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

ZABEZPEČENÉ ODEČTY CHYTRÝCH ELEKTROMĚRŮ

SECURE READING OF SMART METERS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — Denisa Chaloupková

AUTHOR

VEDOUCÍ PRÁCE — Ing. David Kohout

SUPERVISOR

BRNO 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Studentka: Denisa Chaloupková ***ID:*** 211792

Ročník: 3 ***Akademický rok:*** 2021/22

NÁZEV TÉMATU:

Zabezpečené odečty chytrých elektroměrů

POKYNY PRO VYPRACOVÁNÍ:

Téma je zaměřeno na realizaci zabezpečených automatických odečtů chytrých elektroměrů pomocí push módu standardu DLMS. Prostudujte a popište možnosti zabezpečení push módu dle DLMS standardu, včetně náležitostí obsažených v objektech, které s push módem souvisí.

Výstupem bakalářské práce bude návrh a implementace zabezpečených automatických odečtů chytrých elektroměrů s využitím push módu (alespoň jedné varianty) standardu DLMS s využitím knihovny Gurux ve vhodném programovacím jazyku (doporučeným jazykem je Java). Server (elektroměr) bude implementován na platformu Raspberry Pi a proběhne ověření zabezpečeného push módu při komunikaci se stranou klienta. Také budou porovnány datové objemy při využití zabezpečené a nezabezpečené varianty push módu.

DOPORUČENÁ LITERATURA:

[1] Gurux.DLMS | Gurux for DLMS smart meters. [online]. [cit. 2021-09-13]. Dostupné z: https://www.gurux.fi/Gurux.DLMS.Notify

[2] MATOUŠEK, Petr. Technical report: Analysis of DLMS Protocol [online]. VUT FIT-TR-2017-13, Brno, CZ, 2017 [cit. 2021-02-22]. Dostupné z: https://www.fit.vut.cz/research/publication/11616

Termín zadání: 7.2.2022 ***Termín odevzdání:*** 31.5.2022

Vedoucí práce: Ing. David Kohout

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Práce se zabývá implementací zabezpečeného push módu standardu DLMS. Představuje standard DLMS/COSEM, obsahuje teorii k zabezpečení zpráv a popisuje, jak funguje push a jeho atributy. Pro tyto účely je využívaná knihovna Gurux v jazyce Java a upraven příklad gurux.dlms.push.listener. Jako server je použita platforma Raspberry Pi. Pro ověření zabezpečení push zpráv byla využita aplikace Wireshark a nástroj Gurux DLMS Translator dostupný v aplikaci Gurux GXDLMSDirector. Následně jsou push zprávy porovnány z hlediska jejich velikosti podle použitého druhu zabezpečení. Čím objemnější je obsah odesílaných dat ve zprávě, tím jsou dodatečná data, potřebná k zabezpečení zprávy, v poměru s celkovým objemem menší.

KLÍČOVÁ SLOVA

datové objemy, DLMS, knihovna Gurux DLMS, push zprávy, zabezpečení

ABSTRACT

The work deals with the implementation of the secure push mode of the DLMS standard. It introduces the DLMS/COSEM standard, contains a theory for message security and describes how push works and its attributes. The Gurux library in Java is used and the example gurux.dlms.push.listener is modified. The Raspberry Pi platform is used as the server. Wireshark and the Gurux DLMS Translator available in Gurux GXDLMSDirector were used to verify security of a push message. Subsequently, push messages are compared in terms of their size according to the type of security used. The larger the content of the data sent in the message, the smaller the additional data needed to secure the messages in proportion to the total volume.

KEYWORDS

DLMS, Gurux DLMS library, push messages, security, size of data

CHALOUPKOVÁ, Denisa. *Zabezpečené odečty chytrých elektroměrů*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2022, 43 s. Bakalářská práce. Vedoucí práce: Ing. David Kohout

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Denisa Chaloupková
VUT ID autora:	211792
Typ práce:	Bakalářská práce
Akademický rok:	2021/22
Téma závěrečné práce:	Zabezpečené odečty chytrých elektroměrů

Prohlašuji, že svou závěrečnou práci jsem vypracovala samostatně pod vedením vedoucí/ho závěrečné 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 autorka uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušila autorská práva třetích osob, zejména jsem nezasáhla nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědoma následků porušení ustanovení § 11 a následujících autorského zákona č. 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.

Brno ....

podpis autorky*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Ráda bych poděkovala vedoucímu bakalářské práce panu Ing. Davidu Kohoutovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Device Language Message Specification/Companion Specification for Energy Metering, zkráceně DLMS/COSEM, je globálně uznávaný standard pro chytré měření energií. Jedním ze způsobů komunikace serveru a klienta je odesílání push zpráv. Server je schopen odesílat push zprávy bez toho, aby si je klient vyžádal. Proto je push mód vhodný k odesílání např. alarmů nebo plánovaného odečtu energií. Jako každá jiná komunikace, i push zprávy vyžadují zabezpečení.

Tato práce se zabývá implementací zabezpečených push zpráv. Je využita knihovna Gurux v jazyce Java a upraven příklad gurux.dlms.push.listener. Jako server je využívaná platforma Raspberry Pi.

Práce na začátku krátce představuje standard DLMS/COSEM, jeho využití, správce a dokumentaci.

V další části se práce zabývá zabezpečením dat popsaného v Zelené knize. Jsou popsány způsoby autentizace, druhy kryptografických algoritmů, kontext zabezpečení, přístupová práva, atd. Dále jsou popsány důležité části COSEMu, push IC a popsáno zabezpečení COSEM dat podle Modré knihy.

Poslední kapitola je zaměřena na představení Gurux knihovny a její využití pro odesílání push zpráv. Je zde ověřeno zabezpečení odesílaných push zpráv pomocí aplikace Wireshark a nástroje Gurux DLMS Translator. Zachycené push zprávy jsou porovnány z hlediska jejich velikosti v závislosti na použité zabezpečení.

1 DLMS/COSEM

Device Language Message Specification/Companion Specification for Energy Metering, zkráceně DLMS/COSEM, je globální standard pro chytré řízení energie a vody, pokročilé řízení a inovativní měření [1]. Specifikace byla vyvinuta a je udržována DLMS User Association (DLMS UA) [2]. Specifikace je rozdělena na 4 části. Části jsou barevně rozlišeny a popsány v barevných knížkách, tzv. Coloured books. Zelená kniha (Green book) popisuje architekturu a protokoly [3], Modrá kniha (Blue book) popisuje COSEM a OBIS (Object Identification System) [4], Žlutá kniha (Yellow book) popisuje postup testování a Bílá kniha (White book) obsahuje definice pojmů.

Specifikace DLMS/COSEM specifikuje datový model a komunikační protokoly pro výměnu dat se zařízením pro měření. Dodržuje tříkrokový postup:

1. Modelling (modelování) - zahrnuje model rozhraní měřícího zařízení a pravidla pro identifikaci dat, popsáno v Blue book,
2. Messaging (zprávy) - zahrnuje služby pro mapování modelu rozhraní na protokolové datové jednotky (APDU) a kódování těchto APDU (Application Protocol Data Unit), popsáno v Green book,
3. Transporting (přenos) - zahrnuje přenos zpráv přes komunikační kanál, popsáno v Green book.

Aplikační vrstva DLMS/COSEM (AL) specifikuje služby pro navázání logických spojení mezi klientem a serverem (případně servery) a služby pro přístup k atributům a metodám COSEM objektů [3].

Specifické profily komunikačních médií DLMS/COSEM určují jak mohou být přenášeny zprávy aplikační vrstvy přes různá komunikační média. Každý komunikační profil specifikuje sadu protokolových vrstev požadovaných pro podporu DLMS/COSEM AL „on top“ [3].

Rozsáhlé zavádění chytrých měřicích systémů vyžaduje silné mechanismy zabezpečení informací, které chrání soukromí spotřebitelů energií, obchodní zájmy poskytovatelů energie, služeb a bezpečnost energetické infrastruktury [3].

DLMS/COSEM poskytuje vestavěné bezpečnostní mechanismy. Poskytuje mechanismy pro identifikaci a autentizaci klientů a serverů, specifická přístupová práva k atributům a metodám COSEM objektů v rámci aplikačních asociací (AA) vytvořených mezi klientem a serverem. Dostupné jsou také šifrované APDU, které umožňují ochranu vyměňovaných zpráv mezi klienty a servery [3].

2 Informační bezpečnost v DLMS/COSEM

2.1 DLMS/COSEM bezpečnostní koncept

Ke zdrojům serverů DLMS/COSEM (atributy a metody COSEM objektů) mohou klienti přistupovat v rámci Application Associations (AA). Během zřízení AA se klient a server musí identifikovat. Server může také vyžadovat, aby se uživatel klienta sám identifikoval. Dále může server požadovat autentizaci klienta a klient může požadovat autentizaci serveru.

Jakmile je vytvořena AA, xDLMS služby mohou být použity pro přístup k atributům a metodám COSEM objektu v závislosti na zabezpečení a přístupových právech. xDLMS APDU nesoucí service primitives mohou být kryptograficky chráněna. Požadovaná ochrana je určena kontextem zabezpečení a přístupovými právy [3].

2.2 Identifikace a autentizace

Každý DLMS/COSEM klient a každý server (logical device) je vázán se Service Access Point (SAP) v protokolové vrstvě podporující AL. Tyto SAP jsou přítomny v PDU nesoucí xDLMS APDU v rámci AA. Mechanismus identifikace uživatele klienta umožňuje serveru rozlišovat mezi různými uživateli a zaznamenávat jejich aktivity při přístupu k měřiči [3].

Autentizační mechanismy určují protokol, který mají komunikační entity použít k autentizaci během ustanovení AA [3]. Existují 3 různé autentizační mechanismy, každý má jinou úroveň autentizačního zabezpečení:

- No security (Lowest Level Security),
- Low Level Security (LLS),
- High Level Security (HLS).

2.2.1 No security (Lowest Level Security) autentizace

V tomto případě dovoluje klientovi získat některé základní informace od serveru. Neprobíhá zde žádná autentizace, klient může přistupovat k atributům a metodám COSEM objektů v rámci kontextu zabezpečení a přístupovým oprávněním v daném AA.

2.2.2 Low Level Security autentizace

V tomto případě se klient autentizuje předáním hesla, které server zná. Heslo je předáno aktuálnímu objektu „Asociation SN / LN“ modelující AA, které má být

zřízeno. Tyto objekty také poskytují prostředky ke změně hesla. Pokud je předané heslo akceptováno, ustanoví se AA, v opačném případě je odmítnuto.
LLS autentizace je podporována funkcí COSEM-OPEN service následovně:

- klient přenese heslo k serveru použitím COSEM-OPEN.request,
- server zkontroluje, zda je heslo správné,
- pokud ano, klient je autentizován a AA může být ustanoveno,
- pokud ne, AA bude odmítnuto,
- výsledek zřízení AA zašle server zpět pomocí COSEM-OPEN.response společně s diagnostickými informacemi.

2.2.3 High Level Security autentizace

V tomto případě se musí autentizovat klient i server, aby mohla být ustanovena AA. Tato autentizace je rozdělena do čtyřkrokového procesu, který je podporován funkcí COSEM-OPEN service a metodou reply_to_HLS_authentication z IC „Association SN / LN“:

1. Klient odešle serveru výzvu CtoS a v závislosti na autentizačním mechanismu další informace.
2. Server odešle klientovi výzvu StoS a v závislosti na autentizačním mechanismu další informace.
   Pokud je výzva StoC a CtoS stejná, klient ji odmítne a ukončí proces ustanovení AA.
3. Klient zpracuje StoC a doplňující informace podle pravidel HLS autentizačního mechanismu platný pro daný AA a výsledek odešle serveru. Server zkontroluje, zda f(StoC) je správný a pokud je, akceptuje autentizaci klienta.
4. Server zpracuje CtoS a doplňující informace podle pravidel HLS autentizačního mechanismu platný pro daný AA a výsledek zašle klientovi. Klient pak zkontroluje, zda je f(CtoS) správný a pokud je, akceptuje autentizaci serveru.

Krok 1 a 2 je podporován funkcí COSEM-OPEN service. Po 2. kroku server udělí přístup k metodě reply_to_HLS_authentication, za předpokladu, že navrhovaný aplikační kontext a kontext xDLMS jsou přijatelné. Kroky 3 a 4 jsou podporovány metodou reply_to_HLS_authentication. Pokud jsou oba tyto kroky provedeny úspěšně, ustanoví se AA s vyjednaným aplikačním kontextem a xDLMS kontextem.

HLS autentizace vyžaduje kryptografické zpracování výzev, které si vyměňuje klient a server [3]. Je k dispozici několik autentizačních mechanismů a jsou vypsány v tabulce 2.1.

Tab. 2.1: Seznam autentizačních mechanismů.

Autentizace	**mechanism_id**
Bez zabezpečení	0
LLS	1
HLS	2
HLS MD5	3
HLS SHA-1	4
HLS GMAC	5
HLS SHA-256	6
HLS ECDSA	7

2.3 Kryptografické algoritmy

DLMS/COSEM používá kryptografii k ochraně informací. Existují tři základní typy kryptografických algoritmů:

- kryptografické hash funkce, které nepotřebují klíče (i když mohou být použity v módu, ve kterém se klíče používají). Hash funkce se často používá jako komponenta algoritmu k poskytování bezpečnostní služby;
- algoritmy symetrického klíče, které používají jeden klíč, který je sdílený odesílatelem a příjemcem, k aplikaci ochrany, odstranění ochrany a kontrole ochrany. Poměrně snadno se implementují a poskytují vysokou propustnost;
- algoritmy asymetrického klíče používají veřejný a soukromý klíč, které spolu matematicky souvisí. Ve srovnání s algoritmy symetrického klíče je implementace těchto algoritmů složitá a vyžaduje více výpočtů.

Pro použití kryptografie je potřeba ustanovit klíče pro jednotlivé strany. V DLMS/-COSEM se využívá hodně kryptografických algoritmů. Ty nejdůležitější jsou:

- AES – Advanced Encryption Standard, v DLMS/COSEM využívá velikost klíčů 128 bit, nebo 256 bit. Využiváný pro šifrovaný přenos dat nebo přenosu dohodnutého klíče.
- ECDSA – Algoritmus digitálního podpisu na eliptických křivkách, používá se pro digitální podpisy. Dle DLMS Security Suite 1 nebo 2 se používají křivky P-256 s SHA-256 nebo P-384 s SHA-384.
- ECDH – Elliptic Curve Diffie-Hellman, využívá se k dohodnutí šifrovacích klíčů;
- SHA – Secure Hash Algorithm, využívaný pro digitální podpisy, výměnů klíčů autentizaci. Využivané SHA-256 a SHA-384.

2.4 Kontext zabezpečení

Kontext zabezpečení definuje bezpečnostní atributy relevantní pro kryptografii. Je spravován v objektech Security setup. Obsahuje tyto položky:

- security suite, který určuje dostupné bezpečnostní algoritmy,
- bezpečnostní politika, která určuje druhy zabezpečení, které mají být aplikovány na všechny xDLMS APDU vyměňované v rámci AA.
- bezpečnostní materiály, které zahrnuje bezpečnostní klíče, inicializační vektory, certifikáty, atd.

2.5 Přístupová práva

Přístupová práva jsou specifikována v objektech Association SN/LN. Konkrétně v atributu object_list z objektu Association LN nebo v atributu access_rights_list z objektu Association SN. Element access_mode v access_rights určuje druh přístupu a kryptografickou ochranu. Je to datový typ enum.

V případě Association LN verze 3 a Association SN verze 4 je enum hodnota interpretována jako unsigned8. Význam každého bitu je v tabulce 2.2:

Tab. 2.2: Hodnoty přístupových práv [3].

Bit	**Přístup k atributu**	**Přístup k metodě**
0	přístup ke čtení	přístup
1	přístup k zápisu	nepoužíváno
2	ověřený žádost	ověřený žádost
3	šifrovaná žádost	šifrovaná žádost
4	digitálně podepsaná žádost	digitálně podepsaná žádost
5	ověřená odpověď	ověřená odpověď
6	šifrovaná odpověď	šifrovaná odpověď
7	digitálně podepsaná odpověď	digitálně podepsaná odpověď
Příklady	enum (3): read write enum (6): přístup pro zápis s ověřenou žádostí enum (255): přístup pro čtení/zápis s ověřenou, šifrovanou a digitálně podepsanou žádostí a odpovědí	enum (1): access enum (2): nepoužíváno enum (5): přístup s ověřenou žádostí enum (253): přístup s ověřenou, šifrovanou a digitálně podepsanou žádostí a odpovědí

2.6 Generování náhodných čísel

Různé algoritmy používáné v DLMS/COSEM mohou požadovat generování náhodných čísel. Proto musí být poskytnut silný generátor náhodných čísel (RNG). RNG by měl být nedeterministický. Pokud takový není k dispozici, systém použije dostečnou entropii k vytvoření kvalitního seed pro deterministický RNG.

2.7 Komprese

Kompresi lze použít na COSEM data nebo xDLMS APDU. Tento proces lze kombinovat se symetrickým šifrováním. Kompresní algoritmus vychází z ITU-T V.44:2000 a musí splňovat tyto požadavky:

- nízké výpočetní zatížení;
- nízké nároky na paměť;
- nízká latence.

2.8 Security suite

Security uite určuje sadu kryptografických algoritmů dostupných pro různá kryptografická primitiva a velikosti klíčů. DLMS/COSEM Security Suites vychází z NSA Suite B a obsahují kryptografické algoritmy pro autentizaci, šifrování, dohody o klíčích, digitální podpisy a hashe. Kromě toho jsou k dispozici kompresní algoritmy a algoritmy zapouzdření klíče. Tabulka 2.3 ukazuje DLMS/COSEM security suites [3].

Tab. 2.3: DLMS/COSEM security suites [3]

Security Suite ID	**Security suite jméno**	**Šifrování**	**Digitální podpis**	**Dohodnutí klíče**	**Hash**	**Přenos klíče**	**Komprese**
0	AES-GCM-128	AES-GCM-128	-	-	-	AES-128	-
1	ECDH-ECDS-AES-GCM-128-SHA-256	AES-GCM-128	ECDSA s P-256	ECDH s P-256	SHA-256	AES-128	V.44
2	ECDH-ECDSA-AES-GCM-256-SHA-384	AES-GCM-256	ECDSA s P-384	ECDH s P-384	SHA-384	AES-256	V.44
Rezervováno	-	-	-	-	-	-	-

2.9 Ochrana xDLMS APDU

Ochranu na COSEM data lze aplikovat stejně, jako u xDLMS APDU. Když AP vyvolá atribut COSEM objektu nebo metodu související s xDLMS service primitive, parametry služby zahrnují Security_Options parametr. Tento parametr informuje AL o druhu šifrovaného APDU a ochraně, která má být použita, a zahrnuje nezbytný bezpečnostní materiál. AL nejprve sestaví APDU odpovídající service primitive a pak sestaví zašifrované APDU.

Když AL obdrží zašifrované APDU od vzdáleného partnera, dešifruje APDU a obnoví originální nezašifrované APDU. Když dešifrování proběhne úspěšně, vyvolá příslušné service primitive. Mezi další parametry služby patří parametry Security_Status a Protection_Element, které informují AP o druhu použitého šifrovaného APDU, o ochraně, která byla ověřena a odstraněna, a může zahrnovat bezpečnostní materiál [3].

3 COSEM

COSEM je model rozhraní komunikujícího zařízení pro měření energií, který poskytuje pohled na funkčnost dostupnou prostřednictvím komunikačního rozhraní. Poskytuje sémantiku pro měřící aplikaci. COSEM model využívá objektově orientovaný přístup. Instance COSEM třídy rozhraní (IC) je nazývána COSEM objekt rozhraní. Objekty modelují funkčnost měřícího zařízení prostřednictvím implementace rozhraní.

COSEM model představuje měřič energie jako server používaný klientskými aplikacemi, které získávají data z měřiče, poskytují kontrolní informace měřiči a spouštějí známé akce v rámci měřiče prostřednictvím řízeného přístupu k atributům a specifickým metodám objektů, které tvoří rozhraní serveru.

3.1 Fyzické a logické zařízení

COSEM modeluje měřící zařízení na fyzické (physical device), které představuje konkrétní fyzický měřič. Fyzické zařízení obsahuje jedno nebo více logických zařízení (logical devices). Každé logické zařízení obsahuje COSEM objekty, které modelují funkcionalitu daného logického zařízení. Každé logické zařízení má své unikátní pojmenování – logical device name (LDN). LDN je definováno jako řetězec o 16 oktetech. První tři oktety nesou unikátní identifikátor výrobce, které spravuje DLMS UA.

COSEM server je strukturovaný do tří hierarchických úrovní fyzické zařízení:

- fyzické zařízení,
- logické zařízení,
- dostupné COSEM objekty.

3.2 COSEM třídy rozhraní a objekty

COSEM IC se řídí objektově orientovaným přístupem. Objekt je kolekce atributů a metod. Atributy reprezentují vlastnosti objektu. Každý atribut obsahuje význam, datový typ a hodnotu. Hodnota atributu může ovlivnit chování objektu a metody umožňují provádět operace s atributy objektu.

3.3 Přistupování k objektu prostřednictvím aplikační asociace

Aby bylo možné přistupovat ke COSEM objektům na serveru, musí nejprve klient vytvořit spojení se serverem, které se označuje jako aplikační asociace (AA). Tento proces identifikuje partnery a charakterizuje kontext, ve kterém budou AA komunikovat. Kontext zahrnuje kontext aplikace, kontext ověřování a xDLMS kontext.

Informace jsou obsaženy ve speciálním COSEM objektu - Association. Existují dva typy objektu Association v závislosti na odkazování na jméno: logical name (LN) nebo short name (SN).

Každé logické zařízení obsahuje alespoň jeden objekt třídy Association LN nebo Association SN.

3.4 Logical name a Short name

K atributům a metodám COSEM objektů lze přistupovat pomocí odkazování. Existují dva různé způsoby, jak odkazovat na COSEM objekty.

3.4.1 LN odkazování

LN odkazování je způsob, jak získat přístup k atributům a metodám COSEM objektů pomocí identifikátoru COSEM IC a instance této třídy, ke které tyto atributy a metody patří.

Odkaz na atribut je class_id, hodnota logical_name, atrribute_index. Odkaz na metodu je class_id, hodnota logical_name, method_index.

Indexy atributů a metod jsou uváděny v definici každé IC. Jsou to kladná čísla začínající od 1.

3.4.2 SN odkazování

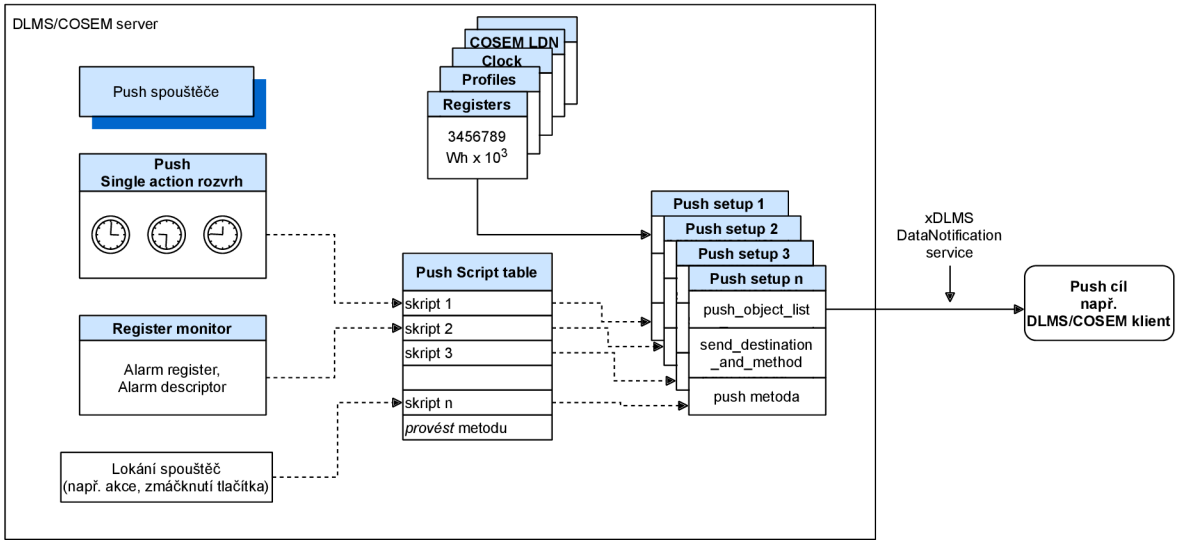
SN odkazování je způsob, jak získat přístup k atributům a metodám COSEM objektů po prvním namapování na krátká jména. Tento druh odkazování je určen pro použití v jednoduchých zařízeních.

V tomto případě každý atribut a metoda COSEM objektu je identifikována 13bitovým celým číslem. Syntaxe je stejná jako u DLMS pojmenované proměnné. Když se používá SN odkazování, každý atribut a metoda objektu musí být namapovány na krátká jména [5].

3.5 Push interface class

DLMS zprávy mohou být posílány k cíli, aniž by byly vyžádány. Push může být iniciován pomocí spouštěčů, např. když nastane plánovaný čas, lokálně monitorovaná hodnota přesáhne stanovenou hranici, nebo nastane nějaká údalost (zmáčknutí tlačítka, otevření krytu elektroměru, atd.).

Push mechanismus se řídí vzorem publish/subscribe. Publishing je v DLMS/COSEM modelováno pomocí atributu object_list objektů „Association“, které poskytují seznam COSEM objektů a jejich atributy dostupné v dané AA. Subscription je modelováno zápisem příslušných atributů objektů „Push setup“. Požadovaná data jsou poslána pomocí služby xDLMS DataNotification po aktivaci daného spouštěče. COSEM model push operace je zobrazen na obrázku 3.1.



Obr. 3.1: COSEM model push operace [4].

Základním prvkem modelovaní push operace je Push setup IC. Atribut push_object_list obsahuje seznam odkazů na COSEM atributy objektů, které mají být přeneseny.

Různé spouštěče volají skript v objektu Push Script table (instance Script table IC), který pak vyvolá push metodu souvisejícího Push setup objektu. Cíl, komunikační médium, protokol, kódování, načasování i případné opakování push operace jsou určeny dalšími atributy Push setup objektu.

Každý spouštěč může způsobit odeslání dat do dedikovaného cíle. Proto je každý spouštěč, nebo skupinu spouštěčů, k dispozici samostatný Push setup objekt definující obsah, cíl push zprávy a také použité komunikační médium.

Pro účely odeslání dat, když se spustí poplach, jsou k dispozici objekty Alarm monitor (instance Register monitor IC). Alarmy jsou drženy objektem Alarm register nebo objektem Alarm descriptor.

Objekty Alarm descriptor jsou monitorovány objekty Alarm „Register monitor“. Atribut akcí poskytuje odkaz na action_up script a možná na action_down skript v objektu Push „Script table“, které následně vyvolávají push metodu požadovaného objektu „Push setup“. Když dojde k poplachu, předdefinovaná sada dat, která může obsahovat kromě jiných dat i objekty Alarm register a Alarm descriptor, se odešle.

Push data jsou odeslány pomocí nevyžádané služby xDLMS typu non-client/server - služba DataNotification. Pokud jsou odeslaná data moc dlouhá, můžou být odeslána v blocích.

Push proces probíhá v aplikačním kontextu AA, na který odkazuje atribut push_client_SAP objektu „Push setup“. Kontext zabezpečení je určen objektem „Security setup“, na který odkazuje objekt „Association“. Nezbytné parametry ochrany dat jsou definovány v atributu push_protection_parameters, který nabízí stejné možnosti jako IC „Data Protection“.

Veškeré informace nezbytné ke zpracování dat obdržené klientem jsou buď:

- načtené klientem např. čtením atributu push_object_list, nebo
- musí být součástí zasílaných dat, nebo
- musí být předdefinovany v klientské aplikaci [4].

3.5.1 Push setup

Push setup objekt obsahuje seznam referencí na atributy COSEM objektů, které mají být přeneseny. Také obsahuje cíl odeslání, push metodu, časové okno komunikace a zpracování opakování. Ve verzi 1 byla přidána možnost ochrany dat, která nabízí stejné možnosti, jaké jsou definovány v Data protection IC (viz kapitola 3.6). Ve verzi 2 je specifikováno 6 nových možností:

- nový mechanismus výběru nového záznamu, který umožňuje výběr záznamů související s posledním potvrzeným záznamem;
- nový mechanismus výběru sloupců, který umožňuje explicitně vybrat sloupce buffer atributů Profile generic objektu;
- atribut repetition_delay nyní poskytuje exponenciální vzorec, který umožňuje prodloužit zpoždění opakování s každým pokusem o opakování;
- nový atribut push_operation_method, který umožňuje specifikovat, zda je primitivum služby DataNotification.request voláno s Service_Class == Unconfirmed nebo Confirmed a jak funguje operace push opakování;

- nový atribut confirmation_parameters, který umožňuje omezit časový interval, ve kterém lze vybrat záznamy související s posledním potvrzeným záznamem;
- nový atribut last_confirmation_date_time, který obsahuje datum a čas kdy byla služba DataNotification naposledy potvrzena.

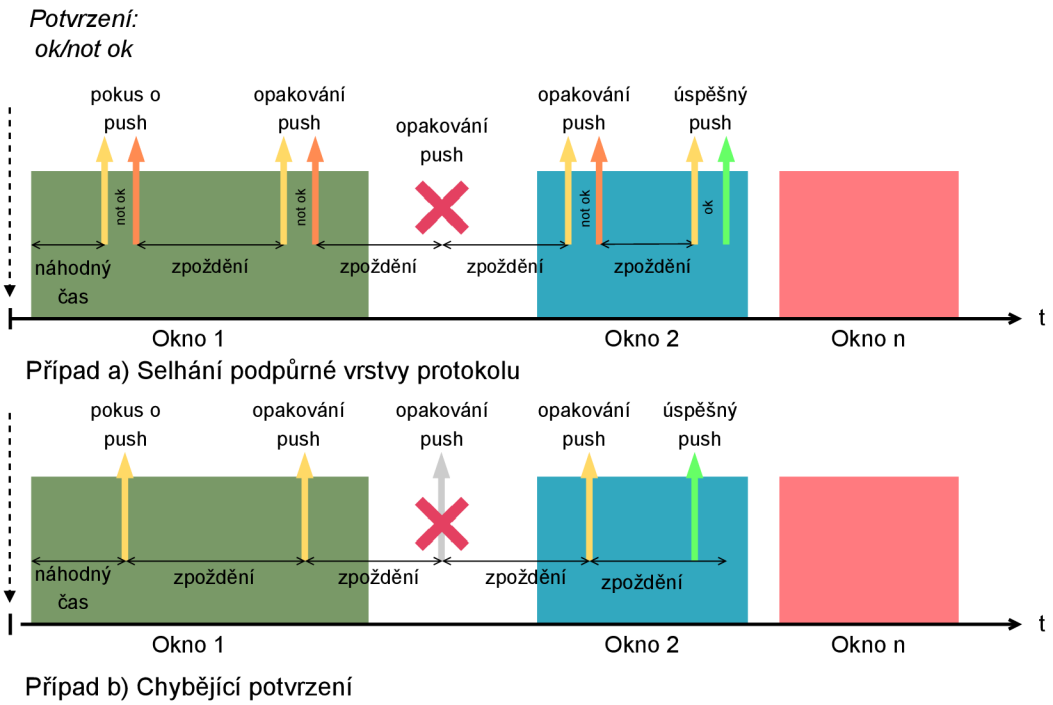
Seznam všech atributů Push setup IC je uveden v tabulce 3.1. Ve využívané knihovně od firmy Gurux obsahuje Push setup IC jen prvních 7 atributů [9].

Tab. 3.1: Push setup atributy.

Push setup atributy	**Datový typ**
1. logical_name	octet-string
2. push_object_list	array
3. send_destination_and_method	structure
4. communication_window	array
5. randomisation_start_interval	long-unsingned
6. number_of_retries	unsigned
7. repetition_delay	structure
8. port_reference	octet-string
9. push_client_SAP	integer
10. push_protection_method	array
11. push_operation_method	enum
12. confirmation_parameters	structure
13. last_confirmation_date_time	date-time

Tato verze IC je určena k usnadnění použití relativního a absolutního výběru dat. Push se uskuteční po vyvolání push metody, spuštěné objektem Push Single action schedule, objektem Alarm Register monitor, nebo vyhrazenou interní nebo externí událostí. Po tom, co je spuštěna push operace pomocí jednoho ze spouštěčů, je tato operace provedena podle nastavení v objektu Push setup. V závislosti na nastavení komunikačního okna se push operace provede okamžitě, nebo hned jak se komunikační okno stane aktivní po náhodném zpoždění.

Jestliže je implementován potvrzovací atribut, může být detekováno, zda byl push neúspěšný. V tomto případě je možné zprávy opakovat. Push operace s komunikačními okny, zpožděním a opakováním je znázorněna na obrázku 3.2.



Obr. 3.2: Push okna, zpoždění, opakování [4].

3.6 COSEM data protection class

Instance této IC umožnují aplikovat kryptografickou ochranu na COSEM data, tj. na hodnoty atributů, vyvolávání metod a jejich návratových parametrů. Toho je dosaženo nepřímým přístupem k atributům nebo metodám jiných COSEM objektů prostřednictvím instancím Data protection IC, které poskytují nezbytné mechanismy a parametry k aplikaci, ověření a odstranění ochrany na COSEM datech. Ochrana COSEM dat je v souladu s ochranou na xDLMS APDU(viz kapitola 2.9).
Příklady použití ochrany COSEM dat zahrnují, ale nejsou omezeny na:

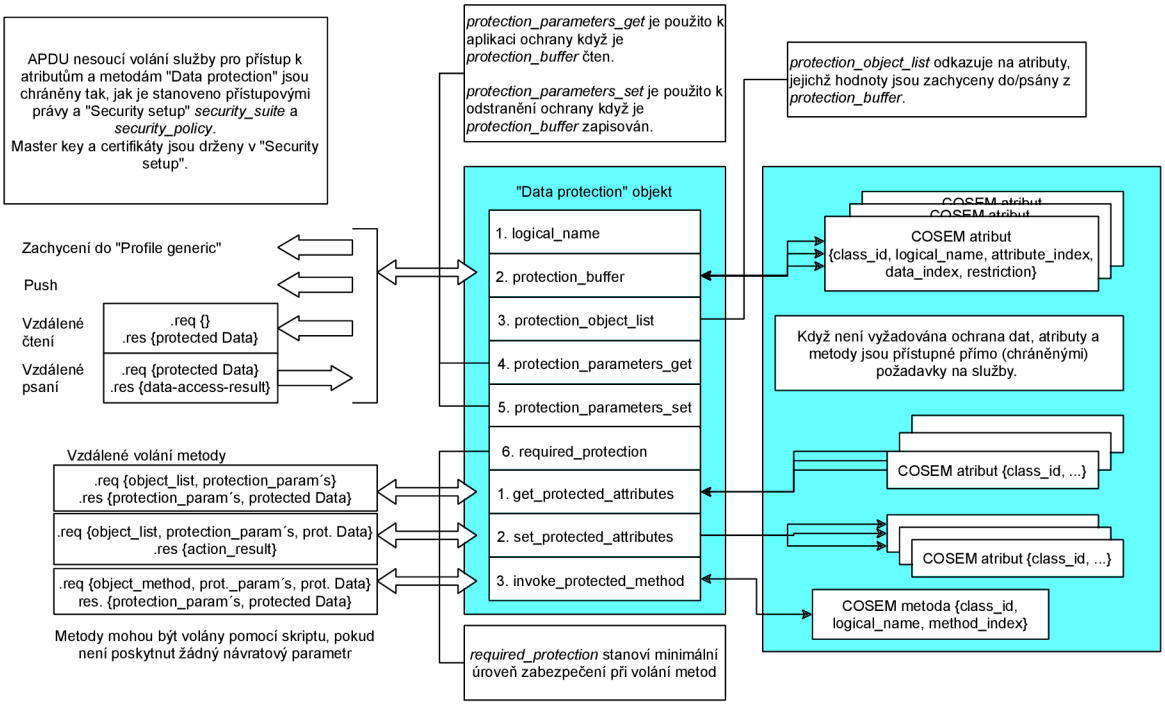
- čtení nebo zápis předdefinované sady hodnot chráněných atributů;
- uložení předdefinované sady hodnot chráněných atributů do Profile generic objektů pro pozdější vyhledávání;
- odesílání předdefinové sady hodnot chráněných atributů pomocí push metody;
- čtení nebo zápis vybraných atributů jiných COSEM objektů s ochranou;
- volání metody jiného COSEM objektu s voláním ochráněné metody a návratovými paramtery.

Ochrana může zahrnovat jakoukoliv kombinaci autentizace, šifrování, digitálního podpisu a může být aplikována vrstveným způsobem. Strany aplikující a odstraňující ochranu jsou DLMS/COSEM server a další identifikovaná strana, kterou může být

DLMS/COSEM klient nebo třetí strana.

Použitím ochrany dat mezi serverem a třetí stranou umožňuje zachovat důvěrnost citlivých dat vůči klientovi, jehož prostřednictvím třetí strana přistupuje k serveru. Podepisování dat třetí stranou podporuje nepopiratelnost.

Parametry ochrany jsou vždy řízeny klientem, některé prvky jsou podle potřeby vyplněny serverem. Sada zabezpečení je určena objektem Security setup, odkazovaným z aktuálního objektu Association SN nebo Association LN [4]. Obrázek 3.3 ukazuje COSEM model ochrany dat a vztah „Data protection“ objektu s jinými COSEM objekty.



Obr. 3.3: COSEM model ochrany dat [4].

Pro přístup k atributům jiných COSEM objektů s chráněnými daty jsou k dispozici dva mechanismy:

- čtení nebo zápis atributu protection_buffer. Tento atribut může být zachycen v „Profile generic“ objektech nebo odeslán pomocí „Push setup“ objektů;
- vyvolání metody get_protected_attributes nebo set_protected_attributes.

Pro přístup k metodě jiného COSEM s chráněnými daty je k dispozici metoda invoke_protected_method.

APDU nesoucí vyvolání služby pro přístup k atributům a metodám „Data protection“ objektů jsou chráněny tak, jak je stanoveno přístupovými právy k těmto atributům a metodám, a pomocí „Security setup“ security_suite a security_policy.

Master key a certifikáty jsou drženy v „Security setup“, tak jak popisuje bezpečnostní sada. Ochrana COSEM dat se aplikuje a odstraňuje v různých případech následovně:

1. Když je atribut protection_buffer čten/zachycen v „Profile generic“ objektu/ odeslán pomocí push operace:
   - jsou zachyceny atributy určené pomocí protection_object_list;
   - ochrana podle protection_parameters_get je aplikována na sadu atributů a výsledek je vložen do protection_buffer;
   - hodnota protection_buffer je vrácena/zachycena v „Profile generic“ bufferu/ odeslána push operací pomocí „Push setup“ objektů.
2. Když je psán protection_buffer:
   - chráněná data jsou napsány do protection_buffer a
   - ochrana podle protection_parameters_set je odstraněna a výsledné hodnoty atributu jsou zapsány do atributů specifikovaných v protection_object_list.
3. Když je volána metoda get_protected_attributes:
   - atributy, určené prvkem object_list z get_protected_attributes_request, jsou zachyceny;
   - je použita ochrana podle atributu required_protection a odezvy protection_parameters. Pokud protection_parameters nesplňuje required_protection, volání metody selže;
   - hodnoty chráněných atributů jsou vráceny.
4. Když metoda set_protected_attributes je volána:
   - ochrana na protected_attributes je ověřena a odstraněna pomocí protection_parameters, které musí splňovat required_protection;
   - výsledné hodnoty atributu jsou vloženy do atributů specifikovaných prvkem object_list.
5. Když je volána metoda invoke_protected_method:
   - ochrana od parametrů volání chráněné metody je odstaněna pomocí parametrů ochrany v požadavku, který musí splňovat required_protection;
   - metoda, která je určena prvkem object_method z invoke_protected _method_request, je volána s tímto parametrem volání metod;
   - na návratových parametrech je aplikována ochrana používající response protection parametry, které musí splňovat required_protection. Pokud protection_parameters nesplňují required_protection, volání metody selže;
   - jsou vráceny návratové parametry chráněné metody.

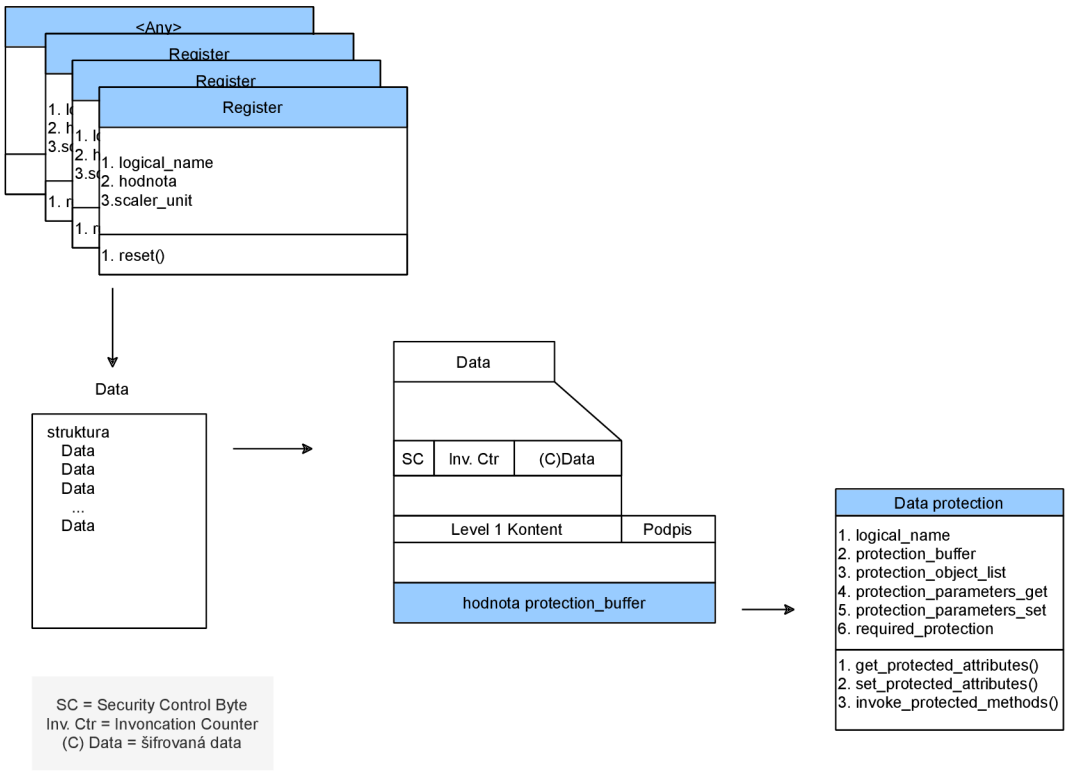
Obrázek 3.4 ukazuje příklad, jak jsou chráněna data v protection_buffer konstruována z atributů určených z protection_object_list podle protection_parameters_get.

Když je načten atribut protection_buffer, provedou se následující kroky:

1. předpoklady: protection_object_list, protection_parameters_get, master klíč, dohoda o klíči a certifikáty digitálního podpisu podle potřeby;
2. zachycení atributů COSEM objektů určené pomocí protection_object_list a vytvoření Dat, struktury obsahující jednotlivá data zachycených atributů;
3. ochrana dat podle protection_parameters_get. Na obrázku 3.3 jsou aplikované dvě vrstvy zabezpečení. První vrstva je kombinace komprese, šifrování a autentizace určené podle Security control byte SC, z tohoto pak vznikne (C)Data. Druhá vrstva je digitální podpis aplikovaný na (C)Data.
4. vložení chráněných dat do protection_buffer;
5. vrácení hodnoty protection_buffer.

Může být nutné přečíst také protection_parameters_get k získání ochraných parametrů k ověření nebo odstranění ochrany příjemcem.

Čítač volání, použitý při aplikování nebo odstranění ochrany, souvisí s použitým klíčem. Když je aplikována ochrana, čítač se navýší. Když se změní klíč, čítač je vynulován.



Obr. 3.4: Vytváření protection_buffer [4].

4 Implementace zabezpečených push zpráv

4.1 Gurux knihovna

Pro práci byla vybrána knihovna DLMS v jazyce Java od Gurux Ltd, která je šířena pod duální licencí a dostupná ke stažení na GitHub [10]. Knihovna je open source a vztahuje se na ni licence GNU GPL v2 [11].

Finská společnost Gurux Ltd. byla založena v roce 1998 a specializuje se na DLMS protokol. Poskytuje také pomoc na fóru, které je pravidelně čteno Gurux developery a aktivními uživateli [12].

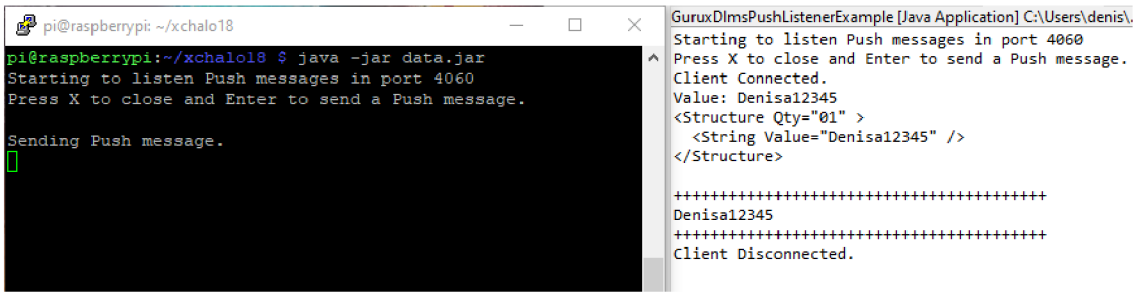
Pro práci s knihovnou bylo zvoleno vývojové prostředí Eclipse IDE 2021-09 [13].

4.2 Program

Pro odesílání push zpráv je upraven příklad gurux.dlms.push.listener, který je dostupný ke stažení na GitHubu. Tento příklad odesílá push zprávy a přijaté push zprávy zpracovává a vypisuje do konzole. Jako server je využita platforma Raspberry Pi a jako klient počítač. V programu je nastaveno, na jakém portu naslouchá pro příchozí push zprávy a na jaký port a IP adresu se push zprávy odesílají. V případě, že jsou push zprávy šifrovány, využívá se Security Suite 0 s výchozími klíči.

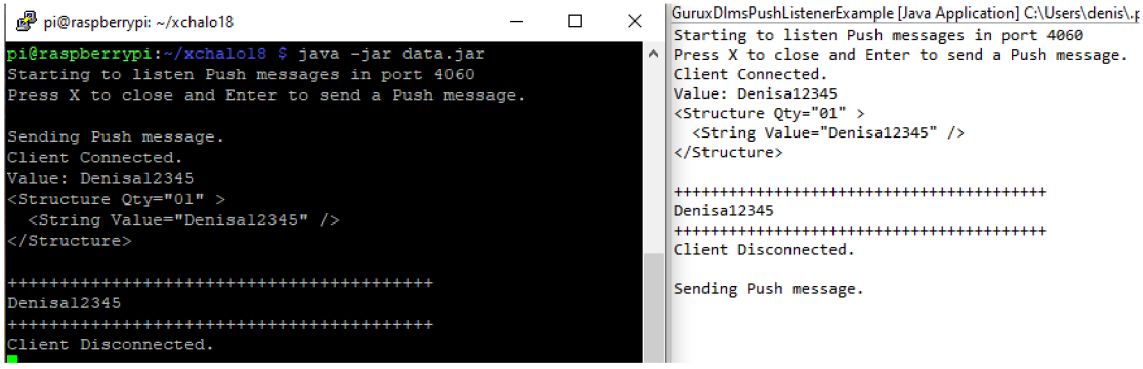
Pro vzdálené připojení k Raspberry Pi je použita apliakce PuTTY [14] a pro přesouvání souborů na Raspberry Pi je použita aplikace WinSCP verze 5.19 [15]. Aplikace PuTTY je využita pouze pro ovládání, na funkčnost komunikace mezi Raspberry Pi a počítačem nemá vliv. Na obrázku 4.1 je vidět komunikace s Raspberry Pi a PC:

1. V konzoli je vypsáno, na jakém portu zařízení naslouchá pro příchozí push zprávy a instrukce pro ovládání programu.
2. Po stiknutí enter na Raspberry Pi vypíše program do konzole hlášení o odesílání push zprávy.
3. Probíhá navázání spojení s klientem, následně odesílání push zprávy a ukončení spojení s klientem.
4. Na konzoli klienta se vypíše hlášení o navázání spojení s druhou stranou, následně se vypíše obsah přijaté push zprávy a nakonec hlášení o ukončení spojení.



Obr. 4.1: Odesílání push zprávy z Raspberry Pi na PC.

Tato komunikace funguje oboustranně, jak lze vidět na obrázku 4.2.



Obr. 4.2: Odesílání push zprávy z PC na Raspberry Pi.

Pro účely testování velikostí zpráv byly vybrány tři objekty:

- GXDLMSClock,
- GXDLMSData,
- GXDLMSRegister.

Objekt GXDLMSClock je používán pro ukládání aktuálního času. Tento objekt má 9 atributů, v push zprávě je obsažen druhý atribut Time. Objekt GXDLMSData je používán pro ukládání obecných dat, jako jsou konfigurační data a parametry. Tento objekt má pouze dva atributy a v push zprávě je obsažen druhý atribut Value. Objekt GXDLMSRegister je používán k uložení hodnoty s přidruženým scalerem a jednotkou. Tento objekt má 3 atributy, v push zprávě je obsažen druhý atribut Value, který nese samotnou hodnotu odečtu, a třetí atribut Scaler and unit (měřítko hodnoty a jednotka) [16]. Měřítko udává pozici desetinné čárky u naměřené hodnoty pomocí mocnin deseti. Jednotka pak udává, jaká je měřená veličina.

Výpis 4.1 ukazuje vytvoření objektů a naplnění hodnotami, které mají být odeslány v push zprávě. Objekt clock je naplněn aktuálním časem těsně před odesláním push zprávy.

Výpis 4.1: Vytvoření instancí a naplnění hodnotami.

```
GXDLMSClock clock = new GXDLMSClock();

GXDLMSRegister register =
    new GXDLMSRegister("0.0.96.9.0.255");
register.setUnit(Unit.TEMPERATURE);
register.setValue(562);
register.setScaler(10);

GXDLMSData data = new GXDLMSData("0.0.42.0.0.255");
data.setValue("Denisa12345");
```

Výpis 4.2 ukazuje vytvoření objektu GXDLMSSecureNotify, který se stará o odesílání push zpráv a nastavení jejich zabezpečení. Lze vybrat až 4 zabezpečení push zprávy:

- bez zabezpečení,
- zabezpečení autentizací,
- zabezpečení šifrováním,
- zabezpečením autentizovaným šifrováním.

Výpis 4.2: Nastavení zabezpečení.

```
GXDLMSSecureNotify cl =
  new GXDLMSSecureNotify(true, 1, 1, InterfaceType.WRAPPER);
cl.getCiphering().setSecurity(Security.ENCRYPTION);
```

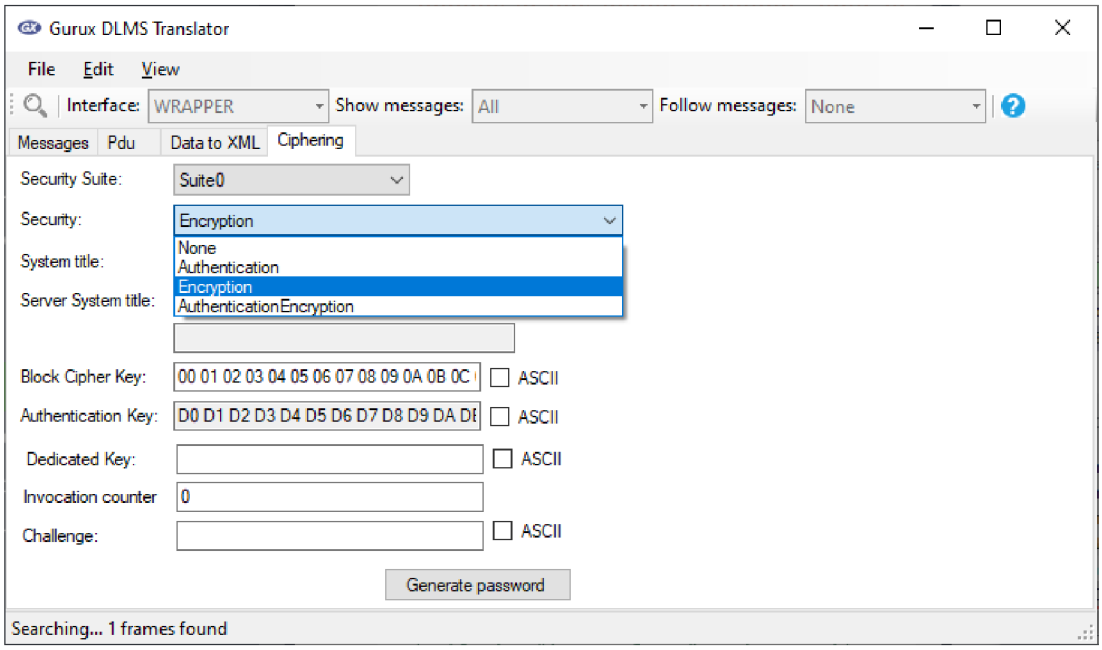
Třída GXDLMSPushListener zpracovává přijaté push zprávy. Po stáhnutí příkladu program používá jen objekt GXDLMSClock a v kontruktoru třídy GXDLMSPushListener je specifikováno, který objekt a atribut je v push zprávě. Proto je konstruktor upraven pro každý objekt zvlášť.

4.3 Ověření šifrování

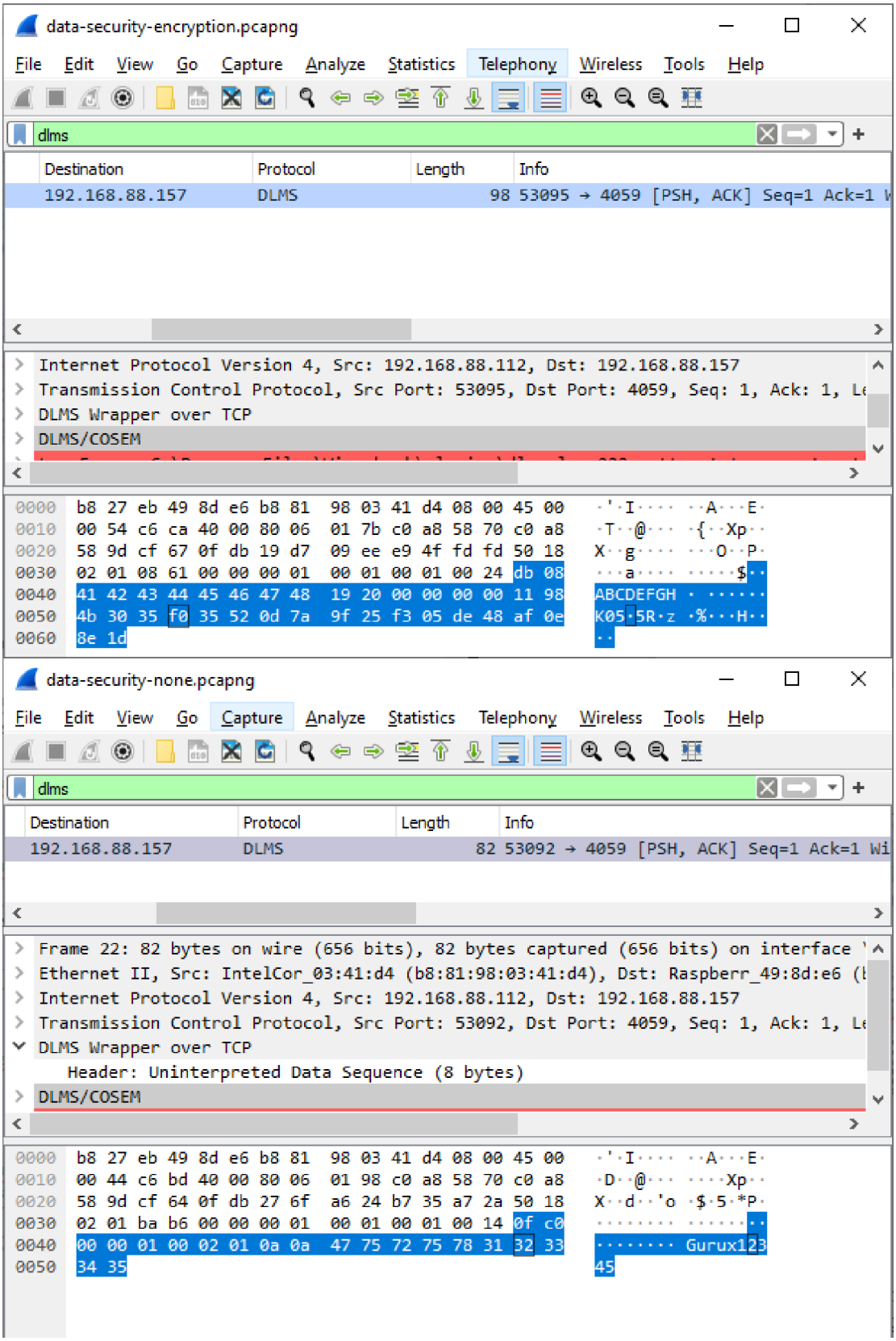
Pro ověření funkčnosti odesílání šifrovaných zpráv a porovnání velikostí šifrovaných a nešifrovaných zpráv je použit program Wireshark [17] s doplňkem pro překlad DLMS zpráv, který je dostupný na GitHubu [18].

Na obrázku 4.4 je vidět záznam komunikace mezi Raspberry Pi a počítačem. Horní okno Wiresharku zobrazuje šifrovanou push zprávu a dolní okno nešifrovanou. DLMS PDU jsou zvýrazněny modře. U nešifrované zprávy je vidět čitelná odeslaná hodnota.

Šifrované zprávy je možné dešifrovat pomocí nástroje DLMS Translator v aplikaci Gurux GXDLMSDirector, dostupné na stránkách firmy Gurux [19]. Tento nástroj překládá surová data do čitelné formy a dešifruje šifrované zprávy podle nastavení v záložce Ciphering, která je zobrzena na obrázku 4.3. Nastavuje se zde druh Security Suite, druh zabezpečení a použité klíče.

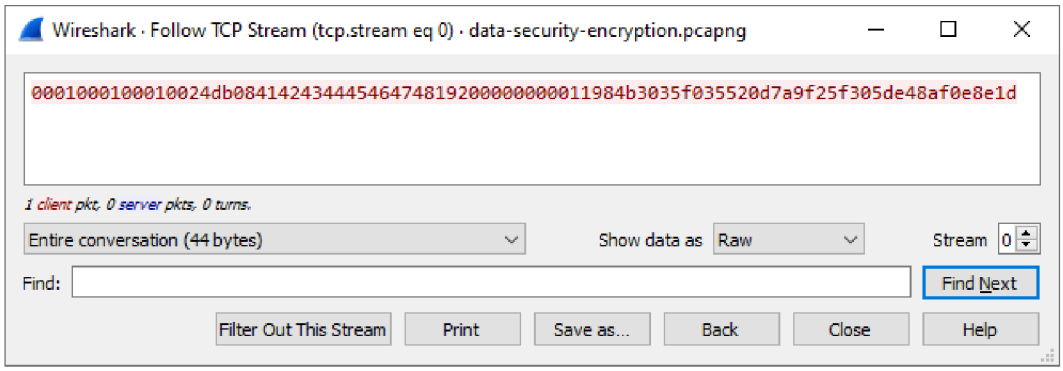


Obr. 4.3: Záložka Ciphering v Gurux DLMS Translator.



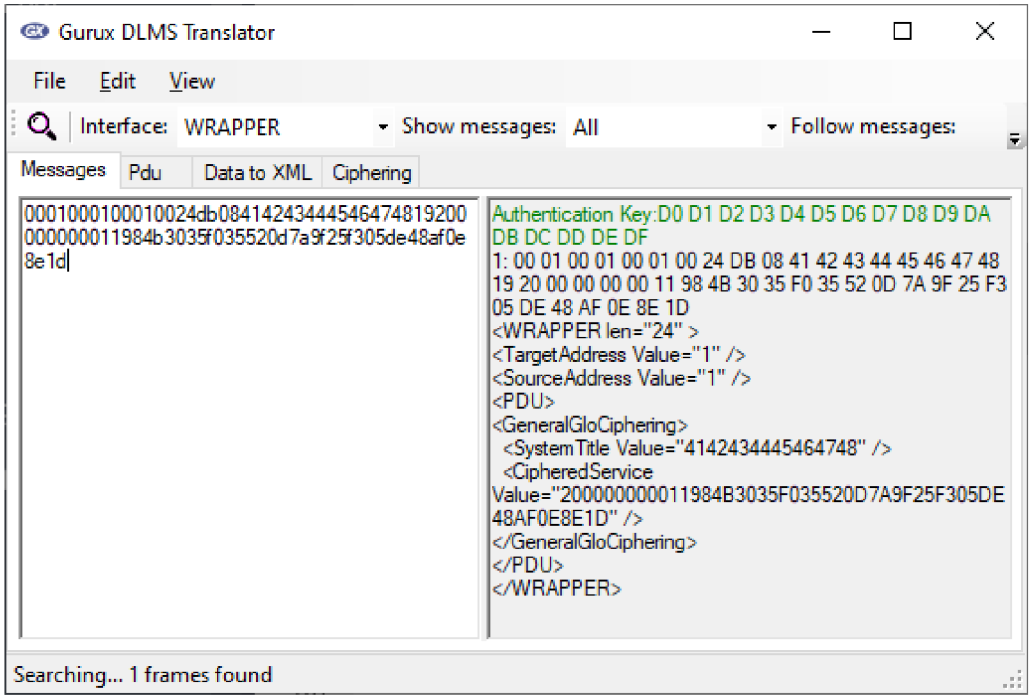
Obr. 4.4: Šifrovaná a nešifrovaná push zpráva.

Pro překlad a dešifrování zpráv je potřeba získat surová data. Ta jsou k získání v programu Wireshark v sekci Conversations po zachycení komunikace mezi Raspberry Pi a počítačem, volba zobrazení dat jako „raw“. Pro lepší hledání konkrétní komunikace je použit filtr pro zobrazení komunikace s použitým protokolem DLMS. Obrázek 4.5 ukazuje získání neupravených dat pro šifrovanou zprávu.



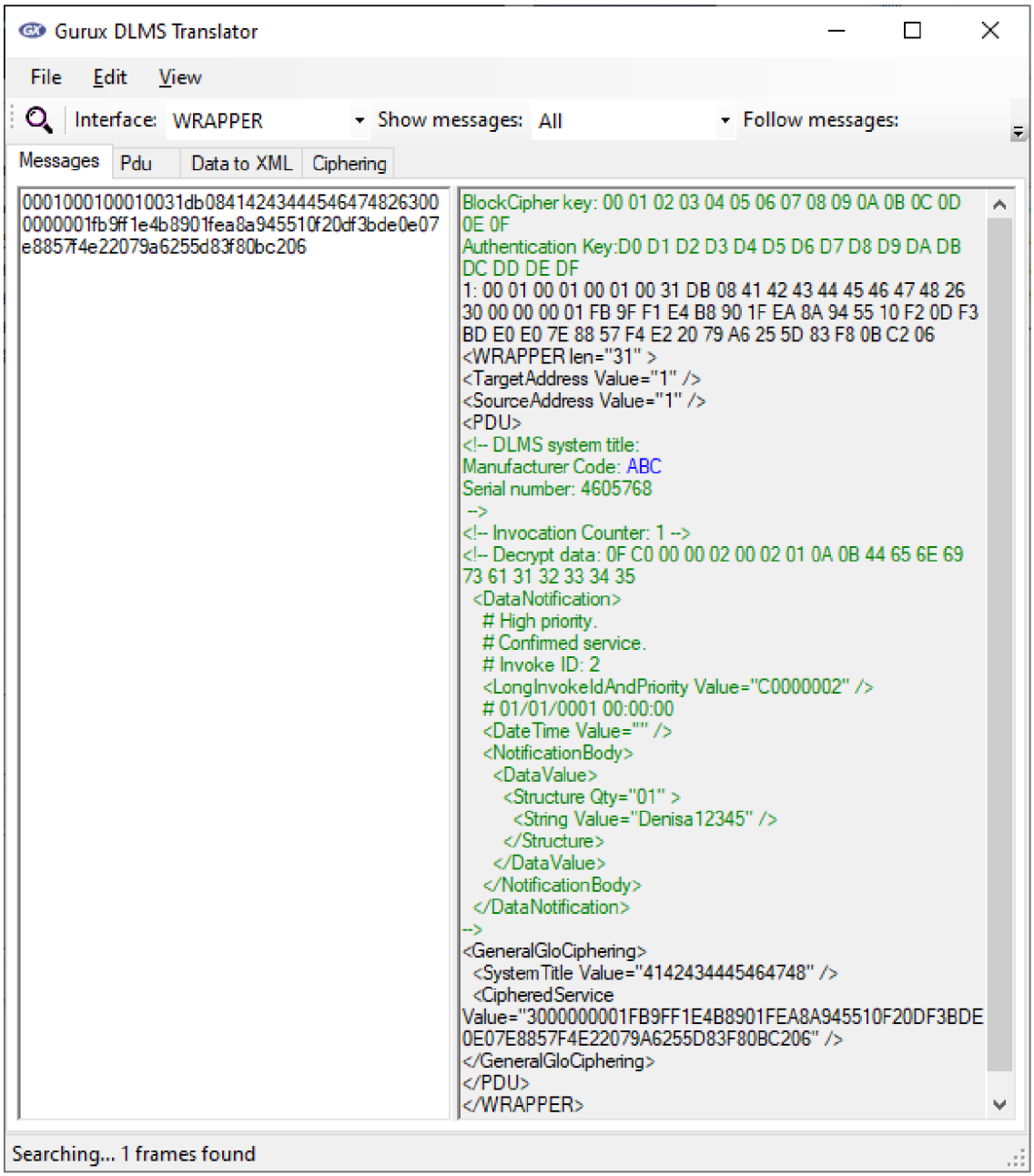
Obr. 4.5: Získání surových dat ve Wireshark.

Získaná data lze přeložit pomocí levého okna překladače, které je na obrázku 4.6. V nastavení Ciphering není nastaven použitý klíč, proto je v pravém okně zobrazena šifrovaná zpráva. Je zde vidět, že je využit wrapper, hodnoty adres a že je použito obecné globální šifrování. V obecném globálním šifrování lze ještě vidět hodnota System Title, která značí odesílající stranu.



Obr. 4.6: Překlad šifrované zprávy.

Po přidání klíče do nastavení Ciphering je pak zobrazena dešifrovaná zpráva, viz obrázek 4.7. Zelená část jsou dešifrovaná data (Denisa12345), viz výpis 4.1, kde je tato hodnota definovaná v programu.

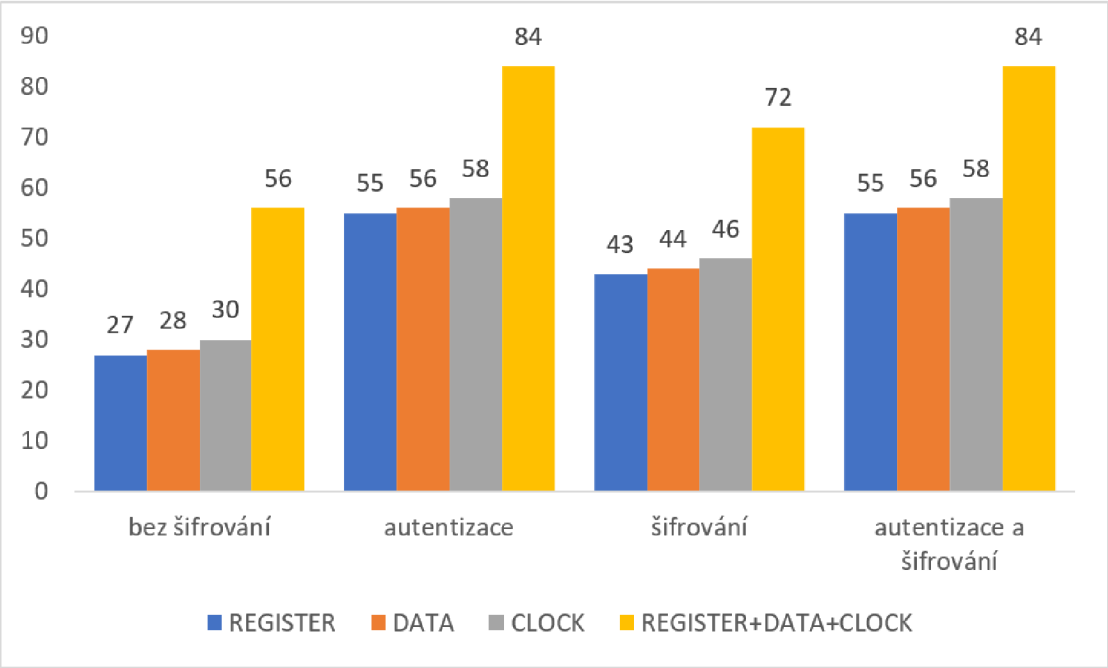


Obr. 4.7: Překlad dešifrované zprávy.

4.4 Datové objemy zpráv

Pro zabezpečení se využívá Security Suite 0. Pro šifrování je tedy využíván standard AES-128 s provozním režimem blokových šifer GCM.

Na obrázku 4.8 je vidět graf velikostí zpráv tří objektů s různým zabezpečením. Uvedené velikosti jsou v bytech.



Obr. 4.8: Graf velikostí odeslaných push zpráv s různým zabezpečením.

Obrázek 4.9 ukazuje zachycená „raw“ data stejné push zprávy s různými druhy zabezpečení. Mezerou jsou odděleny bloky znázorňující strukturu na obrázku 4.10.

Bez zabezpečení

```
0001000100010014                                0fc00000010002010a0a47757275783132333435
```

Autentizace

```
0001000100010030 db0841424344454647482510000000000 0fc00000010002010a0a47757275783132333435 3447d5429442a09ee327e446
```

Šifrování

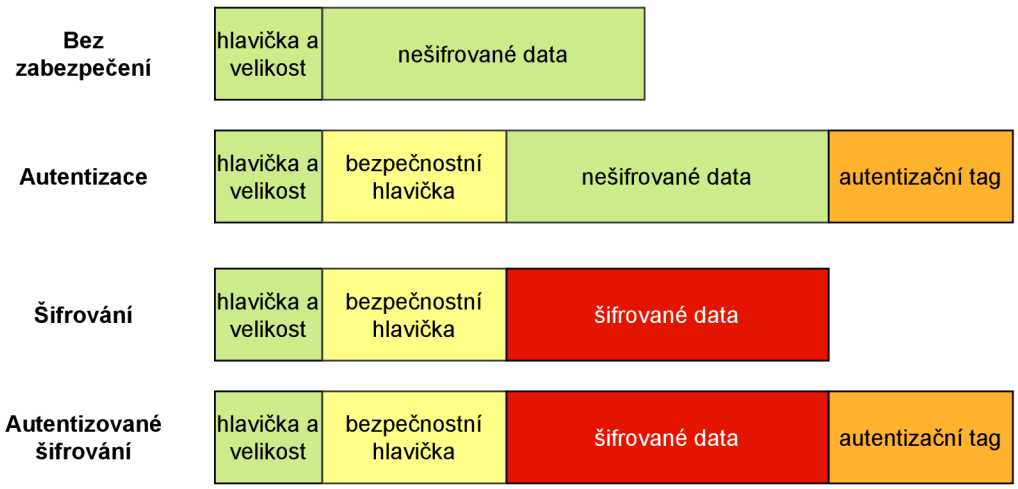
```
0001000100010024 db0841424344454647481920000000000 11984b3035f035520d7a9f25f305de48af0e8e1d
```

Autentizované šifrování

```
0001000100010030 db08414243444546474825300000000 11984b3035f035520d7a9f25f305de48af0e8e1d 2fc1b36c2b00433675fef0be
```

Obr. 4.9: Příklady zachycených surových dat odeslané push zprávy při různém druhu zabezpečení.

Struktura zpráv záleží na využitém druhu zabezpečení. Bez zabezpečení zpráva obsahuje hlavičku, velikost a nešifrovaná data. Při použití autentizace se na konec zprávy přidává autentizační tag. Bezpečnostní hlavička se přidává při použití autentizace nebo šifrování a obsahuje SC byte s druhem zabezpečení (SC-A, SC-E, SC-EA) a čítač volání.



Obr. 4.10: Struktura zpráv při různém zabezpečení.

Tabulka 4.1 ukazuje velikosti různých push zpráv při různých zabezpečeních a procentuální zvětšení proti nezabezpečené zprávě. Hlavička a délka, bezpečnostní hlavička a autentizační tag mají konstantní velikost. Data jsou jediná část zprávy, která mění svou velikost. Šifrování dat nemění jejich velikost, protože je využíván GCM mód, který nemění šifrováním jejich velikost. Čím jsou data větší, tím je menší poměrné zvětšení při zabezpečení zprávy. Objekty jsou v tabulce seřazeny od největšího procentuálního zvětšení po nejmenší. Nejmenší zvětšení má push zpráva, která obsahuje všechny tři objekty. Proto je nejvhodnější odesílat více objektů, výsledná zabezpečená zpráva pak nebude o moc zvětšena.

Tab. 4.1: Velikosti zpráv v bytech.

	Hlavička a délka	**Bezpečnostní hlavička**	**Data**	**Autentizační tag**	**Celkem**	**Zvětšení %**
REGISTER						
Bez šifrování	8	0	19	0	27	100
Autentizace	8	16	19	12	55	203
Šifrování	8	16	19	0	43	159
Autentizované šifrování	8	16	19	12	55	203
DATA						
Bez šifrování	8	0	20	0	28	100
Autentizace	8	16	20	12	56	200
Šifrování	8	16	20	0	44	157
Autentizované šifrování	8	16	20	12	56	200
CLOCK						
Bez šifrování	8	0	22	0	30	100
Autentizace	8	16	22	12	58	193
Šifrování	8	16	22	0	46	153
Autentizované šifrování	8	16	22	12	58	193
REGISTER+DATA+CLOCK						
Bez šifrování	8	0	48	0	56	100
Autentizace	8	16	48	12	84	150
Šifrování	8	16	48	0	72	129
Autentizované šifrování	8	16	48	12	84	150

Závěr

Bakalářská práce se v úvodu věnuje druhům zabezpečení v DLMS/COSEM, jak COSEM chrání data a jak funguje push. V novější verzi protokolu obsahuje Push Setup IC více atributů, než je ve využívané knihovně implementováno. Jedním z těchto chybějících atributů je parametr zabezpečení push, který specifikuje způsob zabezpečení určený jen pro push a předává je službě DataNotfication.

Pro praktickou část byla využita knihovna od firmy Gurux v jazyce Java a byl upraven jejich příklad pro odesílání a zpracovávání přijatých push zpráv. Byly vybrány testovací objekty, které byly vkládany do push zpráv. Protože bylo využito více objektů, byl upraven konstruktor pro třídu GXDLMSPushListener tak, aby očekával vybrané objekty a jejich atributy a mohl je zobrazit do konzole.

Pro účely testování byl každý objekt zaslán samostatně a následně všechny dohromady. Pro zabezpečení push zpráv byl využit Security Suite 0 s výchozími klíči a otestovány byly 4 druhy zabezpečení. Push zprávy byly zachytávány pomocí aplikace Wireshark a komunikace probíhala mezi Raspberry Pi a počítačem. Pomocí Wireshark a Gurux DLMS Translator bylo ověřeno funkční zabezpečení.

V poslední části jsou porovnány velikosti odesílaných push zpráv s různým zabezpečením. Je popsána struktura těchto zpráv a v tabulce 4.1 jsou shrnuty velikosti jednotlivých částí zpráv podle typu zabezpečení a je zde procentuálně vyjadřeno zvětšení zpráv proti nezabezpečeným push zprávám. Zprávy se podle typu zabezpečení zvětšují o konstantní velikost částí bezpečnostní hlavička (16 bytů) a autentizační tag (12 bytů). Čím větší byla velikost dat, tím bylo menší poměrné zvětšení celé zabezpečené zprávy. Proto je vzhodné odesílat více objektů v push zprávách, aby byl výsledný nárůst zabezpečení co nejmenší.

Literatura

[1] *Overview. DLMS* [online]. [cit. 2021-11-08]. Dostupné z: <https://www.dlms.com/dlms-cosem/overview>

[2] *DLMS* [online]. [cit. 2021-11-08]. Dostupné z: <https://www.dlms.com/>

[3] *Green Book: Architecture and Protocols. DLMS*. Zug, Switzerland: DLMS User Association, 2020.

[4] *Blue Book: COSEM Inteface Classes and OBIS Object Identification System. DLMS*. Zug, Switzerland: DLMS User Association, 2020.

[5] *MATOUŠEK, Petr. Analysis of DLMS Protocol* [online]. 2017 [cit. 2021-12-07]. Dostupné z: <https://www.fit.vut.cz/research/publication-file/11616/TR-DLMS.pdf>

[6] HOFFMANN, Stefan G., Robin MASSINK a Gerd BUMILLER. New security features in DLMS/COSEM — A comparison to the smart meter gateway. In: *2015 IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA)* [online]. IEEE, 2015, 2015, s. 1-6 [cit. 2022-05-28]. ISBN 978-1-5090-1238-1. Dostupné z: <https://ieeexplore.ieee.org/document/7387098>

[7] LURING, Norman, Daniel SZAMEITAT, Stefan HOFFMANN a Gerd BUMILLER. Analysis of security features in DLMS/COSEM: Vulnerabilities and countermeasures. In: *2018 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)* [online]. IEEE, 2018, 2018, s. 1-5 [cit. 2022-05-28]. ISBN 978-1-5386-2453-1. Dostupné z: <https://ieeexplore.ieee.org/document/8403340>

[8] LIESKOVAN, Tomas, Jan HAJNY a Petr CIKA. Smart Grid Security: Survey and Challenges. In: *2019 11th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)* [online]. IEEE, 2019, 2019, s. 1-5 [cit. 2022-05-28]. ISBN 978-1-7281-5764-1. Dostupné z: <https://ieeexplore.ieee.org/document/8970738>

[9] *Push Setup* [online]. [cit. 2022-05-14]. Dostupné z: <https://www.gurux.fi/Gurux.DLMS.Objects.GXDLMSPushSetup>

[10] *Gurux DLMS library for Java* [online]. GitHub [cit. 2021-12-08]. Dostupné z: <https://github.com/Gurux/gurux.dlms.java>

[11] *Gurux - Dual License* [online]. [cit. 2021-12-08]. Dostupné z: <http://www.gurux.fi/index.php?q=duallicense>

[12] *Gurux - About us* [online]. [cit. 2021-12-08]. Dostupné z: <http://www.gurux.fi/AboutUs>

[13] *Eclipse* [online]. The Eclipse Foundation [cit. 2021-12-09]. Dostupné z: <https://www.eclipse.org/downloads/packages/release/2021-09/r>

[14] *PuTTY* [online]. [cit. 2022-05-24]. Dostupné z: <https://www.putty.org/>

[15] *WinSCP* [online]. [cit. 2022-05-24]. Dostupné z: <https://winscp.net/eng/download.php>

[16] *Gurux.DLMS.Objects. Gurux* [online]. [cit. 2022-05-25]. Dostupné z: <https://www.gurux.fi/Gurux.DLMS.Objects>

[17] *Wireshark* [online]. [cit. 2022-05-26]. Dostupné z: <https://www.wireshark.org/download.html>

[18] *Wireshark dissector for DLMS protocol* [online]. [cit. 2022-05-26]. Dostupné z: <https://github.com/matousp/dlms-analysis>

[19] *Download. Gurux* [online]. [cit. 2022-05-28]. Dostupné z: <http://www.gurux.fi/Download>

Seznam symbolů a zkratek

AA	Application Association
AARE	A-Associate Response
AARQ	A-Associate Request
AES	Advanced Encryption Standard
AL	Application Layer
AP	Application Process
APDU	Application Layer Protocol Data Unit
COSEM	Companion Specification for Energy Management
DLMS	Device Language Message Specification
DLMS UA	DLMS User Association
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
GCM	Galois/Counter Mode
HLS	High Level Security
IC	Interface Class
LDN	Logical Device Name
LLS	Low Level Security
LN	Logical Name
OBIS	Object identification System
PDU	Protocol Data Unit
RNG	Random Number Generator
SAP	Service Access Point
SC	Security Control byte

SC-A	Security Control byte - Authentication applied
SC-AE	Security Control byte - Authentication and Encryption applied
SC-E	Security Control byte - Encryption applied
SC-N	Security Control byte - No protection applied
SHA	Secure Hash Algorithm
SN	Short Name

A Obsah elektronické přílohy

- `/` ..... kořenový adresář přiloženého archivu
  - `Program` ..... zdrojový kód
    - `gurux.dlms.push.listener.example.zip`
    - `clock.jar`
    - `data.jar`
    - `register.jar`
    - `clock-data-register.jar`
  - `Zachycená komunikace` ..... zachycená komunikace pomocí Wiresharku
    - `clock`
      - `clock-security-aut.pcapng`
      - `clock-security-autEncryption.pcapng`
      - `clock-security-encryption.pcapng`
      - `clock-security-none.pcapng`
    - `data`
      - `data-security-aut.pcapng`
      - `data-security-autEncryption.pcapng`
      - `data-security-encryption.pcapng`
      - `data-security-none.pcapng`
    - `register`
      - `register-security-aut.pcapng`
      - `register-security-autEncryption.pcapng`
      - `register-security-encryption.pcapng`
      - `register-security-none.pcapng`
    - `objects-security-auth.pcapng`
    - `objects-security-authEncryp.pcapng`
    - `objects-security-encryp.pcapng`
    - `objects-security-none.pcapng`