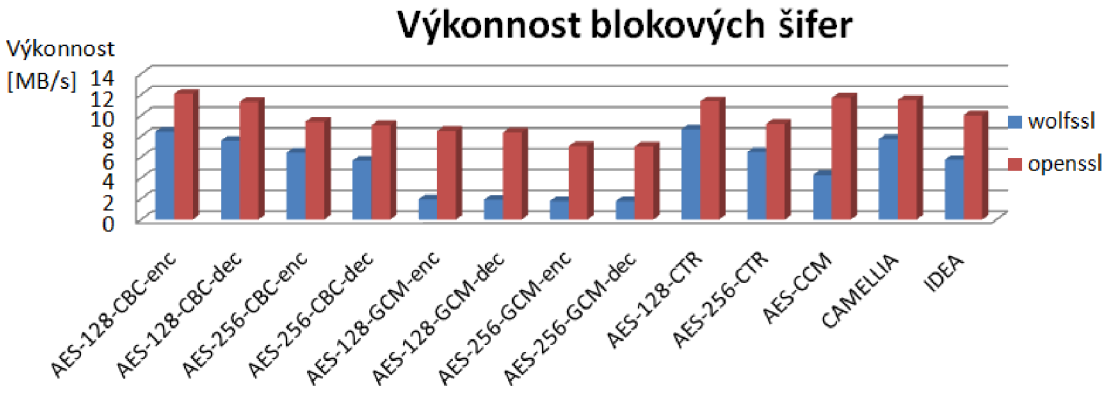
Blokové šifry

V první řadě probíhalo měření jednotlivých blokových šifer z knihovny OpenSSL a wolfSSL pomocí nástrojů, které jsou součástí knihoven. Seznam měřených blokových šifer a jejich jednotlivé výkonnosti pro různé velikosti bloku dat můžeme vidět v tabulce Tab.4.1. Následně byly výsledky testů zobrazeny graficky, které můžete vidět na Obr.4.2.

Tab. 4.1: Výkonnost blokových šifer pomocí nástrojů knihoven

		wolfSSL	openSSL
Bloková šifra	**Velikost bloku dat [MB]**	**Výkonnost [MB/s]**	**Výkonnost [MB/s]**
AES-128-CBC-enc	10	8,43	12,08
AES-128-CBC-dec	10	7,57	11,31
AES-256-CBC-enc	10	6,44	9,41
AES-256-CBC-dec	10	5,65	9,06
AES-128-GCM-enc	5	1,94	8,51
AES-128-GCM-dec	5	1,91	8,37
AES-256-GCM-enc	5	1,78	7,05
AES-256-GCM-dec	5	1,76	7,01
AES-128-CTR	10	8,67	11,36
AES-256-CTR	10	6,49	9,19
AES-CCM	10	4,29	11,71
CAMELLIA	10	7,73	11,48
IDEA	10	5,74	10,02

Z grafu, který můžeme vidět na Obr. 4.2 je patrné, že z testování výkonnosti blokových šifer při šifrování/dešifrování dat pro různé velikosti klíčů, vychází lépe knihovna OpenSSL.



Obr. 4.2: Výkonnost blokových šifer pomocí nástrojů knihoven

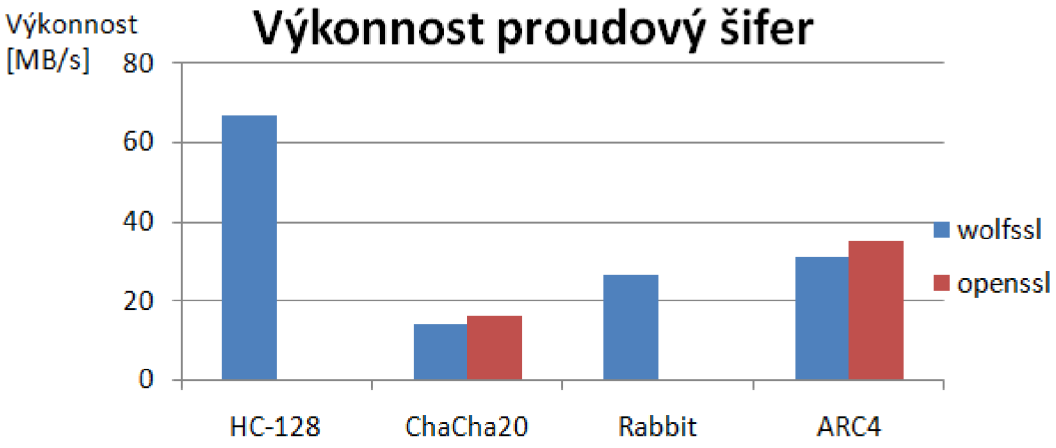
Proudové šifry

Dále proběhlo měření jednotlivých proudových šifer z knihovny OpenSSL a wolfSSL pomocí nástrojů, které jsou součástí knihoven. Seznam měřených proudových šifer a jejich jednotlivé výkonnosti pro různé velikosti bloku dat můžeme vidět v tabulce Tab.4.2. Následně

byly výsledky testů zobrazeny graficky, které můžete vidět na Obr.4.3. Z grafu je patrné, že z testování výkonnosti proudových šifer vychází opět lépe knihovna OpenSSL. Můžeme si zde všimnout, že v tabulce 4.2 u proudových šifer HC-128 a Rabbit nemáme hodnotu výkonnosti, je tomu proto, že knihovna OpenSSL nepodporuje nejnovější proudové šifry HC-128 a Rabbit z důvodu dřívějšího vývoje knihovny.

Tab. 4.2: Výkonnost proudových šifer pomocí nástrojů knihoven

		wolfSSL	OpenSSL
Proudová šifra	**Velikost bloku dat [MB]**	**Výkonnost [MB/s]**	**Výkonnost [MB/s]**
HC-128	70	66,89	-
ChaCha20	15	13,74	16,2
Rabbit	30	26,51	-
ARC4	35	31,21	35,44



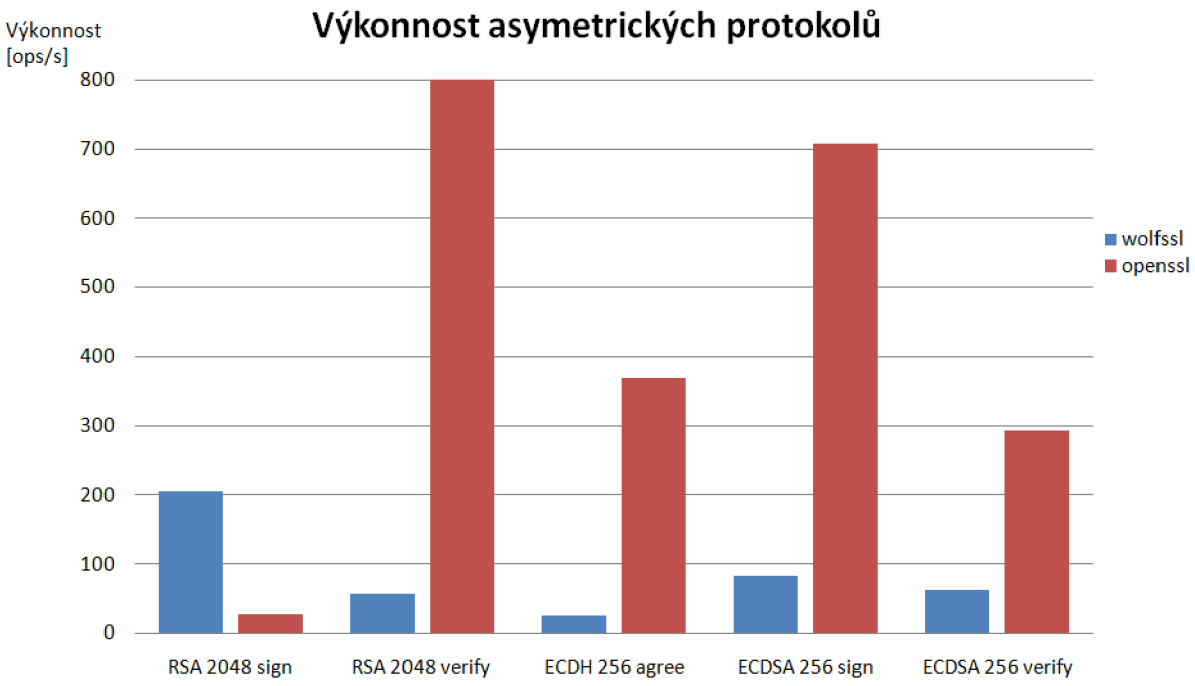
Obr. 4.3: Výkonnost proudových šifer pomocí nástrojů knihoven

Asymetrické kryptografické protokoly

Nakonec proběhlo měření asymetrických protokolů z knihovny OpenSSL a wolfSSL pomocí nástrojů, které jsou součástí knihoven. Seznam měřených asymetrických protokolů s jejich výkonností můžeme vidět v tabulce Tab.4.3. Následně byly výsledky testů zobrazeny graficky, které můžete vidět na Obr.4.4. Z grafu je patrné, že z testování výkonnosti asymetrických protokolů vychází opět lépe knihovna OpenSSL.

Tab. 4.3: Výkonnost asymetrických protokolů pomocí nástrojů knihoven

	wolfSSL	**openSSL**
Asymetrické protokoly	**Výkonnost [ops/s]**	**Výkonnost [ops/s]**
RSA 2048 sign	205,5	26,6
RSA 2048 verify	55,59	805,6
ECDH 256 agree	25,3	369,3
ECDSA 256 sign	83,26	707,7
ECDSA 256 verify	61,49	291,8



Obr. 4.4: Výkonnost asymetrických protokolů pomocí nástrojů knihoven

4.2 Vlastní měření knihovny wolfSSL pomocí podpory crypto algoritmů

Dále bylo provedeno vlastní měření kryptografické knihovny wolfSSL za pomocí podpory crypto algoritmů. Všechny námi zvolené šifry a asymetrické protokoly, které chceme změřit, jsou sepsané v souboru api.c. Tento soubor je součástí přílohy na přiloženém CD. V tomto souboru je více šifer a protokolů, ale pro tuto práci jsme použili pouze ty, co jsou vypsané v tabulkách níže. Tento soubor se nachází v adresáři `./wolfssl/tests`. I přes to, že šifry a asymetrické protokoly byly už sepsané od vývojářů knihovny wolfSSL, byl docela dost velký

problém s překladem jednotlivých funkcí na Raspberry Pi 2. Jednotlivé funkce, jak bylo nejprve testováno na virtuálním stroji, nainstalovaném na mém počítači, který běžel na systému Ubuntu 17-10. V tomto případě nebyl žádný problém vykopírovat si jednotlivé funkce do vlastního projektu, do projektu přidat příslušné `#include` soubory a projekt bez jakéhokoliv problému zkompilovat a poté spustit. Takovýto hladký scénář jsem předpokládal i na zařízení Raspberry Pi2, ale realita byla jiná. Proto vlastní měření funkcí probíhalo v souboru api.c, do kterého se doplnily příkazy pro měření. Po následných úpravách se zadal příkaz `make`, aby došlo k překladu kódu, který jsme upravili. Nakonec byl spuštěn test pomocí příkazu `./unit.test`. Jednotlivá měření probíhala v cyklech pro přesnější výsledky. V kódu je proměnná `NUM_CYCLES`, která nám udává počet opakování. V našem případě hodnota této proměnné byla nastavena na 1000. Další proměnná je `multiply_input_size`, která udává, jak velký bude vstupní text v násobcích této proměnné. Snažili jsme se přiblížit co nejvíce velikosti bloku dat, která je uvedená pro jednotlivé šifry v tabulkách. Samotné měření jednotlivých funkcí probíhalo tak, že se vzal aktuální čas a čas po vykonání funkce, tyto časy se mezi sebou odečetly a vydělily se počtem opakování. Výsledný čas se vypisuje do konzole. Hodnoty výkonnosti pro jednotlivé šifry a asymetrické protokoly jsou v porovnání s testováním výkonnosti pomocí nástrojů knihoven (benchmarků) pomalejší, je to z důvodu lepší optimalizace.

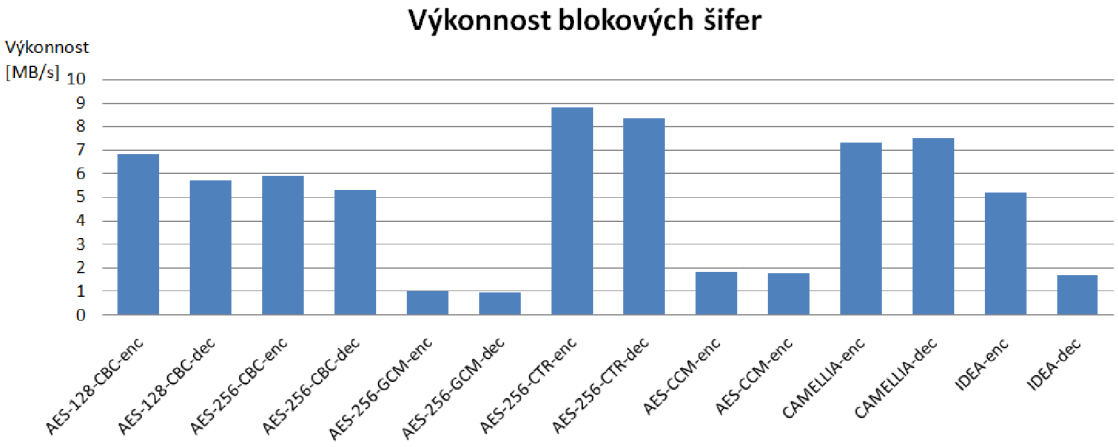
Blokové šifry

V první řadě probíhalo měření jednotlivých blokových šifer z knihovny wolfSSL pomocí podpory crypto algoritmů. Seznam měřených blokových šifer a jejich jednotlivé výkonnosti pro různé velikosti bloku dat můžeme vidět v tabulce Tab.4.4. Následně byly výsledky testů zobrazeny graficky, které můžete vidět na Obr.4.5.

Z grafu, který můžeme vidět na Obr. 4.5 je patrné, že z testování výkonnosti blokových šifer při šifrování/dešifrování dat pro různé velikosti klíčů, vychází nejlépe bloková šifra AES-CTR a dále CAMELLIA.

Tab. 4.4: Výkonnost blokových šifer pomocí podpory crypto algoritmů

		wolfSSL
Bloková šifra	**Velikost bloku dat [MB]**	**Výkonnost [MB/s]**
AES-128-CBC-enc	10	6,8
AES-128-CBC-dec	10	5,7
AES-256-CBC-enc	10	5,9
AES-256-CBC-dec	10	5,3
AES-256-GCM-enc	5	1,01
AES-256-GCM-dec	5	0,93
AES-256-CTR-enc	10	8,81
AES-256-CTR-dec	10	8,35
AES-CCM-enc	10	1,85
AES-CCM-dec	10	1,79
CAMELLIA-enc	10	7,33
CAMELLIA-dec	10	7,51
IDEA-enc	10	5,2
IDEA-dec	10	1,7



Obr. 4.5: Výkonnost blokových šifer pomocí podpory crypto algoritmů wolfSSL

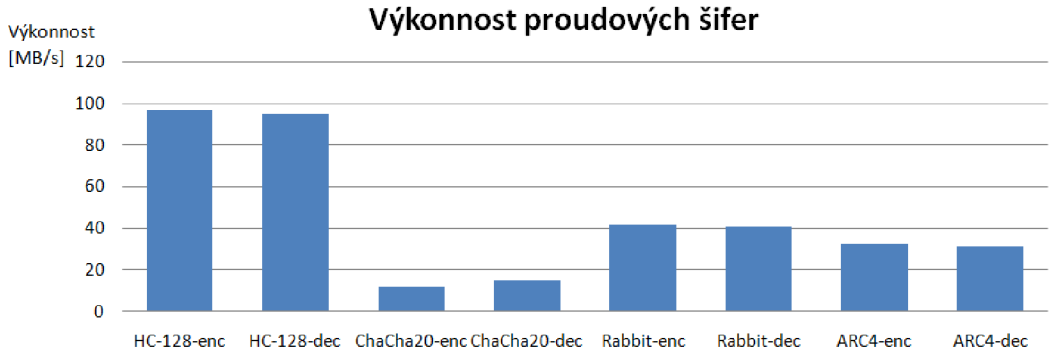
Proudové šifry

Dále probíhalo měření jednotlivých proudových šifer z knihovny wolfSSL pomocí podpory crypto algoritmů. Seznam měřených proudových šifer a jejich jednotlivé výkonnosti pro různé velikosti bloku dat můžeme vidět v tabulce Tab.4.5. Následně byly výsledky testů zobrazeny graficky, které můžete vidět na Obr.4.6.

Tab. 4.5: Výkonnost proudových šifer pomocí podpory crypto algoritmů

		wolfSSL
Proudová šifra	**Velikost bloku dat [MB]**	**Výkonnost [MB/s]**
HC-128-enc	70	96,55
HC-128-dec	70	94,91
ChaCha20-enc	15	11,74
ChaCha20-dec	15	15,06
Rabbit-enc	30	41,86
Rabbit-dec	30	40,77
ARC4-enc	35	32,51
ARC4-dec	35	31,48

Z grafu, který můžeme vidět na Obr. 4.6 je patrné, že z testování výkonnosti proudových šifer při šifrování/dešifrování dat pro různé velikosti klíčů, vychází nejlépe proudová šifra HC-128 a dále Rabbit.



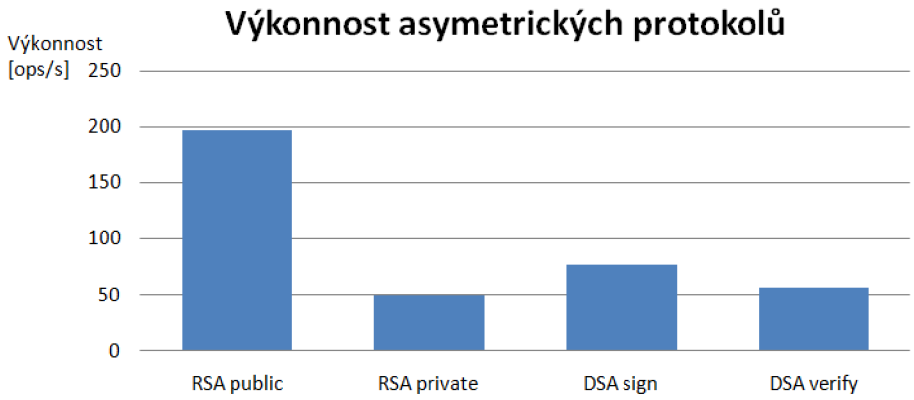
Obr. 4.6: Výkonnost proudových šifer pomocí podpory crypto algoritmů wolfSSL

Asymetrické kryptografické protokoly

Nakonec probíhalo měření jednotlivých asymetrických protokolů z knihovny wolfSSL pomocí podpory crypto algoritmů. Seznam měřených asymetrických protokolů a jejich jednotlivé výkonnosti můžeme vidět v tabulce Tab.4.6. Následně byly výsledky testů zobrazeny graficky, které můžete vidět na Obr.4.7.

Tab. 4.6: Výkonnost asymetrických protokolů pomocí podpory crypto algoritmů

	wolfSSL
Asymetrické protokoly	**Výkonnost [ops/s]**
RSA public	196,58
RSA private	49,88
DSA sign	76,47
DSA verify	56,93



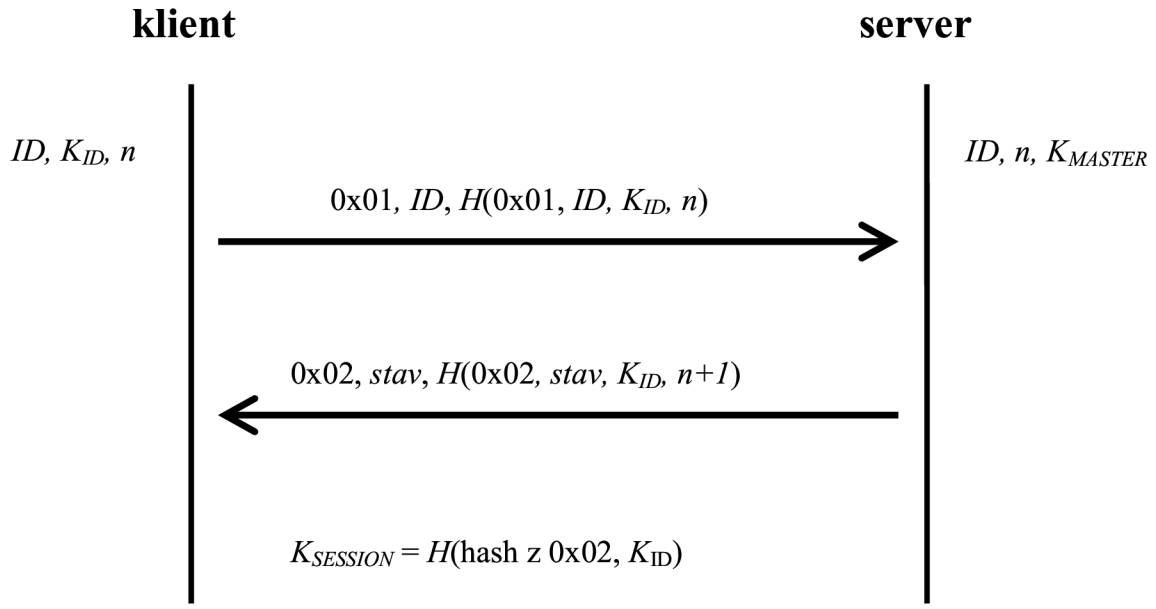
Obr. 4.7: Výkonnost asymetrických protokolů pomocí podpory crypto algoritmů wolfSSL

5 NÁVRH KOMUNIKAČNÍHO SCÉNÁŘE V IoT

Tato část práce se bude zabývat návrhem komunikačního scénáře v IoT, který bude poskytovat jednoduchý autentizační protokol klient-server založený na sdíleném tajemství. Dále tento scénář bude implementován a simulován na zařízení Raspberry Pi 2, kde také proběhnou jednotlivá měření.

5.1 Návrh protokolu

Náš navržený protokol se skládá z klienta a serveru. Každý klient má jedinečné 4 bajtové identifikační číslo označené *ID*, jedinečné 32 bajtové heslo označené K_{ID} a 4 bajtové sekvenční číslo *n*, které obsahuje i server. Abychom ušetřili paměť na serveru a server tak nemusel uchovávat hesla pro každého klienta, tak se heslo vždy dopočítá pomocí $H(K_{MASTER},ID)$ tedy z hlavního hesla K_{MASTER} a jedinečných identifikačních čísel klientů. Dále se protokol skládá ze zpráv, které se posílají mezi klientem a serverem. Jednotlivé zprávy začínají návěštím, které nám představuje 1 bajtové číslo. Toto návěští slouží k rozpoznání zpráv v dané komunikaci. Součástí protokolu nesmí chybět hashovací funkce. Pro náš scénář byla použita hashovací funkce SHA256, která je v návrhu protokolu označena *H(hashovanýřetězec)*. Pro vyhodnocení ověření serverem používáme 1 bajtové číslo označené *stav*, pokud je hodnota čísla *stav* 0x01, je ověření považováno za úspěšné. Po úspěšné autentizaci dojde mezi klientem a serverem k ustanovení klíče pro následující šifrovanou komunikaci. Označení tohoto klíče je $K_{SESSION}$.



Obr. 5.1: Autentizační protokol klient-server

Na obrázku Obr. 5.1 můžeme vidět návrh našeho protokolu, který obsahuje dvě zprávy a to zprávu 0x01 a zprávu 0x02. Klient ve své paměti uchovává své identifikační číslo *ID*, klíč K_{ID} a sekvenční číslo *n*. Na straně serveru je uloženo pro každého klienta jedinečné identifikační číslo *ID*, sekvenční číslo *n* a jedno hlavní heslo K_{MASTER}. Na serveru se pokaždé dopočítává klíč pro klienta K_{ID}, jak bylo zmíněno výše v textu, je to proto, abychom ušetřili paměť na serveru a server tak nemusel uchovávat hesla pro každého klienta.

První zpráva poslaná klientem na server je zpráva 0x01. Zpráva se skládá z návěští, *ID* klienta a hashe. Hash se skládá opět z návěští a ID klienta proto, aby se zabránilo manipulaci s daty během přenosu. Hash dále tvoří klíč klienta K_{ID} a sekvenční číslo *n*, které zde slouží k zabránění případným útokům. Jakmile server obdrží zprávu 0x01 od klienta, dojde pomocí *ID* ze zprávy 0x01 a hlavního hesla K_{MASTER} dopočítání klíče pro klienta K_{ID}. Server si vytvoří totožný hash jako u klienta a dojde k porovnání otisků. Předpokládá se zajištění synchronizace sekvenčního čísla *n* jak na straně serveru, tak i na straně klienta.

Dále dojde k vytvoření odpovědi od serveru klientovi. Jedná se o zprávu 0x02, která je složená z návěští, indikátoru *stav*, podle kterého poznáme, zda byla autentizace úspěšná či neúspěšná. Pokud autentizace proběhne úspěšně, k sekvenčnímu číslu se přičte jednička a ke zprávě se připojí hash. Hash bude složený z návěští, indikátoru *stavu*, klíče klienta K_{ID} a zvětšeného sekvenčního čísla *n*. Následně klient přijme zprávu 0x02 od serveru. Po přijetí zprávy se ověří *stav*. Pokud je stav vyhodnocen jako úspěšný, dojde u klienta k zvětšení sekvenčního čísla *n* o jedna, sestavení stejného hashe jako na straně serveru a dojde k porovnání otisků.

Při úspěšné autentizaci mezi klientem a serverem dojde k ustanovení klíče $K_{SESSION}$, který slouží pro následnou šifrovanou komunikaci. Klíč $K_{SESSION}$ se vytvoří na straně klienta, tak i na straně serveru a bude obsahovat otisk hashe ze zprávy 0x02 a klíč klienta K_{ID}.

5.2 Implementace a měření protokolu

V této kapitole jsme provedli implementaci a měření navrženého protokolu, který byl popsán výše. Implementace a měření jsme provedli na zařízení Raspberry Pi 2. Návrh programu byl prováděn ve vývojovém prostředí Eclipse. Program je navržen pro obě strany komunikace dohromady, aby simuloval komunikaci, mezi klientem a serverem, kde komunikaci začíná klient. Proto se v kódu vyskytují funkce, které by v reálné situaci provedl server, ale pro následné měření byly použity zde, např. *imagMessage02()*. V kódu se vyskytuje funkce *send()*, tato funkce slouží pro simulaci odesílání zprávy. Samotný kód je součástí přílohy na přiloženém CD.

Přehled použitých funkcí v programu:

Autentizace se spustí a provede zavoláním této funkce:

Authentication()

Vytvoření zprávy *0x01,ID, h(0x01, ID, K_{ID}, n)*, tuto zprávu klient odesílá na server pro zahájení autentizace:

createMessage01()

Funkce, která probíhá na straně serveru, slouží k vytvoření odpovědi (zprávy 0x02) klientovi *0x02, stav, h(0x02, stav, K_{ID}, n+1):*

imagMessage02()

Tato funkce ověří zprávu 0x02 od serveru, když je odpověď vyhodnocena klientem správně, přičte se k sekvenčnímu číslu *n* jednička a dopočítá $K_{SESSION}$:

verify02()

Hashovací funkce SHA256 byla použita z knihovny wolfSSL. Výběr této hashovací funkce byl proveden po srovnání s hashovací funkcí z knihovny OpenSSL. Kde hashovací funkce SHA256 z knihovny wolfSSL byla řádově o desítky ms rychlejší.

V našem navrženém protokolu jsou použity čtyři hash funkce:

- Ve funkci *createMessage01()*doba trvání je 64µs
- Ve funkci *imagMessage02()* doba trvání je 12µs
- Ve funkci *verify02()* jsou použité dvě hash funkce, obě mají stejnou dobu trvání 8µs

Měření jednotlivých částí probíhalo tak, že do programu byly doplněny funkce pro měření. Ve funkci *Authentication()*se provádí měření vytvoření zprávy 0x01 na straně klienta, dále vytvoření zprávy 0x02 na straně serveru a nakonec se provede měření ověření přijaté zprávy 0x02 na straně klienta a ustanovení hesla pro následnou šifrovanou komunikaci. Výsledky měření jsou uvedeny v tabulce Tab. 5.1.

Tab. 5.1: Výsledky měření navrženého protokolu

	Velikost zprávy [bajtů]	Doba trvání [µs]
Zpráva 0x01 od klienta	37	581
Odpověď 0x02 od serveru	34	101
Ověření 0x02 a ustanovení $K_{SESSION}$	-	146

5.3 Bezpečnostní analýza

Při návrhu protokolu jsme počítali s případnými útoky, které by mohly být použity na náš navržený protokol. Proto jsme do návrhu protokolu zahrnuli určitá bezpečnostní opatření, abychom útokům předešli. První opatření proti možnému útoku je zvolený dostatečně dlouhý klíč klienta K_{ID}, který je dlouhý 32 bajtů. Pokud by se útočník dostal k tejnému klíči klienta K_{ID} nemá zdaleka vyhráno. Pro tento případ bylo do protokolu přidáno sekvenční číslo n, které je 4 bajtové. Dále by útočník musel znát hash funkce, která byla pro tento protokol použita a hlavně složení hashovaného řetězce. Pokud by se podařilo útočníkovi zachytit zprávu 0x01 a chtěl by se vydávat za klienta, opět by to neprošlo a útočník by byl serverem odmítnut, protože naše sekvenční číslo n se zvětšuje, když server posílá zprávu 0x02. Tedy při autentizaci mělo sekvenční číslo n jinou hodnotu, než při odpovědi serveru. Pokud by se útočníkovi podařilo nějakým způsobem obdržet zprávu 0x02, stále není schopen dopočítat $K_{SESSION}$ pro následnou šifrovanou komunikaci bez znalosti K_{ID}.

5.4 Použití navrženého protokolu v prostředí IoT

V této části se pokusíme použít náš navržený protokol v reálné situaci v prostředí IoT. Tento protokol by se dal použít v několika IoT scénářích z důvodu jeho rychlosti, to nám zajišťuje použitá hash funkce.

5.4.1 Chytré parkování

Náš navržený protokol by se dal využít v reálné situaci v prostředí IoT pro chytré parkování. Návrh tohoto scénáře můžeme vidět na Obr. 5.2. Náš navržený protokol by se

využil pro komunikaci mezi senzory a IoT platformou. Kde tedy senzory by byly umístěny ve vozovce v jednotlivých parkovacích pruzích, a pomocí bezdrátové sítě by komunikovali přes navržený protokol se serverem, který by běžel na zařízení Raspberry Pi2. Jednotlivé senzory by v sobě uchovávaly svoje identifikační číslo, klientské heslo a sekvenční číslo. Při úspěšné autentizaci mezi klientem a serverem by došlo k ustanovení hesla $K_{SESSION}$, které by dále sloužilo pro následnou šifrovanou komunikaci, ve které by se přenášela informace o stavu parkovacího místa (VOLNO/OBSAZENO) mezi klientem a serverem. Pro následnou šifrovanou komunikaci by bylo vhodné použít proudovou šifru AES. Server by si dále zpracovával data a pracoval s nimi podle potřeb.

Obr. 5.2: Chytré parkování převzato z [34]

5.4.2 Přístup fanoušků na stadion

Dále by se dal náš navržený protokol využít v reálné situaci v prostředí IoT pro přístup fanoušků na stadiony. Navržený protokol by se využil pro komunikaci mezi klientem (mobilním zařízením) a serverem (turniket se zařízením Raspberry Pi 2). Zde by komunikace probíhala pomocí rozhraní Bluetooth. Klient na mobilním zařízení bude mít zapnuté rozhraní Bluetooth, dále bude mít klient ve svém zařízení elektronickou vstupenku. Turniket (zařízení Raspberry Pi 2) naslouchá a čeká na klienta, jakmile se klient přiblíží, naváže se spojení se serverem a proběhne autentizace. Při úspěšné autentizaci mezi klientem a serverem by došlo k ustanovení hesla $K_{SESSION}$, které by dále sloužilo pro následnou šifrovanou komunikaci, ve které by se ověřila platnost elektronické vstupenky a došlo by k následnému povolení či zamítnutí vstupu. Pro následnou šifrovanou komunikaci by bylo vhodné použít proudovou šifru AES. Následně by byl fanouškovi povolen vstup na stadion.

ZÁVĚR

Cílem diplomové práce bylo seznámení s kryptografickými metodami a knihovnami, které jsou vhodné pro zabezpečení komunikace v Internetu věcí (IoT), kde jsou využité výpočetně a paměťově omezená zařízení. Dále bylo za úkol zhodnocení kryptografických knihoven. V diplomové práci byly použity následující kryptografické knihovny openSSL a wolfSSL a další kryptografické knihovny nanoSSL a matrixSSL byly pouze zkoumány. Dále bylo provedeno porovnání kryptografických knihoven, podle podporovaného algoritmu a standardu, například blokové a proudové šifry, hashovací algoritmy a podpora jednotlivých protokolů. Z tohoto srovnání vyšla nejlépe knihovna openSSL, další v pořadí byla knihovna wolfSSL a nanoSSL, nejhůře dopadla ve srovnání knihovna matrixSSL.

Dále bylo provedeno experimentální měření kryptografických knihoven OpenSSL a wolfSSL v IoT prostředí na zařízení Raspberry Pi 2, za pomocí nástrojů, které obsahují knihovny od vývojářů. Měřila se výkonnost jednotlivých proudových a blokových šifer, dále výkonnost asymetrických protokolů. Výsledky měření byly zaznamenány do tabulek a následně zobrazeny graficky. Z těchto testů nám opět vyšla nejlépe knihovna OpenSSL.

Dále bylo provedeno vlastní měření knihovny wolfSSL pomocí podpory crypto algoritmů. Měřila se opět výkonnost jednotlivých proudových a blokových šifer, dále výkonnost asymetrických protokolů. Výsledky měření byly opět zaznamenány do tabulek a následně zobrazeny graficky. Pro blokové šifry vychází nejlépe šifry AES-CTR a CAMELLIA. Pro proudové šifry vychází nejlépe šifry HC-128 a Rabbit. Hodnoty výkonnosti pro jednotlivé šifry a asymetrické protokoly jsou v porovnání s testováním výkonnosti pomocí nástrojů knihoven (benchmarků) pomalejší, je to z důvodu lepší optimalizace.

Dále byl proveden návrh komunikačního scénáře v IoT, který poskytuje jednoduchý autentizační protokol klient-server založený na sdíleném tajemství. Při návrhu protokolu jsme udělali několik opatření pro případ, kdyby chtěl útočník náš protokol napadnout. Dále tento scénář byl implementován a simulován na zařízení Raspberry Pi 2, kde proběhlo jeho měření. Jeho rychlost nám zajišťuje použitá hash funkce. V našem protokolu byly použity 4 hash funkce v jednotlivých částích kódu. Výsledky byly zapsané do tabulky a vycházeli takto, zprávy 0x01 od klienta velká 37 bajtů trvá 581µs, odpověď od serveru velká 34 bajtů 101µs a ověření zprávy 0x02 a ustanovení hesla $K_{SESSION}$, pro následnou šifrovanou komunikaci 146µs. V poslední části práce proběhlo zamyšlení, kde by se náš navržený protokol dal použít v IoT prostředí. Jedno z možných řešení je využití při chytrém parkování. Zde by jednotlivé senzory pomocí bezdrátové sítě komunikovali přes náš navržený protokol se serverem, který by běžel na zařízení Raspberry Pi 2.

Literatura

[1] INTERNET OF THINGS - COUNCIL Internet of Things [online]. 2015, poslední aktualizace 08. 12. 2015 [cit. 2016-11-22] Dostupné z: http://www.theinternetofthings.eu/.

[2] Internet of Things History. [online]. Postscapes.com, 1. 2. 2016 [cit. 2018-05-17]. Dostupné z: http://www.postscapes.com/internet-of-things-history

[3] ASHTON, Kevin. *That 'Internet of Things' Thing. In the real world, things matter more than ideas*. [online]. RFID Journal, 2017 Emerald Expositions, LLC. 22. 6. 2009 [cit. 2018-05-16]. Dostupné z: http://www.rfidjournal.com/articles/view?4986

[4] GERSHENFELD, Neil A. *When Things Start to Think. Henry Holt & Company*, 1999, 224 stran. ISBN 978-0-80505874-1 [cit. 2018-05-17].

[5] ROSE, Karen, ELDRIDGE, Scott, CHAPIN, Lyman. *The Internet of Things: An Overview: Understanding the Issues and Challenges of a More Connected World* [online]. The Internet Society, 15. 10. 2015 [cit. 2018-05-16]. Dostupné z: http://www.internetsociety.org/doc/iot-overview.

[6] HANES, David, SALGUEIRO, Gonzalo, GROSSETETE, Patrick, BARTON, Robert, HENRY, Jerome. *IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things.* Cisco Press, 2017, stran. ISBN 978-1-58714-456-1 [cit. 2018-05-17].

[7] BURIAN, Pavel. *Internet inteligentních aktivit. Grada,* 2014, 336 stran. ISBN 978-80-247-5137-5 [cit. 2018-05-17].

[8] GÁLA, Libor, POUR, Jan, ŠEDIVÁ, Zuzana. *Podniková informatika: počítačové aplikace v podnikové a mezipodnikové praxi.* Grada Publishing, 2015, 240 stran. ISBN 978-80-247-5457-4 [cit. 2018-05-17].

[9] EVANS, Dave. *The Internet of Things. How the Next Evolution of the Internet Is Changing Everything.* [online]. Cisco, 2011 [cit. 2018-05-15]. Dostupné z: https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf

[10] Statista.com. Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025. [online]. Statista, 15. 5. 2018 [cit. 2018-05-15]. Dostupné z: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

[11] Internet věcí (Internet of Things). KODYS. *INTERNET VĚCÍ (INTERNET OF THINGS)* [online]. 2015 [cit. 2016-11-22]. Dostupné z URL: http://www.kodys.cz/internet-of-things.html.

[12] KERAMIDAS, Georgios, VOROS, Nikolaos, HÜBNER, Michael. *Components and Services for IoT Platforms: Paving the Way for IoT Standards.* Springer International Publishing, 2017, 383 stran. ISBN 978-3-319-42302-9. [cit. 2018-05-15].

[13] ROSE, Karen, ELDRIDGE, Scott, CHAPIN, Lyman. *The Internet of Things: An Overview: Understanding the Issues and Challenges of a More Connected World* [online]. The Internet Society, 15. 10. 2015 [cit. 2018-05-18]. Dostupné z: http://www.internetsociety.org/doc/iot-overview.

[14] NOVOTNÝ, Vít. *Architektura sítí. Fakulta elektrotechniky a komunikačních technologií VUT v Brně*, Brno, 2002. [cit. 2018-05-15].

[15] PETERKA, J. *Referenční model ISO/OSI - jeho vznik*. Computerworld, č. 12/92, rok 1992. ISSN 1210-992 [cit. 2018-05-15].

[16] DUFFY, Paul. *Beyond MQTT: A Cisco View on IoT Protocols.* [online]. Cisco, 30. 4. 2013 [cit. 2018-05-18]. Dostupné z: https://blogs.cisco.com/digital/beyond-mqtt-a-cisco-view-on-iot-protocol

[17] TRIPATHY, B. K., ANURADHA, J. *Internet of Things (IoT): Technologies, Applications, Challenges and Solutions.* CRC Press, 2017, 334 stran. ISBN 978-1-138-03500-3. [cit. 2018-05-15].

[18] ZIGBEE WIRELESS STANDARD. [online]. DIGI, [cit. 2018-05-18]. Dostupné z: https://www.digi.com/resources/standards-and-technologies/rfmodems/zigbee-wireless-standard.

[19] Z-Wave, the Smartest Choice for your Smart Home. [online]. Z-WAVE, [cit. 2018-05-16]. Dostupné z: http://www.z-wave.com/about.

[20] VOJÁČEK, Antonín. *Základní úvod do oblasti internetu věcí (IoT).* [online]. Automatizace.hw.cz, 16. 9. 2016 [cit. 2018-05-17]. Dostupné z: http://automatizace.hw.cz/zakladni-uvod-do-oblasti-internetu-veci-iot.html.

[21] SigFox tutorial-network architecture, interfaces, protocol stack. [online]. Rrfwireless-world.com, 2012 [cit. 2018-05-15]. Dostupné z: http://www.rfwireless-world.com/Tutorials/Sigfox-tutorial.html.

[22] Mikropilot pro vzdálený odečet vodoměrů. [online]. Iotcluster.cz, 7. 10. 2016 [cit. 2018-05-15]. Dostupné z: http://www.iotcluster.cz/mikropilot-pro-vzdaleny-odecet-vodomeru/

[23] LoRaWAN. *What is it? A technical overview of LoRa and LoRaWAN*. [online]. LoRa Alliance, 2015 [cit. 2018-05-17]. Dostupné z: https://www.lora-alliance.org/portals/0/documents/whitepapers/LoRaWAN101.pdf

[24] Designing the Internet of Things. People Internet vs. Device Internet. [online]. Micrium, 2018 [cit. 2018-05-17]. Dostupné z: https://www.micrium.com/iot/internet-protocols/

[25] *Mqtt.org* [online]. MQTT community, 2014 [cit. 2016-11-22]. Dostupné z: http://mqtt.org.

[26] Constrained Application Protocol for Internet of Things. CHEN, Xi. Constrained Application Protocol for Internet of Things [online]. 2014 [cit. 2016-11-22]. Dostupné z: http://www1.cse.wustl.edu/~jain/cse574-14/ftp/coap/

[27] E. Rescorla, T. Dierks, *The Transport Layer Security (TLS) Protocol, Version 1.1,RFC 4346, Updated by RFCs 4366, 4680, 4681,* 2006 [cit. 2016-11-22]. Dostupné z: http://tools.ietf.org/html/rfc4346

[28] E. RESCORLA a N. MODANUGU. RFC 4347: *Datagram Transport Layer Security)*. 2006 [cit. 2016-11-22]. Dostupné z: http://tools.ietf.org/pdf/rfc4347.pdf

[29] IoT Standards and Protocols. [online]. Postscapes, 2018 [cit. 2018-05-17]. Dostupné z: https://www.postscapes.com/internet-of-things-protocols/

[30] OpenSSL Software Foundation. *OpenSSL* [online]. 2015 [cit. 2016-11-22]. Dostupné z: https://www.openssl.org

[31] wolfSSL github.com/wolfSSL/wolfssl/wiki [online]. [cit. 2016–12–05]. Dostupné z: https://github.com/wolfSSL/wolfssl/wiki

[32] Mocana Corporation. *nanoSSL* [online] 2015 [cit. 2016-11-16]. Dostupné z: https://cdn2.hubspot.net/hubfs/397758/Downloads/IoT_Datasheets/nanossl_ds_151112.pdf?t=1481650328428

[33] MatrixSSL github.com/matrixssl/matrixssl.github.io [online]. [cit. 2016–12–05]. Dostupné z: https://github.com/matrixssl/matrixssl.github.io

[34] Smart city v praxi [online]. 2017 [cit. 2018-05-12]. Dostupné z URL: http://www.smartcityvpraxi.cz/

Použité zkratky

AES	Advanced Encryption Standard
API	Application Program Interface
DES	Data Encryption Standard
IoT	Internet of Things
SHA	Secure Hash Algorithm
SSL	Secure Sockets Layer
TLS	Transport Layer Security
PKCS	Public Key Cryptographic Standards
RSA	Rivest, Shamir, Adleman
DSA	Digital Signature Algorithm
M2M	Machine to Machine
IP	Internet Protocol
WLAN	Wireless Local Area Network
GSM	Global System for Mobile communication
HTTP	Hyper Text Transfer Protocol
FTP	File Transfer Protocol
SMTP	Simple Mail Transfer Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
CDMA	Carrier Division Multiple Access
LTE	Long Term Evolution
WiFi	Wireless Fidelity
WPAN	Wireless Personal Area Network
WMAN	Wireless Metropolitan Area Network
WAN	Wide Area Network

Obsah přiloženého CD

CD přiložené k diplomové práci obsahuje text práce. Dále obsahuje program autentizačního protokolu vytvořeného v prostředí Eclipse. Soubor api.c, na kterém probíhal test výkonnosti knihovny wolfSSL.