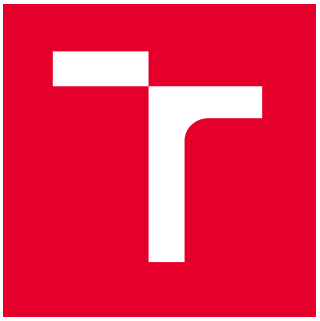


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

ZAJIŠTĚNÍ DOSTUPNOSTI A BEZPEČNOSTI V PRŮMYSLOVÝCH SÍTÍCH

ENSURING AVAILABILITY AND SECURITY IN INDUSTRIAL NETWORKS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Zdeněk Zatloukal**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Eva Holasová**
SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Bc. Zdeněk Zatloukal ***ID:*** 219360

Ročník: 2 ***Akademický rok:*** 2023/24

NÁZEV TÉMATU:

Zajištění dostupnosti a bezpečnosti v průmyslových sítích

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je implementace firmware v jazyce Python 3.7+ do zařízení s operačním systémem založeným na distribuci s linuxovým jádrem. Toto zařízení bude schopno lokálně komunikovat pomocí protokolu Modbus RTU/TCP (jedno fyzické rozhraní na protokol) a obsluhovat simultánně až 4 klienty na jednom rozhraní, přičemž bude zajištěna dostupnost všech Modbus klientů najednou včetně jejich registrů. Předpokladem je zpracování min. 20 veličin do 1 vteřiny s max. ztrátovostí dat 0,5 % za 24 hodin (dlouhodobá statistika). Vzdálená obousměrná komunikace bude probíhat s cloudovým server pomocí protokolu MQTT. Předpokladem komunikace je umožnění dvou režimů VPN (IPSec) a šifrované verze MQTT. Zařízení bude využívat HTTPS záložní kanál pro případ výpadku MQTT. Požadavkem je opětovné připojení k serveru v případě výpadku spojení a následného zaslání historicky naměřených dat. V teoretické části se student seznámí s problematikou průmyslových sítí, komunikačními protokoly MQTT a Modbus RTU/TCP a kybernetickou bezpečnost. Na základě požadavků navrhne architektonický model a provede formální kontrolu. V praktické části provede implementaci a hardening celého řešení. Následně bude realizováno experimentální testování pro ověření splnění požadavků včetně případné optimalizace. V závěru student vytvoří dokumentaci k celému řešení, včetně technické specifikace a vyhodnocení užitných parametrů.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucí práce.

Termín zadání: 5.2.2024 ***Termín odevzdání:*** 21.5.2024

Vedoucí práce: Ing. Eva Holasová

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

ABSTRAKT

V současné době je klíčovým aspektem průmyslové automatizace zajištění vysoké úrovně bezpečnosti, spolehlivosti a dostupnosti dat průmyslových komunikačních sítí. Vzhledem k rostoucím kybernetickým hrozbám je nezbytné rozvíjet a implementovat pokročilé strategie pro ochranu průmyslových infrastruktur.

Tato práce si klade za cíl provést komplexní analýzu a vývoj bezpečnostních řešení pro průmyslové sítě se zaměřením na komunikační protokoly Modbus RTU/TCP a MQTT, které jsou zásadní pro efektivní řízení moderních průmyslových operací.

V práci jsou využity metodologické přístupy kombinující teoretický výzkum s praktickými experimenty. Zahrnuje analýzu stávajících protokolů, návrh zabezpečení komunikace prostřednictvím těchto protokolů a jejich následné testování v kontrolovaném průmyslovém prostředí pro ověření bezpečnosti, efektivity a spolehlivosti.

Analýza odhalila podstatné nedostatky ve zabezpečení stávajících systémů a vedla k integraci nových bezpečnostních opatření do průmyslového rozvaděče, která značně zlepšila ochranu dat a komunikační infrastruktury. Implementovaná řešení demonstrují významný pokrok v ochraně průmyslových sítí proti různým typům útoků.

Tato práce představuje významný přínos v oblasti bezpečnosti průmyslových sítí. Výsledky práce poskytují cenné informace pro další vývoj bezpečnostních technologií v průmyslu a nabízejí praktické návody pro inženýry a techniky, kteří se zabývají zlepšováním stavu bezpečnosti průmyslových systémů.

KLÍČOVÁ SLOVA

bezpečnost, dostupnost dat, grafické uživatelské rozhraní, HTTPS, IPsec, komunikace v reálném čase, kybernetická bezpečnost, Modbus RTU, Modbus TCP, MQTT, průmyslová zařízení, průmyslové sítě, VPN, Wireguard, zabezpečení průmyslových zařízení

ABSTRACT

Currently, a key aspect of industrial automation is ensuring a high level of security, reliability, and data availability in industrial communication networks. Given the rising cyber threats, it is essential to develop and implement advanced strategies for protecting industrial infrastructures.

This thesis aims to perform a comprehensive analysis and development of security solutions for industrial networks, focusing on the Modbus RTU/TCP and MQTT communication protocols, which are crucial for the effective management of modern industrial operations.

The work employs methodological approaches that combine theoretical research with practical experiments. It includes an analysis of existing protocols, the design of communication security through these protocols, and their subsequent testing in a controlled industrial environment to verify security, efficiency, and reliability.

The analysis revealed significant deficiencies in the security of existing systems and led to the integration of new security measures into the industrial switchgear, which significantly improved the protection of data and communication infrastructure. The implemented solutions demonstrate significant progress in protecting industrial networks against various types of attacks.

This thesis represents a significant contribution to the field of industrial network security. The results of the work provide valuable information for the further development of security technologies in the industry and offer practical guides for engineers and technicians who are working on improving the security status of industrial systems.

KEYWORDS

cybersecurity, data availability, graphical user interface, HTTPS, IPsec, industrial devices, industrial networks, Modbus RTU, Modbus TCP, MQTT, real-time communication, security, security of industrial devices, VPN, Wireguard

ZATLOUKAL, Zdeněk. *Zajištění dostupnosti a bezpečnosti v průmyslových sítích*. Diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2024. Vedoucí práce: Ing. Eva Holasová

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. Zdeněk Zatloukal
VUT ID autora:	219360
Typ práce:	Diplomová práce
Akademický rok:	2023/24
Téma závěrečné práce:	Zajištění dostupnosti a bezpečnosti v průmyslových sítích

Prohlašuji, že svou závěrečnou práci jsem vypracoval 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 autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné 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 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 autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucí mé diplomové práce, paní Ing. Evě Holasové, za odborné vedení, konzultace, nekonečnou trpělivost, stále přítomnou ochotu a podnětné návrhy, bez kterých by tato diplomová práce nemohla vzniknout. Dále bych chtěl poděkovat společnosti EASYCON Solution s.r.o. za poskytnuté zázemí a věcné podněty k práci. Speciální díky patří Michalovi Novotnému za pravidelné konzultace, které mi otevřely nové úhly pohledu při řešení problematiky této práce.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Vzhledem k rostoucí integraci a automatizaci průmyslových procesů, se stále více pozornosti přikládá vývoji a implementaci průmyslových sítí, které jsou základem pro efektivní a bezpečný chod moderních průmyslových zařízení. Tato diplomová práce nabízí komplexní analýzu klíčových aspektů návrhu, implementace a zabezpečení průmyslových sítí, které jsou zásadní pro zajištění bezpečného a spolehlivého provozu v současném průmyslovém prostředí.

První kapitola je zaměřena na definici průmyslových sítí a jejich důležitosti. Zabývá se srovnáním průmyslových a informačních sítí, popisuje architekturu průmyslových řídicích systémů v rámci Purdue modelu a současný trend Průmyslu 4.0. Ten přináší nové výzvy a přístupy v oblasti průmyslových technologií. Následující kapitola se věnuje bezpečnosti průmyslových sítí. Prozkoumává historii útoků na průmyslové sítě a analyzuje různé vektory útoku, jež tyto sítě ohrožují. Podrobně se také zabývá metodami obrany proti útokům, včetně využití technologií jako je TLS (Transport Layer Security), VPN (Virtual Private Network) a MACsec (Media Access Control Security) pro zvýšení bezpečnosti komunikace v různých vrstvách architektury TCP/IP (Transmission Control Protocol/Internet Protocol).

Třetí kapitola je zaměřena na komunikační protokoly využívané v průmyslových sítích, především protokoly Modbus (v jeho variantách TCP –Transmission Control Protocol a RTU – Remote Terminal Unit) a MQTT (Message Queuing Telemetry Transport). Dále obsahuje popis těchto protokolů, jejich bezpečnostní aspekty a využití v praxi. Tato kapitola klade zvláštní důraz na analýzu bezpečnostních rizik a možnosti jejich mitigace. Následující kapitola se soustředí na návrh a bezpečnostní aspekty síťové komunikace rozvaděče ESCON. Popisuje, jak byla navržena architektura a zabezpečení tohoto zařízení, včetně specifických bezpečnostních opatření a hardeningu zařízení.

V páté a šesté kapitole je rozebrána implementace a testování komunikace přes Modbus RTU a Modbus TCP. Jsou zde popsány praktické kroky zapojení, testování a vyhodnocení efektivity a bezpečnosti implementovaných řešení. Kapitola sedmá pak rozebírá detailněji implementaci a bezpečnost MQTT a HTTPS (Hyper Text Transfer Protocol Secure) komunikací, včetně vývoje a testování softwarových komponent, které tyto protokoly podporují.

Nad rámec zadání práce se kapitola osmá zaměřuje na komplexní systémy zabezpečení průmyslových zařízení, zahrnující aktivní ochranu a monitorování zařízení v technologické síti průmyslových objektů. Zvláštní pozornost je v této kapitole věnována grafickému uživatelskému rozhraní pro zjednodušenou konfiguraci Wireguard komunikace, která umožňuje uživatelům intuitivně a efektivně konfigurovat zabezpečený síťový tunel.

1 Definice průmyslových sítí

Průmyslové sítě, nazývané také jako sítě operačních technologií (Operational technology networks – OT networks), slouží k monitorování a řízení fyzických zařízení, procesů nebo událostí. Tyto sítě jsou navržené pro využití v reálném čase za účelem realizace přesné komunikace zahrnující deterministickou komunikaci při kontrole a monitorování dat. Z důvodu vysokých požadavků na přesnost, rychlost a dostupnost komunikace nejsou do průmyslových protokolů plně integrovány základní bezpečnostní požadavky (autentizace, šifrování). Autentizace je proces ověření identity osoby, entity nebo systému za účelem získání přístupu do systému. Oproti tomu je šifrování proces transformace strojově či lidsky čitelných informací do podoby s nulovou informační hodnotou. Tyto sítě jsou vzhledem k čím dál vyššímu počtu konvergovaných systémů, tj. systémů spojujících různé přenosové technologie a služby do jedné inteligentní sítě (např. do IP sítí), více vystaveny kybernetickým útokům. Tuto situaci je třeba řešit v nižších vrstvách síťových architektur. Více o této problematice je popsáno v kapitole 2 – Bezpečnost průmyslových sítí [34].

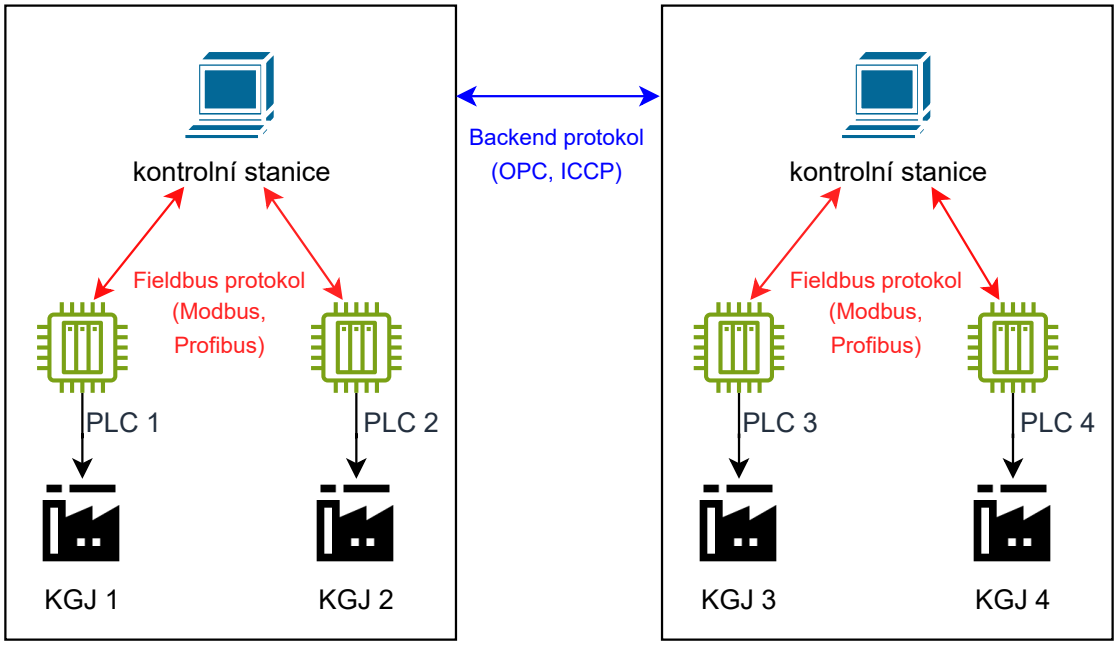
Průmyslové sítě zahrnují systémy monitorování a řízení průmyslových procesů, tzv. Industrial Control Systems (ICS), jež jsou složeny ze široké škály různých zařízení a komponent se specifickou rolí v systému:

- Programovatelný logický automat (PLC) – programovatelný řídicí mikroprocesor kontrolovaného stroje,
- Jednotka vzdáleného řízení (RTU) – vzdálený programovatelný řídicí mikroprocesor pro sběr a přenos dat do centrálního kontrolního systému,
- Inteligentní elektronické zařízení (IED) – více procesorové zařízení s vyšší komplexitou (provádí více operací najednou – monitorování, měření, komunikace s řídicími systémy),
- Inženýrské pracoviště (EW) – stolní počítač sloužící se specifickým softwarem pro správu PLC, RTU, IED,
- Rozhraní mezi člověkem a softwarem (HMI) – software pro kontrolu a monitoring kontrolovaných výrobních procesů,
- Datové úložiště (DH) – aplikace sbírající data v reálném čase z běžících procesů s agregací do centrální databáze za účelem analýzy,
- Komunikační brány (CG) – zařízení zajišťující překlad protokolů na hranici dvou heterogenních systémů,
- Hlavní řídící jednotka (MTU) – zařízení spravující komunikaci s PLC nebo RTU, pomocí komunikace stylu controller-device iniciuje komunikaci, sbírá data a zpracovává je,
- Systém pro dohledové řízení a sběr dat (SCADA) – systém pro centrální kontrolu a monitorování dat získaných ze vzdálených lokalit [7].

Funkcionality jednotlivých prvků v systému ICS uvedených výše lze shrnout do několika vět. Kontrolní zařízení (MTU, IED) interagují s kontrolovanými zařízeními (RTU, PLC), které přímo kooperují s průmyslových procesem. Zařízení pro uživatelské řízení (HMI, Frontend procesor, Inženýrské pracoviště, Datové úložiště) jsou využívané pro řízení a správu dat systému ICS. SCADA a komunikační brány propojují jednotlivé komponenty mezi sebou a zajišťují jejich vzájemnou komunikaci.

Průmyslové protokoly

V průmyslové automatizaci se používají specializované průmyslové protokoly přizpůsobené požadavkům na řízení průmyslových procesů. Průmyslové protokoly se dělí na dva typy – Fieldbus protokoly a Backend protokoly, viz obr. 1.1 [34].



Obr. 1.1: Znázornění rozdílu mezi Fieldbus a Backend protokoly

Fieldbus protokol obecně popisuje síťové systémy navržené k přenosu digitálních informací mezi fyzickými zařízeními. Je využíván v systémech pracujících v reálném čase, kde dochází k výměně dat mezi různými průmyslovými komponentami. Také zajišťuje komunikaci mezi fyzickými zařízeními, jako jsou senzory, akční prvky, programovatelné logické automaty a kontrolními systémy pro sběr dat, řízení a dohled (kontrolní stanice SCADA). Fieldbus protokoly se řadí do kategorie průmyslových sítí, které se zaměřují na minimalizaci kabeláže, zjednodušení diagnostiky a poskytování flexibilní komunikace v reálném čase. Spojení těchto funkcí umožňuje syn-

chronizaci a koordinaci průmyslových procesů, efektivitu při sběru dat a optimalizaci automatizačních úloh. Příkladem protokolů typu fieldbus jsou:

- PROFIBUS (Process Field Bus) – využití v automatizaci procesů a výroby,
- Modbus – jeden z nejrozšířenějších řídicích průmyslových protokolů,
- Foundation Fieldbus – používán zejména v chemickém a ropném průmyslu,
- Interbus – využíván především v Evropě pro aplikace v průmyslové automatizaci [34].

Backend protokol je termín používaný pro popis komunikačního protokolu, který je primárně zaměřen na výměnu dat a instrukcí mezi backendovými systémy (serverové aplikace nebo databázové systémy), které spolu komunikují v distribuované síťové architektuře. Na rozdíl od Fieldbus protokolů, které umožňují interakci mezi řídícími a kontrolovanými zařízeními na úrovni fyzických zařízení, backend protokoly řeší komunikaci na vyšší úrovni abstrakce a často zahrnují složitější datové struktury a komunikační procesy. Backend protokoly jsou zásadní pro koordinaci a efektivitu operací v rámci většího organizačního celku. Typicky nezajišťují přímý přístup k zařízením nebo senzorům, ale umožňují, aby se jednotlivé řídicí systémy navzájem informovaly, sdílely data nebo si vyžádaly data od sebe navzájem. Díky tomu mohou být údaje analyzovány, agregovány a přeposílány mezi různými systémy a aplikacemi, což umožňuje pokročilé řízení, monitorování a optimalizaci průmyslových procesů. Příkladem takového protokolu je protokol Open Process Communications (OPC) nebo Inter-Control Center Protocol (ICCP) [34].

1.1 Porovnání se sítěmi informačních technologií

Porovnání průmyslových sítí se sítěmi informačních technologií je komplexní téma, které se týká různých aspektů – od účelu těchto sítí přes využívaná zařízení až po zabezpečení komunikace. V následujícím textu jsou detailně rozebrány různé oblasti průmyslových sítí a sítí informačních technologií. Konkretně se jedná o oblasti:

- účel – jaký je účel a využití průmyslových sítí a sítí informačních technologií,
- využívaná zařízení – jaká zařízení jsou v těchto sítích využívaná,
- rychlost komunikace – jak zvládají porovnávané sítě přenášet větší objemy dat,
- využívané protokoly – jaké protokoly jsou využívané v průmyslových sítích a sítích informačních technologií,
- bezpečnost komunikace – jak je zajištěna dostupnost, důvěrnost a integrita dat,
- škálovatelnost – jaké jsou možnosti rozšíření a integrace nových zařízení do těchto sítí.

Účel

Průmyslové sítě jsou navrženy s primárním zaměřením na řízení, monitorování a automatizaci průmyslových procesů v reálném čase. Tento přístup zajišťuje bezprostřední odezvu na měnící se podmínky průmyslového prostředí. Ta je zásadní pro udržení optimálního výkonu a dostupnosti dat. Na druhou stranu, sítě informačních technologií jsou orientovány na široké spektrum datové komunikace, od aplikací operujících v reálném čase (VoIP, streamování videa) po nekritickou komunikaci (e-maily, cloudové služby). Díky této flexibilitě mají široké využití v běžném životě a podnikání [10].

Využívaná zařízení

V průmyslových sítích se nachází specifická zařízení, jako jsou senzory, akční prvky, PLC a další, která jsou zásadní pro automatizaci a kontrolu průmyslových operací. Tyto systémy jsou často optimalizovány pro konkrétní účely a vyžadují vysokou úroveň spolehlivosti. Oproti tomu, sítě informačních technologií slouží univerzálnějším zařízením, jako jsou počítače, mobilní telefony a tablety, které podporují širokou škálu aplikací a služeb [10].

Rychlost komunikace

Průmyslové sítě jsou navrženy tak, aby splňovaly striktní požadavky na komunikaci v reálném čase, zajišťovaly dostupnost dat a tak zabezpečovaly kontinuální a bezpečný provoz průmyslových systémů. Sítě informačních technologií mohou naopak tolerovat vyšší latenci a jsou primárně zaměřeny na maximalizaci propustnosti a efektivity datového přenosu. Zatímco průmyslové sítě musí čelit výzvám spojeným s rušením a fyzickým omezením průmyslového prostředí, sítě informačních technologií jsou konstruovány s důrazem na škálovatelnost a přizpůsobivost [10].

Využívané protokoly

Průmyslové sítě se spoléhají na specializované protokoly navržené pro průmyslovou automatizaci, jako jsou např. Modbus, Profibus, EtherCAT. Na druhou stranu sítě informačních technologií využívají obecnější a univerzální internetové protokoly jako HTTP, FTP, SMTP a DNS. Tato rozdílná orientace odráží specifické požadavky každé domény na komunikaci a integraci systémů [10].

Zabezpečení komunikace

V průmyslových sítích byl tradičně kladen větší důraz na fyzické zabezpečení a izolaci od vnějších sítí. Nicméně s narůstajícím propojením průmyslových sítí a sítí

informačních technologií je stále více kladen důraz na sofistikovanější zabezpečení na úrovni protokolů a aplikací. Sítě informačních technologií, které jsou přístupné z internetu, vyžadují rozsáhlá bezpečnostní opatření, včetně šifrování, autentizace, firewallů a ochrany integrity dat, aby se chránily proti širokému spektru kybernetických hrozeb [10].

Škálovatelnost provozu

Škálovatelnost provozu je klíčovým faktorem, který odlišuje průmyslové sítě od sítí informačních technologií. Ovlivňuje schopnost systému rozšiřovat a přizpůsobovat se narůstajícím požadavkům bez ztráty výkonu a kvality služby. Průmyslové sítě, navržené specificky pro konkrétní výrobní procesy a aplikace, jsou charakteristické svým stabilním a předvídatelným chováním. Tyto sítě jsou optimalizovány pro kontinuální operace s předvídatelným množstvím zařízení a komunikačními požadavky, což omezuje jejich dynamickou škálovatelnost. Přidání nových zařízení nebo rozšíření funkcionality v průmyslových sítích může vyžadovat plánované vylepšení infrastruktury a softwaru. Oproti tomu jsou sítě informačních technologií inherentně navrženy tak, aby byly vysoce dynamické a škálovatelné za účelem adaptace na rychle se měnící požadavky digitálně propojeného světa. Díky pokrokům v cloudových technologiích, virtualizaci a síťových protokolech mohou sítě informačních technologií efektivně spravovat prudký nárůst datového provozu a počtu připojených zařízení. Tato flexibilita umožňuje sítím informačních technologií rychle se přizpůsobit novým aplikacím, službám a uživatelským požadavkům, od podnikových systémů až po individuální spotřebitelské technologie [10].

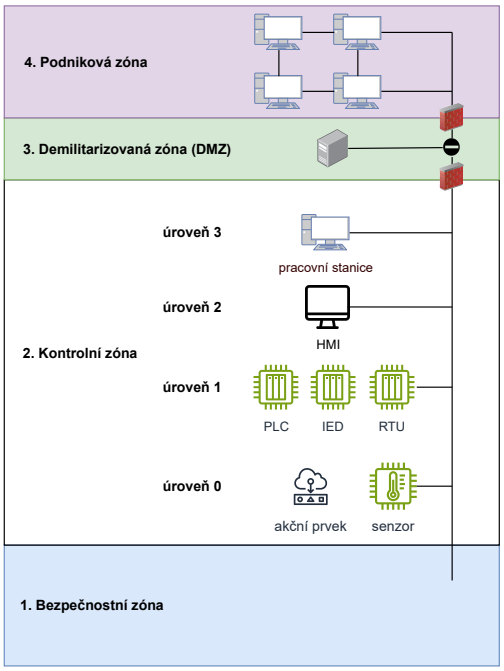
1.2 Architektura ICS – Purdue model

Průmyslové řídicí systémy (zkratka ICS, z anglického *Industrial Control Systems*) jsou množina různých počítačů a elektronických a mechanických zařízení používaných k řízení fyzických procesů. Jsou obvykle velmi složité a zahrnují heterogenní hardware (senzory, akční prvky, fyzické systémy, kontroléry) a softwarové komponenty (SCADA řídicí systémy). Protokoly využívané v těchto systémech slouží specifickému účelu a nemají definované žádné bezpečnostní prvky. Tento stav umožňuje snadnější realizaci bezpečnostních hrozeb [7].

Referenčním modelem architektury ICS je Purdue model, také známý jako Purdue Enterprise Reference Architecture. Tato architektura byla navržena pro zlepšení kybernetické bezpečnosti a řízení průmyslových automatizačních systémů. Purdue model definuje sedm úrovní hierarchie (viz obr. 1.2), které reprezentují různé

vrstvy a oblasti v průmyslovém prostředí – od podřízených zařízení a řídicích systémů na spodních úrovních až po firewally a správu informací na vyšších úrovních. Toto hierarchické uspořádání usnadňuje implementaci bezpečnostních opatření, protože umožňuje izolaci různých částí sítě a konkrétní ochranu na každé úrovni [66]. Níže v číslovaném seznamu jsou podrobně popsány jednotlivé úrovně Purdue modelu:

1. **Bezpečnostní zóna** – Zabezpečení systémů pro průmyslovou kontrolu, zahrnující specifické bezpečnostní mechanismy a systémy pro detekci a prevenci incidentů. Tato zóna obsahuje:
   - systémy pro monitorování síťových anomálií, jako jsou systémy pro detekci průniku (IDS),
   - zařízení pro aktivní ochranu, například systémy pro prevenci průniku (IPS),
   - další bezpečnostní nástroje a služby navržené k ochraně průmyslových kontrolních systémů a infrastruktury.
2. **Kontrolní zóna** – Obsahuje systémy operačních technologií pro řízení a monitorování fyzických procesů a zařízení. Tato zóna je dále rozdělena do následujících podúrovní:
   - **Úroveň 0** – Zahrnuje primární senzory a akční prvky (např. ventily, spínače), které přímo interagují s fyzickými procesy.
   - **Úroveň 1** – Zahrnuje kontrolní zařízení jako jsou programovatelné logické kontroléry (PLC) a vzdálené terminálové jednotky (RTU), které přijímají data ze senzorů a vydávají příkazy akčním prvkům.
   - **Úroveň 2** – Zahrnuje systémy pro ovládání a vizualizaci procesů, včetně rozhraní určeného pro komunikaci s řídící jednotkou (HMI) a kontrolních stanic, poskytující uživatelské rozhraní pro operátory.
   - **Úroveň 3** – Zahrnuje systémy pro automatizaci výrobních procesů, řídicí a optimalizační software, který podporuje plánování výroby, sledování výkonnosti a zajištění kvality.
3. **Demilitarizovaná zóna (DMZ)** – DMZ slouží jako bezpečnostní mezistupeň mezi Podnikovou zónou a operačními technologiemi. Tato zóna obsahuje systémy a služby, které zprostředkovávají výměnu dat mezi sítěmi operačních technologií (OT) a sítěmi informačních technologií (IT). Cílem je minimalizovat riziko kybernetických útoků ze sítě informačních technologií směrem ke kritickým systémům operačních technologií.
4. **Podniková zóna** – Tato zóna zahrnuje podnikové informační technologie (IT), včetně tradičních kancelářských systémů a zařízení, jako jsou osobní počítače, mobilní telefony a servery podnikových aplikací. Je to prostředí, kde se provádí plánování, finanční analýzy, správa a další nefyzické procesy podniku [66].



Obr. 1.2: Znázornění referenčního modelu podnikové architektury – Purdue model

Purdue model je užitečný nástroj pro průmyslové organizace, které se snaží zlepšit bezpečnost svých systémů. Pomáhá pochopit, jaké kroky jsou nutné k ochraně jejich zařízení a systémů před kybernetickými hrozbami, jak zabezpečit kritickou infrastrukturu a jak minimalizovat rizika spojená s provozem v průmyslovém prostředí [66].

Model Purdue pro průmyslovou kontrolu a automatizaci je uznávaným standardem pro hierarchickou organizaci průmyslových sítí a systémů. Tento model strukturuje sítě do různých úrovní podle jejich funkce a bezpečnostních požadavků, což umožňuje efektivnější správu a zabezpečení průmyslových a podnikových procesů [66].

1.3 Architektura ICS – Průmysl 4.0

Termínem Průmysl 4.0 jsou zastřešovány progresivní technické metody v průmyslové oblasti. Hlavní vizí Průmyslu 4.0 je chytrá továrna, která prostřednictvím různých senzorů na strojích monitoruje prostředí a provádí rozhodnutí směřující k optimalizaci výroby a produkce. Další vizí je propojení celého dodavatelského řetězce – od výrobce přes poskytovatele komunikace až k zákazníkům. Výroba by měla být schopna být střediskem zisku, které dokáže pružně reagovat na změněnou situaci

buď na trhu (změna preferencí odběratelů) nebo na základě změn aktuálních odběratelských potřeb [25].

Právě tyto změny aktuálních odběratelských potřeb jsou velkých tématem hlavně v energetice. Protože vzhledem k faktu, že se elektrická energie obtížně skladuje, je potřeba aktuální výrobu energie srovnávat s aktuální poptávkou po energiích. To je z nákladového hlediska zajímavý koncept pro každého výrobce. Příkladem takového modelu je koncept regulátora České energetické přenosové soustavy ČEPS a.s., který pomocí služeb výkonové rovnováhy (SVR) výše popsanou situaci řeší na úrovni energetiky [8].

Koncept Průmyslu 4.0 obsahuje čtyři základní koncepty:

- interoperabilita – schopnost chytrých strojů a lidí společně komunikovat přes HMI (rozhraní pro komunikaci člověka a stroje),
- decentralizace – celý systém neřídí centrální prvek, ale tento systém je rozdělen do samosprávných celků, které rozhodují za účelem maximalizace výroby,
- řízení v reálném čase s minimálními odchylkami,
- modularita – schopnost přizpůsobit se novým požadavkům [25].

RAMI 4.0 model

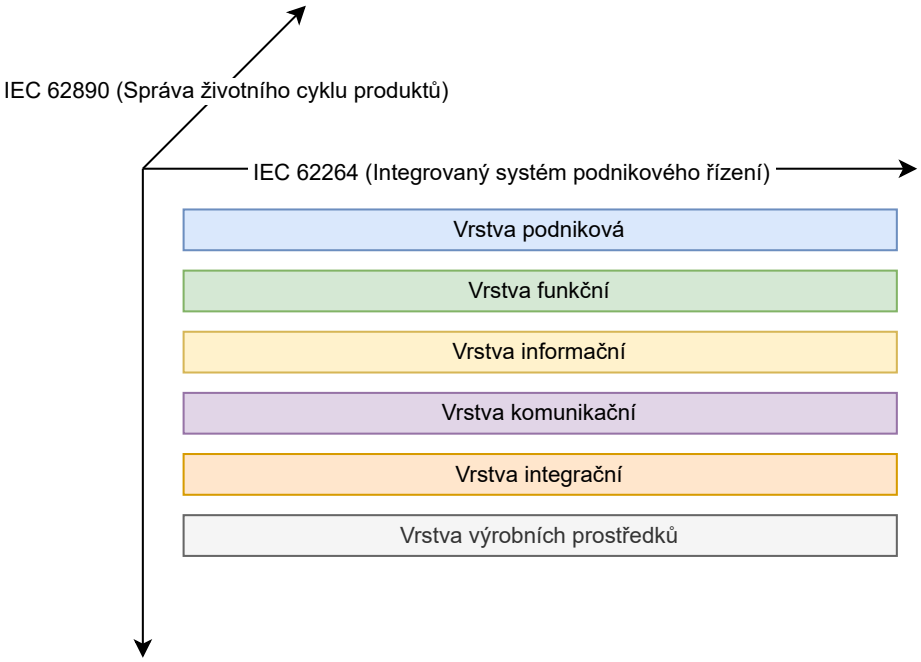
RAMI 4.0 (**R**eference **A**rchitecture **M**odel for **I**ndustry 4.0) je konceptuální model navržený pro podporu digitální transformace průmyslu a implementaci principů Průmyslu 4.0. Jedná se o evoluci předchozího RAMI 3.0 modelu a slouží k lepší organizaci a integraci průmyslových systémů v době digitální revoluce [18].

Model RAMI 4.0 je trojrozměrný model (viz obr. 1.3), který zahrnuje veškeré aspekty Průmyslu 4.0. První horizontální osou jsou hierarchické vrstvy podle normy *IEC 62264 (Integrovaný systém podnikového řízení)*. Tato hierarchická struktura symbolizuje různé role a úkoly v průmyslových závodech a výrobních zařízeních. Pro reprezentaci konceptu Průmyslu 4.0 byly tyto role rozděleny do několika úrovní. Počínaje výrobkem a končícím propojením do Internetu věcí a poskytováním služeb [27].

Druhá horizontální osa reprezentuje životní cyklus zařízení a produktů podle normy *IEC 62890 (Správa životního cyklu produktů a systémů používaných v procesním průmyslu)*. Tato osa rozlišuje dvě kategorie produktů – typ a instance. Typ zahrnuje období, kdy je výrobek navržen, testován a schvalován. Instance je poté označení výrobku, kdy je výrobek vyroben, testován a vyráběn sériově [29].

Na vertikální ose RAMI modelu se nachází 6 vrstev, které slouží k popisu charakteristik zařízení. Každá vrstva je analyzována samostatně v rámci virtuálního mapování. Těmito vrstvami jsou:

- **vrstva podniková** – zaměřuje se na strategické rozhodování a řízení podniku,



Obr. 1.3: Zjednodušené znázornění referenčního modelu RAMI 4.0

- **vrstva funkční** – definuje funkce a operace, které jsou potřebné pro provoz zařízení a výrobních procesů,
- **vrstva informační** – slouží k centralizovanému ukládání a zpracování dat, která jsou shromážděna z různých zdrojů včetně senzorů a výrobních prostředků,
- **vrstva komunikační** – zajišťuje komunikaci mezi různými zařízeními a systémy v průmyslovém prostředí,
- **vrstva integrační** – zodpovídá za propojení a integraci různých průmyslových systémů a zařízení,
- **vrstva výrobních prostředků** – zahrnuje samotné výrobní prostředky, stroje, senzory a další fyzická zařízení, která provádějí výrobní operace [18].

Porozumění jednotlivým vrstvám na všech osách je zásadní pro pochopení, plánování a provádění transformace průmyslových systémů směrem k plné integraci technologií Průmyslu 4.0. RAMI 4.0 definuje kroky potřebné k realizaci plného využití Průmyslu 4.0:

- identifikace – je potřeba identifikovat všechna zařízení pomocí jediného standardu,
- sémantika — pro komunikaci mezi zařízeními různých výrobců je nutná jednotná sémantika, včetně společné syntaxe pro data,
- kvalita služeb – nutnost definice kritických služeb, nekritických služeb a jejich

kvalitu, v sítích informačních technologií též známé jako kvalita služeb (QoS), která určuje prioritu každé služby [53].

Výše popsaný model, díky svému vrstvovému rozdělení průmyslových systémů, pomáhá organizacím lépe porozumět a plánovat výrobní procesy a komunikační technologie v kontextu Průmyslu 4.0. Tento model usnadňuje integraci a řízení výrobního prostředí, které přináší zefektivnění výroby vedoucí ke konkurenční výhodě na trhu [53].

2 Bezpečnost průmyslových sítí

V předchozí kapitole 1.1 bylo uvedeno, že jedním z problémů průmyslových protokolů je bezpečnost. Tato skutečnost je způsobená tím, že průmyslové sítě mají vysoce specifické protokoly a je kladen výrazný důraz na co nejnižší latenci při zpracování dat v průmyslových sítích. Proto v průmyslu nedošlo k takovému výraznému rozvoji bezpečnosti jako v sítích informačních technologií, kde vlivem stále většího využití internetu v běžném životě začalo být důležité chránit osobní údaje, finanční prostředky a přístupy k účtům do libovolných aplikací. Z tohoto důvodu mají sítě informačních technologií z hlediska bezpečnosti náskok.

Průmyslové sítě jsou často součástí kritické infrastruktury státu a každý útok na ně může výrazně negativně ovlivnit kvalitu života jeho obyvatel. Ohroženy jsou např. nemocnice, elektrárny, vodní zdroje apod.

V důsledku OT/IT konvergence se objevují nová rizika pro průmyslové sítě, jelikož už nejsou izolovány od okolního světa, ale jsou propojeny skrze internet. Tato konvergence zdůraznila důležitost dvou různých bezpečnostních triád – triády CIA, která je základem IT bezpečnosti, a triády AIC, která je relevantní pro OT bezpečnost. Jednotlivá písmena jsou seřazena sestupně podle stupně důležitosti a symbolizují:

- a – availability = dostupnost komunikace,
- I – integrity = integrita dat,
- C – confidentiality = důvěrnost dat.

Dostupnost komunikace se týká zajištění přístupnosti a použitelnosti komunikačních systémů a služeb v okamžiku, kdy jsou potřeba. To znamená, že komunikace a služby jsou dostupné pro uživatele, kteří je potřebují, bez zásadních výpadků nebo omezení. Integrita dat zajišťuje zachování a zajištění nedotčenosti dat během celého procesu a manipulace s nimi. To znamená, že data zůstávají nepoškozena a nezměněna během přenosu, ukládání nebo zpracování. Důvěrnost dat se týká ochrany informací před neoprávněným přístupem nebo zobrazením. Zajišťuje, že data jsou přístupná pouze oprávněným osobám nebo systémům a zůstávají utajena před těmi, kteří nemají potřebné oprávnění k jejich zobrazení nebo použití [62].

Rozdíl mezi CIA (IT) a AIC (OT) triádou spočívá v důrazu na různé aspekty zabezpečení informací. Pro Informační technologie je prioritní důvěrnost dat (osobní údaje, bankovní data, obchodní tajemství, aj.). Pro Operační technologie je naopak nejdůležitější dostupnost služby (elektrárny, zdravotnické stroje, apod.), jež jednotlivá zařízení poskytují.

2.1 Historie útoků na průmyslové sítě

Pro lepší pochopení důležitosti zabezpečení průmyslových sítí je v následujícím textu stručný popis neznámějších útoků na ICS/SCADA systémy. Prvním útokem byl v roce 1982 trojský kůň směřovaný na Trans-sibiřské plynové potrubí, v kterém způsobil obrovskou explozi [7].

V roce 2010 asi nejznámější počítačový červ *Stuxnet* vyřadil celý íránsky jaderný program. Tento červ byl vytvořen, aby útočil na systémy SCADA, přeprogramoval jejich funkcionalitu a své změny skryl [21].

V roce 2015 trojský kůň jménem *BlackEnergy3* vyřadil 30 rozvoden na Ukrajině, což způsobilo masivní několika hodinový výpadek tamní přenosové soustavy [6].

Jedním z posledních případů útoku je malware *TRITON*, který přeprogramovává PLC zařízení, aby přešly do nefunkčního stavu (idle state). Tímto může následně vzniknout situace ohrožující lidské životy [12].

Předpoklad, že počet těchto útoků v budoucnu bude klesat, je nerealistický. Firma Kasperky, která chrání nejenom průmyslová zařízení, oznámila, že v druhé polovině roku 2016 bylo skoro 40 % jimi chráněných zařízení napadeno nějakou formou malwarového útoku. Výše popsané je jasnou signaturou, že počet těchto útoků není již v současnosti zanedbatelný, a že je potřeba jej řešit prostředky aktivní ochrany [7].

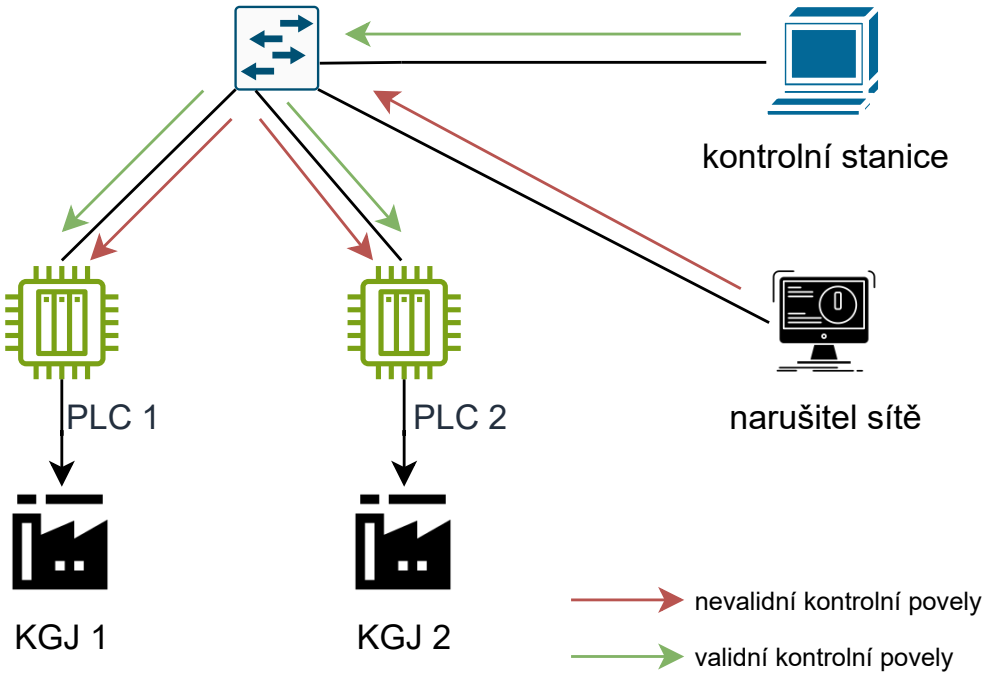
2.2 Vektory útoku

V sítích informačních technologií je primárním cílem zajištění důvěrnosti, integrity a dostupnosti, neboli CIA triády. V sítích operačních technologií je situace podobná, ovšem v opačném pořadí – cílem je zajistit AIC triádu. Prioritou je dostupnost, která má v průmyslu ještě vyšší význam než v běžných sítích informačních technologií, následovaná integritou a důvěrností dat. V tomto kontextu je **dostupnost nejdůležitějším aspektem ICS/SCADA systémů**, protože na těchto systémech mnohdy závisí lidské životy. Dostupnost teploty jádra v jaderné elektrárně je víc důležitá, než důvěrnost této informace [7].

Útoky, které mohou ohrozit průmyslové sítě jsou:

- falešné kontrolní povely (útok na integritu),
- změna dat v přenášených datových jednotkách (útok na integritu),
- DDoS útoky (útok na dostupnost),
- falešná kontrolní stanice (útok na integritu),
- zachycení přenosu dat (útok na důvěrnost) [68].

Na obrázku 2.1 je možné vidět jeden ze způsobů provedení útoku na integritu sítě integrováváním kontrolní stanice, která odesílá falešné kontrolní povely. Stejně tak



Obr. 2.1: Znázornění vektoru útoku – falešná kontrolní stanice

by mohl tento útočník podobným způsobem zachytávat data nebo měnit kontrolní povely v přenášených datových jednotkách pomocí útoku muže uprostřed realizovaného kompromitováním ARP záznamů v síti. Dalším možným útokem by mohlo být zahlcení kontrolního PLC, které ovládá KGJ (kogenerační jednotku), např. pomocí TCP SYN Flood útoku při ustanovení Modbus TCP komunikace.

2.3 Bezpečnost průmyslových protokolů

Bezpečnost průmyslových protokolů lze rozdělit na 2 části. První z nich je ochrana proti vzniku náhodných chyb a řešení chyb komunikace. V této oblasti průmyslové protokoly excelují. Tolerance chybových stavů a dostupnost jsou u nich na vysoké až excelentní úrovni. Oproti tomu stojí ochrana proti úmyslným útokům. V této oblasti průmyslové protokoly nemají tak silné nebo dokonce žádné prostředky ochrany (šifrování, autentizace dat, aj.). Tento stav se začal zlepšovat po roce 2006, kdy začaly vznikat zabezpečené verze průmyslových protokolů vetšinou na úrovni transportní vrstvy pomocí protokolu Transport Layer Security (TLS). Pro ukázku, viz níže seznam protokolů, včetně jejich zabezpečené varianty:

- Modbus (1979) – Modbus/TCP Security (2008),
- DNP3 (1990) – DNP3-SA (2020),
- Profinet (2003) – PROFINET Security Classes (2019),

- OPC (1996) – OPC UA (2006),
- IEC 104 (2000) – IEC 62361 (2007),
- IEC 61850 (2003) – IEC 62361 (2007) [7].

Modbus zavedený v roce 1979, reprezentuje fundamentální protokol v průmyslové automatizaci, zpřístupňující robustní a univerzální rámec pro komunikaci mezi řídicími systémy a průmyslovými zařízeními. Původní verze Modbus bohužel nebyla vybavena sofistikovanými bezpečnostními opatřeními, což vedlo k potenciálním bezpečnostním rizikům. Rozšíření Modbus/TCP Security, představené v roce 2008, adresovalo tyto bezpečnostní deficity prostřednictvím integrace šifrování, kontrol mechanismů integrity dat a zajištění dostupnosti dat [67].

Distributed Network Protocol 3 (DNP3), uvedený na trh v roce 1990, byl navržen specificky pro energetický sektor a přináší přesnost a spolehlivost potřebné pro správu energetických distribučních sítí. Bezpečnostní rozšíření DNP3-SA, zveřejněné v roce 2020, rozšiřuje původní protokol o významné bezpečnostní prvky zaměřené na ochranu kritické infrastruktury, posilující obranu proti sofistikovaným kybernetickým útokům [48].

Protokol Profinet, uvedený v roce 2003, představuje evoluční krok v průmyslové komunikaci, umožňující rychlé a spolehlivé výměny dat mezi zařízeními. V reakci na rostoucí bezpečnostní hrozby byly v roce 2019 představeny PROFINET Security Classes, které poskytují vyspělé nástroje pro ochranu komunikačních kanálů včetně šifrování a mechanismů pro kontrolu integrity [47].

OPC Unified Architecture (OPC UA), vyvinutý v roce 2006 jako nástupce OPC, je současným standardem pro interoperabilitu mezi průmyslovými systémy. Tento protokol nabízí rozšířené bezpečnostní schopnosti a podporu pro široké spektrum průmyslových aplikací [23].

IEC 60870-5-104 (IEC 104) je komunikační protokol, který se používá především pro vzdálenou kontrolu a monitorování zařízení v energetických systémech. Protokol je částí širší série IEC 60870-5, která se zaměřuje na protokoly pro monitorování, řízení a automatizaci v průmyslovém sektoru [26].

Standard IEC 61850 slouží pro komunikaci a integraci v automatizačních systémech. Byl vyvinut k zajištění interoperability a komunikace mezi různými zařízeními v elektrických rozvodnách. Tento standard je užíván v oblasti distribuce a přenosu elektrické energie a umožňuje efektivní výměnu dat mezi zařízeními, jako jsou ochranná relé, měřicí transformátory, vypínače a další průmyslové komponenty. IEC 61850 definuje modely dat, služby a komunikační protokoly, které usnadňují integraci zařízení a systémů od různých výrobců [24].

Následný standard IEC 62361 z roku 2007 definoval metody pro posílení kybernetické bezpečnosti předchozích standardů IEC 60870-5 a IEC 61850 za účelem zajištění celkové ochrany energetických a automatizačních systémůf [28].

Průmyslové komunikační protokoly představují zásadní prvek v architektuře moderních automatizovaných systémů, které poskytují mechanismy pro řízení, monitorování a sběr dat z průmyslových zařízení. Tyto protokoly mají značný dopad na operativní efektivitu a bezpečnost kritické infrastruktury. Každý z těchto protokolů hraje důležitou roli v oblasti své sféry, proto vznikly jejich bezpečnostní rozšíření, které poskytují nezbytné ochranné prvky v reakci na neustále se vyvíjející kybernetické hrozby [7].

2.4 Obrana proti kybernetickým útokům

Tato kapitola je věnována obraně proti kybernetickým útokům na různých úrovních síťové infrastruktury. V jednotlivých podkapitolách jsou detailně rozebrána:

- zabezpečení na úrovni transportní vrstvy,
- ochrana na úrovni síťové vrstvy skrze virtuální soukromé sítě a monitorování provozu,
- implementace šifrování dat spojové vrstvy,
- strategie omezení fyzického přístupu k síti.

2.4.1 Ochrana na úrovní transportní vrstvy – TLS

Protokol TLS (Transport Layer Security) je široce využívaný protokol jak v sítích operačních technologií, tak v sítích informačních technologií. Hlavním účelem TLS je zajištěni důvěrnosti, integrity a dostupnosti dat mezi koncovými uzly [36].

Tento protokol je bezpečnostní nadstavbou transportního protokolu TCP, který přenáší data aplikačních protokolů jako HTTP, SMTP, FTP, atd. TLS protokol funguje na modelu klient-server a obsahuje 2 vrstvy (TLS Record protokol, TLS Handshake protokol) [68].

TLS Record protokol

TLS Record Protocol funguje na základě dvojice abstraktních entit nazývaných zprávy a záznamy. Zprávy jsou datové jednotky posílané mezi stranami, zatímco záznamy jsou konkrétní bloky dat vytvořené z těchto zpráv a obsahují hlavičku, která zahrnuje informace o typu obsahu, délce zprávy a další potřebné informace pro dekódování. Záznamy jsou šifrovány a chráněny autentizačními mechanismy, což zajišťuje integritu a důvěrnost přenášených dat. Tyto záznamy jsou pak předány do podřízeného protokolu, který zajišťuje kompresi, šifrování a autentizaci komunikace [9].

TLS Handshake protokol

TLS Handshake protokol je klíčovým prvkem v navazování zabezpečené komunikace mezi klientem a serverem v rámci Transport Layer Security (TLS). Tento proces umožňuje oběma stranám se dohodnout na společných šifrovacích parametrech a výměně klíčů pro následnou šifrovanou komunikaci. Probíhá ve třech fázích:

- v první fázi dochází k navázání spojení a vyjednání parametrů,
- v druhé fázi probíhá autentizace stran a výměna šifrovacích klíčů,
- v poslední fázi se ověřuje integrita spojení.

Tímto procesem se zajišťuje bezpečnost komunikace a vytváří se společné prostředí pro šifrovaný přenos dat mezi klientem a serverem [58].

2.4.2 Ochrana na úrovní síťové vrstvy – Virtuální soukromé sítě

Ochrana průmyslových sítí může být zajištěna pomocí prostředků ochrany na úrovni síťové vrstvy. Jednou z takových možností je Virtual Private Network, zkráceně VPN, česky Virtuální soukromá síť. Tato síť je komunikační prostředí, do kterého je přístup určen pouze konkrétním stranám, které se vůči sobě autentizují a vytváří společné komunikační prostředí, tzv. VPN tunely, které komunikující stranám zajišťují dostupnost, integritu a důvěrnost dat [16]. V následujících odstavcích je popis několika nejznámější VPN protokolů (OpenVPN, IPSec, SSTP, PPTP, WireGuard).

OpenVPN

OpenVPN je open-source software umožňující vytvoření šifrovaného a bezpečného spojení prostřednictvím virtuální privátní sítě (VPN), což poskytuje uživatelům zabezpečený přístup k internetu nebo ke vzdáleným sítím. Jeho hlavními výhodami jsou vysoká úroveň šifrování dat, podpora různých platforem a operačních systémů, schopnost práce za firewallem a NATem, flexibilita v konfiguraci a otevřený zdrojový kód. To umožňuje komunitě přispívat k jeho vylepšování a zvyšování bezpečnosti [49].

IPsec

V kontextu zabezpečení průmyslových sítí představuje IPsec (Internet Protocol security) základní stavební kámen pro ochranu datové komunikace na síťové vrstvě. Jeho primárním cílem je zajistit autentizaci, integritu a důvěrnost dat přenášených přes IP sítě. IPsec je široce využíván v různých aplikačních scénářích, včetně virtuálních privátních sítí (VPN), kde umožňuje bezpečné tunelování dat mezi oddělenými sítěmi [19].

IPsec se skládá ze tří hlavních protokolů, AH (Authentication Header), ESP (Encapsulating Security Payload) a SA (Security Association). Protokol AH poskytuje mechanismus pro ověření autentičnosti dat. Protokol ESP šifruje data k zajištění jejich důvěrnosti a také podporuje autentizaci. Poslední protokol SA definuje, jaké budou protistrany používat prostředky pro sestavení a udržení zabezpečené komunikace. Klíčovým aspektem IPsec je jeho schopnost konfigurovat politiky bezpečnosti a provádět vzájemnou autentizaci mezi koncovými body komunikace. To je realizováno pomocí IKE (Internet Key Exchange) protokolu, který umožňuje bezpečnou výměnu klíčů a dynamické spravování bezpečnostních asociací [15].

V průmyslových sítích, kde je kladen velký důraz na dostupnost a integritu dat, poskytuje IPsec cenné řešení pro zabezpečení mezisíťové komunikace. Jeho implementace umožňuje zajistit, že data přenášená mezi zařízeními, jako jsou senzory, programovatelné logické automaty a řídicí servery, jsou chráněna proti odposlechu a manipulaci [15].

IPsec představuje základní prvek pro zabezpečení průmyslových sítí, který umožňuje efektivní ochranu datové komunikace v prostředích vyžadujících vysokou úroveň bezpečnosti. Jeho schopnost zajistit důvěrnost, integritu a autentizaci dat činí z IPsec nepostradatelný nástroj pro zabezpečení dnešních sítí [19].

SSTP

SSTP poskytuje mechanismus pro přenos paketů síťového protokolu Point-to-Point (PPP) přes HTTP kanál, který je šifrován pomocí protokolu Secure Sockets Layer (SSL). Díky tomu je možné SSTP využít pro bezpečnou komunikaci mezi klientem a VPN serverem přes internet, dokonce i pokud jsou mezi nimi síťové bariéry, jako jsou firewally, které omezují použití jiných VPN protokolů [37].

Jednou z hlavních výhod SSTP je jeho schopnost snadno překonávat NAT (Network Address Translation) a firewall omezení. Tato skutečnost činí z tohoto protokolu vynikající volbu pro uživatele za přísně omezenými síťovými prostředími. SSTP využívá standardní port HTTPS (TCP port 443), díky čemuž jeho provoz vypadá na první pohled jako běžný webový provoz, což mu umožňuje snadno proklouznout skrze většinu síťových kontrol. Další klíčovou vlastností je vysoký stupeň šifrování, který SSTP nabízí díky použití SSL/TLS, zaručující silnou ochranu přenášených dat [37].

PPTP

PPTP (Point-to-Point Tunneling Protocol) je jeden z nejstarších a nejrozšířenějších protokolů pro vytváření VPN. Navržen pro snadnou implementaci a širokou

kompatibilitu s různými operačními systémy, PPTP poskytuje základní úroveň šifrování pro přenos dat mezi uživatelským zařízením a VPN serverem. Jeho hlavními výhodami jsou jednoduchá konfigurace, nízká náročnost na výpočetní výkon a podpora v mnoha operačních systémech. Nicméně, nedostatek silného šifrování a známé bezpečnostní slabiny učinily PPTP méně vhodným pro citlivé přenosy dat v dnešní době [22].

WireGuard

WireGuard je moderní VPN protokol známý pro svou jednoduchost a vysokou úroveň bezpečnosti, který je široce používán v internetu věcí a peer-to-peer komunikaci. Protokol kombinuje moderní algoritmy s efektivními technikami výměny klíčů pro zajištění bezpečného, rychlého a spolehlivého přenosu dat [65].

Šifrovací algoritmy – Využíván algoritmus ChaCha20 – symetrický proudový šifrovací algoritmus, navržený Danielem J. Bernsteinem. Je známý pro svou vysokou míru bezpečnosti a výkonnost na platformách bez hardwarové akcelerace blokových šifer. ChaCha20 pracuje s klíčem o délce 256 bitů a využívá jedinečnou metodu permutací pro generování šifrovacího proudového klíče, což je zvláště efektivní v prostředích s omezenými zdroji, jako jsou průmyslová zařízení [13].

Algoritmy na výměnu klíčů – Využíván algoritmus Curve25519, navržený také Danielem J. Bernsteinem, je algoritmus pro výměnu klíčů založený na stejnojmenné eliptické křivce, který se vyznačuje vysokou bezpečností a efektivitou. Curve25519 je optimalizovaný pro rychlou výměnu klíčů a minimalizaci potenciálních slabých míst, jako je odolnost vůči útokům opakováním. Tento algoritmus je široce přijímán pro svou schopnost rychle a bezpečně navázat šifrovaný komunikační kanál bez nutnosti předchozího sdílení klíčů [13].

Autentizace a integrita – Využíván algoritmus Poly1305 pro MAC (Message Authentication Code) autentizaci, který se používá ve spojení s ChaCha20 pro zajištění integrity a autentizace dat. Tato kombinace je vysoce odolná proti různým typům útoků a poskytuje záruky integrity dat, které jsou zásadní pro bezpečné síťové operace. Dalším nástrojem zajištění integrity dat je BLAKE2s, moderní hašovací funkce, je nástupcem BLAKE2 a je navržen tak, aby byl rychlejší v softwarové implementaci než MD5 a SHA-1, přičemž nabízí vyšší bezpečnostní úroveň [13].

WireGuard je navržen s důrazem na bezpečnost, rychlost a jednoduchost. To z něj dělá ideální volbu pro moderní VPN řešení. Jeho použití moderních šifrovacích technologií v kombinaci s moderním návrhem architektury zajišťuje, že VPN spojení jsou nejen bezpečná, ale také výkonná a efektivní [65].

2.4.3 Ochrana na úrovní síťové vrstvy – Monitorování provozu

Jedním z nástrojů monitorovaní provozu je **IDS (Intrusion Detection System)**, bezpečnostní mechanismus navržený k monitorování sítí a systémů za účelem identifikace podezřelých, potenciálně škodlivých nebo neobvyklých aktivit. Tento systém funguje pasivně a slouží k detekci a signalizaci možných hrozeb, aniž by aktivně zasahoval do síťového provozu [5].

Princip fungování IDS spočívá v analýze síťového provozu nebo sledování událostí v systému. Pomocí předem definovaných pravidel a vzorů IDS identifikuje odchylky od normálního chování a upozorňuje administrátory nebo vytváří záznamy o identifikovaných hrozbách [5].

Existují dva hlavní typy IDS:

1. Systém detekce proniknutí založený na síti (NIDS): Monitoruje síťový provoz a identifikuje potenciálně škodlivé aktivity na základě vzorů dat přenášených po síti.
2. Systém detekce proniknutí založený na hostiteli (HIDS): Instaluje se na konkrétním zařízení a monitoruje aktivitu tohoto zařízení, sleduje události na operačním systému, aplikacích atd. [5].

Monitorování provozu není jenom o pasivním sledování provozu a oznamování bezpečnostních událostí. Je možné taky aktivně řešit tyto bezpečnostní problémy pomocí **IPS (Intrusion Prevention System)**. Tento systém je navržený k aktivní ochraně sítí a systémů před neoprávněnými přístupy, útoky a hrozbami. IPS je pokročilejší verzí systému IDS, který nejen detekuje, ale také aktivně zasahuje a reaguje na identifikované hrozby v reálném čase [20].

Princip fungování IPS spočívá v kontrole a monitorování síťového provozu nebo událostí v systému. Podobně jako IDS, IPS analyzuje data pomocí pravidel a vzorů a identifikuje neobvyklé nebo škodlivé aktivity. Nicméně, na rozdíl od IDS, IPS má schopnost přijímat automatické akce, které aktivně brání sítě nebo systém před identifikovanými hrozbami [20].

IPS může provádět různé akce pro ochranu sítě, jako je blokování konkrétních IP adres, uzavření nebo otevření portů, filtrování provozu a další. Tyto automatické reakce jsou zaměřeny na okamžitou eliminaci nebo omezení škodlivého provozu, což pomáhá chránit síť nebo systém před různými typy útoků [20].

Rozdíl mezi IDS a IPS spočívá v jejich schopnostech reakce. IDS pouze identifikuje a hlásí hrozby, zatímco IPS nejen identifikuje, ale také aktivně zasahuje a brání síť nebo systém proti těmto hrozbám. IPS je proto považován za pokročilejší a dynamičtější ochranu proti útokům na sítě a systémy [5].

Typy reportování událostí

Součástí správného monitorování sítě je také okamžité reportování, které může být realizováno různými kanály, jako jsou e-maily, URL webhooky a logovací soubory. Každá z těchto metod představuje jiný způsob komunikace a má své specifické vlastnosti a použití.

Emailové reportování je tradiční a stále populární metoda upozornění, která umožňuje okamžité poslání detailního popisu incidentu konkrétním adresátům. Tato metoda je univerzálně přístupná a snadno konfigurovatelná. Email může obsahovat různé formáty zpráv, od prostého textu přes formátovaný HTML obsah až po přílohy, které mohou obsahovat další podrobnosti nebo logy. Emailové reporty jsou často automaticky generovány bezpečnostními systémy ihned po detekci problému.

Webhooky jsou HTTP callbacky, které fungují na principu push oznámení. Když dojde k určité události, systém aktivní ochrany sítě odesílá HTTP POST požadavek na konfigurované URL. Tento požadavek může být přijat webovým serverem nebo jakoukoliv službou, která webhooky podporuje, a může spustit řadu akcí, jako je spuštění skriptů, aktualizace databáze nebo upozornění týmu. Webhooky umožňují integraci bezpečnostních reportů s různými externími nástroji a službami v reálném čase [64].

Logování je proces zaznamenávání detailních informací o aktivitách sítě do souborů zvaných logy. Tyto logy jsou uloženy pro pozdější analýzu a jsou klíčové pro forenzní šetření po bezpečnostních incidentech. Logy mohou obsahovat informace o síťovém provozu, přístupových pokusech, detekovaných hrozbách a akcích, které systém přijal jako reakci na detekované incidenty. Logovací soubory jsou důležitým zdrojem informací pro sledování dlouhodobých trendů a pro optimalizaci bezpečnostních opatření [69].

Aktivní ochrana sítě tedy využívá kombinaci těchto reportovacích metod k zajištění komplexního přístupu k informování a reakci na bezpečnostní události.

2.4.4 Ochrana na úrovní spojové vrstvy – MACsec

Protokol MACsec (Media Access Control security) představuje standardizovanou technologii pro zabezpečení provozu na spojové vrstvě ISO/OSI modelu v ethernetových sítích. Jeho hlavním účelem je poskytnout šifrování a ověřování integrity dat přenášených mezi síťovými zařízeními na úrovni spojové vrstvy [4].

MACsec využívá symetrického šifrování na úrovni ethernetové komunikace. To znamená, že data jsou šifrována přímo na fyzickém rozhraní zařízení. Tímto způsobem se zabezpečuje celý ethernetový rámec dat včetně hlaviček, čímž je zajištěna důvěrnost, integrita a autentičnost datové komunikace [4].

V průmyslových sítích má implementace MACsec protokolu značný vliv na zvýšení bezpečnosti a spolehlivosti datové komunikace. V prostředích jako průmyslová automatizace, energetika nebo zdravotnictví, kde je kladen důraz na ochranu dat a minimalizaci rizik spojených s neoprávněným přístupem nebo útoky na síť, hraje MACsec klíčovou roli. Tato technologie umožňuje minimalizovat možnosti útoků typu Muže uprostřed (Man-in-the-middle), odhalení nebo úpravu dat a zajišťuje, že pouze oprávněné zařízení mohou přistupovat k síti a komunikovat s ostatními zařízeními [4].

2.4.5 Ochrana na úrovní fyzické vrstvy – Omezení přístupu k síti

Jednou z možností mírnění hrozeb útoku je kontrola přístupu ke komunikaci/síti. Průmyslové sítě mají výhodu oproti klasickým sítím informačních technologií, že ve většině případů mohou být zcela odděleny od vnějšího světa, tzn. internetu, avšak s rostoucí oblibou Internetu věcí v průmyslu, začíná být tato metoda zabezpečení stále více vzdálená od realizovatelnosti. Z tohoto důvodu je tato metoda pouze zmíněna a nikoliv rozebrána do detailu.

3 Komunikační protokoly

3.1 Modbus

Protokol Modbus, navržený v roce 1979, je doposud nejpoužívanějším protokolem v architektuře průmyslových sítí. Modbus je aplikační protokol, což znamená, že operuje na 7.vrstvě ISO/OSI modelu. Funguje na principu request (žádost) – response (odpověď) a je jednoduše adaptovatelný na různé síťové architektury (sběrnice, paketové sítě, apod.). To umožnilo vzniknout jeho dvou nejznámější verzím – Modbus TCP (viz kapitola 3.1.1 Modbus TCP) a Modbus RTU (viz kapitola 3.1.2 Modbus RTU) [34].

Datové typy Modbus protokolu se dělí do 4 kategorií. V tab. 3.1 je uvedeno volné přeložení jednotlivých typů přenášených dat do češtiny, včetně jejich originálního anglického pojmenování a významu.

Označení	**Význam**
Čtený bit (Discrete Input)	Jeden bit určený pouze ke čtení.
Cívka (Coil)	Jeden bit, který lze číst i zapisovat.
Čtený registr (Input Register)	16bitový registr určený pouze ke čtení.
Zápisový registr (Holding Register)	16bitový registr, který lze číst i zapisovat.

Tab. 3.1: Typ přenášených dat Modbus protokolem [39]

Kódy funkcí (Function codes) protokolu Modbus se dělí do 3 kategorií. První kategorií jsou *Veřejné kódy*, které jsou pevně definovány a validovány Modbus-IDA komunitou, která usiluje o rozšíření využití protokolu Modbus do různých segmentů průmyslu. Veřejné kódy jsou v rozsahu 01–64, 73–99, 111–127, z nichž nejsou zatím implementovány všechny. Nejznámější funkce, jejich názvy a popisy lze nalézt v tab. 3.2.

Kód	**Název funkce**	**Popis**
01	Read Coils	Čtení jednoho nebo více bitů
02	Read Discrete Inputs	Čtení jednoho nebo více bitů
03	Read Holding Registers	Čtení jednoho nebo více 16bitových registrů
04	Read Input Registers	Čtení jednoho nebo více 16bitových registrů
05	Write Single Coil	Zápis jednoho bitu
06	Write Single Register	Zápis jednoho 16bitového registru
15	Write Multiple Coils	Zápis více bitů
16	Write Multiple Registers	Zápis více 16bitových registrů

Tab. 3.2: Kódy základních funkcí protokolu Modbus [39]

Další kategorií jsou *kódy Definováno uživatelem*. Jsou v rozsahu 65–72, 100–110 a je umožněno každému poskytovateli implementovat tyto funkce dle specifik jejich zařízení a aplikačního využití. Poslední kategorií jsou *Rezervované kódy*, které jsou rezervovány pro konkrétní společnosti a nejsou dostupné pro libovolné využití. Tyto kódy jsou podmnožinou *Veřejných kódů* a jsou v rozsahu 8–10, 13–14, 41–42, 90–91 a 125–127 [63].

Modbus specifikace definuje různé typy dat umístěné v různých blocích a přiděluje jim lokální rozsah adres. Nemusí to však vždy automaticky vést k jednoduchému adresování pro účely dokumentace nebo porozumění alokované paměti dostupné přes Modbus daného zařízení. Pro usnadnění popisu umístění paměťových bloků bylo zavedeno číslování, které přidává číselné předpony k adrese daných dat, viz tab. 3.3 [63].

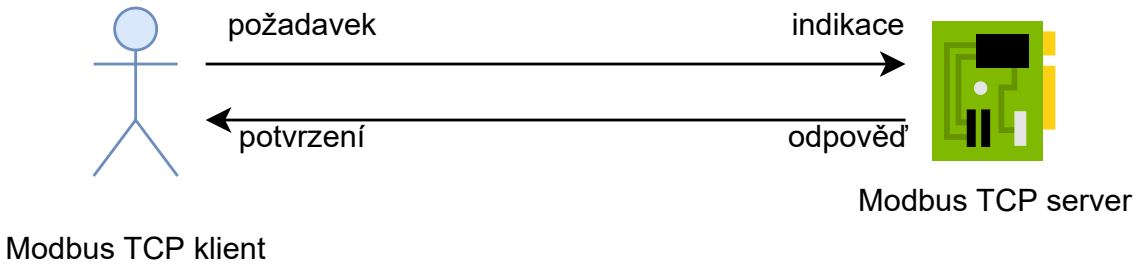
Datový blok	**Prefix**
Cívky (Coils)	0
Diskrétní vstupy (Discrete Inputs)	1
Čtené registry (Input Registers)	3
Zapisované registry (Holding Registers)	4

Tab. 3.3: Ukázka prefixů jednotlivých Modbus datových bloků [39]

Namísto specifikace položky (např. *registr 14 na adrese 13*) by manuál k zařízení uváděl danou datovou položku na adrese 4 014, 40 014 nebo 400 014. V každém případě se jako první číslo uvádí číslo 4, což identifikuje držené registry, a adresa je dále specifikována pomocí zbývajících čísel. Rozdíl mezi notacemi 4XXX, 4XXXX a 4XXXXX závisí na rozsahu adresního prostoru používaného zařízením. V případě, že jsou využity všechny registry, je vhodné použít zápis 4XXXXX, což určuje rozsah od 400 001 do 465 536. Pokud se však využívá jen malé množství registrů, častou praxí je použít rozmezí od 4 001 do 4 999 [63].

3.1.1 Modbus TCP

V této kapitole bude prozkoumán protokol Modbus TCP, jeden z klíčových protokolů používaných v průmyslové automatizaci, který umožňuje komunikaci mezi různými zařízeními v průmyslových sítích. Tento protokol je rozšířením klasického Modbus RTU protokolu, přizpůsobený pro použití v moderních TCP/IP sítích. Tento protokol je zásadní pro integraci průmyslových zařízení do více rozvinutých síťových infrastruktur. Na obr. 3.1 je znázorněno, jak probíhá komunikace mezi komunikujícími stranami. Modbus TCP požadavek je zpráva zaslaná klientem za účelem zahájení přenosu požadovaných dat. Modbus TCP indikace je přijatý Modbus TCP požadavek na straně serveru. Modbus TCP odpověď je zpráva s požadovanými daty klientem. A Modbus TCP potvrzení je odpověď s přijatými daty na straně klienta [40].



Obr. 3.1: Znázornění Modbus TCP komunikace založené na modelu klient server.

Modbus TCP zachovává základní principy Modbus protokolu, včetně jeho jednoduché struktury a spolehlivosti, přičemž přidává výhody, které přináší použití TCP/IP. Tento protokol je založen na klient-server modelu, kde server, obvykle PLC (programovatelný logický kontrolér) reaguje na požadavky klienta, např. systému řízení nebo monitorovací aplikace. Díky využití TCP/IP umožňuje Modbus TCP efektivní a rychlou komunikaci mezi zařízeními rozptýlenými v různých geografických lokacích [42].

V rámci Modbus TCP komunikace se zprávy odesílají v paketech, které jsou standardně strukturovány a obsahují adresu zařízení, funkční kód a data. Funkční kódy definují typ operace, jako je čtení registru, zápis do registru, čtení digitálních vstupů nebo výstupů a další. Díky této flexibilitě je Modbus TCP vhodný pro širokou škálu aplikací v průmyslu, od sledování a řízení procesů po sběr dat [42].

Další významnou výhodou Modbus TCP je jeho schopnost snadné integrace do stávajících síťových struktur, což značně usnadňuje nasazení v průmyslových prostředích.

Navíc, vzhledem k jeho otevřenému standardu, je Modbus TCP široce podporován mnoha výrobci průmyslových zařízení, které umožňují vytvářet heterogenní systémy s komponenty od různých dodavatelů [42].

Bezpečnost protokolu

Ačkoliv Modbus TCP nabízí výhody v oblasti průmyslové komunikace, má také své nevýhody – obzvlášť co se týče bezpečnosti. Standardní Modbus protokoly, včetně Modbus TCP, obecně neposkytují šifrování ani autentizaci, což je v současné době považováno za zásadní slabost, zejména vzhledem k rostoucím hrozbám kybernetické bezpečnosti. To znamená, že pro zajištění bezpečné komunikace je nutné implementovat další bezpečnostní vrstvy, jako je například použití virtuálních privátních sítí (VPN) nebo implementace bezpečnosti na vyšších vrstvách (TLS) [39].

Kromě bezpečnostních aspektů je také důležité při implementaci Modbus TCP v heterogenním prostředí zvážit otázky kompatibility a interoperability. I když standardní protokol nabízí určitou míru flexibility, mohou se vyskytnout problémy s kompatibilitou mezi zařízeními od různých výrobců vyžadující důkladné testování a konfiguraci pro zajištění hladkého provozu systému při implementaci řídicích systémů [39].

Modbus TCP je silným a flexibilním nástrojem pro průmyslovou automatizaci, poskytující efektivní způsob komunikace v průmyslových aplikacích. Jeho jednoduchost, spolehlivost a široká podpora činí z Modbus TCP populární volbu pro mnoho průmyslových aplikací. Nicméně, výzvy spojené s bezpečností a interoperabilitou vyžadují pozornost a pečlivé plánování, aby se zajistila bezpečná a efektivní implementace v průmyslových sítích [39].

3.1.2 Modbus RTU

Modbus RTU (Remote Terminal Unit) představuje jeden z nejrozšířenějších sériových komunikačních protokolů používaných v průmyslové automatizaci. Tento protokol je zvláště vhodný pro aplikace, kde jsou zdroje omezené a je vyžadována spolehlivá sériová komunikace. Jeho hlavní přednosti – jednoduchost, robustnost a snadná implementace – jej činí ideální volbou pro široké spektrum průmyslových aplikací. Modbus RTU využívá sériovou komunikaci pro přenos dat mezi různými zařízeními v systému, což zahrnuje PLC (programovatelné logické automaty), senzory, a akční prvky. Data jsou přenášena v binárním formátu, jenž minimalizuje velikost datových balíčků a zvyšuje rychlost komunikace, která je zásadní v reálném čase vyžadujících průmyslových prostředích. Navíc, díky své struktuře a minimálním požadavkům na konfiguraci je Modbus RTU ideální pro jednoduché i komplexní systémy, čímž usnadňuje integraci nových zařízení do stávajících sítí bez nutnosti rozsáhlých úprav nebo drahého hardwaru [41].

Modbus RTU, založený na master-slave architektuře, je jedním z klíčových protokolů v průmyslové automatizaci, který optimalizuje komunikaci mezi různými zařízeními v systému. V tomto modelu jedno master zařízení, typicky programovatelný logický automat (PLC) nebo pokročilý počítačový systém, koordinuje a řídí komunikaci s jedním nebo více slave zařízeními, jako jsou senzory a akční prvky. Tato zařízení reagují na dotazy od master zařízení, poskytují data o svém provozním stavu nebo vykonávají specifické operace podle přijatých instrukcí [41].

Komunikační proces v Modbus RTU spočívá v přenosu binárně kódovaných zpráv, které jsou strukturované pro minimalizaci přenosové režie a maximalizaci efektivity. Každá zpráva se skládá z:

- adresy slave zařízení (Slave ID) – umožňuje master zařízení specifikovat, které zařízení má reagovat,
- funkčního kódu – definuje operaci (například čtení nebo zápis registru),
- datové části – obsahuje samotnou informaci pro nebo od slave zařízení,
- kontrolního součet (CRC) – zajišťuje integritu přenášených dat.

Výběr sériového rozhraní (např. RS-232 nebo RS-485) je klíčový pro adaptaci Modbus RTU na specifické podmínky průmyslového prostředí, kde mohou být problémy s dlouhými kabely a elektromagnetickým rušením. Jednou z významných vlastností Modbus RTU je jeho deterministický charakter, který je zásadní pro aplikace vyžadující vysokou úroveň spolehlivosti a předvídatelnosti. V systémech založených na Modbus RTU musí každá žádost od master zařízení vyvolat odpověď od slave zařízení, což zajišťuje, že systémový stav je neustále aktualizován a odráží reálné provozní podmínky. Tato funkce je kritická v aplikacích jako jsou procesní řídicí systémy, kde časová přesnost a konzistence odpovědí mohou přímo ovlivnit bezpeč-

nost a efektivitu operací. Vzhledem k tomu, že Modbus RTU je závislý na fyzickém a elektrickém prostředí, ve kterém operuje, vyžaduje pečlivé plánování a implementaci, zvláště v složitých nebo rozsáhlých průmyslových instalacích [41].

Bezpečnost protokolu

Modbus RTU, přestože přináší významné přínosy pro průmyslovou automatizaci, vykazuje několik kritických omezení, která vyplývají ze základních charakteristik jeho návrhu. Jako sériový komunikační protokol, Modbus RTU čelí výzvám, které jsou spojeny s omezenou rychlostí a dosahem komunikace. Tyto limitace mohou být zásadní v aplikacích vyžadujících rychlou odezvu nebo v rozsáhlých průmyslových instalacích, kde je třeba přenášet data přes dlouhé vzdálenosti [41].

Bezpečnostní aspekty Modbus RTU představují další významnou oblast obav. Standardně tento protokol nezahrnuje šifrování ani pokročilé autentizační mechanismy. To z něj činí zranitelný cíl pro kybernetické útoky. Tento nedostatek je obzvláště kritický v kontextu současného nárůstu sofistikovanosti a frekvence kybernetických útoků, které cílí na průmyslové kontrolní systémy. Jako reakce na tuto bezpečnostní hrozbu jsou využívány izolované komunikační sítě a bezpečnostní brány, které slouží jako filtry pro monitorování a kontrolu síťového provozu, zabraňující tak neautorizovanému přístupu a šíření škodlivého kódu [41].

Správa zařízení v Modbus RTU sítích vyžaduje pečlivou adresaci a konfiguraci, aby se zabránilo adresním konfliktům a zajistila spolehlivá komunikace. V rozsáhlejších systémech s množstvím zařízení je klíčové udržet koherenci konfigurace, což může být komplikované kvůli manuální správě potřebné pro každé zařízení. V tomto ohledu je důležité zvážit návrh síťové architektury tak, aby byla zajištěna efektivní správa a škálovatelnost. Díky své spolehlivosti a jednoduchosti stále zůstává Modbus RTU důležitým prvkem v průmyslové automatizaci. Moderní implementace musí aktivně řešit výše zmíněné výzvy, zejména v oblastech bezpečnosti a správy sítě. Rozhodnutí o použití Modbus RTU by měla být prováděna s důrazem na potřebné bezpečnostní protokoly a infrastrukturu, aby se minimalizovala rizika a zlepšila celková efektivnost a spolehlivost průmyslových operačních systémů [41].

3.2 Analýza knihoven Modbus

Tato kapitola se zaměřuje na srovnání a detailní analýzu několika populárních Python knihoven pro práci s protokolem Modbus. Mezi zkoumané knihovny patří pymodbus, MinimalModbus, Modbus-tk, uModbus a PymodbusTCP.

Cílem analýzy je zhodnotit tyto knihovny z hlediska jejich implementační jednoduchosti, rychlosti komunikace a bezpečnosti kódu, což jsou klíčové faktory pro výběr správné knihovny při vývoji aplikací v oblasti průmyslové automatizace a řízení zařízení. Porovnání těchto aspektů bude poskytnuto na základě průzkumu funkcí, dokumentace a praktických zkušeností s použitím těchto knihoven. Analyzované faktory budou důkladně zkoumány s cílem poskytnout ucelený přehled výhod a nevýhod každé z knihoven a usnadnit tak výběr ideálního nástroje pro specifické potřeby v oblasti Modbus komunikace v Pythonu.

pymodbus

Knihovna pymodbus představuje komplexní nástroj pro práci s Modbus protokolem v jazyce Python. Tato knihovna nabízí podporu pro varianty protokolu Modbus (TCP, RTU, ASCII), což umožňuje komunikaci s různými zařízeními. Obsahuje rozsáhlou dokumentaci, která usnadňuje implementaci a poskytuje užitečné příklady kódu. Díky své rozmanitosti a flexibilitě je vhodná pro širokou škálu průmyslových aplikací. pymodbus podporuje jak klasické synchronní, tak i asynchronní operace, což umožňuje efektivní práci s různými typy komunikačních scénářů. Nicméně, kvůli širokému spektru funkcí může být jeho implementace náročnější pro začátečníky. Nabízí robustní a komplexní možnosti konfigurace a manipulace s daty, což umožňuje pokročilou manipulaci s Modbus daty. pymodbus je aktivně udržovaný projekt s pravidelnými aktualizacemi a díky rozsáhlé funkcionalitě je vhodný pro pokročilé uživatele a projekty, které vyžadují široké možnosti práce s protokolem Modbus [51].

MinimalModbus

MinimalModbus je kompaktní knihovna pro práci s Modbus protokolem v Pythonu. Navzdory svému minimalistickému přístupu poskytuje robustní sadu funkcí pro komunikaci pomocí Modbus RTU/TCP, což usnadňuje implementaci pro začátečníky i pokročilé uživatele. Jeho snadné použití a intuitivní rozhraní zjednodušuje integraci do různých projektů, zejména těch, které vyžadují jednoduchou a spolehlivou komunikaci s průmyslovými zařízeními. MinimalModbus poskytuje přehlednou dokumentaci a příklady kódu, což usnadňuje rychlé pochopení jeho funkcionality a implementaci do projektů. Tato knihovna podporuje různé formáty dat a umožňuje snadnou manipulaci s Modbus registry a hodnotami, což je užitečné pro čtení

a zápis dat. Díky svému zaměření na základní funkce pro komunikaci s Modbus protokolem může být vhodným výběrem pro menší projekty nebo tam, kde je potřeba jednoduché a spolehlivé řešení. Avšak kvůli své minimalistické povaze může mít omezené možnosti pro pokročilé scénáře, které vyžadují rozsáhlejší funkcionality. Své silné stránky spočívají v jednoduchosti použití a spolehlivosti, což z ní činí vhodnou volbu pro základní komunikační úkoly s Modbus protokolem v Pythonu [14].

Modbus-tk

Modbus-tk je knihovna v Pythonu, která se specializuje na tvorbu grafických uživatelských rozhraní pro komunikaci pomocí Modbus protokolu. Tato knihovna poskytuje grafické prvky a nástroje pro vytváření vizualizací dat a ovládacích prvků, což je užitečné pro vývoj aplikací s uživatelskými rozhraními. Díky této funkčnosti je Modbus-tk vhodný pro projekty, které potřebují interaktivní prostředí pro monitorování a ovládání Modbus zařízení. Nabízí jednoduché rozhraní pro komunikaci s Modbus zařízeními a umožňuje vytváření uživatelsky přívětivých aplikací. Při použití Modbus-tk je třeba brát v úvahu, že jeho zaměření na uživatelská rozhraní může být omezením pro projekty, které se soustředí spíše na komunikaci a manipulaci s daty Modbus než na vizualizaci [43].

uModbus

Knihovna uModbus představuje jednoduchý nástroj pro práci s Modbus protokolem v Pythonu. Je navržena s důrazem na jednoduchost použití, což usnadňuje začlenění do projektů a zjednodušuje komunikaci s Modbus zařízeními. Své minimalistické zaměření reflektuje v omezené, avšak stabilní sadě funkcí pro komunikaci přes Modbus RTU/TCP. Díky své jednoduchosti poskytuje uModbus přehledné rozhraní pro čtení a zápis dat z Modbus zařízení, což může být výhodné pro menší projekty nebo tam, kde je prioritou snadná integrace a spolehlivá komunikace. Avšak kvůli svému minimalistickému přístupu může mít omezené možnosti pro pokročilejší komunikační scénáře nebo projekty, které vyžadují rozsáhlejší funkcionality [57].

PymodbusTCP

PymodbusTCP je knihovna v Pythonu zaměřená na implementaci komunikace s protokolem Modbus v TCP variantě. Poskytuje robustní sadu funkcí pro usnadnění vytváření klienta nebo serveru Modbus TCP. Své silné stránky nachází v jednoduchosti implementace, což umožňuje rychlé začlenění do projektů a snadnou manipulaci s Modbus daty. PymodbusTCP nabízí základní rozhraní pro komunikaci s Modbus zařízeními, což může být užitečné pro projekty vyžadující pouze základní funkce pro

komunikaci. Avšak kvůli svému zaměření na TCP verzi Modbus protokolem může mít omezené možnosti pro další varianty protokolu, jako je například Modbus RTU nebo ASCII. Tato knihovna je vhodná pro uživatele, kteří hledají jednoduché řešení pro komunikaci s Modbus TCP zařízeními v Pythonu. Její jednoduchost a přehlednost při implementaci mohou být výhodné zejména pro menší projekty, ale může být omezená pro rozsáhlejší aplikace, které vyžadují více pokročilých funkcionalit. PymodbusTCP je užitečným nástrojem pro projekty, které se soustředí na komunikaci s Modbus TCP a vyžadují jednoduchost implementace a spolehlivost [52].

Výsledek analýzy knihoven Modbus

Kompletní sumarizovaný výsledek vlastností jednotlivých knihoven, který zahrnuje detailní srovnání jejich klíčových charakteristik je vizualizován v tab. 3.4. Tato tabulka poskytuje srovnání pěti populárních Python knihoven určených pro práci s Modbus protokoly. Je strukturována do tří hlavních sloupců:

- **Analyzovaná knihovna** – Zde jsou uvedeny názvy knihoven, které byly zahrnuty do analýzy.
- **Výhody** – Tento sloupec detailně popisuje přednosti každé knihovny, včetně jejich obecných charakteristik.
- **Nevýhody** – Jsou zde uvedeny hlavní nevýhody každé knihovny, jako jsou omezení v podpoře protokolů, frekvence aktualizací, složitost dokumentace, nebo omezení funkcionality.

Analyzovaná knihovna	Výhody	Nevýhody
pymodbus	jednoduchost, Modbus TCP/RTU/ASCII, flexibilita	struktura dokumentace, nízká úroveň abstrakce,
MinimalModbus	jednoduchost, nízká výpočetní náročnost, dokumentace	pouze Modbus RTU, omezené funkcionality
Modbus-tk	úroveň abstrakce, Modbus TCP/RTU/ASCII, GUI	nedostatečnost dokumentace, frekvence aktualizací
uModbus	jednoduchost, open-source, podpora Modbus RTU	omezené funkcionality, slabá podpora Modbus TCP
PymodbusTCP	jednoduchost, flexibilita, práce s registry (daty)	podpora pouze pro Modbus TCP

Tab. 3.4: Souhrn vlastností Modbus knihoven pro Python

Důležitější je avšak následující tab. 3.5, kde jsou v jednotlivých sloupcích uvedeny požadavky na knihovny a každá buňka je vyplněna slovy Ano nebo Ne v závislosti na tom, zda daná knihovna splňuje požadavky:

- Jednoduchost práce s knihovnou – Hodnoceno na základě uživatelských recenzí a dokumentace.
- Dobrá dokumentace – Hodnoceno na základě kvality a dostupnosti dokumentace.
- Pravidelné aktualizace – Zjištění skutečnosti, zda knihovna dostává pravidelné aktualizace.
- Podpora Modbus RTU i TCP – Zjištění skutečnosti, zda knihovna podporuje oba požadované typy protokolů Modbus.

Tato tabulka poskytuje rychlý přehled o tom, které knihovny jsou nejvhodnější pro specifické potřeby a požadavky v rámci projektů zahrnujících Modbus protokoly.

Na základně výsledků analýzy a zmíněně tabulky byla pro realizaci datové komunikace využívající protokol Modbus TCP a Modbus RTU byla zvolena knihovna pymodbus. Tato knihovna vyniká svou rozsáhlou podporou pro obě varianty Modbus protokolu, což umožňuje efektivní a flexibilní manipulaci s daty Modbus. Pymodbus poskytuje uživatelům bohatou sadu funkcí a detailní možnosti konfigurace, což umožňuje přizpůsobení komunikace specifickým potřebám projektu. Tato robustnost a flexibilita byly klíčové pro její výběr, protože umožňují jednoduché zapojení do širokého spektra průmyslových aplikací.

Na rozdíl od knihovny MinimalModbus, která se zaměřuje převážně na jednoduchost a spolehlivost pro Modbus RTU bez široké podpory různých variant protokolu a konfiguračních možností, pymodbus nabízí komplexnější řešení. Modbus-tk, i přestože je bohatá na uživatelská rozhraní, se nezdá být ideální volbou pro základní datovou komunikaci bez dalších uživatelských interakcí. UModbus a PymodbusTCP, ačkoli jsou jednoduché a přístupné, mají omezené funkčnosti a podporují pouze jednu variantu Modbus protokolu, což je výrazně limituje ve srovnání s pymodbus.

Výběr knihovny pymodbus byl tudíž logickým krokem díky její schopnosti pokrýt obě varianty Modbus protokolu a nabídnout širokou škálu funkcionalit pro efektivní manipulaci s daty a adaptabilitu v různých komunikačních scénářích. Tato schopnost integrace a konfigurace dle potřeb uživatele činí pymodbus ideální volbou pro náročné průmyslové aplikace, které vyžadují robustní a spolehlivé řešení pro komunikaci s průmyslovými zařízeními.

Knihovna	**Jednoduchost**	**Dokumentace**	**Aktualizace**	**Podpora RTU/TCP**
MinimalModbus	Ano	Ano	Ne	Ne
Modbus-tk	Ano	Ano	Ano	Ano
uModbus	Ano	Ano	Ne	Ne
PymodbusTCP	Ano	Ano	Ano	Ne
pymodbus	Ano	Ano	Ano	Ano

Tab. 3.5: Porovnání Modbus knihoven

3.3 MQTT protokol

MQTT (Message Queuing Telemetry Transport) je protokol navržený pro optimalizaci komunikace v síťových prostředích s omezenými zdroji, což je typické pro aplikace v oblasti Internetu věcí (IoT). Tento protokol využívá publikování/odběr (publish/subscribe) model, který umožňuje vysokou míru flexibility ve správě zpráv mezi zařízeními. V rámci tohoto modelu mohou zařízení publikovat informace na definovaná témata (topics), zatímco ostatní zařízení se mohou k těmto tématům přihlásit a přijímat relevantní zprávy, což minimalizuje potřebu trvalých spojení a snižuje nároky na šířku pásma [54].

3.3.1 Popis komunikace

Architektura MQTT umožňuje klientům publikovat zprávy na definovaná témata, zatímco jiní klienti mohou být na tato témata přihlášeni a přijímat odpovídající zprávy. Klíčovým prvkem tohoto systému je MQTT broker, který funguje jako centrální uzel, řídící distribuci zpráv mezi klienty a spravující připojení, odběry a publikace [54].

MQTT je navržen tak, aby byl maximálně lehký a efektivní, což je nezbytné v prostředích s omezenou šířkou pásma nebo s omezenými výpočetními zdroji. Díky své minimální režii a nízkým požadavkům na síťové zdroje je MQTT ideální pro aplikace vyžadující rychlé a spolehlivé zpracování zpráv ve velkém měřítku, což umožňuje jeho použití ve složitých průmyslových aplikacích i v jednodušších domácích automatizačních systémech [54].

MQTT nabízí tři úrovně záruky doručení zpráv, které umožňují aplikacím optimalizovat komunikaci podle konkrétních požadavků:

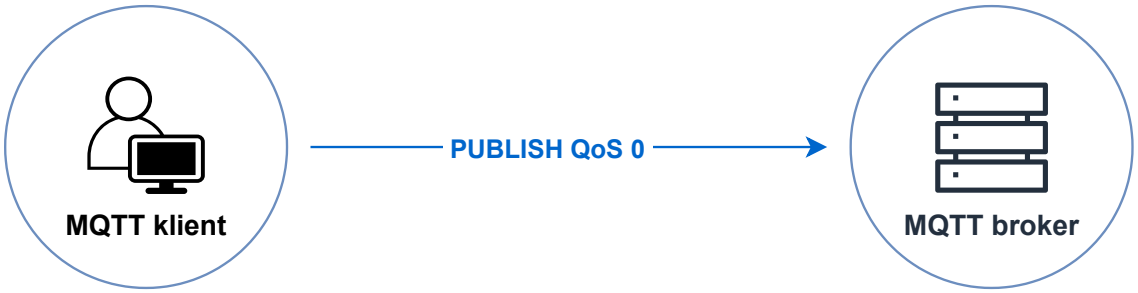
- Kvalita služby úrovně 0 (*QoS 0*) -– zprávy jsou doručeny nejvýše jednou, což je nejrychlejší a nejméně spolehlivý režim, vhodný pro méně důležité zprávy, kde není nutné zaručit doručení.
- Kvalita služby úrovně 1 (*QoS 1*) -– zprávy jsou doručeny alespoň jednou, což zajišťuje spolehlivější doručení než režim na úrovni 0, ale může dojít k duplikaci zpráv.
- Kvalita služby úrovně 2 (*QoS 2*) -– každá zpráva je garantovaně doručena právě jednou. Tato úroveň je nejpomalejší, ale zajišťuje nejvyšší úroveň spolehlivosti a je vhodná pro kritické údaje [61].

Kvalita služby úrovně 0

Úroveň kvality služby (Quality of Service, QoS) 0 v MQTT, označovaná jako *maximálně jednou* (at most once), je základní úroveň doručení zpráv, která garantuje

nejmenší režii při komunikaci. Tato úroveň je nejspolehlivější z hlediska rychlosti doručení, protože **nevyžaduje potvrzení o přijetí zprávy od příjemce**, viz obr. 3.2. To znamená, že zprávy jsou odesílány pouze jednou a klient nevyžaduje, ani nečeká potvrzení o přijetí zprávy brokerem [61].

V praxi to znamená, že zpráva může být doručena vůbec nebo jednou, což je přijatelné pro případy, kdy ztráta nějakých zpráv není kritická. Například, v aplikacích, kde jsou zprávy časté a každá individuální zpráva není nezbytně nutná pro srozumitelnost celkového kontextu (například telemetrická data z čidel měřících teplotu, která jsou posílána každou minutu). Tato úroveň je také nejefektivnější co se týče využití síťových zdrojů, protože snižuje zatížení sítě tím, že eliminuje potřebu výměny mnoha kontrolních zpráv, které jsou jinak potřebné pro zajištění vyšších úrovní spolehlivosti doručení. Pro aplikace, kde je priorita rychlost přenosu před absolutní spolehlivostí doručení nebo kde není dopad ztráty zprávy zásadní, je QoS 0 vhodnou volbou. Je to ideální pro masové posílání dat, kde může být občasné vynechání zprávy akceptovatelné ve srovnání s náklady na zajištění spolehlivějšího doručení zpráv [61].



Obr. 3.2: Znázornění 0.stupně QoS MQTT

Kvalita služby úrovně 1

Kvalita služby (Quality of Service, QoS) úroveň 1 v MQTT, označovaná jako *alespoň jednou* (at least once), zajišťuje, že zpráva je doručena příjemci alespoň jednou, ale **může dojít k duplikaci zpráv**. Tato úroveň QoS vyžaduje od příjemce potvrzení o přijetí zprávy, viz obr. 3.3. To znamená, že pokud odesílatel neobdrží potvrzení (PUBACK) v určitém časovém limitu, zprávu odešle znovu [61].

Při použití QoS 1 odesílatel uchovává zprávu v lokální paměti až do doby, kdy obdrží potvrzení o jejím přijetí. Tento proces začíná, když klient odešle zprávu MQTT s přiřazeným identifikátorem zprávy (Packet Identifier). Příjemce po obdržení této zprávy odpoví zprávou PUBLISH ACK, která obsahuje stejný identifikátor zprávy. Jakmile je toto potvrzení přijato odesílatelem, může být zpráva z lokální paměti odstraněna. Tento mechanismus zajišťuje, že zpráva bude doručena alespoň jednou,

ale kvůli možnému zasílání duplicitních zpráv v důsledku ztráty potvrzení nebo jiných síťových problémů není zajištěna jedinečnost doručení zprávy. QoS 1 je vhodná pro případy, kde je ztráta zprávy nepřijatelná, ale duplicitní zprávy nezpůsobují významný problém. Tato úroveň je často používána v aplikacích, kde je potřeba zaručit doručení informace, jako jsou různé druhy upozornění a ovládací zprávy v domácích automatizačních systémech, bezpečnostních systémech, nebo v průmyslových monitorovacích aplikacích, kde je důležité, aby zprávy byly doručeny, i když může docházet k jejich opakování [61].

Obr. 3.3: Znázornění 1.stupně QoS MQTT

Kvalita služby úrovně 2

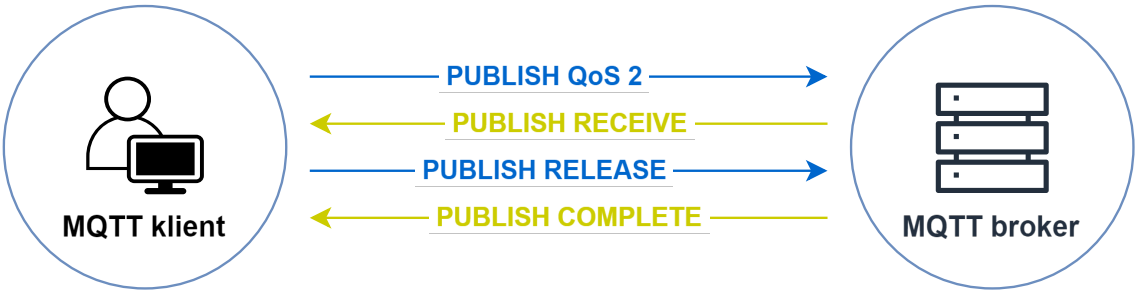
Kvalita služby (Quality of Service, QoS) úroveň 2 v MQTT, označovaná jako *právě jednou* (exactly once), je nejvyšší úroveň záruky doručení zpráv v protokolu MQTT. Tato úroveň zaručuje, že každá zpráva je doručena **právě jednou a eliminuje možnost duplicitního doručení**, viz obr. 3.4. To je klíčové pro aplikace, kde je důležité zachovat přesnou sekvenci a jedinečnost komunikačních operací [61].

Proces doručení zprávy na úrovni QoS 2 je zajištěn dvoufázovým potvrzovacím procesem, který zahrnuje následující kroky:

1. **Fáze – Odeslání a potvrzení přijetí zprávy:**
   - Odesilatel začne proces tím, že odesílá zprávu s příznakem QoS 2, označenou unikátním identifikátorem zprávy.
   - Přijímač přijme tuto zprávu a odesílá zpět potvrzení o přijetí (PUBLISH RECEIVE), které také obsahuje identifikátor zprávy.
   - Po přijetí PUBLISH RECEIVE odesilatel ukládá stav zprávy a odesílá zprávu PUBLISH RELEASE, což signalizuje, že zpráva byla potvrzena a je připravena být zpracována příjemcem [44].
2. **Fáze – Zpracování a finální potvrzení:**
   - Přijímač po obdržení PUBLISH RELEASE zpracuje zprávu a odesílá finální potvrzení doručení (PUBLISH COMPLETE) zpět odesilateli.

- Odesilatel po obdržení PUBLISH COMPLETE může zprávu bezpečně odstranit z dočasné paměti, protože je potvrzeno její úplné zpracování příjemcem [44].

Tento komplexní proces zajišťuje, že zprávy nebudou ztraceny a nebudou doručeny vícekrát. To je zásadní v aplikacích, kde je důležitá přesnost a spolehlivost datové komunikace. Jsou to například finanční transakce, kritické monitorovací systémy nebo operace, které vyžadují synchronizaci přesného stavu mezi zařízeními. Vzhledem k vyšší režii spojené s touto úrovní doručení je QoS 2 obvykle používán v situacích, kde jsou požadavky na integritu dat vyšší než požadavky na rychlost komunikace. Jelikož proces vyžaduje více výměn zpráv mezi odesilateli a přijímači, může mít negativní vliv na výkon v prostředích s vysokou latencí nebo omezenou šířkou pásma [61].



Obr. 3.4: Znázornění 2.stupně QoS MQTT

Oznámení nečekané chyby v komunikaci

Funkce *last will and testament* (poslední vůle) v protokolu MQTT je významný mechanismus pro řízení a reakci na neočekávané odpojení klientů z MQTT sítě. Tato funkce umožňuje klientům specifikovat zprávu, která bude automaticky odeslána ostatním připojeným klientům, když dojde k neočekávanému přerušení spojení [60].

Klient specifikuje svou poslední vůli při navazování spojení s MQTT brokerem. Tato zpráva obsahuje:

- tématický kanál, na který má být zpráva odeslána,
- obsah zprávy,
- požadovanou úroveň kvality služby.

MQTT broker uchovává informace o poslední vůli klienta v paměti po dobu trvání jeho spojení. Broker nevyužívá tuto zprávu, dokud nedojde k nečekanému odpojení. Pokud se klient odpojí bez řádného ukončení spojení (například v důsledku výpadku sítě nebo selhání zařízení), broker aktivuje a publikuje zprávu poslední vůle na předem určené téma. Poslední vůle je zásadní pro umožnění rychlé reakce

na chyby nebo selhání v systému a pomáhá udržet integritu systému v případě neočekávaných událostí [60].

3.3.2 Bezpečnost protokolu

Ačkoliv MQTT představuje efektivní řešení pro komunikaci v IoT a průmyslových aplikacích, jeho standardní verze neobsahuje šifrování ani pokročilé metody autentizace, což může představovat bezpečnostní rizika v prostředích, kde je důležitá ochrana dat a soukromí. Tyto omezení mohou způsobovat zranitelnosti vůči odposlechům nebo útokům typu "man-in-the-middle", které mohou ohrozit integritu a důvěrnost přenášených dat [11].

Pro zajištění vyšší úrovně bezpečnosti byla vyvinuta šifrovaná varianta MQTT, označovaná jako MQTTs, která implementuje TLS/SSL šifrování. Tato verze protokolu poskytuje robustní zabezpečení datových toků mezi klienty a MQTT brokerem, čímž výrazně zvyšuje bezpečnost komunikace. Šifrování TLS/SSL chrání data přenášená v rámci MQTT protokolu před neoprávněným přístupem a útoky, zajišťuje integritu dat a autentizaci komunikujících stran. Implementace MQTTs vyžaduje pečlivé řízení bezpečnostních certifikátů a klíčů, které je klíčové pro udržení důvěryhodnosti a efektivity šifrování. Správná konfigurace TLS/SSL, včetně správného výběru šifrovacích algoritmů a nastavení parametrů, je nutné pro optimalizaci bezpečnosti bez kompromisu na výkonu systému [11].

S rostoucím významem IoT a s tím související potřebou bezpečné komunikace v síti je zřejmé, že MQTT a jeho šifrované varianty budou nadále hrát klíčovou roli v digitalizaci průmyslu a automatizaci. Zabezpečení těchto protokolů musí být prioritou pro organizace, které se snaží chránit své operace a data před kybernetickými hrozbami, a zároveň udržet schopnost efektivní a spolehlivé komunikace mezi zařízeními v rozsáhlých IoT sítích [3].

3.4 HTTPS protokol

HTTPS (Hypertext Transfer Protocol Secure) je rozšířený komunikační protokol používaný na internetu, který je navržen pro zabezpečenou výměnu informací mezi klientem a serverem. Protokol HTTPS je v podstatě rozšířením standardního HTTP, do kterého přidává vrstvu šifrování, aby poskytoval vyšší úroveň zabezpečení. Zabezpečení je realizováno pomocí protokolů SSL (Secure Sockets Layer) nebo jeho nástupce TLS (Transport Layer Security), které zajišťují šifrování komunikace [59].

Popis a bezpečnost protokolu

HTTPS je klíčovým prvkem pro zajištění bezpečnosti na internetu, základem jeho funkcionality je šifrování dat mezi klientem, jako je webový prohlížeč, a serverem. Toto šifrování zajišťuje, že informace odesílané po internetu jsou chráněny před neoprávněným přístupem třetích stran. Funkce šifrování je nezbytná pro ochranu osobních, finančních a dalších citlivých dat, která by jinak mohla být snadno předmětem útoku. Realizace šifrování probíhala dříve skrze protokol SSL, dnes se však již využívá jeho nástupce TLS. Tyto protokoly nejenže zajišťují šifrování dat, ale také garantují integritu dat přenášených mezi komunikačními partnery, což chrání data před neoprávněnou modifikací během přenosu [59].

Dalším zásadním prvkem zabezpečení protokolu HTTPS je autentizace serverů pomocí certifikátů X.509. Tyto certifikáty jsou vystavovány důvěryhodnými certifikačními autoritami (CA), a ujišťují uživatele o autentičnosti serveru, se kterým komunikují. Tento mechanismus autentizace pomáhá eliminovat riziko spojení s falešným nebo podvodným serverem, což je častá taktika využívaná ve phishingových útocích. Certifikáty tedy hrají klíčovou roli v ochraně proti podvodům a manipulaci s daty [59].

Tato infrastruktura zajišťuje, že komunikace mezi webovým prohlížečem a serverem je bezpečná, nejen šifrováním obsahu přenosu, ale i potvrzováním pravosti a integrity komunikačního partnera. Výsledkem je robustní zabezpečené webové prostředí, kde uživatelé mohou bez obav vykonávat online transakce, sdílet citlivé informace a prohlížet internet s důvěrou v ochranu svých dat [17].

Reálné využití

V současném digitálním věku se HTTPS stal nezbytným standardem pro webové aplikace, zejména v sektorech, kde se manipuluje s osobními a finančními informacemi, jako jsou e-commerce, online bankovnictví a další relevantní služby. Protokol HTTPS zajišťuje šifrování dat mezi klientem a serverem, které chrání přenášené

informace před neoprávněným přístupem a útoky typu muže uprostřed (man-in-the-middle). Tato ochrana zajišťuje zachování důvěry uživatelů a jejich bezpečnost při online transakcích a komunikaci [59].

Implementace a správa HTTPS však s sebou nese řadu výzev. Správa X.509 certifikátů a jejich pravidelné obnovování vyžadují pečlivou administrativu, zejména v rozsáhlých organizacích s mnoha doménami. Správná konfigurace serverů a certifikátů je kritická pro zajištění nepřetržitého zabezpečení a vyžaduje systematické plánování a monitorování. Kromě administrativních a logistických výzev přináší šifrování a dešifrování dat také zvýšené nároky na výpočetní zdroje, což může vést k zatížení serverů a zpomalení jejich odezvy, zejména v případě vysokého objemu datového provozu [59].

Navzdory těmto výzvám jsou přínosy HTTPS nepopiratelné. Zabezpečení komunikace, ověřování identity serverů prostřednictvím certifikátů a ochrana integrity dat jsou klíčové aspekty, které HTTPS nabízí. Tyto funkce jsou nezbytné pro ochranu uživatelů a pro udržení důvěry ve virtuálním prostředí. Vzhledem k neustále se zvyšujícímu počtu kybernetických hrozeb a útoků je více než kdy jindy důležité, aby byly tyto bezpečnostní protokoly udržovány na nejvyšší možné úrovni, aby se zajistilo bezpečné a spolehlivé internetové prostředí [17].

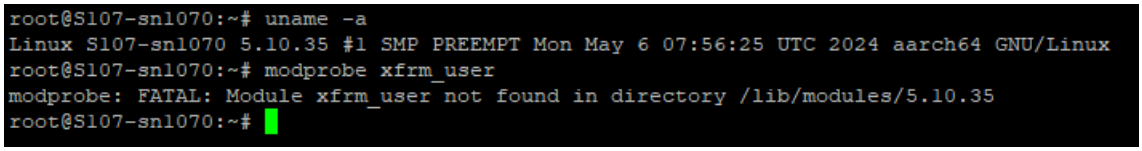
4 Síťová komunikace rozvaděče ESCON

Tato práce staví na již připraveném rozvaděči ESCON-C4 společnosti EASYCON Solution s.r.o. Výchozím stavem je vyrobený rozvaděč se zprovozněným IPsec tunelem z routeru RUTX11 (VPN gateway) na Easycon servery, kde se nachází MQTT broker a HTTP server. V práci jsou poté v následujících kapitolách a jejich podkapitolách popsány jednotlivé dodělávané komunikační a zabezpečovací prvky. Konkrétně se jedná o návrh, realizaci a testování:

- Modbus RTU komunikace,
- Modbus TCP komunikace,
- MQTT + HTTPS komunikace,
- Systému zabezpečení pro ESCON rozvaděč,
- GUI pro automatickou konfiguraci VPN tunelu.

Síťová komunikace rozvaděče ESCON zahrnuje veškerou výměnu dat mezi zařízeními a systémy, které jsou součástí rozvaděče. Tato komunikace se týká především správy, monitoringu a ovládání průmyslových zařízení a procesů, které jsou do tohoto rozvaděče integrovány. V podkapitole 4.1 je tato architektura detailně popsána a podkapitola 4.2 se dále zaměřuje na návrh zabezpečení síťové komunikace rozvaděče pomocí vlastního IPS a VPN tunelu WireGuard. Zde je nutné uvést, že zabezpečení pomocí VPN protokolu WireGuard bylo zvoleno, protože nebylo možné plně nakonfigurovat IPsec tunel z důvodu nepodpory potřebného modulu *xfrm_user* v aktuálně nejnovějším linuxovém jádře operačního systému zařízení Unipi (stav ke dni 6.5.2024), viz obr. 4.1. Hlavní funkcí modulu *xfrm_user* je poskytování rozhraní mezi uživatelským prostředím a jádrem Linuxu pro konfiguraci a správu zabezpečení na síťové vrstvě. Pomocí tohoto modulu mohou aplikace v uživatelském prostoru služby IPsec provádět následující operace:

- nastavení bezpečnostních politik (Security Policies) – tyto politiky určují, jaký druh zabezpečení bude použit pro určité pakety na základě jejich zdroje, cíle a dalších parametrů,
- správa bezpečnostních asociací (Security Associations) – tyto asociace jsou dohody, které specifikují konkrétní parametry pro šifrování a autentizaci komunikace mezi dvěma nebo více hostiteli.



Obr. 4.1: Zobrazení verze linuxového jádra a nepřítomnosti modulu *xfrm_user*

V kapitole 4.2.2, kde jsou analyzovány technologie IPsec a WireGuard (vybraný bude zabezpečovat komunikaci v lokální síti rozvaděče), jsou uvedeny další argumenty, proč je využití protokolu WireGuard v konečném důsledku výhodnější, než využití protokolové sady IPsec.

4.1 Návrh architektury rozvaděče ESCON

V této části práce je navržena architektura komunikace rozvaděče ESCON, viz obr. 4.2. Tato architektura, je navržena tak, aby poskytovala robustní a efektivní řešení pro správu a kontrolu průmyslových procesů. Základem jsou klíčové komponenty a protokoly, které zajišťují bezproblémovou komunikaci a vzájemnou spolupráci mezi různými zařízeními v síti. Mezi základní komunikační uzly této architektury patří:

- router RUTX11,
- řídicí a měřicí jednotka Unipi Patron S107,
- ethernetový přepínač,
- průmyslové elektroměry,
- programovatelné logické automaty (PLC),
- energetické jednotky (KGJ) a
- Easycon cloud.

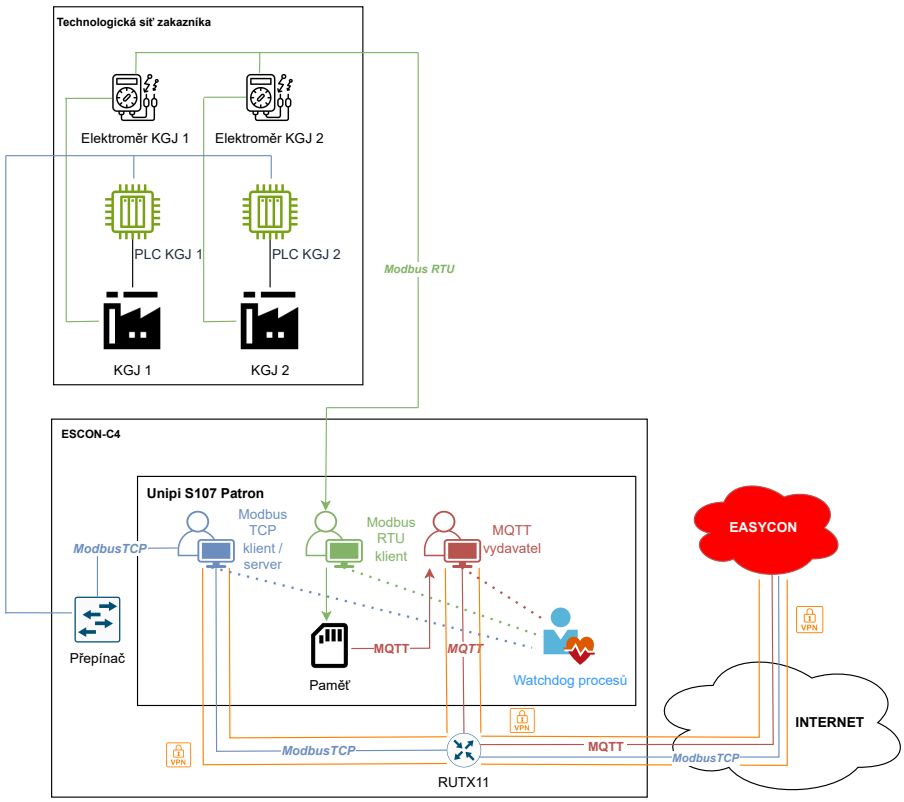
Router RUTX11

Tento router slouží jako hlavní brána pro přístup k internetu. Jeho úkolem je zabezpečit online připojení pro vzdálený monitoring a řízení systému, stejně jako pro propojení s externími cloudovými službami (Easycon cloud, Remote Management System). Díky pokročilým možnostem konfigurace a podpoře různých síťových protokolů poskytuje RUTX11 flexibilní a spolehlivou cestu pro data mezi lokální sítí a veřejným internetem.

Řídicí a měřicí jednotka Unipi Patron S107

Jako centrální řídicí jednotka, Unipi Patron S107 nejenže koordinuje komunikaci přes MQTT, ale také ukládá a zpracovává data z různých částí systému. Tento modul také slouží jako platforma pro integraci a správu rozsáhlého spektra senzorů a akčních prvků, což umožňuje centralizované monitorování a řízení průmyslových procesů. Na této jednotce běží Modbus TCP klient, Modbus TCP server, Modbus RTU klient, MQTT vydavatel a dohledový systém (Watchdog procesů).

Modbus TCP klient může aktivně žádat o data od Modbus serverů nebo zařízení (např. senzory, elektroměry, PLC). Tyto informace mohou být důležité pro monitoring stavu zařízení, diagnostiku a pro provádění operativních úprav v reálném čase.



Obr. 4.2: Softwarová architektura procesů rozvaděče ESCON-C4

Klient může posílat příkazy k ovládání různých zařízení na dálku, např. zapínat nebo vypínat motory, nastavovat parametry procesů nebo resetovat systémy. Posbíraná data jsou dalším procesem, Modbus TCP serverem, propagována do Easycon cloudu.

Modbus RTU klient je založen na protokolu Modbus RTU, který umožňuje sériovou komunikaci (sběrnice RS485). Klient čte požadovaná data z elektroměrů na základě definovaných registrů, identifikátorů Modbus RTU serverů (Slave ID), přenosové rychlosti a kontrolního součtu. Tyto údaje slouží k zajištění, že čtené hodnoty z definovaných registrů jsou správné a neporušené. Tyto data mohou zahrnovat teploty, tlaky, výkony, napětí, stav zařízení a jsou ukládána do paměti jednotky do té doby než je další proces MQTT vydavatel odešle do Easycon cloudu ke zpracování a uložení do databáze.

MQTT vydavatel (publisher) je zásadní komponenta v této architektuře, jelikož je jedním z hlavních protokolů používaných v Internetu věcí (IoT) pro lehkou a efektivní komunikaci mezi zařízeními. MQTT vydavatel má za úkol odesílat zprávy, které obsahují data z různých senzorů nebo zařízení, do MQTT brokeru (broker v Easycon cloudu). V případě problémů s MQTT vydavatelem, resp. MQTT komunikací nebo s implementovanými VPN tunely je přítomna záložní HTTPS komunikace. Tato

záložní komunikace v případě výpadku MQTT komunikace nebo úplného výpadku internetu přebírá iniciativu ve zpracování a odesílání dat do Easycon Cloudu. Více informací o té funkcionalitě, viz kap. 5.3 – Implementace MQTT a HTTPS komunikace.

Dohledový systém (Watchdog procesů) je zásadní pro zajištění nepřetržitého a bezpečného provozu systému. Monitoruje běh kritických procesů a v případě detekce problémů nebo selhání automaticky restartuje systém nebo upozorní operátory. Dohledový systém představuje klíčovou bezpečnostní a stabilizační komponentu v průmyslových a informačních systémech. Tento systém je nezbytný pro zajištění nepřetržitého a bezpečného provozu systému, jelikož aktivně monitoruje běh kritických procesů a zasahuje v případě jejich selhání nebo nefunkčnosti. Watchdog neustále ověřuje, zda jsou všechny klíčové procesy v běhu a zda fungují v rámci definovaných parametrů. To zahrnuje procesy řízené pomocí komunikačních protokolů jako Modbus TCP a Modbus RTU, stejně jako operace spojené s MQTT vydavateli.

V kontextu neustále rostoucích požadavků na průmyslové systémy, kde dostupnost a bezpečnost jsou na prvním místě, je dohledový systém (watchdog) nezbytný pro udržení kontinuity, efektivity a bezpečnosti operací. Jeho schopnost interakce s komunikačními prvky a systémy je klíčová pro jakýkoliv průmyslový systém, včetně ESCON rozvaděče.

Ethernetový přepínač

Ethernetový přepínač představuje centrální distribuční bod v lokální síti, kde je koordinován tok dat mezi připojenými zařízeními. Umožňuje simultánní přenos datových paketů mezi více zařízeními, jako jsou PLC systémy a další senzorové jednotky. Tímto způsobem zefektivňuje využití síťové kapacity a zároveň minimalizuje riziko kolizí a přetížení sítě. Přes tento přepínač jsou propojovány zařízení Unipi a router RUTX11 s technologickými sítěmi zákazníka.

Průmyslové elektroměry

Průmyslové elektroměry jsou speciálně navržené měřicí zařízení určené k monitorování a zaznamenávání spotřeby elektrické energie v průmyslových aplikacích. Tyto elektroměry jsou klíčové pro zajištění efektivního provozu v různých průmyslových prostředích. Poskytují vysokou přesnost měření, která je zásadní pro správné účtování energie a analýzu spotřeby. Kromě základních měření jako je celková spotřeba (kWh), jsou průmyslové elektroměry schopné monitorovat celou další řadu parametrů, například aktuální činný výkon nebo jalový výkon výrobního energetického zařízení.

Na tyto elektroměry je napojena řídicí a měřicí jednotka Unipi Patron S107, která z těchto elektroměrů pomocí Modbus RTU protokolu dokáže číst požadované veličiny.

Programovatelné logické automaty (PLC)

Programovatelné logické automaty jsou základním stavebním kamenem průmyslové automatizace a kontrolních systémů. PLC jsou používány k automatizaci specifických procesů, strojů nebo technologií v rámci průmyslového zařízení. Fungují na základě programovatelného paměťového zařízení, které ukládá instrukce a provádí funkce, jako je logika, sekvenční kontrola, časování, počítání a aritmetika, řízené vstupy a výstupy. V kontextu práce slouží PLC jako Modbus TCP server, jenž odesílá požadovaná data Modbus TCP klientovi a řídí se jeho příkazy. Na základě těchto příkazů jsou řízeny energetické jednotky.

Energetické jednotky

Energetické jednotky, známé také jako kogenerační jednotky (KGJ), jsou klíčové pro efektivní výrobu a využití energie v průmyslových zařízeních. Kogenerační jednotky jsou využívány především tam, kde je potřeba tepla a elektrické energie, jako jsou chemické závody, papírny, nemocnice a další velká zařízení, která vyžadují obě formy energie pro svůj chod. Tyto jednotky mají vetšinou přidělená PLC, jenž slouží pro jejich řízení pomocí již zmíněných protokolů Modbus TCP nebo Modbus RTU [33].

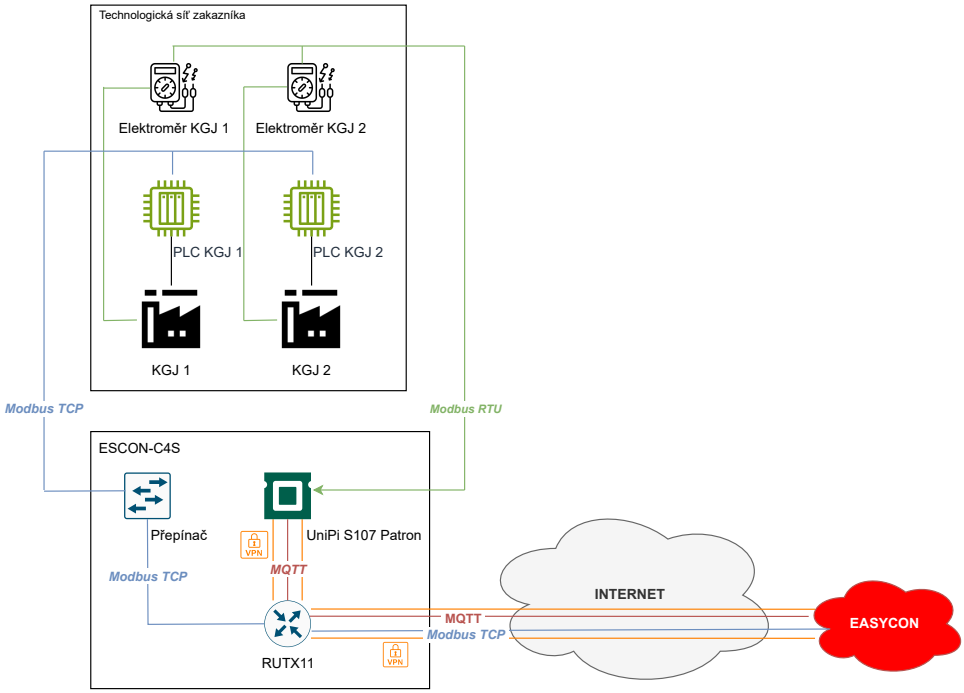
Easycon cloud

Easycon cloudem je myšlen informační systém společnosti EASYCON Solution s.r.o., která se specializuje na digitalizaci, monitorování, řízení a efektivitu výrobních procesů prostřednictvím své IoT platformy.

4.2 Návrh bezpečnosti rozvaděče ESCON

Bezpečnost komunikace je klíčovou součástí celkové bezpečnosti a efektivity průmyslových řídicích procesů. V rámci tohoto tématu je třeba důkladně prozkoumat různé aspekty, které ovlivňují bezpečný provoz rozvaděčů a celých průmyslových zařízení. To zahrnuje návrh, instalaci, provoz a údržbu rozvaděčů, stejně jako implementaci ochranných a monitorovacích systémů. Následující odstavce detailněji popisují návrh zabezpečení z hlediska síťové bezpečnosti.

Ideové schéma zapojení ESCON rozvaděče v procesu monitorování a řízení průmyslových jednotek lze vidět na obr. 4.3. V horní části schématu jsou dva elektroměry, každý je přiřazen k odpovídající kogenerační jednotce. Tyto elektroměry jsou propojeny za účelem sběru a analýzy dat o spotřebě a výrobě elektrické energie k jednotce UniPi S107 Patron. Komunikační spojení mezi elektroměry, PLC a dalšími komponentami je realizováno prostřednictvím dvou hlavních protokolů – Modbus TCP a Modbus RTU. Ve střední části je ethernetový přepínač, který slouží jako centrální bod pro propojení různých zařízení v síti. Dále je zobrazena jednotka UniPi S107 Patron, řídící jednotka s podporou pro MQTT. V dolní části obrázku je zobrazen router RUTX11, který má integrovanou podporu VPN pro zabezpečenou vzdálenou komunikaci. Celý systém je připojen k internetu s možností vzdáleného přístupu ze strany cloudu EASYCON přes zavedený VPN tunel.

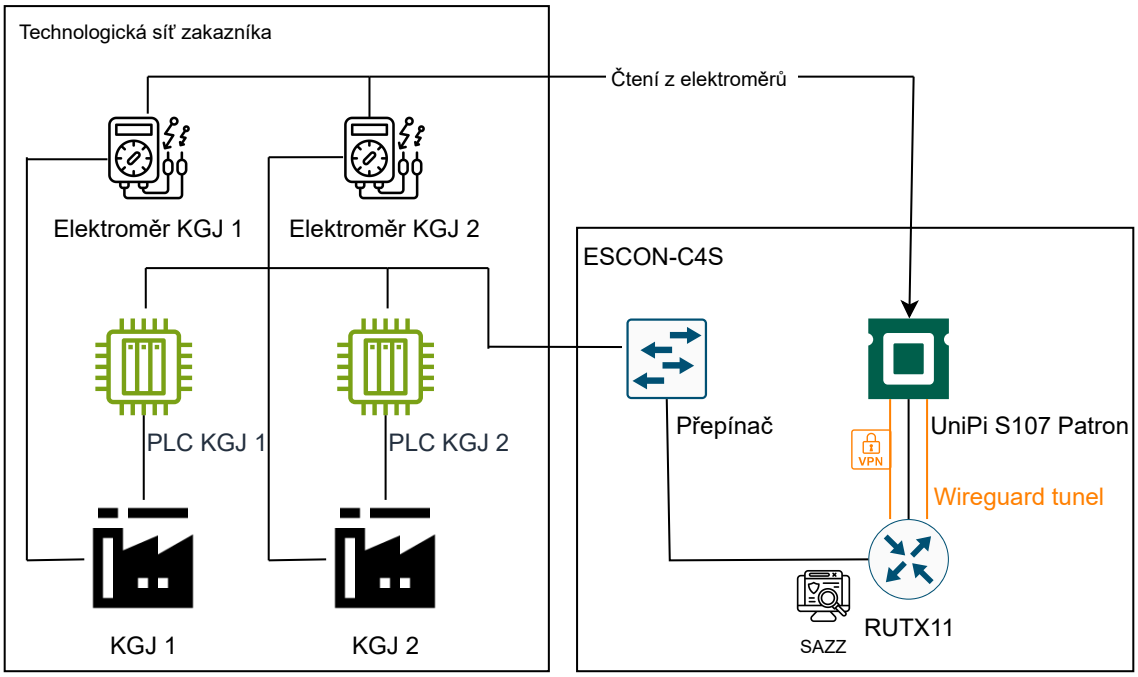


Obr. 4.3: Ideové schéma zapojení rozvaděče ESCON-C4 do průmyslového procesu

Součástí práce je dle požadavku zadavatele práce (společnost EASYCON Solution s.r.o.) implementovat zabezpečení komunikace přímo v lokální síti, kde se nachází průmyslová zařízení a řídicí jednotky, viz obr. 4.4. Pro zajištění této skutečnosti bylo navrženo zabezpečení síťové vrstvy (3. vrstva ISO/OSI modelu) pomocí:

- prostředků aktivní ochrany pro monitoring a řešení případných nežádoucích zařízení v lokální síti – viz kapitola 6.1 – Systém pro aktivní zabezpečení průmyslových zařízení a
- VPN tunelů pro zajištění šifrování a integrity dat mezi routerem RUTX11 a jednotkou Unipi S107 Patron – viz kapitola 6.2 – GUI pro zjednodušenou konfiguraci WireGuard komunikace.

Tato bezpečnostní opatření by měly doplnit již aktivní bezpečnostní prvky jako jsou IPsec tunely z EASYCON serverů na router RUTX11 v ESCON rozvaděčích.



Obr. 4.4: Znázornění umístění bezpečnostních funkcí rozvaděče ESCON-C4

4.2.1 Návrh systému pro aktivní zabezpečení průmyslových zařízení

Navržený systém aktivního zabezpečení průmyslových zařízení, zkráceně SAZZ, jehož implementace je popsaná v kapitole 6.1, je postaven na principu **inventarizace sítě** a funkcionalit **IPS**. Představuje komplexní řešení pro ochranu síťové infrastruktury před potenciálními hrozbami a neautorizovaným přístupem. Klíčovou složkou

tohoto systému je proces systematického sběru a analýzy informací o všech zařízeních připojených k síti. Tento proces je známý jako inventarizace sítě a jeho hlavním úkolem je poskytovat přesný a aktuální přehled o všech zařízeních, které jsou součástí síťové infrastruktury – zahrnující servery, routery, přepínače a další klíčové prvky.

Inventarizace sítě obnáší nejen identifikaci a dokumentaci všech síťových zařízení, ale také sledování jejich konfigurace, umístění, vlastností a vzájemného propojení. Specializované nástroje pro monitoring sítě jsou vybaveny schopností provádět pravidelná skenování sítě a sbírat důležité informace, jako jsou IP a MAC adresy, informace o výpočetních zdrojích, operačních systémech a další relevantní data, která charakterizují každé zařízení a jeho roli v síti.

Vedle inventarizace se systém aktivního zabezpečení opírá o schopnosti IPS, které zahrnují monitorování síťového provozu a analýzu datových toků v reálném čase s cílem rozpoznat a zastavit potenciální útoky dříve, než mohou způsobit škodu. IPS jsou schopny detekovat a blokovat nejen známé útočné vzory, ale mohou také identifikovat abnormální chování nebo anomálie v síťovém provozu, které mohou naznačovat přítomnost neautorizovaných nebo invazivních zařízení.

Cíle systému aktivního zabezpečení zahrnují:

- identifikaci všech validních aktivních zařízení v síti,
- uchování a správu informací o těchto zařízeních,
- detekci neautorizovaných nebo neznámých zařízení, která se pokusí připojit k síti, s možností jejich případně blokace.

V kombinaci s inventarizací sítě je IPS nezbytným nástrojem pro udržení bezpečnosti, umožňuje rychle reagovat na bezpečnostní incidenty.

Návrh implementace

Cílem systému pro aktivní zabezpečení průmyslových zařízení je sjednotit funkcionality **inventarizace sítě** a **IPS** (viz kapitola 2.4.3) do jednoho unikátního ochranného systému, který by měl chránit prostředí lokálních sítí, ke kterým bude připojen router v ESCON rozvaděči.

systému pro aktivní zabezpečení průmyslových zařízení je navržen jako Python program, jenž bude v reálném čase monitorovat síť a poskytovat v pravidelných intervalech informace o provozu sítě. Detailní informace jsou dostupné pro každé zařízení (IP, MAC, výrobce, název hosta v síti). Tyto informace jsou agregovány ze sítě, je vytvořen tzv. otisk sítě a tento otisk je předán administrátorům pomocí zvoleného způsobu reportování. Systém také umožňuje logování událostí pro pozdější analýzu (uchování dat přímo na kontrolním zařízení po dobu minimálně 1 týdne). Pokud by došlo k nežádoucí anomálii v síti, okamžitě je informována důvěryhodná

entita.

Typy reportů systému pro aktivní zabezpečení průmyslových zařízení jsou:

- email – reportovaný email s konkrétními informacemi o akcích systému nebo přítomnosti nežádoucího zařízení v sítí,
- URL – informace o konkrétních akcích systému nebo přítomnosti nežádoucího zařízení v sítí může být poslána také pomocí HTTPS požadavku na vzdálený server,
- logy – informace o konkrétních akcích systému nebo přítomnosti nežádoucího zařízení je implicitně vždy logována na zařízení, na němž běží popisovaný systém.

Cílem realizace není pouze pasivní monitoring síťového provozu a oznámení o vyskytnutích anomáliích, ale i aktivní ochrana. Při výskytu útočníka v sítí může program:

- deaktivovat DHCP (v případě, že je aktivní),
- blokovat nevalidní IP adresy,
- zablokovat nevalidní MAC adresy,
- uzavření fyzických portů,
- blokace aplikačních portů.

Navrhované bezpečnostní techniky jsou svou povahou reaktivní. Zatímco ideálním stavem je předejít jakémukoli bezpečnostnímu incidentu před jeho vznikem, realita je taková, že mnoho hrozeb je identifikováno až ve chvíli, kdy k nim dojde. Aktivace zmíněných bezpečnostních opatření až po detekci útočníka pomáhá minimalizovat škody a zabraňuje dalšímu šíření útoku v síti.

Útočník je detekován pomocí systému, který kombinuje funkcionality síťového monitorování a systému IPS. Tento systém porovnává otisk sítě provedený při prvotní inventarizaci sítě, při níž označí legitimní zařízení, které jsou po ukončení inventarizace uloženy v databázi. Následně, pokud se v síti objeví útočník, systém porovná nové zařízení s uloženými legitimními zařízeními, a pokud nenajde shodu, začne systém reagovat pomocí zmíněných metod.

Přínosem navrhovaných metod je, že například pomocí deaktivace DHCP se zabrání automatickému přidělování IP adres zařízením. To může omezit schopnost útočníka připojit svá zařízení k síti. Další silnou obrannou možností je blokování IP adres, MAC adres a portů ve firewallu na přístupovém prvku routeru RUTX11. Tyto akce zamezí komunikaci útočníka s ostatními částmi sítě nebo s kontrolními servery. Uzavřením fyzických portů je útočníkovi znepřístupněna libovolná komunikace se sítí. Blokování aplikačních portů může zastavit určité typy útoků, které vyžadují konkrétní síťové služby, a blokování IP a MAC adres může izolovat kompromitovaná zařízení nebo známé zdroje útoků.

Zavedení a aktivace těchto bezpečnostních opatření během incidentu umožňuje rychle reagovat na hrozby a minimalizovat potenciální škody.

4.2.2 Návrh zabezpečení síťové komunikace pomocí VPN

Tato práce si klade za cíl zajistit důvěrnost a integritu dat mezi zařízeními v rozvaděči ESCON, k tomu účelu bude sloužit VPN protokol. K tomuto účelu byly vybráni dva kandidáti. Protokolová sada IPsec, jenž je již v současnosti využívaná ze strany zadavatele práce, a WireGuard, jenž je novějším protokolem cílící na vytváření moderních rychlých a bezpečných VPN tunelů.

Při porovnávání těchto dvou technologií je možné zjisti, že WireGuard, jenž je integrován přímo do jádra Linuxu od verze 5.6, reprezentuje pokročilý přístup k vytváření VPN tunelů využívající nejnovější a nejbezpečnější šifrovací algoritmy. IPSec oproti tomu byl implementovaný již v kernelu verze 2.4 z počátku 21. století, ačkoliv byl aktualizován, je stále založen na starším kódu. WireGuard se odlišuje vysokou úrovní bezpečnosti, což je esenciální pro výběr technologie VPN tunelů. Díky moderním kryptografickým postupům je WireGuard považován za technologii nejnovější generace, v kontrastu s tradičními bezpečnostními prvky IPSec. Navíc WireGuard dosahuje lepší přenosové rychlosti oproti IPSec, což má signifikantní vliv na preferenci WireGuard pro aplikace požadující rychlý datový tok, který je nutný pro průmyslové sítě, kde rychlost datové komunikace a dostupnost dat hraje hlavní roli [1].

Minimalistická kódová základna WireGuard zároveň snižuje pravděpodobnost výskytu chyb, významně kontrastuje s rozsáhlým kódem IPSec, což naznačuje, že WireGuard je méně náchylný k chybám a snadněji udržovatelný. Ve vztahu ke konfiguraci a designu je WireGuard charakterizován jako jednoduchý, zatímco IPSec je konfiguračně významně složitější při porovnání konfiguračních souboru IPsec a WireGuard. To podporuje argumentaci ve prospěch využití WireGuard v moderních aplikacích, kde je kladen důraz na jednoduchost, přímou konfiguraci a snadnou údržbu vytvořených VPN tunelů [1].

Základem pro analýzu protokolů IPsec a WireGuard byla práce *Pudelko et al. (2020) Performance Analysis of VPN Gateways*, která podrobně zkoumala různé architektury softwarových VPN brán a jejich vliv na výkon. Klíčovým zjištěním práce bylo, že WireGuard představuje nejslibnější softwarovou implementaci VPN z architektonického hlediska. V implementaci WireGuard dosáhl nejvyšší rychlosti ze všech hodnocených VPN implementací s 6.2 miliony paketů za sekundu (Mpps) a 40 Gbit/s [45].

Z výše uvedených důvodů bude v rámci lokální sítě rozvaděče upřednostněna implementace WireGuard tunelu. Avšak z rozvaděče bude vytvořen IPsec tunel na vzdálený server, aby se zachovala integrita se současnými řešeními společnosti EASYCON Solution s.r.o.

4.2.3 Hardening zařízení Unipi S107 Patron

Za účelem naximálního zvýšení bezpečnosti byly na zařízení UniPi S107 Patron provedeny následující hardeningové kroky:

1. Aktualizace softwaru: Na zařízení byly nainstalovány nejnovější verze operačního systému a softwaru, aby bylo zajištěno, že obsahují aktuální bezpečnostní opravy a vylepšení.
2. Konfigurace firewallu: Byla nastavena pravidla firewallu s cílem omezit přístup k síťovým portům pouze na důvěryhodnou síťovou komunikaci jako MQTT, Modbus TCP a SSH.
3. Šifrování komunikace: Byly zřízeny VPN tunely IPsec a WireGuard.
4. Zabezpečení přihlašovacích údajů: Byly nastaveny silné hesla pro všechny uživatelské účty, což bylo provedeno pomocí zásad hesel doporučených Národním úřadem pro kybernetickou a informační bezpečnost, který jenž v dokumentu *Minimální bezpečnostní standard v1.2*[1], který je podpůrným materiálem pro subjekty, které nespadají pod zákon o kybernetické bezpečnosti definuje definuje heslo o minimální délce 17 znaků nebo více využívající malá a velká písmena, číslice a minimálně 1 speciální znak.
5. Monitoring a audit: Byla zavedena řešení pro monitorování zařízení, včetně konfigurace a nasazení systémů pro detekci podezřelé aktivity a zaznamenávání bezpečnostních událostí, viz kapitola 6.1.

Tato opatření byla provedena s cílem minimalizovat povrch útoku a zvýšit schopnost zařízení odolávat potenciálním hrozbám, čímž byla posílena jeho bezpečnostní robustnost.

[1] `https://nukib.gov.cz/download/publikace/podpurne_materialy/minimalni-bezpecnostni-standard_v1.2.pdf`

5 Implementace komunikačních protokolů

V této části je detailně popsána implementace a testování čtyř klíčových komunikačních protokolů – Modbus RTU, Modbus TCP, MQTT a HTTPS. Každý z těchto protokolů hraje zásadní roli v zajištění spolehlivé výměny dat mezi průmyslovými zařízeními a jejich řídicími systémy. V následujících podkapitolách je také pro každý protokol podrobně sepsán průběh a výsledek testování, jenž je podložený tabulkami, grafy a ukázkami z reálné komunikace mezi dvěma zařízeními.

5.1 Implementace Modbus RTU

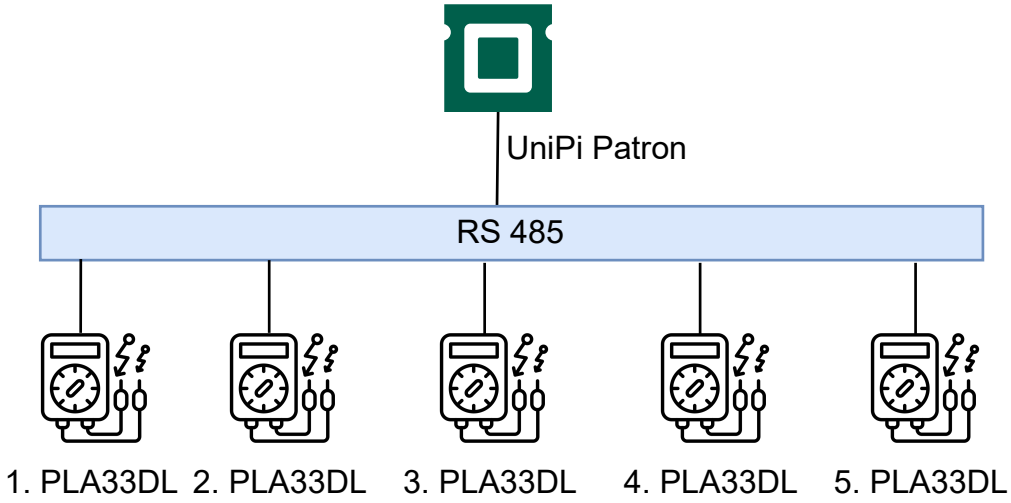
V rámci praktické části práce byl vytvořen skript využívající Modbus RTU protokol, implementovaný na zařízení UniPi S107 Patron. Účelem tohoto skriptu je realizace spolehlivého čtení dat z elektrických měřicích zařízení, specificky elektroměrů a analyzátorů elektrické energie, která jsou běžně využívána v průmyslových kontrolních systémech.

Pro vývoj tohoto skriptu byla vybrána knihovna pymodbus. Výběr knihovny pymodbus byl učiněn na základě důkladné analýzy různých dostupných knihoven, jak je zdokumentováno v kapitole 3.2. V této analýze byla zkoumána kritéria jako výkon, kompatibilita, komunitní podpora, udržitelnost a snadnost použití. Podrobné srovnání a vyhodnocení uvedených knihoven vedlo k závěru, že pymodbus je nejvhodnější volbou pro požadavky a cíle implementace.

Výsledný skript je postaven na robustních a ověřených základech, zajišťujících efektivní a bezproblémovou komunikaci s cílovými elektroměry/analyzátory, což umožňuje získávat přesná a spolehlivá data potřebná pro energetický management a další aplikace.

5.1.1 Zapojení analyzátoru PLA33DL pro testování s UniPi

Napájecí svorky jsou označeny čísly 11 (L) a 12 (N). Po připojení napětí na tyto svorky je možné zařízení zapnou. Zařízení je napájeno 230V AC. Pro připojení UniPi na sériovou sběrnici využijeme svorky 13 (A, +, nebo RxTx+) a 14 (B, -, nebo RxTx-). Následně je možné se zařízením komunikovat. Pro měření napětí je možné použít svorky 1 – 3 (L) a 4 (N), teda je možné měřit až tři fáze. V testovacím prostředí bylo zapojeno 5 zařízení na jednou sériovou sběrnici. Následně k ní bylo připojeno i zařízení UniPi pro čtení dat, viz obr.5.1.



Obr. 5.1: Znázornění zapojení řídicí jednotky UniPi ke sběrnici RS485 s elektroměry

5.1.2 Testování Modbus RTU komunikace s PLA33DL

Pro otestování byl vytvořen Modbus RTU skript na řídicí jednotce UniPi. Poté byly vykonány testy, při kterých byly registry opakovaně čteny s různými parametry. Výsledkem testů je určení nastavení, dle kterých bude zařízení číst co nejvíce registrů v sekundové periodě. Testy byly provedeny následovně:

1. Nastavení parametrů
2. Nastavení počtů registrů
3. Nastavení časového limitu (timeout) čtení a přenosové rychlosti dat
4. Spuštění testu
5. Zapsání výsledků

Při každém testu došlo k 1000 opakováním cyklu. Jeden cyklus zabezpečil čtení všech definovaných registrů z definovaných zařízení. Pro test se tedy využilo 5 zařízení PLA33DL a testovali se tyto parametry:

- Timeout s hodnotami:
  - 0,1s
  - 0,075s
  - 0,05s
- Přenosová rychlost s hodnotami:
  - 9600 Bd
  - 19200 Bd

Definovávané čtené registry, včetně jejich adres, délky a typu hodnot jsou uvedeny v tab. 5.1. Následně poté tab. 5.2 představuje soubor naměřených dat, která ukazují

Název registru	Adresa	Délka	Typ hodnoty
Napětí svorky 1	30040	2	Float
Napětí svorky 2	30042	2	Float
Napětí svorky 3	30044	2	Float
Činný výkon fáze 1	30024	2	Float
Činný výkon fáze 2	30026	2	Float
Činný výkon fáze 3	30028	2	Float

Tab. 5.1: Popis čtených Modbus registrů při testování komunikace

vliv časového limitu a počtu registrů na celkový čas a čas jednoho cyklu při přenosové rychlosti 9600 Bd. Tabulka je rozdělena do čtyř sloupců:

- Timeout [s] – Hodnota časového limitu v sekundách, se třemi různými nastaveními – 0,1 s, 0,075 s a 0,05 s.
- Počet registrů – Počet registrů, které jsou čteny během operace, s hodnotami od 3 do 6.
- Celkový čas [s] – Celkový čas potřebný k vykonání operací v sekundách.
- Čas jednoho cyklu [s] – Průměrný čas jednoho cyklu v sekundách, což odpovídá celkovému času dělenému počtem cyklů (nebo operací).

Timeout [s]	Počet registrů	Celkový čas [s]	Čas jednoho cyklu [s]
0,1	6	1651	1,651
0,1	5	1418	1,418
0,1	4	1128	1,128
0,1	3	827,8	0,8278
0,075	6	1619,4	1,6194
0,075	5	1349,8	1,3498
0,075	4	1084,4	1,0844
0,075	3	797	0,797
0,05	6	1510,5	1,5105
0,05	5	1264,5	1,2645
0,05	4	1010,1	1,0101
0,05	3	764,7	0,7647

Tab. 5.2: Popis výsledků pro přenosovou rychlost 9600 Bd

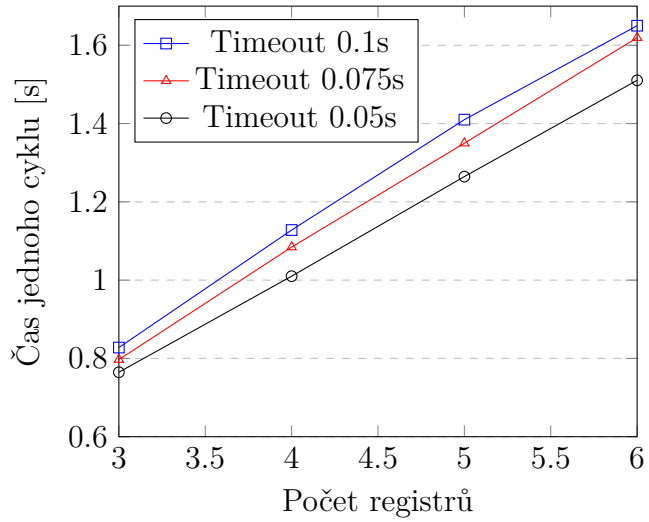
Graf na obr. 5.2 vycházející z tab. 5.2 ilustruje závislost času jednoho cyklu na počtu čtených registrů při různých nastaveních timeoutů, a to při přenosové rychlosti 9600 Bd. Graf je zobrazen s vertikální osou Y, která ukazuje čas jednoho

cyklu v sekundách (od 0,6 do 1,7 s) a horizontální osou X, která ukazuje počet registrů od 3 do 6.

Data jsou reprezentována třemi různými sériemi:

- Modrá křivka se čtverečky odpovídá časového limitu 0,1 sekundy. Zde je vidět, že čas jednoho cyklu roste s počtem čtených registrů od 0,8278 s pro 3 registry až po 1,65 s pro 6 registrů.
- Červená křivka s trojúhelníky zobrazuje časový limit 0,075 sekundy. Tato série ukazuje mírně nižší časy než modrá křivka, s hodnotami od 0,797 s pro 3 registry do 1,6194 s pro 6 registrů.
- Černá křivka s kruhy představuje časový limit 0,05 sekundy, což jsou nejnižší zaznamenané časy, od 0,7647 s pro 3 registry do 1,5105 s pro 6 registrů.

Graf ukazuje, že kratší časové limity vedou k rychlejším cyklům a že čas cyklu se zvyšuje s počtem čtených registrů. Dále je z grafu zřejmé, že čas jednoho cyklu má tendenci růst lineárně s nárůstem počtu registrů, bez ohledu na nastavený časový limit.



Obr. 5.2: Závislost času jednoho cyklu na počtu registrů pro různé časové limity při přenosové rychlosti 9600 Bd.

Tabulka 5.3 prezentuje výsledky týkající se přenosové rychlosti 19200 Bd v obdobné struktuře jako u tabulky 5.2. Následně graf na obr. 5.3 vycházející z tab. 5.3 znázorňuje závislost času jednoho cyklu na počtu čtených registrů pro různé nastavené časové limity. Osu X grafu tvoří počet registrů, které se pohybují od 3 do 6, zatímco osa Y reprezentuje čas jednoho cyklu v sekundách, a to v rozmezí od 0,5 s do 1,5 s.

Ve grafu jsou zobrazeny tři různé série dat:

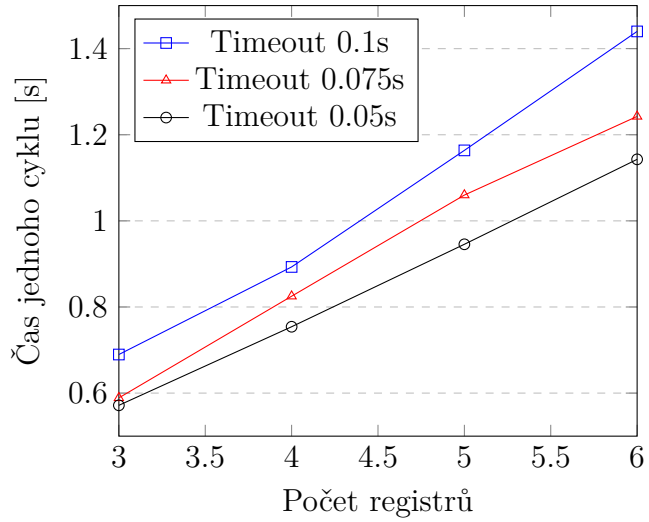
- Modrá křivka s čtverečkovými značkami představuje časový limit 0,1 sekundy.

Timeout [s]	Počet registrů	Celkový čas [s]	Čas jednoho cyklu [s]
0,1	6	1449	1,449
0,1	5	1163,7	1,1637
0,1	4	893	0,893
0,1	3	689,6	0,6896
0,075	6	1242,6	1,2426
0,075	5	1059,9	1,0599
0,075	4	825,	0,825
0,075	3	588,9	0,5889
0,05	6	1142,9	1,1429
0,05	5	945,6	0,9456
0,05	4	754,1	0,7541
0,05	3	571,6	0,5716

Tab. 5.3: Popis výsledků pro přenosovou rychlost 19200 Bd

- Červená křivka s trojúhelníkovými značkami představuje časový limit 0,075 sekundy.
- Černá křivka s kruhovými značkami představuje časový limit 0,05 sekundy.

Z grafu je patrné, že s rostoucím počtem registrů se pro všechny tři časové limity zvyšuje čas jednoho cyklu. Nejnižší hodnoty časů cyklu jsou dosaženy při nejnižším časového limitu 0,05 sekundy, což naznačuje, že kratší časové limity mohou efektivně zkracovat dobu trvání cyklu. Celý graf slouží k ilustraci vlivu počtu registrů a nastaveného časového limitu na rychlost zpracování dat při přenosové rychlosti 19200 Bd.



Obr. 5.3: Závislost času jednoho cyklu na počtu registrů pro různé časové limity pro přenosovou rychlost 19200 Bd.

5.1.3 Vyhodnocení řešení implementace Modbus RTU komunikace

Pokud mají zařízení (elektroměry, analyzátory) možnost komunikace s rychlostí **19200 Bd**, doporučuje se tuto rychlost využít. V takovém případě je Unipi schopno do 1 sekundy přečíst z každého z pěti připojených zařízení až 5 registrů. To odpovídá rychlosti čtení **25 registrů za sekundu (25 reg/s)**, jak je uvedeno v tab. 5.3, 3.řádek zdola.

Pokud mají zařízení pouze možnost komunikace s rychlostí **9600 Bd**, lze z pěti zařízení přečíst do jedné sekundy až 4 registry z každého, což představuje rychlost čtení přibližně **20 registrů za sekundu (±20 reg/s)**. V tomto případě však hrozí riziko, že nebude dodržena požadovaná sekundová perioda. Proto se doporučuje nastavit časový limit čtení na 0,1 s, aby se minimalizovalo riziko chybného čtení. Tímto způsobem lze číst pouze 3 registry z každého zařízení, což sníží rychlost čtení na **15 registrů za sekundu (15 reg/s)**, jak je dokumentováno v tab. 5.2, 4. řádek shora.

5.2 Implementace Modbus TCP

Byl vytvořen Modbus TCP skript na zařízení Unipi pro čtení dat z PLC, používající knihovnu pymodbus. Tato knihovna byla zvolena díky analýze jejích funkcí a kompatibility s Modbus TCP protokolem, která byla provedena v kapitole 3.2. Skript umožňuje efektivní sběr dat a je navržen tak, aby zajistil spolehlivou komunikaci mezi zařízením UniPi a připojenými PLC, což významně zlepšuje sběr a řízení energetických zařízení.

5.2.1 Implementace

K implementaci Modbus TCP serveru, který by splňoval požadavky zadání této práce byl vybrán programovací jazyk Python a knihovna pymodbus.

Klíčové požadavky požadavky, které bude nutné ověřit při testování implementace (viz kapitola 5.2.2):

- schopnost obsluhovat až 4 klienty simultánně,
- zpracování minimálně 20 veličin do 1 sekundy — Modbus TCP Server musí být schopen rychle reagovat na požadavky klientů a aktualizovat/registrovat hodnoty efektivně,
- maximální ztrátovost dat 0,5 % za 24 hodin — Modbus TCP Server musí být vysoce dostupný a spolehlivý.

Implementace zahrnuje asynchronní TCP server postavený na knihovně pymodbus, který zprostředkovává komunikaci s Modbus klienty a zároveň spravuje data uchovávaná v lokální databázi SQLite. Databáze je inicializována funkcí, která vytvoří tabulku pro uchování registračních hodnot společně s časovým razítkem každého záznamu. Asynchronní funkce pro aktualizaci dat periodicky upravuje hodnoty v registrech a zapisuje změny do databáze. Tato funkce demonstruje možnost sledování a analýzy změn registru v reálném čase. Systém také obsahuje obsluhu klientů, kde se zaznamenávají připojení a odpojení klientů, což je kritické pro monitoring stavu síťové komunikace. Tato implementace navržena tak, aby efektivně simulovala průmyslové provozní podmínky v laboratorním měřítku, což umožňuje praktické testování a vývoj průmyslových komunikačních protokolů a databázových operací v kontrolovaném prostředí.

5.2.2 Testování komunikace

V této sekci se zaměříme na experimentální ověření schopností Modbus TCP serveru. Specifikace požadavků, které jsou detailněji popsané v seznamu 5.2.1, klade na server výkonnostní a spolehlivostní nároky.

Server musí být schopen efektivně a bez ztráty kvality služeb obsloužit až čtyři klienty současně. Tato vlastnost, testovaná v podsekci 5.2.2 – Testování splnění požadovaného počtu klientů, je kritická pro aplikace, kde je vyžadována vysoká míra komunikace s více zařízeními v reálném čase.

Jedním z dalších klíčových kritérií je schopnost serveru zpracovat minimálně 20 veličin do jedné sekundy pro každého klienta. Tato rychlost zpracování, testovaná v podsekci 5.2.2 – Testování splnění požadované doby zpracování dat, umožňuje serveru rychle reagovat na změny a aktualizovat nebo registrovat hodnoty s minimálním zpožděním, což je nezbytné pro udržení plynulosti procesů a minimalizaci reakční doby systému.

V neposlední řadě je třeba, aby server vykazoval maximální ztrátovost dat ve výši 0,5 % za 24 hodin. Vysoká dostupnost a spolehlivost jsou zásadní pro udržení kontinuálního a bezpečného provozu v průmyslových aplikacích, kde každé procento ztráty dat může mít vážné důsledky. Testování tohoto požadavku je popsáno v podsekci 5.2.2 – Testování splnění požadované ztrátovosti dat.

Testování splnění požadovaného počtu klientů

V rámci experimentálního ověření schopnosti Modbus TCP serveru zvládat simultánní spojení bylo realizováno testování s deseti klienty. Tyto klienti se k serveru připojovaly a opětovně se odpojovaly v definovaných intervalech. Logovací data, prezentovaná na obr. B.1, poskytují důkaz o dynamickém charakteru síťové komunikace, která je zvláště zřetelná při pozorování frekvence připojování a odpojování klientů. Tato pravidelnost v komunikaci odpovídá očekávanému modelu výkonu pro síťová zařízení nastavená na periodický přenos dat.

Detailní analýza zaznamenaných dat umožňuje identifikaci okamžiků, kdy byl server současně v interakci s více klienty, což svědčí o jeho schopnosti efektivně spravovat několik paralelních komunikačních relací. V časovém okamžiku 20:22:54 jsou čtyři Modbus TCP klienti připojeni k serveru souběžně, čímž byl splněn stanovený požadavek na minimální počet klientů, které Modbus TCP server musí být schopen obsloužit simultánně. Tato skutečnost potvrzuje, že server splňuje kritéria pro paralelní připojitelnost klientů, což je podstatné pro potvrzení jeho operativní spolehlivosti dle předepsaných požadavků.

Testování splnění požadované doby zpracování dat

Dalším klíčovým požadavkem pro Modbus TCP komunikaci je zpracování minimálně 20 veličin do jedné sekundy. Tento požadavek byl ověřen pomocí skriptu, který postupně prováděl čtení a zápis dat z Modbus TCP serveru (zařízení Unipi S107 Patron).

Test byl automatizovaný (Python skript na routeru RUTX11) a zahrnoval scénáře s 1, 3, 5, 10, 15, 20, 25 a 30 klienty. Jednotlivá klientská připojení byla realizována pomocí vláken procesu vytvořeného testovacího Python skriptu, kde každý klient měl jedno vlákno, v němž běžel klientský program na čtení, zápis nebo obojího najednou v závislosti na právě prováděném typu testu. V každém vlákně se za běhu provádělo měření dob a počítání výsledků pro jednotlivé operace každého klienta. Po ukončení všech vláken a předání výsledku hlavnímu vláknu procesu, hlavní vlákno spočítalo celkové výsledky.

Během testu každý klient žádal server o zpracování 20 registrů v 50 iteracích, mezi kterými byla vložena pauza délky jedné sekundy. To znamená, že postupně s každým dalším testem stoupalo zatížení serveru. Ten byl postupně zatížen 1 000, 3 000, 5 000, 10 000, 15 000, 20 000, 25 000 a 30 000 operacemi. V poslední fázi testu, kdy probíhalo současně zápis a čtení, byl počet operací dvojnásobný, čímž server čelil v poslední testu při počtu 30 klientů celkově 60 000 operacím během přibližně jedné minuty, což odpovídá průměru 1 000 operací za sekundu. Detailní výsledky testu jsou podrobně popsány v tab. 5.4, 5.5 a 5.6, kde jsou shrnuta data měření doby čtení a zápisu z Modbus TCP serveru s cílem hodnotit výkonnost a spolehlivost serveru za různých provozních podmínek. Data postupně ukazují, jak se mění mediánová, průměrná, minimální a maximální doba čtení a zápisu v závislosti na počtu klientů připojených k serveru.

V tab. 5.4 jsou prezentovány výsledky měření **doby čtení** z Modbus TCP serveru, z nichž je zřejmé, že s rostoucím počtem klientů se prodlužuje jak průměrná, tak mediánová doba odezvy. Tento trend naznačuje, že s rostoucím počtem klientů dochází k většímu zatížení serveru, což má za následek delší doby čekání na dokončení čtení. Metriky zahrnují mediánovou, průměrnou, minimální a maximální dobu čtení vyjádřenou v sekundách. Pro jednoho klienta je jak mediánová, tak průměrná doba čtení 0,1105 sekundy, přičemž minimální doba činí 0,1 sekundy a maximální 0,1216 sekundy. S nárůstem počtu klientů roste variabilita doby čtení; například u tří klientů je medián 0,1313 sekundy a průměr 0,1277 sekundy, s minimem 0,0989 sekundy a maximem 0,1726 sekundy. Tento vzrůstající trend doby čtení pokračuje až k 30 klientům, kde medián dosahuje hodnoty 0,9065 sekundy a průměrná doba 0,8047 sekundy, přičemž maximální doba se zvyšuje na 1,8356 sekundy. Tato data ilustrují, jak zvýšení počtu klientů ovlivňuje dobu čtení dat z serveru.

Druhá tabulka, odkazovaná jako tab. 5.5, obsahuje data týkající se měření **doby zápisu** na Modbus TCP server. I zde lze pozorovat trend rostoucích dob čekání s narůstajícím počtem klientů, což ukazuje na omezení serveru ve schopnosti efektivně zvládat větší počet současných zápisových požadavků. Výsledky měření ilustrují nárůst jak mediánové, tak průměrné doby zápisu, což signalizuje zvýšené zatížení serveru a zpomalení reakce při vyšším počtu simultánně zapojených klientů. Kon-

Počet klientů	Mediánová doba	Průměrná doba	Minimální doba	Maximální doba
1	0,111	0,111	0,100	0,122
3	0,131	0,128	0,099	0,173
5	0,187	0,162	0,098	0,293
10	0,281	0,233	0,098	0,593
15	0,308	0,314	0,098	0,889
20	0,422	0,410	0,098	1,176
25	0,661	0,657	0,099	1,547
30	0,907	0,805	0,099	1,836

Tab. 5.4: Výsledky měření doby čtení z Modbus TCP serveru.

krétně, mediánová doba zápisu pro jednoho klienta je 0,1104 sekundy, zatímco pro 30 klientů vzrostla na 0,7982 sekundy. Podobným způsobem se průměrná doba zápisu zvýšila z 0,1104 sekundy pro jednoho klienta na 0,6948 sekundy pro 30 klientů.

Je také možné pozorovat, že minimální doby zápisu zůstávají relativně konstantní bez ohledu na počet klientů, což naznačuje stabilní základní latenci systému. Naproti tomu maximální doby zápisu narůstají, což ilustruje rostoucí variabilitu v odezvách systému za vyššího zatížení. Tento vzestupný trend maximálních dob, od 0,1262 sekundy pro jednoho klienta až po 1,457 sekundy pro 30 klientů, odhaluje limitace serveru v situacích špičkového zatížení. Tyto poznatky jsou klíčové pro pochopení kapacitních omezení serveru, které je nutné zohlednit při integraci této komunikace do průmyslového prostředí.

Počet klientů	Mediánová doba	Průměrná doba	Minimální doba	Maximální doba
1	0,110	0,110	0,100	0,126
3	0,114	0,115	0,098	0,172
5	0,132	0,126	0,097	0,234
10	0,200	0,172	0,097	0,434
15	0,219	0,265	0,097	0,689
20	0,321	0,286	0,097	0,945
25	0,559	0,558	0,098	1,228
30	0,798	0,695	0,098	1,457

Tab. 5.5: Výsledky měření doby zápisu z Modbus TCP serveru.

Ve třetí tab. 5.6 jsou prezentovány kombinované výsledky měření **doby čtení a zápisu**, které ukazují výrazné zvýšení doby odezvy při současném provozu obou

operací. Tato data reflektují nejvyšší zaznamenané doby odezvy, což poukazuje na významné zatížení serveru, který musí zvládat simultánní zpracování požadavků na čtení i zápis od většího počtu klientů.

Data ukazují, že jak mediánová, tak průměrná doba operací narůstá s počtem klientů, což signalizuje zvyšující se zatížení systému při simultánním zpracování většího počtu požadavků. Pro jednoho klienta jsou mediánová i průměrná doba shodné a činí 0,2126 sekundy. To indikuje stabilní latenci při nízkém zatížení systému. S narůstajícím počtem klientů však obě tyto metriky rostou – to je doprovázeno zpomalením serveru. Zvláště výrazný nárůst je patrný u 25 klientů, kde mediánová doba dosahuje hodnoty 2,0713 sekundy a průměrná 2,0701 sekundy.

Minimální doby zůstávají relativně konstantní až do počtu 20 klientů. To naznačuje, že systém má schopnost zpracovat některé operace rychle, i při vyšším počtu klientů. Naopak maximální doby dramaticky rostou, kvůli výskytu extrémně dlouhých operací za vyšších zatížení. Tento trend je zřejmý, když maximální doba pro jednoho klienta je 0,2277 sekundy a pro 30 klientů vzrůstá na 3,2015 sekundy.

Tyto výsledky jsou klíčové pro pochopení dopadu simultánního zpracování požadavků na výkonnost Modbus TCP serveru. S rostoucím počtem klientů server zpomaluje, tak může ovlivnit celkovou spolehlivost a odezvu systému. Při návrhu a optimalizaci systémů by měly být tyto údaje zohledněny pro zajištění adekvátní škálovatelnosti a výkonnosti v reálných provozních podmínkách.

Počet klientů	Mediánová doba	Průměrná doba	Minimální doba	Maximální doba
1	0,213	0,213	0,196	0,228
3	0,214	0,211	0,195	0,256
5	0,271	0,247	0,195	0,410
10	0,454	0,360	0,195	0,853
15	0,468	0,590	0,195	1,513
20	0,880	0,858	0,195	1,947
25	2,071	2,070	0,880	2,590
30	2,643	2,391	0,880	3,202

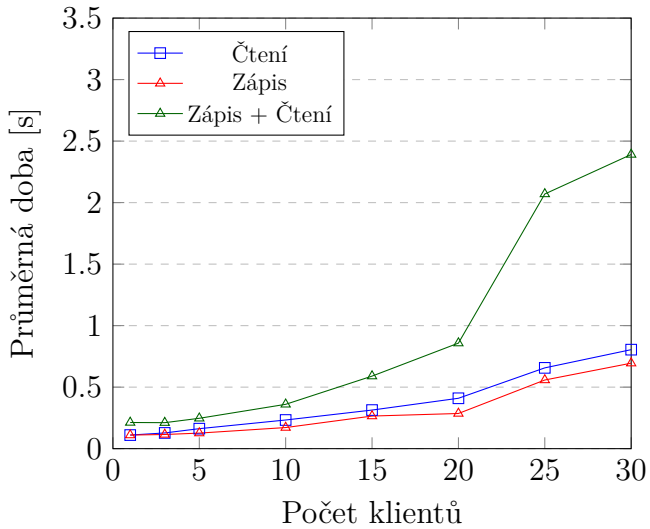
Tab. 5.6: Výsledky měření doby čtení a zápisu z Modbus TCP serveru.

Graf na obr. 5.4 znázorňuje průměrné doby operací na Modbus TCP serveru v závislosti na počtu klientů, přičemž jsou rozlišeny tři typy operací: čtení, zápis a kombinované čtení a zápis.

Z grafu je zřejmé, že s rostoucím počtem klientů systematicky narůstají průměrné doby všech typů operací. Průměrná doba čtení (modrá křivka) je konzistentně vyšší než průměrná doba zápisu (červená křivka) ve všech měřených bodech. To naznačuje, že čtení může být časově náročnější operace ve srovnání se zápisem na serveru. Výjimku tvoří začátek grafu, kde obě křivky startují s velmi podobnými hodnotami pro jednoho klienta. Křivka kombinovaných operací (zelená křivka) pak ukazuje vyšší průměrné doby, což reflektuje kumulativní efekt obou operací.

Průměrné doby zápisu a čtení se rozcházejí s rostoucím počtem klientů. To naznačuje rostoucí význam optimalizace čtecích operací při vyšším zatížení. Důležité je také poznamenat, že kombinované doby zápisu a čtení rostou nadproporcionálně, při čemž se nelineárně zvyšuje zatížení serveru při simultánním zpracování obou typů operací.

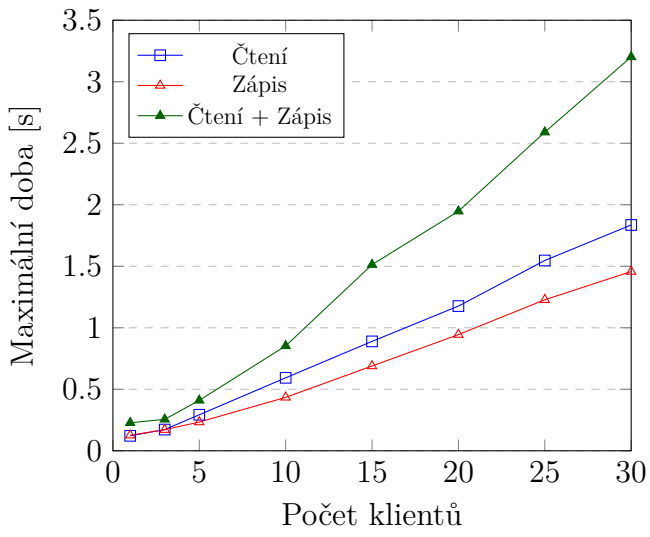
Uvedený graf efektivně demonstruje, jak zvýšení počtu klientů významně ovlivňuje doby operací na Modbus TCP serveru, které má přímý dopad na plánování kapacity a strategie škálování systémů. Tyto údaje jsou kritické pro návrh systémů, které musí být schopné udržet kritickou odezvu i při vyšším počtu klientů, což je nezbytné pro zajištění spolehlivosti a efektivity v průmyslových aplikacích založených na Modbus TCP.



Obr. 5.4: Průměrné doby operací v závislosti na počtu klientů

Graf 5.5 prezentuje analýzu vztahu mezi počtem klientů a maximálními doby reakce Modbus TCP serveru na operace čtení, zápisu a jejich kombinace. Tato data jsou klíčová pro porozumění zátěži serveru v závislosti na simultánní aktivitě klientů.

Modrá čára (Čtení) demonstruje, jak s rostoucím počtem klientů lineárně narůstají maximální doby reakce pro operace čtení. Tento trend reflektuje zvyšující se zatížení serveru a odpovídající prodloužení doby odezvy, což poukazuje na potenciální omezení škálovatelnosti systému při zpracování čtecích požadavků. Červená čára (Zápis) ukazuje, že maximální doby reakce pro zápisové operace rostou s počtem klientů, avšak mírnějším tempem než u čtení. Tento fenomén naznačuje možnou vyšší efektivitu zpracování zápisových požadavků serverem nebo nižší datovou náročnost těchto operací ve srovnání s čtením. Zelená čára (Čtení + Zápis) ilustruje maximální doby reakce pro kombinované operace. Výrazný nárůst maximálních dob s počtem klientů poukazuje na značnou náročnost souběžného zpracování čtecích a zápisových operací, jenž mohou vést k významnému snížení výkonu systému při vysokém počtu simultánních uživatelů. Avšak i při počtu 10 klientů byla **kombinace čtení a zápisu splněna do jedné sekundy pro všechny klienty**, čímž bylo dokázáno splnění podmínky minimálního počtu klientů (4 klienti), včetně požadavku na rychlost zpracování dat (20 registrů za sekundu) pro každého klienta.



Obr. 5.5: Maximální doby operací v závislosti na počtu klientů

Testování splnění požadované ztrátovosti dat

V rámci tohoto testu byl stanoven požadavek na maximální ztrátovost dat ve výši 0,5 % za 24 hodin. Test probíhal nepřetržitě po dobu 25 hodin, konkrétně od 22:00 dne 17. dubna 2024 do 23:00 dne 18. dubna 2024. Během této doby bylo každou sekundu z Modbus TCP serveru na Modbus TCP klienta přenášeno 20 registrů. Tento režim znamenal, že celkem bylo během testu přeneseno více než 1 800 000 hodnot. V obr. B.2 v příloze práce je možné vidět, jak probíhalo sekundové čtení dat z Modbus TCP serveru pro definované registry s proměnlivými hodnotami.

Důležité je zdůraznit, že během testu nedošlo k žádné ztrátě dat. Všechny přenesené hodnoty byly Modbus TCP klientem správně přijaty a zpracovány. Vysoká spolehlivost přenosu a absence ztrát dat naznačují, že systém je velmi robustní a efektivně zvládá požadavky stanovené pro tento test. Výsledky tak demonstrují, že konfigurace serveru a klienta, stejně jako síťová infrastruktura, jsou adekvátně dimenzovány pro zajištění vysoké úrovně datové integrity a dostupnosti v reálném čase.

5.3 Implementace MQTT a HTTPS komunikace

V této diplomové práci je implementován systém pro distribuci dat v reálném čase, který využívá MQTT protokol jako metodu komunikace pro odesílání dat na určený centrální bod – MQTT broker. Systém integruje MQTT a HTTPS komunikační protokoly s dynamickým sledováním stavu a odesíláním dat v závislosti na aktuálním stavu těchto způsobů komunikací. Klíčovým prvkem je MQTT vydavatel, umístěný na zařízení UniPi S107 Patron, který sbírá data z elektroměrů prostřednictvím Modbus RTU protokolu. Tato data jsou následně odesílána pomocí MQTT protokolu na MQTT brokera. Data z brokera poté odebírá MQTT odběratel, který dále určuje účel jejich dalšího využití.

5.3.1 Implementace vydavatele

V této části je popsán MQTT vydavatel, což je Python skript běžící na UniPi S107 Patron, který kontinuálně monitoruje připojené elektroměry. Data jsou sbírána v reálném čase a pravidelně odesílána přes MQTT protokol na nakonfigurovaný MQTT broker. Skript využívá knihovnu paho-mqtt pro publikování dat na specifické téma, které je přístupné pro odběratele dat. Tato metoda umožňuje efektivní a škálovatelné distribuování dat mezi různými částmi systému nebo externími aplikacemi.

V případě výpadku internetového připojení nebo selhání MQTT brokera systém automaticky detekuje tyto události a přepne na záložní komunikační režim HTTPS. K tomu slouží knihovna requests, která je používána pro odeslání nahromaděných dat na HTTPS server. Data jsou balíčkována a odesílána ve formátu JSON, což zajišťuje, že informace zůstávají dostupné a chráněné i během síťových poruch.

V rámci implementovaného systému poskytuje funkce ukládání dat do lokální databáze robustní způsob, jak zajistit integritu a dostupnost dat v případě výpadku internetového připojení. Data jsou v této fázi akumulována do interní paměti zařízení. Jakmile je internetové připojení obnoveno, systém automaticky zahájí proces odesílání nahromaděných dat zpětně přes HTTPS komunikační kanál.

Proces hromadění dat probíhá následovně, pokud je výpadek o délce:

- 0-48 hodin – čtení probíhá sekundově, data se ukládají sekundově,
- 48-196 hodin – čtení probíhá sekundově, data se ukládají každých 15 sekund,
- >196 hodin – čtení probíhá sekundově, data se ukládají minutově.

Tento proces je navržen tak, aby po dlouhou dobu nedošlo k zahlcení paměti zařízení Unipi S107 Patron při zvážení velikosti jednoho záznamu v lokální databázi SQLite3 se třemi sloupci, které mají specifické datové typy jako jsou:

- Časové razítko (TEXT) – formát ISO 8601 (YYYY-MM-DD HH:MM:SS), každé takové časové razítko bude mít 19 znaků. V SQLite každý znak v řetězci

zabírá 1 B. Takže 19 B a 1 B pro terminální nulový znak = 20 B [31].

- Identifikátor veličiny (TEXT) – Nechť je průměrná délka identifikátoru veličiny 20 znaků a 1 B pro terminální nulový znak = 21 B.
- Hodnota veličiny (REAL) – REAL (reálné číslo) vždy uloženo jako 8 B [55].

SQLite používá dodatečnou režii na záznam, včetně metadat pro typy dat a ukazatelů v rámci databázového souboru. Navíc každý záznam obsahuje tzv. hlavičku záznamu, která uchovává informace o velikosti a typu jednotlivých sloupců. Hlavička může být proměnlivě dlouhá v závislosti na počtu sloupců a jejich datových typech. Režie hlavičky může být přibližně 1 B na sloupec a dodatečné bajty na uchování velikosti dat ve sloupci [55]. Z uvedeného tedy plyne odhad celkové velikosti jednoho záznamu v databázi:

- časové razítko = 20 B,
- identifikátor veličiny = 21 B,
- hodnota veličiny = 8 B,
- režie = 5-10 B (v závislosti na implementaci SQLite a velikosti dat) [55].

Jeden záznam by tedy mohl mít velikost kolem 54 až 59 B. Toto číslo je pouze odhadem a může se lišit v závislosti na specifikacích a konfiguraci SQLite. Jednotka Unipi má 6,3 GB volného místa v paměti za běžného provozu, viz obr. 5.6. Velikost, kterou zahrnuje jedna veličina (jeden záznam v tabulce) je maximálně kolem 59 B. To znamená paměťové zatížení je 59 B x 20 čtených veličin za sekundu, což je rovno **1180 B/s**.

```
root@S107-sn1070:~# df -H
Filesystem      Size  Used Avail Use% Mounted on
/dev/mmcblk2p1  7.9G  1.4G  6.3G  18% /
devtmpfs        348M     0  348M   0% /dev
tmpfs           517M     0  517M   0% /dev/shm
tmpfs           517M   59M  459M  12% /run
tmpfs           5.3M   13k  5.3M   1% /run/lock
tmpfs           517M     0  517M   0% /sys/fs/cgroup
tmpfs           158M  173k  158M   1% /tmp
/dev/mmcblk2p1  7.9G  1.4G  6.3G  18% /var
tmpfs           104M     0  104M   0% /run/user/0
root@S107-sn1070:~#
```

Obr. 5.6: Zobrazení volné paměti jednotky Unipi S107 Patron

V následujících rovnicích jsou uvedené výpočty, jak dlouho by mohla být jednotka Unipi S107 Patron bez přístupu internetu a stále by dokázala efektivně sbírat data (bez problémů s nedostatkem paměti). Po ukončení výpadku by za tu dobu sesbíraná data i tak zvládla postupně odesílat na cílový HTTPS server. Implementace takového mechanismu je zásadní především v průmyslových aplikacích, kde je přesnost a úplnost datové historie nezbytná pro analýzu a rozhodování.

Maximální doba sekundové ukládání dat

$$\text{Celkový čas} = \frac{6\,300\,000\,000\,\text{B}}{1180\,\text{B/s}} = 5\,344\,068\,\text{s},$$
$$\text{Převod na hodiny} = \frac{5\,344\,068\,\text{s}}{3600\,\text{s}} = 1484,46\,\text{hod},$$
$$\text{Převod na dny} = \frac{1484,46\,\text{hod}}{24\,\text{hod}} \approx 62\,\text{dní}.$$

Maximální doba 15 sekundové ukládání dat

$$\text{Celkový čas} = \frac{6\,300\,000\,000\,\text{B}}{1180\,\text{B}/15\,\text{s}} = 80\,169\,492\,\text{s},$$
$$\text{Převod na hodiny} = \frac{80\,169\,492\,\text{s}}{3600\,\text{s}} = 22\,269,86\,\text{hod},$$
$$\text{Převod na dny} = \frac{22\,269,86\,\text{hod}}{24\,\text{hod}} \approx 927\,\text{dní}.$$

Maximální doba minutového ukládání dat

$$\text{Celkový čas} = \frac{6\,300\,000\,000\,\text{B}}{1180\,\text{B}/60\,\text{s}} = 320\,677\,966\,\text{s},$$
$$\text{Převod na hodiny} = \frac{320\,677\,966\,\text{s}}{3600\,\text{s}} = 89\,077,21\,\text{hod},$$
$$\text{Převod na dny} = \frac{89\,077,21\,\text{hod}}{24\,\text{hod}} \approx 3708\,\text{dní}.$$

5.3.2 Implementace odběratele

Tato práce byla také cílena na implementaci MQTT odběratele, který zároveň funguje jako HTTPS server s využitím Flask frameworku. Systém je navržen tak, aby efektivně zpracovával příchozí zprávy z MQTT brokera a HTTPS požadavky. MQTT klient je konfigurován s adresou brokera a odebírá data na specifikovaném tématu. Klient se připojuje k brokeru na dané adrese a odebírá zprávy na určeném tématu, které je definováno pro distribuci dat.

Paralelně s MQTT klientem běží Flask aplikace v samostatném vlákně, která poskytuje HTTPS endpoint pro přijímaná data. Tento endpoint umožňuje přijímání dat přes HTTPS POST požadavky ve formátu JSON. Tento mechanismus poskytuje alternativní cestu pro příjem dat od systémů, které nemohou využívat MQTT, čímž se zvyšuje flexibilita a integrace systému s různými technologiemi.

Flask server a MQTT klient operují nezávisle díky multithreadingu, což zajišťuje, že komunikační rozhraní mohou fungovat současně bez vzájemného blokování. MQTT callback funkce monitorují a reagují na změny v připojení MQTT klienta.

Tyto funkce zaznamenávají informace o úspěšných připojeních, selháních a nečekaných odpojeních a automaticky se pokoušejí o znovupřipojení, pokud dojde k výpadku.

Na obrázku C.1 je možné vidět implementaci konzolového výstup MQTT odběratele, který přijímá a zobrazuje proud datových zpráv v reálném čase. Každá zpráva obsahuje identifikátor veličiny a hodnotou. Barevné zvýraznění různých částí zprávy napomáhá uživatelské přehlednosti a rychlé orientaci v přijatých datech.

5.3.3 Testování komunikace

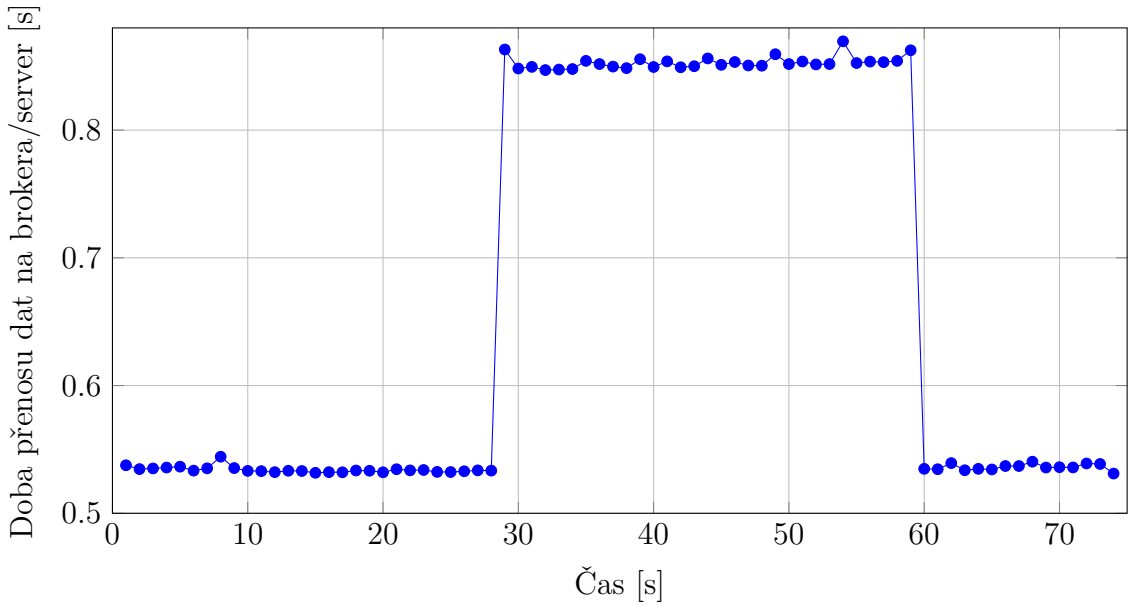
V kontextu průmyslového nasazení byla efektivita protokolu MQTT ověřena skrze jeho schopnost splnit stanovená kritéria pro zpracování a přenos dat. Je doloženo, že protokol je schopen přenášet až dvacet rozdílných veličin za sekundu, což potvrzují časová razítka, zobrazená na obr. C.1, kde je použita zelená barva pro jejich zvýraznění. Systém MQTT navíc efektivně zvládá synchronizaci mezi alespoň čtyřmi klienty, což je zajištěno díky využití MQTT brokera. Během dvaceti-čtyřhodinového testu bylo úspěšně přeneseno více než 1,7 milionu zpráv bez zjištěné ztráty, což svědčí o významné spolehlivosti a datové přesnosti. Kontinuita komunikace byla zajištěna prostřednictvím záložního přenosu dat přes HTTPS, což přispívá k odolnosti systému v případě výpadku primárního komunikačního kanálu.

Jak je ilustrováno na obrázcích C.2 a C.3 v příloze, systém prokázal schopnost plynule přecházet mezi MQTT a HTTPS protokoly. V momentě výpadku MQTT brokera přechází systém hladce na odesílání dat přes HTTPS, jak ukazuje obr. C.2, čímž je zajištěna nepřetržitá dostupnost dat. Při obnovení MQTT komunikace systém automaticky přepne zpět na primární komunikační kanál, jak je vidět na obr. C.3, což je přednostní metoda pro rychlý a efektivní přenos dat.

V případě simultánního výpadku obou komunikačních cest, MQTT i HTTPS, jsou data dočasně uchovávána v paměti zařízení. Jakmile se připojení obnoví, jsou data hromadně odeslána, což ilustruje obr. C.4, zobrazující log přijatých dat po krátkodobém výpadku trvajícím kolem 12 sekund. Tento mechanismus zajišťuje, že i při přechodných komunikačních poruchách je integrita a dostupnost dat zachována.

V grafu na obr. 5.7 je zobrazena doba zpoždění přenosu dat pomocí MQTT a HTTPS. Rychlejší MQTT, které bylo odesíláno v čase 0-28 s, je po simulované nedostupnosti klienta nahrazeno během jedné sekundy záložní HTTPS komunikací, přičemž jsou okamžitě přenášena data pomocí HTTPS. To přináší vyšší latenci, jak je vidět na vyšších hodnotách doby přenosu v čase 29-59 s. Poté se obnovilo spojení s MQTT brokerem a data se začaly opětovně přenášet pomocí MQTT protokolu.

Tento kontrast zdůrazňuje výhodu rychlé komunikace a dostupnosti dat při využití protokolu MQTT v průmyslových aplikacích, avšak i HTTPS, jako záložní komunikační zdroj, splňuje požadavek přenosu dat na cílený server v požadovaném intervalu do 1 sekundy.



Obr. 5.7: Graf zpoždění při přenosu dat pomocí MQTT/HTTPS

5.3.4 Zajištění bezpečnost MQTT

V rámci informační bezpečnostní architektury byla implementována nešifrovaná komunikace MQTT v prostředí zabezpečeném kompletně pomocí VPN tunelů IPsec a WireGuard namísto využití protokolu MQTT šifrovaného skrze TLS. Toto rozhodnutí bylo učiněno s ohledem na několik faktorů, které jsou rozhodující pro zajištění bezpečnosti informačních systémů. VPN tunel, vytvářející šifrovanou komunikační cestu mezi klientem a serverem, poskytuje kompletní šifrování datového toku, čímž zajistí, že veškeré informace, včetně zpráv MQTT, jsou ochráněny robustním šifrováním na úrovni síťové vrstvy. Toto řešení přináší výhodu univerzálního šifrování, které je nezávislé na aplikaci a je schopné zabezpečit širokou škálu dalších datových toků bez nutnosti specifického nastavení na aplikační úrovni.

Dalším důvodem pro tento výběr byla otázka výkonu a efektivnosti. Nešifrovaný MQTT protokol eliminuje potřebu dalšího zpracování dat spojeného s TLS šifrováním a dešifrováním na úrovni aplikace, čímž redukuje zátěž systémů a snižuje latenci komunikace. Také z pohledu správy systému je centralizované šifrování poskytované VPN preferováno, jelikož zjednodušuje administraci bezpečnostních politik a klíčů

tím, že se šifrování koncentruje do jediného bodu. Tato centralizace vede ke snížení složitosti konfigurace a zajišťuje, že bezpečnostní postupy jsou jednotné a konzistentní napříč celou organizací. Závěrem lze konstatovat, že i když je volba bezpečnostních opatření vždy specifická pro dané aplikační prostředí a vychází z detailního posouzení rizik, v případě našeho systému je použití VPN s nešifrovaným MQTT vyhodnoceno jako dostatečné pro zajištění požadovaného stupně bezpečnosti.

Výše uvedené potvrzuje i práce *ALKHAFAJEE et al. (2021) Security and Performance Analysis of MQTT Protocol with TLS in IoT Networks*, publikovaná na 4.mezinárodní irácké konferenci o inženýrských technologiích a jejich aplikacích, která potvrzuje, že při rozhodování o implementaci komunikačního protokolu pro průmyslová zařízení je důležité zvážit rovnováhu mezi bezpečností a výkonností. Zatímco použití TLS s MQTT zvyšuje úroveň bezpečnosti přenášených dat, má to za následek významné zvýšení režie systému. Toto zvýšení režie zahrnuje větší čas potřebný k navázání spojení, vyšší množství přenášených dat a zvýšenou spotřebu energie, což je problematické pro zařízení s omezenými výpočetními a energetickými zdroji [2].

Nešifrovaná verze MQTT provozovaná v rámci VPN (Virtual Private Network) tunelu nabízí celkové šifrování všech dat přenášených mezi klientem a serverem. VPN tunel efektivně izoluje komunikaci od vnějších hrozeb a poskytuje bezpečné prostředí i bez aplikace TLS na každé zprávě individuálně. Kromě toho, VPN je efektivnější ve využití výpočetních zdrojů z důvodu:

- operování na nižší vrstvě ISO/OSI modelu (nižší režie),
- rychlejší handshake bez nutnosti spravování certifikátů,
- možnosti využití proudových šifer (WireGuard).

Uvedené body jsou zvláště důležité pro průmyslová zařízení, která jsou často navržena pro dlouhodobý provoz s omezenými výpočetními kapacitami a nízkou spotřebou energie. Technologie VPN jim v tomto případě umožňuje dlouhodobě a rychle komunikovat [35].

Pro průmyslová zařízení, která potřebují rychlou a efektivní komunikaci bez zbytečného zpoždění, je nešifrovaný MQTT přes VPN vhodnější volbou než MQTT s TLS. Pokud je dodržena bezpečnost sítě a průmyslová zařízení jsou správně konfigurována a udržována v rámci bezpečného VPN je toto řešení poskytnout adekvátní ochranu bez zbytečného negativního dopadu na výkon systému [2].

6 Implementace zabezpečení síťové komunikace průmyslových zařízení

V rámci praktické části této práce byl vyvíjen a implementován systém pro monitorování a zabezpečení průmyslových sítí. Systém je založen na softwaru vyvinutém v programovacím jazyce Python a využívá pokročilých metod síťového skenování a monitorování. Jeho primárním cílem je detekce a následná reakce na potenciální bezpečnostní hrozby, které by mohly ovlivnit integritu a dostupnost průmyslové komunikační infrastruktury.

Detailní popis **nástroje pro aktivní zabezpečení průmyslových zařízení** je uveden v kap. 6.1 ukazuje jeho schopnost analyzovat síťový provoz, rozpoznávat neautorizovaná zařízení a neobvyklé vzory komunikace, které mohou signalizovat bezpečnostní rizika. Dále je vysvětlen mechanismus automatického zásahu, kdy v případě detekce hrozby může systém reagovat okamžitým omezením přístupu dotčeného zařízení nebo segmentu sítě.

Jako klíčová metoda pro zajištění síťové komunikace je implementováno **GUI pro zjednodušenou konfiguraci WireGuard komunikace** viz kap. 6.2, moderního síťové nástroje pro virtuální privátní sítě. WireGuard je zvolen pro svou efektivitu, silné šifrovací schopnosti a snadnou konfiguraci. V diplomové práci je popsán proces integrace WireGuard do průmyslového rozvaděče a je demonstrováno, jak tato technologie poskytuje šifrovanou komunikaci mezi zařízeními, což zvyšuje bezpečnost datového toku v konkrétní síti.

6.1 Systém pro aktivní zabezpečení průmyslových zařízení

Praktická část této práce se zaměřením na systém pro aktivní zabezpečení průmyslových zařízení (anglický překlad – system for active security of industrial devices), zkráceně SAZZ (anglická zkratka – SASID), se věnuje implementaci systému pro monitorování a zabezpečení průmyslových sítí. V následujících podkapitolách je popsán nástroj vytvořený v jazyce Python, který integruje různé metody skenování sítě a jejího monitorování s cílem identifikace a reakce na potenciální bezpečnostní hrozby.

Systém je navržen tak, aby byl kompatibilní s Linux-based operačními systémy, převážně založené na linuxové distribuci OpenWRT, která je primárně určená pro směrovače na embedded zařízeních. Umožňuje integraci s různými typy sítí a zařízení bez nutnosti rozsáhlých úprav existující síťové infrastruktury. Klíčové vlastnosti

výsledného kódu zahrnují modulární návrh, který strukturuje kód do modulů a tříd pro snadnou rozšiřitelnost a údržbu, a nabízí vysokou míru flexibility s podporou různých módů skenování a možností specifikace síťových rozhraní a dalších parametrů. Systém také zahrnuje funkci automatické instalace závislostí pro případ, že některé požadavky nejsou splněny, a vyžaduje ověření administrátorských práv před spuštěním aplikace, čímž zvyšuje bezpečnostní standardy. Dále poskytuje detailní logování a reportování prostřednictvím integrace s různými způsoby reportování, jako jsou email, URL a logy, což usnadňuje efektivní sdílení výsledků analýzy. Při vývoji byl kladen maximální důraz na přizpůsobivost, automatizaci a uživatelskou přívětivost (krátké konfigurační soubory, uživatelské argumenty programu).

Monitorování sítě, včetně implementovaných aktivních prvků jakožto reakce na neznámá invazivní zařízení byl vytvořen ve třech módech z nich si uživatel může zvolit jaký typ je pro něho nejvhodnější z hlediska právních, technických a proporcionálních aspektů monitorované sítě. Aktivní, respektive agresivní varianty monitorování sítě totiž nemohou být využity v každé síti.

6.1.1 Architektura a funkcionalita systému

Systém, který je postaven na modulární architektuře umožňující flexibilní rozšiřování jeho funkcí, využívá kombinaci pasivního odposlechu a aktivního skenování sítě, které se adaptují na specifické potřeby a omezení průmyslového prostředí. Tento přístup umožňuje efektivně monitorovat síťový provoz a včas detekovat anomálie či potenciální bezpečnostní incidenty.

Tento softwarový nástroj má několik základních módů:

- pasivní odposlech,
- aktivní skenování,
- agresivní skenování.

Při pasivním odposlechu pracuje zařízení v módu, kdy nenavazuje síťovou komunikaci, ale poslouchá provoz procházející přes něj v dané síti. Tento mód je ideální pro sítě, v nichž je zakázáno provozovat libovolné aktivní skenování sítě pomocí například nástroje nmap. Pro realizaci pasivního odposlechu byla využita knihovna scapy pro python3, která umožňuje zachytávat síťový provoz, jenž prochází přes sítové zařízení. V případě této práce je jako síťové zařízení brán směrovač RUTX11 od firmy Teltonika, na němž bude spuštěn nástroj SAZZ.

Během aktivního skenování systém využívá metody, které umožňují detailnější rozbor síťové topologie (ARP ping). Aktivní skenování je určeno pro sítě, kde má uživatel tohoto softwarového nástroje možnost provádět v omezené míře aktivní průzkum sítě.

Poslední možností je agresivní skenování, které umožňuje hloubkovou analýzu a identifikaci potenciálních hrozeb. Systém implementuje agresivní skenovací techniky využívající pokročilé skenovací strategie a techniky (např. skenování operačních systémů, dostupnost portů, stealth skenování). Tento mód je určený pro situace, kdy uživatel může v daných sítích, k nimž je připojeno síťové zařízení, vykonávat libovolnou činnost. Pro realizaci agresivního módu byl použit nástroj nmap bez limitací zásahu do skenované sítě. Agresivní mód je díky možnost práce s firewallem a příkazovou řádkou operačního systému routeru RUTX11 silným nástrojem pro hledání a blokování nelegitimních účastníků komunikace, o kterých zároveň dokáže sesbírat maximum možných informací.

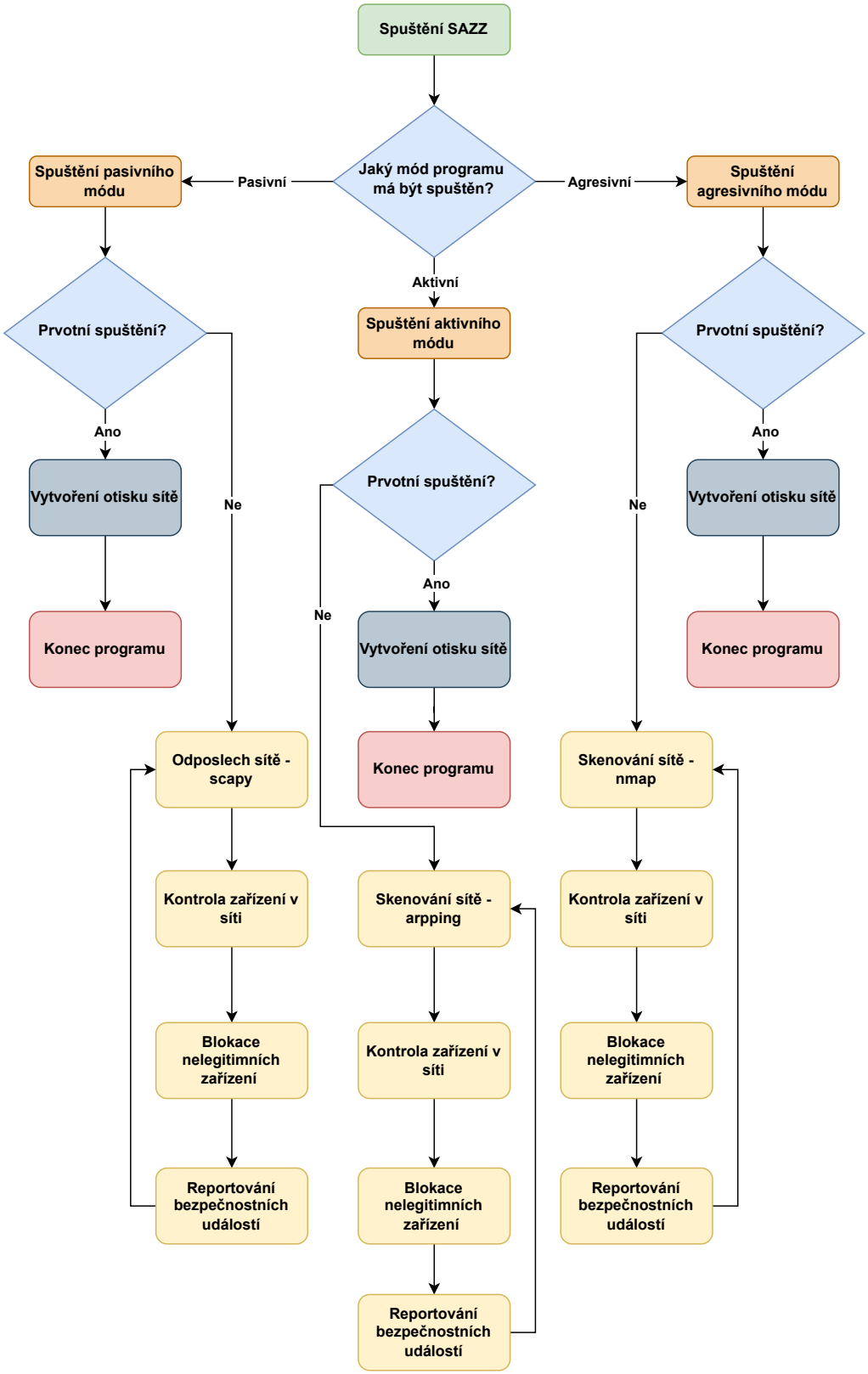
Schéma systému SAZZ, viz obr. 6.1, zahrnuje několik fází a rozhodovacích bodů, které určují konkrétní postupy při spuštění a fungování systému. Prvním krokem v procesu je samotné spuštění systému SAZZ. Po jeho inicializaci je uživatel vyzván k výběru jednoho ze tří provozních módů – agresivního, pasivního nebo aktivního. Tento výběr určuje následné kroky a metodiky použité při monitorování a zabezpečení sítě.

V pasivním módu systém začíná vytvořením otisku sítě, což je krok prováděný pouze při prvotním spuštění. Při druhotném spuštění se provádí odposlech sítě pomocí nástroje knihovny scapy, která umožňuje detailní analýzu síťového provozu. Další fází je kontrola zařízení připojených do sítě, při které jsou identifikována a zablokována nelegitimní zařízení. Tento proces je zakončen reportováním všech případných bezpečnostních událostí, které byly během kontroly zaznamenány, a opětovným spuštěním odposlechu sítě v nekonečné smyčce.

Aktivní mód začíná obdobně vytvořením otisku sítě při prvotním spuštění. Síť je následně skenována pomocí nástroje arpping, který poskytuje informace o aktivních zařízeních na základě ARP dotazů. Po dokončení skenování se provádí kontrola zařízení v síti a identifikace a blokace nelegitimních zařízení. Na závěr jsou reportovány všechny bezpečnostní události a program pokračuje jeho další iterací smyčky.

Agresivní mód rovněž začíná vytvořením otisku sítě při prvotním spuštění. Síť je poté skenována pomocí nástroje nmap, což umožňuje získání detailních informací o připojených zařízeních a jejich stavu. Po skenování následuje kontrola zařízení v síti a blokace nelegitimních zařízení. Stejně jako v aktivním a pasivním módu, i zde je proces zakončen reportováním bezpečnostních událostí a opětovným spuštěním v další iteraci smyčky.

Každý z těchto módů je navržen tak, aby využíval specifické nástroje a metodiky pro monitorování a ochranu sítě, přičemž všechny zahrnují kroky pro skenování sítě, kontrolu zařízení, blokaci nelegitimních zařízení a reportování bezpečnostních událostí. Tímto způsobem SAZZ zajišťuje komplexní ochranu průmyslových zařízení před bezpečnostními hrozbami.



Obr. 6.1: SAZZ - funkční blokový diagram programu

6.1.2 Proces integrace systému zabezpečení na síťovém zařízení

Pokud se nenachází na zařízení je nutné doinstalovat tento software:

- python3
- pip

Níže jsou zmíněny i potřebné python moduly nicméně tyto moduly a jejich dostupnost si program hlídá automaticky a v případě potřeby je doinstaluje.

Potřebné python moduly:

- scapy
- wheel
- requests
- sqlite3
- python3-nmap

6.1.3 Proces spuštění programu pro monitorování sítě

1. Připojit se na zařízení s operačním systémem OpenWRT s root oprávněními
2. Nahrát složky *main*, *scans*, *utils* a *templates* ze zdrojového repositáře, ve kterých se nachází python soubory do adresáře */root*
3. Aktualizovat balíčky – *opkg update*
4. Instalace python – *opkg install python3*
5. Instalace pip – *opkg install python3-pip*
6. Instalace nmap – *opkg install nmap*
7. Přesun do adresáře /root/main
8. Spuštění programu – *python3 network_scanner.py* více o spuštění možnostech spuštění je popsáno níže v kapitole.

6.1.4 Možnosti spuštění programu

Součástí implementace je i konfigurační soubor, kde si může uživatel, mimo zvolení uživatelských parametrů při spuštění, detailně upravit funkcionalitu systému pro aktivní zabezpečeni průmyslové sítě. Konfigurační soubor *project_config.ini* představuje základní konfigurační mechanismus pro aplikaci určenou k monitorování a zabezpečení průmyslových sítí. Tento soubor je strukturován do několika sekcí, kde každá se zabývá specifickými aspekty konfigurace.

U první sekce Language se volí jazyk podle normy ISO 639-1 pro emailovou komunikaci a uživatelské rozhraní programu. Podle tohoto nastavení jsou texty, reporty a logy přizpůsobeny zvoleným požadavkům dle numeronymu lokalizace l10n [30].

Další sekcemi jsou také:

- FilePaths – cesty k databázím a logovacím souborům.
- Security – bezpečnostní parametry pro blokování zařízení.
- Database – detaily pro připojení k databázi a její strukturu.
- NetworkSettings – síťová nastavení včetně velikosti datových bloků.
- EmailReporting – konfigurace pro odesílání reportů emailem.
- UrlReporting – sekce pro URL reportování, která zůstává nespecifikovaná.
- PortsOfInterest – seznam klíčových síťových portů pro monitorování síťového provozu.

Uživatelské spuštění

- Iniciální spuštění systému (`python network_scanner.py -0 -m Passive`)– inicializuje databáze a aktualizuje data. Systém je spuštěn v pasivním módu, což znamená sledování síťového provozu bez aktivního zasahu.
- Standardní spuštění systému (`python network_scanner.py -m Passive -b`) – nastavuje systém do pasivního módu, ale navíc aktivuje funkci blokování, což umožňuje systému blokovat podezřelé síťové zařízení.
- Podporované argumenty pro spuštění skriptu (viz obr. 6.2):

  -h, –help Zobrazí nápovědu s popisem dostupných argumentů a jejich funkcí.

  -0, –init Spustí program v inicializačním módu pro nastavení databází a aktualizace dat.

  -m, –mode Nastaví mód provozu programu. Možnosti jsou `Passive`, `Active`, `Aggressive`.
  - příklad volby módu: *–mode Active*

  -if, –interfaces Určuje rozhraní pro sledování nebo skenování
  - výchozí jsou všechna rozhraní
  - příklad volby rozhraní: *-if eth0,eth1,wwan0*

  -t, –timeout Nastaví časový limit pro sledování nebo skenování v sekundách
  - výchozí je 90 sekund
  - příklad volby časového limitu: *-t 150*

  -p, –period Definuje interval volání sledování/skenování v sekundách
  - výchozí je 5 sekund
  - příklad volby periody: *–period 10*

  -pr, –protocol Specifikuje transportní protokol pro skenování portů
  - výchozí jsou UDP a TCP
  - podporované protokoly jsou UDP a TCP
  - příklad volby protokolu: *–protokol TCP*

  -r, –report Volba metody reportování

  - výchozí je e-mail
  - možnosti reportování jsou emailem nebo URL
  - příklad možnosti zvolení způsobu reportování *–report url*

-i, –incremental Povolí inkrementální přidání zařízení do seznamu platných (výchozí je `False`).

-a, –add Umožní ruční přidání zařízení do seznamu platných zařízení.

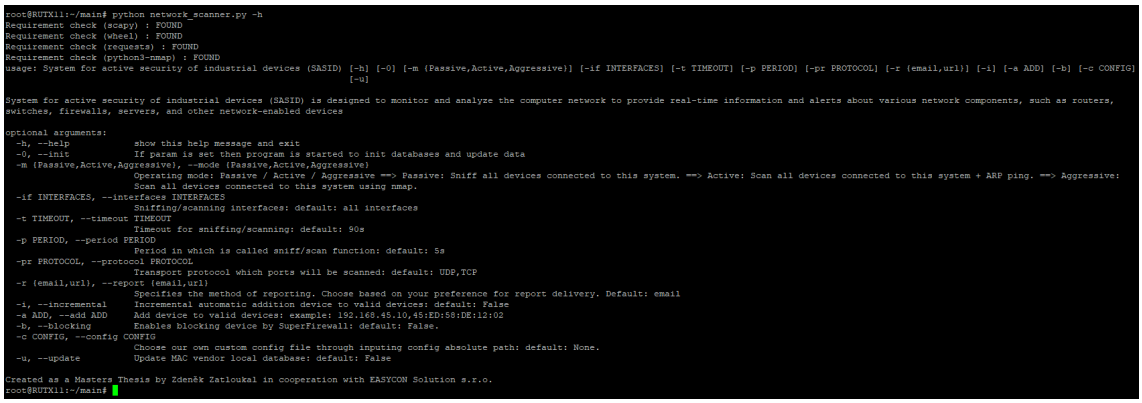
  - příklad přidání nového zařízení do databáze:
    -a 192.168.1.1,11:22:33:44:55:66

-b, –blocking Zapne blokování zařízení SuperFirewallem (výchozí je `False`).

-c, –config Nabídne možnost použití vlastního konfiguračního souboru (výchozí je `None`).

  - příklad výběru konfiguračního souboru: *-c /etc/muj_config.ini*

-u, –update Aktualizuje lokální databázi výrobců MAC adres (výchozí je `False`).



Obr. 6.2: SAZZ - Ukázka help menu programu

6.1.5 Aktivní ochrana sítě - Reportování

Email

V rámci modulu pro reportování e-mailových zpráv byla zavedena funkce pro zasílání podrobných reportů o stavu průmyslových sítí a identifikovaných zařízeních. Tato funkcionalita vybírá sekce e-mailu (například předmět, obsah zprávy, pozdrav a další) z předem definovaných šablon v češtině nebo angličtině, na základě jazykového nastavení specifikovaného v konfiguračním souboru aplikace.

Obsah e-mailu je strukturován do HTML formátu, aby bylo dosaženo vizuálně atraktivního zobrazení informací. Hlavní text a seznam identifikovaných zařízení jsou prezentovány v HTML tabulce, doplněné o zakončující pozdrav. Alternativně, pro

e-mailové klienty, které z bezpečnostních důvodů nepodporují HTML, byla implementována verze e-mailu v prostém textu. Tato verze obsahuje stejné informace jako HTML verze, ale je přizpůsobena pro jednoduchý textový formát, čímž je zajištěna dostupnost informací i v případě, že e-mailový klient HTML nepodporuje.

Pro odeslání e-mailu je použit SMTP server, jehož parametry jsou načteny z konfiguračního souboru. E-mail je odeslán na adresy určené pro příjem reportů, které jsou rovněž definovány v konfiguračním souboru. Celý proces je zakončen voláním funkce pro logování úspěšného odeslání reportu, což umožňuje zaznamenávat informace o počtu adresátů a jejich e-mailových adresách.

Obrázek D.1 prezentuje snímek obrazovky e-mailové komunikace, ve kterém je vizualizován sumární výstup inicializačního skenu systému pro aktivní zabezpečení průmyslových zařízení. V úvodu je standardizovaný textový pozdrav, který naznačuje formalitu a přímý účel zprávy, informující o spuštění daného programu v rámci uživatelovy sítě.

Hlavní částí snímku je tabulka seřazená do tří sloupců, kategorizujících jednotlivá zařízení podle jejich IP adresy, MAC adresy a identifikace výrobce (MAC vendor). Zde uvedené IP adresy reprezentují síťová rozhraní zařízení, zatímco MAC adresy slouží jako unikátní identifikátory síťových rozhraní na linkové vrstvě. Sloupec VENDOR některá zařízení specifikuje, například TP-LINK TECHNOLOGIES CO.,LTD., zatímco pro jiná je uvedeno Not Found, což svědčí o nedostatku informací pro jednoznačnou identifikaci nebo o absenci daných údajů v databázi.

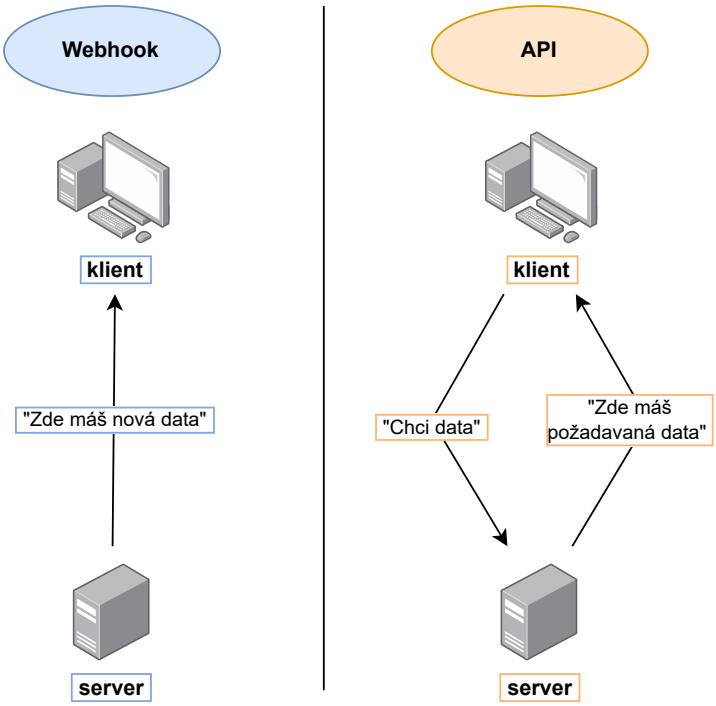
Zobrazení na obr. D.2 je e-mailový komunikační projev, který figuruje jako varovná zpráva indikující detekci neznámého zařízení v rámci síťové infrastruktury. V úvodu je uživateli podán formální pozdrav, po němž následuje výklad kontextu: informuje o operaci Systému pro aktivní zabezpečení průmyslových zařízení. Zpráva dává najevo, že v rámci sítě došlo k identifikaci zařízení, jehož přítomnost nebyla předpokládaná ani očekávaná, následkem čehož bylo zařízení automaticky zablokováno.

Tabulka vložená v těle e-mailu poskytuje systémově strukturované údaje, konkrétně IP adresu, MAC adresu a název výrobce daného zařízení. Tato tabulka obsahuje jediný záznam, který svědčí o izolovaném incidentu. Prezentovaná IP adresa (192.168.1.113) je adresa lokální sítě, zatímco MAC adresa, 60:15:92:70:16:10, představuje jedinečný identifikátor síťového rozhraní zařízení. Segment VENDOR odhaluje, že zařízení bylo vyrobeno společností Unipi Technology s.r.o.

URL

V této práci je kladen důraz na automatizaci procesu reportování pomocí bezpečného HTTPS protokolu. Využití knihovny requests pro jazyk Python umožňuje efektivní

odesílání dat na server prostřednictvím Webhooku. Webhook je speciální druh API, jenž je optimalizován pro automatické zasílání informací reagujících na specifické události bez nutnosti interakce od klienta. Tato vlastnost činí webhooky extrémně vhodnými pro situace, kdy je vyžadována rychlá odezva, jako je okamžitá aktualizace systémového stavu nebo odesílání upozornění v případě anomálií v síti [64].



Obr. 6.3: Znázornění principu reportování přes Webhook

Rozdíl mezi Webhooky a API je znázorněn na obr. 6.3. Automatizace reportování prostřednictvím webhooků znamená vysokou účinnost v síťových systémech, které vyžadují okamžitou reakci. Webhooky, fungující jako zjednodušená forma API, poskytují hlavní výhodu v podobě schopnosti bezprostředního zasílání dat bez explicitního požadavku ze strany klienta, což je ideální pro potřeby síťového monitoringu [64].

Pro implementaci webhooků je nezbytné určit cílové URL, které obsahuje specifickou adresu serveru, port a endpoint pro příjem dat. Zabezpečení komunikace je zajištěno autentizačním klíčem vloženým do hlavičky HTTPS požadavku, což slouží k verifikaci identity zařízení serverem a chrání před neautorizovaným přístupem.

Informace odesílané webhookem jsou ve formátu JSON, který je standardem pro výměnu dat mezi různými systémy a aplikacemi. Využití HTTPS POST metody pak zajistí bezpečný a strukturovaný přenos těchto informací na určené URL.

Význam webhooků v kontextu průmyslových aplikací spočívá v jejich schopnosti odesílat informace v reálném čase, což je klíčové pro efektivní správu propojených systémů.

Logy zařízení

Tato výchozí funkcionalita slouží jako jednoduchý mechanismus pro zaznamenávání událostí do logovacího systému systému pro monitorování a zabezpečení průmyslových sítí. Zprávy jsou formátovány tak, že kontextové údaje a popis události pro efektivní sledování a analýzu provozu systému, umožňuje administrátorům rychle identifikovat a reagovat na významné události a zajišťuje auditovatelnost a transparentnost bezpečnostních operací systému.

6.1.6 Aktivní ochrana sítě - Firewall

Firewall v systému pro aktivní zabezpečení průmyslových zařízení je realizovám pomocí třídy s názvem `SuperFirewall`. Je navržena jako inteligentní bezpečnostní mechanismus, který integruje uživatelské konfigurační soubory, správu databáze a reportovací funkce pro poskytnutí komplexního řešení ochrany síťových zdrojů. Tato třída umožňuje sledování a správu síťových zařízení na základě jejich MAC adresy a definovaných bezpečnostních politik.

Konstruktor třídy

Inicializuje třídu s konfiguračními daty a připravuje správu databáze a reportovací systémy. Vytváří databázové tabulky pro evidenci známých, neznámých a blokovaných zařízení a implementuje kontrolu stavu firewallu spolu s vytvořením speciálních pravidel pro řetězec SASID (speciálně vytvořený řetězec ve firewallu za účelem realizace správy blokování zařízení).

Metoda pro výpočet počtu blokování

Určuje počet blokování, které má být provedeno na základě definovaného multiplikátoru.

Metoda pro vytvoření a přidání pravidel řetězce

Ověřuje existenci řetězce SASID v iptables a případně jej vytváří. Dále vkládá nezbytná pravidla do firewallu pro směrování síťového provozu.

Metoda pro kontrolu stavu firewallu

Prověřuje aktuální stav firewallu ve vztahu k databázovým záznamům blokovaných zařízení a provádí akce blokování nebo odblokování na základě časového plánování v databázi.

Metoda pro odstranění pravidel z iptables

Odstraňuje pravidla pro zadanou MAC adresu ze seznamu iptables, čímž dochází k odblokování zařízení. Zároveň aktualizuje záznamy v databázi a systému pro reportování.

Metoda pro blokování zařízení podle kombinace MAC adresy a IP adresy

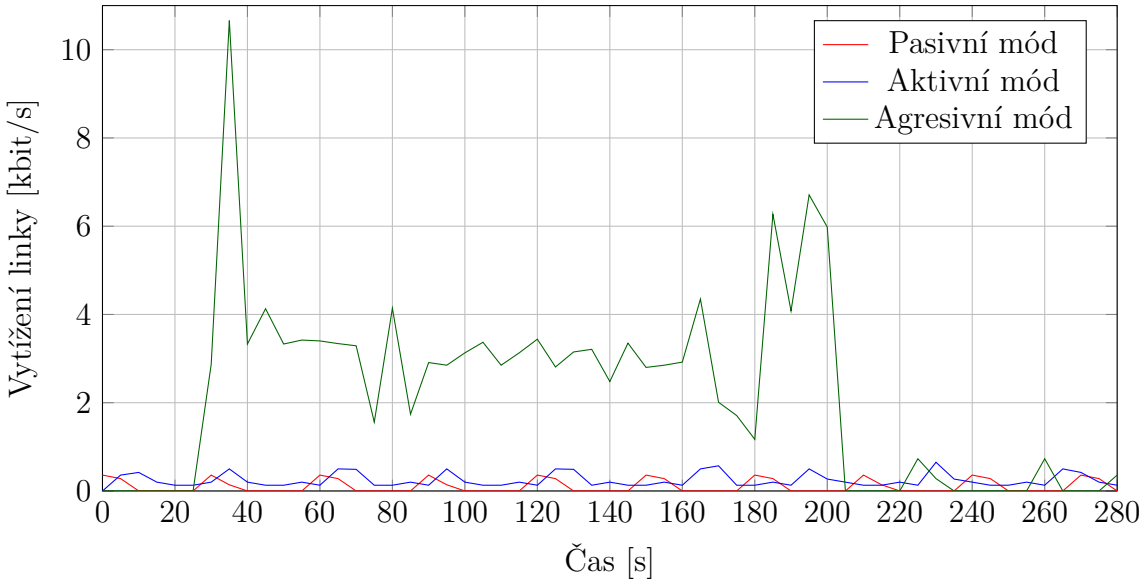
Blokuje zařízení podle MAC adresy v iptables na určitý počet hodin, ukládá tato pravidla a řídí časovače pro automatické odblokování zařízení. Ochrana proti podvržení MAC adresy (MAC spoofing) je zajištěna při prvotní inventarizaci sítě, kdy jsou uloženy do databáze informace o konkrétních zařízeních, které jsou legitimní na dané síti (kombinace IP adresa + MAC adresa). Pokud se v síti objeví nějaké další zařízení, tak pokud se v sítí objeví zařízení, které má stejnou MAC adresu, ale bude mít jinou IP adresu, tak ho systém zablokuje pomocí IP adresy.

Každá z těchto metod představuje nezbytné funkce potřebné pro efektivní řízení bezpečnostních opatření v rámci síťové infrastruktury, což reflektuje potřebu po integraci rozličných nástrojů do jednotného automatizovaného řešení schopného reagovat na potenciální hrozby v reálném čase a poskytovat zprávy pro administrátory.

6.1.7 Vytížení linky při spuštění systému SAZZ

V grafu 6.4 jsou ilustrovány změny v datovém přenosu linky směrovače RUTX11 v závislosti na třech různých provozních módech systému SAZZ – pasivní, aktivní a agresivní. Osa x reprezentuje časový průběh experimentu od 0 do 280 sekund. Osa y ukazuje vytížení linky v kbit/s.

Z grafu je patrné, že v pasivním módu (červená čára v grafu) dochází k nízkému až občas nulovému vytížení linky s výjimkou sporadických špiček vlivem stálé automatické síťové režie v lokální síti. Podobná situace platí i pro aktivní mód (modrá čára v grafu), v kterém je vytížení linky mírně vyšší a stálejší oproti pasivnímu módu, ale opět poměrně nízké oproti agresivnímu (zelená čára v grafu), který vykazuje nejvýraznější zvýšení zatížení linky. Zobrazený graf účinně demonstruje, jak se vytížení linky mění v závislosti na provozních módech.



Obr. 6.4: Vytížení linky v průběhu různých provozních módů systému SAZZ

Uvedený graf potvrzuje požadovaný cíl, aby systém SAZZ zasahoval do sítě takovým způsobem, jak si nadefinuje uživatel dle požadavků správce sítě, v níž tento systém bude v provozu. To znamená, že v pasivním módu systém nezasahuje do sítě vůbec, v aktivním módu systém zasahuje do sítě minimálně (vlivem periodických ARP pingů) a nakonec v agresivním módu je systém nejvíce invazivní vzhledem k vytížení připojené sítě.

6.2 GUI pro zjednodušenou konfiguraci WireGuard komunikace

Toto GUI slouží pro jednoduchou konfiguraci WireGuard tunelu v lokální síti rozvaděče. Bylo při implementační činnosti pojmenování příhodně na WireGUIard. WireGUIard je založen na programovacím jazyce Python a jeho knihovně tkinter, která umožňuje vytvářet grafická uživatelská rozhraní v jazyce Python.

Konfigurace VPN probíhá přes uživatelské rozhraní WireGUIard aplikace pro správu WireGuard VPN, viz obr. E.1. V horní části jsou tlačítka pro připojení, kontrolu a nastavení WireGuard. V sekci pro výběr serveru je možné zvolit předkonfigurovaný server (v tomto případě RUTX11) a pole pro heslo.

Pod touto sekci je textový výpis, který informuje o již úspěšném připojení k SSH serveru s IP adresou 172.16.27.160. Dále jsou zobrazeny informace o úspěšné kontrole požadavků a o generování klíčů pro VPN – soukromý a veřejný klíč jsou zde v nezahešované formě z důvodu kontroly funcionality aplikace. Navíc pokud konfigurujeme

jednu stranu WireGuard tunelu, potřebuje zároveň spustit konfiguraci i na druhém zařízení protože potřebujeme znát veřejný klíč z druhého zařízení, se kterým bude navázáno VPN spojení.

Ve spodní části obrázku je další konfigurační panel s názvem *Nastavení WireGuard*. Zde jsou pole pro zadání názvu rozhraní VPN, IP adresy pro WireGuard, lokální IP adresu protistrany (Peer), IP adresu protistrany a veřejného klíče protistrany. Závěrem je zde tlačítko *Implementovat*, které slouží k aplikaci a uložení provedených nastavení na připojeném zařízení. Tím způsobem je na zařízení umožněna šifrovaná komunikaci oběma zvolenými stranami.

Na obr. 6.5 jsou prezentovány terminálové výstupy konfigurace VPN rozhraní za pomocí technologie WireGuard na dvou zařízeních, konkrétně na routeru RUTX11 a zařízení Unipi. Na obou terminálech je aplikován příkaz `wg`, což je standardní nástroj pro konfiguraci WireGuard rozhraní, jenž je využívám i grafickým uživatelským rozhraním WireGUIard. Na základě výpisu je patrné, že konfigurace obou zařízení zahrnuje definici veřejného klíče a naslouchacího portu pro definované rozhraní *wg0*. Důležitou částí konfigurace je specifikace *peer*, tedy protějšku v VPN tunelu, včetně jeho IP adresy, portu a veřejného klíče. V obou případech jsou zobrazeny statistiky o posledním handshake, což je proces inicializace komunikace mezi zařízeními. Zobrazeno je také množství přenesených dat.

Další částí výstupu jsou výsledky příkazů `ping`, na kterém je demonstrována úspěšná konektivita a nízká latence mezi oběma zařízeními v rámci VPN tunelu. V porovnání s naměřenou latencí nešifrované komunikace se WireGuard jeví jako efektivní a robustní řešení pro nastavení VPN, při ponechání jednoduché konfigurace díky vytvořenému grafické rozhraní.

Měření odezvy zabezpečené komunikace pro 100 ICMP pingů:

- průměrná doba odezvy = 2.958 ms,
- mediánová doba odezvy = 2.626 ms,
- jitter (standardní odchylka doby odezvy) = 1.513 ms

Vyšší hodntota jitteru je přimárně způsobená delším trváním prvního pingu, který je výrazně delší než všechny ostatní doby odezvy z důvodu nutného provedení WireGuard Handshake, jenž trvá cca 8-10 ms.

Pokud bychom přepočetli výše uvedené hodnoty bez prvního pingu, výsledek by byl následující:

- průměrná doba odezvy = 2.700 ms
- mediánová doba odezvy = 2.623 ms
- jitter = 0.203 ms

Tento výsledek není o tolik odlišný jak s ponecháním prvního pingu, pouze se sníží výrazně sníží hodnota jitteru z důvodu zanedbání první výrazně se odlišující hodnoty od průměru.

Měření odezvy nezabezpečené komunikace pro 100 ICMP pingů:

- průměrná doba odezvy = 1.826 ms,
- mediánová doba odezvy = 1.830 ms,
- jitter = 0.037 ms.

Při zabezpečené komunikaci pomocí protokolu WireGuard byla průměrná doba odezvy s prvním pingem 2.958 ms a bez prvního pingu 2.700 ms. Mediánová doba odezvy byla v obou případech srovnatelná, přibližně okolo 2.6 ms. To naznačuje, že i přes šifrování většina paketů dosahovala konzistentních časů. Významné snížení jitteru po odstranění prvního pingu z měření (z 1.513 ms na 0.203 ms) ukazuje, že proces navazování spojení WireGuard (tzv. handshake), který je nutný pro zahájení šifrované komunikace, představuje většinu zpoždění a variability v odezvě.

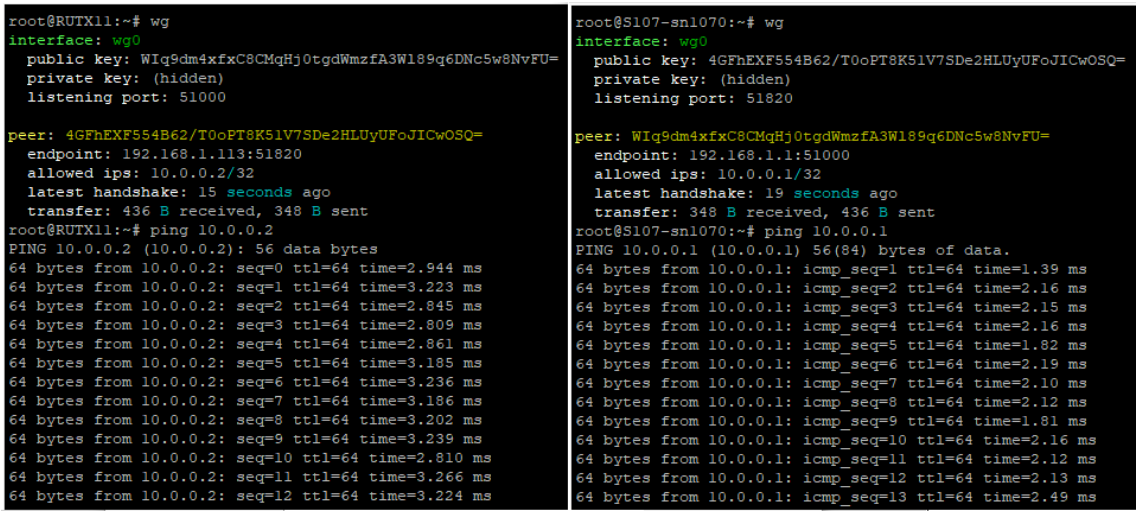
U nezabezpečené komunikace jsou hodnoty průměrné doby odezvy 1.826 ms a mediánové doby odezvy 1.830 ms, což je srovnatelné s výsledky zabezpečené komunikace bez prvního pingu. Nízký jitter 0.037 ms svědčí o vysoké předvídatelnosti a konzistence v časech odezvy, což je typické pro nezabezpečenou komunikaci, kde není přítomna dodatečná latence způsobená šifrovacími procesy.

Na základě těchto dat lze konstatovat, že šifrovaná komunikace pomocí WireGuard má mírně vyšší průměrnou dobu odezvy oproti nezabezpečené komunikaci, což je očekávané v důsledku šifrovací a dešifrovací operací, které vyžadují dodatečný čas. Avšak, WireGuard vykazuje dobrou výkonnost s minimálním dopadem na latenci, jakmile je spojení navázáno, s jasno vylepšenými bezpečnostními výhodami, které šifrování poskytuje.

Ve finálním hodnocení je důležité zdůraznit, že bezpečnostní výhody šifrování WireGuard obvykle převáží mírné zvýšení latence. Proto je využití WireGuard v průmyslových aplikacích, kde je bezpečnost prioritou, vysoce doporučeno i přes zaznamenaný malý nárůst doby odezvy při šifrované komunikaci.

Síťový provoz zníměný výše a znázorněný na obr.6.5 byl analyzován za použití programu Wireshark. Jak ilustruje obr. 6.6, zachycené datové pakety jsou přenášeny prostřednictvím protokolu WireGuard, což je patrné v sloupci *Protocol*. Každý paket je na obrázku číslován v sloupci *No.* a opatřen časovou značkou ve sloupci *Time*.

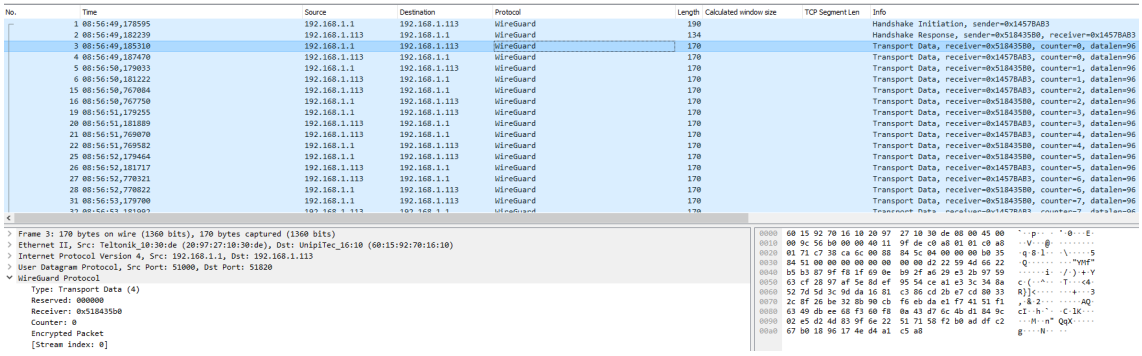
Komunikace mezi koncovými body je zobrazena pomocí zdrojových a cílových IP adres, jež jsou uvedeny ve sloupcích *Source* a *Destination*. Velikost každého paketu je uvedena ve sloupci *Length* a dodatečné technické parametry, jako jsou velikost TCP okna (*Calculated window size*) a délka TCP segmentu (*TCP Segment Len*),



Obr. 6.5: WireGUIard - Ukázka komunikace přes vytvořený WireGuard tunel

poskytují hlubší pohled na charakteristiky síťové komunikace.

Detailní informace o struktuře konkrétního paketu, které jsou zobrazeny v dolní části snímku obrazovky, obsahují údaje specifické pro protokol WireGuard, včetně typu dat, rezervovaného prostoru, identifikátorů odesílatele a příjemce, sekvenčního čísla a šifrovaného obsahu paketu.



Obr. 6.6: WireGUIard - Zachycená komunikace přes vytvořený WireGuard tunel ve Wireshark

6.2.1 Nastavení směrování přes WireGuard tunel

Za předpokladu, že je nutné směrovat specifickou komunikace přes Wiregaurd tunel, např. MQTT komunikace. Tak musí být na zařízení Unipi nastaveno nastaveno statické IP směrování, které zajistí, že veškerá komunikace s cílovou sítí půjde přes WireGuard tunel (rozhraní *wg0*). Příkaz pro přidání takového směrovacího pravidla vypadá takto:

`ip route add <CIDR> dev wg0`
Tento příkaz přidává novou trasu do směrovací tabulky, aby provoz určený pro sítě specifikované pomocí notace CIDR (Classless Inter-Domain Routing) byl přesměrován přes zařízení wg0, tedy rozhraní WireGuard tunelu.

Detailní rozbor jednotlivých částí příkazu:

- *ip route add* – tato část příkazu říká, že chceme přidat nový záznam do směrovací tabulky.
- *<CIDR>* – síť ve formátu CIDR, například 192.168.1.0/24, která určuje cílovou adresu a masku, kam má být síťový provoz směrován.
- *dev wg0* – tento argument určuje, že provoz pro danou síť má být směrován přes rozhraní wg0

Následně protistrana WireGuard tunelu (router RUTX11) musí být přizpůsobena, aby provoz přijímaný přes WireGuard tunel byl správně směrován do internetu, případně na vzdálený server přes další VPN tunel. Tedy pro správné směrovaní důležité nastavit firewall pravidla, aby bylo zajištěno, že data z vnitřní sítě (např. 10.0.0.0/24) jsou korektně maskována při výstupu na internetové rozhraní a zároveň aby byl povolen příjem dat z VPN tunelu wg0 na výstupní ethernetové rozhraní eth1. K tomu slouží následující příkazy iptables:

- `iptables -I POSTROUTING 1 -s <CIDR> -o eth1 -j MASQUERADE -t nat`
  Tento příkaz je součástí nastavení Network Address Translation (NAT) a specificky se týká maskování (MASQUERADE). Slouží k tomu, aby zařízení pomocí VPN tunelu (přes rozhraní wg0) mohla komunikovat s externím internetem nebo jinou sítí připojenou k routeru RUTX11 přes rozhraní eth1.
  POSTROUTING specifikuje, že pravidlo bude aplikováno na pakety, jakmile jsou již rozhodnuty k odeslání na specifické rozhraní,
  1 určuje pozici pravidla v řetězci, zde první pozici, což znamená, že toto pravidlo bude zpracováno jako první,
  -s <CIDR> určuje zdrojovou adresu paketů, pro které bude pravidlo platit, <CIDR> by mělo být specifikováno podle IP rozsahu zařízení připojených do tunelu,
  -o eth1 definuje výstupní rozhraní, přes které pakety opouštějí router,
  -j MASQUERADE umožňuje, že všechny odchozí pakety získají jako zdrojovou IP adresu adresu rozhraní eth1,
  -t nat určuje, že pravidlo patří do tabulky NAT.
- `iptables -I FORWARD -i wg0 -o eth1 -j ACCEPT`
  Tento příkaz povoluje směrování paketů mezi rozhraními, zde konkrétně povoluje předávání paketů přijatých na rozhraní VPN tunelu wg0 na rozhraní eth1, které může vést k internetu nebo jiné lokální síti.

FORWARD týká se paketů, které jsou směrovány přes router a nejsou určeny k lokálnímu doručení,

-i wg0 určuje vstupní rozhraní, odkud pakety přicházejí, v tomto případě z VPN tunelu,

-o eth1 specifikuje výstupní rozhraní, kam mají být pakety směrovány,

-j ACCEPT určuje, že pakety splňující kritéria (příchozí z wg0 a odcházející přes eth1) budou přijaty a dále směrovány.

Tyto příkazy společně umožňují, že veškerá komunikace s cílovou sítí půjde přes WireGuard tunel a následně bude přes router směrována dále s využitím NAT pro přístup k dalším sítím nebo internetu. V případě úspěšného nastavení výše uvedených příkazů je možné na routeru RUTX11 vysledovat, jak vypadá stejná komunikace na úrovní lokální sítě a na úrovní WireGuard, viz obr. E.2. Zabezpečená data, která procházející přes WireGuard tunel, při zachytávání síťového provozu routerem RUTX11 na rozhraní:

- *wg0* jsou vidět nešifrovaně, protože toto rozhraní patří do WireGuard tunelu. Tyto data jsou šifrována.
- *br-lan* jsou vidět šifrovaně, protože toto rozhraní nepatří do WireGuard tunelu, tzn. kdokoliv připojen k lokální sítí **není schopen účinně odposlouchat komunikaci** mezi zařízením Unipi a routerem RUTX11.

Závěr

Cílem práce bylo poskytnout ucelený pohled na průmyslové sítě, jejich architekturu, bezpečnostní výzvy a implementaci klíčových technologií nezbytných pro moderní průmyslové operace.

V první kapitole byly zavedeny základní koncepty průmyslových sítí, které byly porovnány s koncepty informačních sítí. Dále byl proveden detailní rozbor architektur průmyslových řídicích systémů, modelu Purdue a modelu RAMI pro Průmysl 4.0. Rozbor těchto modelů umožnil lepší pochopení současných trendů a potřeb v průmyslové automatizaci. Následující kapitola se zaměřila na bezpečnost průmyslových sítí, kde byly prozkoumány historické útoky a identifikovány hlavní bezpečnostní hrozby. Představeny byly poté metody obrany na různých úrovních architektury TCP/IP, jako například technologie TLS, VPN, MACsec a další fyzické bezpečnostní opatření, které zvyšují celkovou odolnost průmyslových systémů proti externím útokům.

Kapitola třetí podrobně analyzovala komunikační protokoly Modbus (varianty TCP i RTU) a MQTT se zaměřením na jejich bezpečnost. Zvláštní pozornost byla věnována praktickým aspektům implementace a možnostem zabezpečení dat přenášených těmito protokoly. V následujících třech kapitolách byla představena praktická aplikace teoretických znalostí na návrhu a implementaci síťové komunikace, bezpečnostních řešení pro průmyslové rozvaděče, včetně testování a optimalizace těchto systémů v reálných podmínkách.

V kapitole sedmé byla poté podrobně rozvinuta implementace a testování komunikačních protokolů MQTT a HTTPS, zdůrazňující význam robustního zabezpečení a spolehlivosti v průmyslových aplikacích. Poslední, avšak neméně důležitá kapitola byla klíčová pro demonstrování integrace a funkcionality systému pro aktivní zabezpečení průmyslových zařízení. Tato kapitola popsala architekturu systému, procesy integrace a spouštění, stejně jako možnosti aktivní ochrany, jako je reportování neočekávaných událostí a firewall. Bylo představeno také grafické uživatelské rozhraní pro zjednodušenou konfiguraci Wireguard komunikace za účelem zvýšení uživatelské přívětivosti a efektivity v nasazování bezpečnostních opatření.

Tato diplomová práce přinesla celkovou analýzu a řešení pro zlepšení bezpečnosti průmyslových sítí, demonstrovala aplikování bezpečnostních prvků v oblasti průmyslové automatizace a poskytla důležité základy pro další výzkum a rozvoj v této oblasti. Výsledky a doporučení prezentované v této práci jsou zásadní pro zlepšení bezpečnosti a efektivity nasazení bezpečnostních opatření v průmyslových systémech v období neustále rostoucích kybernetických hrozeb.

Literatura

[1] ABDULAZEEZ, A. M. et al., 2020. *Comparison of VPN Protocols at Network Layer Focusing on Wire Guard Protocol*. International Journal of Interactive Mobile Technologies (iJIM), roč. 14, č. 18, s. 157-177. ISSN 1865-7923. Dostupné také z: https://doi.org/10.3991/ijim.v14i18.16507. [cit. 2024-04-20].

[2] ALKHAFAJEE, A. R. et al., 2021. *Security and Performance Analysis of MQTT Protocol with TLS in IoT Networks*. In: 2021 4th International Iraqi Conference on Engineering Technology and Their Applications (IICETA). IEEE, s. 206-211. ISBN 978-1-6654-9461-8. Dostupné také z: https://doi.org/10.1109/IICETA51758.2021.9717495. [cit. 2024-04-20].

[3] ANDY, S.; RAHARDJO, B. a HANINDHITO, B., 2017. *Attack scenarios and security analysis of MQTT communication protocol in IoT system*. In: 2017 4th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI). IEEE, s. 1-6. ISBN 978-1-5386-0549-3. Dostupné také z: https://doi.org/10.1109/EECSI.2017.8239179. [cit. 2023-12-04].

[4] ARRIS ENTERPRISES LLC, 2018. *How MACsec works*. Ruckus FastIron Security Configuration Guide, 08.0.60. Dostupné z: https://docs.ruckuswireless.com/fastiron/08.0.60/fastiron-08060-securityguide/GUID-EBB8AA84-C558-4A12-82F5-3A947FD66CBE.html. [cit. 2023-12-02].

[5] ASHOOR, A. S. a GORE, S., 2011. *Difference between Intrusion Detection System (IDS) and Intrusion Prevention System (IPS)*. In: WYLD, D. C. et al., ed. Advances in Network Security and Applications. Berlin, Heidelberg: Springer Berlin Heidelberg, s. 497-501. ISBN 978-3-642-22539-0. Dostupné také z: https://doi.org/10.1007/978-3-642-22540-6_48. [cit. 2023-10-09].

[6] ASSENZA, G., FARAMONDI, L., OLIVA, G. a SETOLA, R., 2020. Cyber threats for operational technologies. Online. *International Journal of System of Systems Engineering*. Roč. 10, č. 2. ISSN 1748-0671. Dostupné z: https://doi.org/10.1504/IJSSE.2020.109127. [cit. 2024-03-17].

[7] CONTI, M.; DONADEL, D. a TURRIN, F., 2021. *A Survey on Industrial Control System Testbeds and Datasets for Security Research*. IEEE Communications Surveys & Tutorials, roč. 23, č. 4, s. 2248-2294. ISSN 1553-877X. Dostupné také z: https://doi.org/10.1109/COMST.2021.3094360. [cit. 2023-10-02].

[8] ČEPS A.S., 2024. *KODEX PŘENOSOVÉ SOUSTAVY — ČÁST II. Podpůrné služby (PpS)*. Dostupné z: https://www.ceps.cz/cs/kodex-ps. [cit. 2023-04-14].

[9] DAWNBRINGER, A., 2019. *Transport Encryption: Understanding Transport Layer Security (TLS)*. Stockholm: Angelique Dawnbringer. Dostupné z: https://dawnbringer.net/blog/160/understanding_tls. [cit. 2023-10-21].

[10] *Difference between OT and IT networks*, 2020. GeeksforGeeks. Dostupné z: https://www.geeksforgeeks.org/difference-between-ot-and-it-networks/. [cit. 2023-09-27].

[11] DINCULEANĂ, D. a CHENG, X., 2019. *Vulnerabilities and Limitations of MQTT Protocol Used between IoT Devices*. Applied Sciences, roč. 9, č. 5. ISSN 2076-3417. Dostupné také z: https://doi.org/10.3390/app9050848. [cit. 2023-12-04].

[12] Di Pinto, A., Dragoni, Y., a Carcano, A. (2018). TRITON: The first ICS cyber attack on safety instrument systems. Proc. Black Hat USA, 2018, 1-26. Dostupné z: https://scadahacker.com/library/Documents/Cyber_Events/Nozomi%20-%20TRITON%20-%20The%20First%20SIS%20Cyberattack.pdf. [cit. 2023-12-04].

[13] DOWLING, B. a PATERSON, K. G., 2018. *A Cryptographic Analysis of the WireGuard Protocol*. In: PRENEEL, B. a VERCAUTEREN, F., ed. Applied Cryptography and Network Security. Cham: Springer International Publishing, s. 3-21. ISBN 978-3-319-93386-3. Dostupné také z: https://doi.org/10.1007/978-3-319-93387-0_1. [cit. 2024-04-24].

[14] *Easy-to-use Modbus RTU and Modbus ASCII implementation for Python*, 2023. MinimalModbus. Dostupné z: https://minimalmodbus.readthedocs.io/en/stable/usage.html#general-on-modbus-protocol. [cit. 2023-12-03].

[15] ELKEELANY, O. et al., 2002. *Performance analysis of IPSec protocol: encryption and authentication*. In: 2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002. IEEE, s. 1164-1168. ISBN 0-7803-7400-2. Dostupné také z: https://doi.org/10.1109/ICC.2002.997033. [cit. 2024-03-27].

[16] FERGUSON, P. a HUSTON, G., 1998. *What is a VPN?*.

[17] FIELDING, R.; NOTTINGHAM, M. a RESCHKE, J., ed., 2022. *RFC 9110: HTTP Semantics*. Dostupné z: https://www.rfc-editor.org/rfc/rfc9110. [cit. 2023-12-04].

[18] FLATT, H. et al., 2016. *Analysis of the Cyber-Security of industry 4.0 technologies based on RAMI 4.0 and identification of requirements*. In: 2016 IEEE

21st International Conference on Emerging Technologies and Factory Automation (ETFA). IEEE, s. 1-4. ISBN 978-1-5090-1314-2. Dostupné také z: https://doi.org/10.1109/ETFA.2016.7733634. [cit. 2023-10-25].

[19] FRANKEL, S. et al., 2005. *Guide to IPsec VPNs*. Dostupné z: https://www.hhs.gov/guidance/sites/default/files/hhs-guidance-documents/nist80077.pdf. [cit. 2023-10-25].

[20] CHAKRABORTY, N., 2013. *Intrusion detection system and intrusion prevention system: A comparative study*. International Journal of Computing and Business Research (IJCBR), vol. 4, no. 2, s. 1-8. Online. Read the Docs. Dostupné z: http://researchmanuscripts.com/May2013/1.pdf. [cit. 2023-12-01].

[21] CHEN, T. M. a ABU-NIMEH, S, 2011. Lessons from Stuxnet. Online. *Computer*. Roč. 44, č. 4, s. 91-93. ISSN 0018-9162. Dostupné z: https://doi.org/10.1109/MC.2011.115. [cit. 2024-02-12].

[22] HAMZEH, K. et al., 1999. *Point-to-point tunneling protocol (PPTP)*. Dostupné z: https://www.rfc-editor.org/rfc/rfc2637. [cit. 2023-10-25].

[23] HUANG, R.; LIU, F. a PAN, D., 2010. *Research on OPC UA security*. In: 2010 5th IEEE Conference on Industrial Electronics and Applications. IEEE, s. 1439-1444. ISBN 978-1-4244-5045-9. Dostupné také z: https://doi.org/10.1109/ICIEA.2010.5514836. [cit. 2024-04-07].

[24] INTERNATIONAL ELECTROTECHNICAL COMMISSION. IEC 61850-7-2, *Communication networks and systems for power utility automation — Part 7-2: Basic information and communication structure — Abstract communication service interface (ACSI)*. Edition 2.0, 2010. [cit. 2024-04-07].

[25] Industry 4.0, 2015. *Časopis Automa časopis pro automatizační techniku*. Roč. 2015, č. 11, s. 66. [cit. 2023-11-05].

[26] INTERNATIONAL ELECTROTECHNICAL COMMISSION, 2006. *IEC 60870-5-104, Telecontrol equipment and systems — Part 5-104: Transmission protocols — Network access for IEC 60870-5-101 using standard transport profiles*. 2006-06. [cit. 2024-05-07].

[27] INTERNATIONAL ELECTROTECHNICAL COMMISSION, 2013. *IEC 62264, Enterprise-control system integration*. [cit. 2024-04-02].

[28] INTERNATIONAL ELECTROTECHNICAL COMMISSION, 2016. *IEC 62361-100:2016, Power systems management and associated information exchange - Interoperability in the long term - Part 100: CIM profiles to XML schema mapping*. [cit. 2024-02-09].

[29] INTERNATIONAL ELECTROTECHNICAL COMMISSION, 2021. *IEC 62890, Industrial-process measurement, control and automation — Lifecyclemanagement for systems and components*. [cit. 2024-05-10].

[30] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2003. *ISO 639-1, Language code*. [cit. 2024-01-09].

[31] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2019. *ISO 8601-1:2019, Date and time format*. Edition 01 2019-02. [cit. 2024-04-09].

[32] JAKABOCZKI, G. a ADAMKO, E., 2015. *Vulnerabilities Of Modbus Rtu Protocol–A Case Study*. Annals Of The Oradea University, Fascicle Of Management And Technological Engineering, vol. 1. Dostupné z: `https://pdfs.semanticscholar.org/a361/bb132af3f9c9d15f64cfd3ab4c9cd5c2873f.pdf`. [cit. 2023-12-03].

[33] *JAK FUNGUJE KOGENERAČNÍ JEDNOTKA*, 2024. ČEZ ENERGO. Dostupné z: `https://www.cezenergo.cz/cs/o-kogeneraci/jak-funguje-kogeneracni-jednotka`. [cit. 2024-04-12].

[34] KNAPP, E. D. a LANGILL, J. T., 2015. *Industrial Network Security*. Druhé vydání. Waltham, MA: Elsevier. ISBN 9780124201149. Dostupné také z: `https://doi.org/10.1016/C2013-0-06836-3`. [cit. 2023-09-27].

[35] KOTULIAK, I.; RYBAR, P. a TRUCHLY, P., 2011. Performance comparison of IPsec and TLS based VPN technologies. Online. In: *2011 9th International Conference on Emerging eLearning Technologies and Applications (ICETA)*. IEEE, s. 217-221. ISBN 978-1-4577-0052-1. Dostupné z: `https://doi.org/10.1109/ICETA.2011.6112567`. [cit. 2024-05-17].

[36] KRAWCZYK, H.; PATERSON, K. G. a WEE, H., 2013. *On the Security of the TLS Protocol: A Systematic Analysis*. In: CANETTI, R. a GARAY, J. A., ed. Advances in Cryptology — CRYPTO 2013. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, s. 429-448. ISBN 978-3-642-40040-7. Dostupné také z: `https://doi.org/10.1007/978-3-642-40041-4_24`. [cit. 2023-09-23].

[37] LAWAS, J. B. R.; VIVERO, A. C. a SHARMA, A., 2016. *Network performance evaluation of VPN protocols (SSTP and IKEv2)*. In: 2016 Thirteenth International Conference on Wireless and Optical Communications Networks (WOCN). IEEE, s. 1-5. ISBN 978-1-4673-8975-4. Dostupné také z: https://doi.org/10.1109/WOCN.2016.7759880. [cit. 2023-12-04].

[38] LUSKA, I., 2012. *Inventarizační systém aktivních síťových prvků*. Brno: Masarykova univerzita, Fakulta informatiky. Diplomová práce. Vedoucí práce Mgr. Karel Slavíček, Ph.D. Dostupné z: https://theses.cz/id/970000/. [cit. 2023-12-04].

[39] MODBUS ORGANIZATION, INC., 2006. *MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1b*. Modbus.org. Dostupné z: https://modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf. [cit. 2023-09-27].

[40] MODBUS ORGANIZATION, INC., 2006. *MODBUS Messaging on TCP/IP Implementation Guide V1.0b*. Modbus.org. Dostupné z: https://modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf. [cit. 2023-09-23].

[41] *Modbus RTU: A Comprehensive Guide to Understanding and Implementing the Protocol*, 2023. Wevolver. Dostupné z: https://www.wevolver.com/article/modbus-rtu-a-comprehensive-guide. [cit. 2023-12-04].

[42] *Modbus TCP: A Comprehensive Guide to the Protocol and Its Applications*, 2023. Wevolver. Dostupné z: https://www.wevolver.com/article/modbus-tcp-a-comprehensive-guide. [cit. 2023-12-04].

[43] *Modbus-tk: Create Modbus app easily with Python*, 2023. GitHub. Dostupné z: https://github.com/ljean/modbus-tk. [cit. 2023-12-03].

[44] *MQTT Version 3.1.1 - OASIS Standard*, 2014. OASIS. Dostupné z: https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html#_Toc398718058. [cit. 2024-04-21].

[45] PUDELKO, M. et al., 2020. *Performance Analysis of VPN Gateways*. In: 2020 IFIP Networking Conference (Networking). Paris, France. Dostupné z: https://ieeexplore.ieee.org/abstract/document/9142755. [cit. 2024-04-15].

[46] NEČAS, M., 2023. *Modulární Honeypot pro IT a OT*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Bakalářská práce. Vedoucí práce Petr Blažek. Dostupné z: http://hdl.handle.net/11012/210903. [cit. 2023-09-23].

[47] NIEMANN, K.-H., 2019. *IT security extensions for PROFINET*. In: 2019 IEEE 17th International Conference on Industrial Informatics (INDIN). IEEE, s. 407-412. ISBN 978-1-7281-2927-3. Dostupné také z: https://doi.org/10.1109/INDIN41052.2019.8972209. [cit. 2024-04-07].

[48] PATWARDHAN, M., 2012. *DNP3: security and scalability analysis*. PhD Thesis. California State University, Sacramento. Dostupné z: http://hdl.handle.net/10211.9/1645.

[49] PHILLIPS, G., 2020. *The 6 Major VPN Protocols Explained*. Make Use Of. Dostupné z: https://www.makeuseof.com/tag/major-vpn-protocols-explained/. [cit. 2023-09-24].

[50] *Protokoly pro aplikační vrstvu průmyslové automatizace*, 2021. Vyvoj.hw.cz. Dostupné z: https://vyvoj.hw.cz/protokoly-pro-aplikacni-vrstvu-prumyslove-automatizace.html. [cit. 2023-12-10].

[51] *PyModbus - A Python Modbus Stack*, 2023. PyModbus. Dostupné z: https://pymodbus.readthedocs.io/en/latest/source/readme.html#pymodbus-in-a-nutshell. [cit. 2023-12-03].

[52] *Quick start guide*, 2023. Welcome to pyModbusTCP's documentation. Dostupné z: https://pymodbustcp.readthedocs.io/en/latest/quickstart/index.html. [cit. 2023-12-03].

[53] Referenční model struktury Industrie 4.0 RAMI 4.0, 2015. *Časopis Automa časopis pro automatizační techniku*. Roč. 2015, č. 11, s. 66.

[54] SONI, D.; MAKWANA, A., 2017. *A survey on mqtt: a protocol of internet of things (iot)*. In: International conference on telecommunication, power analysis and computing techniques (ICTPACT-2017). p. 173-177. Dostupné z: https://www.researchgate.net/publication/316018571_A_SURVEY_ON_MQTT_A_PROTOCOL_OF_INTERNET_OF_THINGSIOT. [cit. 2023-12-04].

[55] *SQLite Documentation*, 2023. Online. SQLite. Dostupné z: https://www.sqlite.org/docs.html. [cit. 2024-05-17].

[56] SVOBODA, J. et al., 2015. *Network monitoring approaches: An overview*. Int J Adv Comput Netw Secur, vol. 5, no. 2, s. 88-93. Dostupné z: https://www.researchgate.net/profile/Ibrahim_Ghafir/publication/305957483_Network_Monitoring_

Approaches_An_Overview/links/57a75c9d08aefe6167bc1f91/Network-Monitoring-Approaches-An-Overview.pdf [cit. 2023-12-01].

[57] *UModbus*, 2018. Read the Docs. Dostupné z: https://umodbus.readthedocs.io/en/latest/. [cit. 2023-12-03].

[58] *What happens in a TLS handshake? | SSL handshake.* Cloudflare. Dostupné z: https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake/ [cit. 2023-12-04].

[59] *What is HTTPS?*, 2023. Cloudflare. Dostupné z: https://www.cloudflare.com/learning/ssl/what-is-https/. [cit. 2023-12-04].

[60] *What is MQTT Last Will and Testament (LWT)? — MQTT Essentials: Part 9*, 2015. HiveMQ. Dostupné z: https://www.hivemq.com/blog/mqtt-essentials-part-9-last-will-and-testament/. [cit. 2024-04-21].

[61] *What is MQTT Quality of Service (QoS) 0,1, & 2? — MQTT Essentials: Part 6*, 2015. HiveMQ. Dostupné z: https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/. [cit. 2024-04-21].

[62] *What is the CIA triad (confidentiality, integrity and availability)?*, 2023. CHAI, Wesley. TechTarget. Dostupné z: https://www.techtarget.com/whatis/definition/Confidentiality-integrity-and-availability-CIA. [cit. 2023-12-11].

[63] *What is the Modbus Protocol & How Does It Work?*, 2023. NI. Dostupné z: https://www.ni.com/en/shop/seamlessly-connect-to-third-party-devices-and-supervisory-system/the-modbus-protocol-in-depth.html. [cit. 2023-12-11].

[64] *What is a webhook?*, 2024. Red Hat. Dostupné z: https://www.redhat.com/en/topics/automation/what-is-a-webhook. [cit. 2024-04-07].

[65] *What is WireGuard?*, 2023. Cybernews. Dostupné z: https://cybernews.com/what-is-vpn/wireguard-protocol/. [cit. 2023-12-04].

[66] WILLIAMS, T. J., 1994. *The Purdue enterprise reference architecture.* Computers in Industry, vol. 24, no. 2-3, s. 141-158. ISSN 01663615. Dostupné také z: https://doi.org/10.1016/0166-3615(94)90017-5. [cit. 2023-10-25].

[67] XUAN, L. a YONGZHONG, L., 2019. *Research and Implementation of Modbus TCP Security Enhancement Protocol.* Journal of Physics: Conference Series,

2019-06-01, vol. 1213, no. 5. ISSN 1742-6588. Dostupné také z: https://doi.org/10.1088/1742-6596/1213/5/052058. [cit. 2024-04-07].

[68] ZATLOUKAL, Z., 2022. *Zabezpečení energetického polygonu*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií. Bakalářská práce. Vedoucí práce Antonín Bohačík. Dostupné z: http://hdl.handle.net/11012/205538. [cit. 2023-09-23].

[69] ZHANG, Y. et al., 2012. *A survey of security visualization for computer network logs*. Security and Communication Networks, vol. 5, no. 4, s. 404-421. ISSN 1939-0114. Dostupné také z: https://doi.org/10.1002/sec.324. [cit. 2024-04-07].

Seznam symbolů a zkratek

AIC	Availability, Integrity, Confidentiality
ARP	Address Resolution Protocol
CIA	Confidentiality, Integrity, Availability
DDoS	Distributed Denial of Service
DNP3	Distributed Network Protocol
GUI	Graphical User Interface
HTTPS	HyperText Transfer Protocol Secure
ICS	Industrial Control Systems
IPSec	Internet Protocol Security
IT	Informační Technologie
KGJ	Kogenerační Jednotka
MACsec	Media Access Control Security
MQTT	Message Queuing Telemetry Transport
OT	Operační Technologie
PLC	Programovatelný Logický Automat
QoS	Quality of Service
RTU	Remote Terminal Unit
SAZZ	Systém pro aktivní zabezpečení průmyslových zařízení
SCADA	Supervisory Control and Data Acquisition
TCP	Transmission Control Protocol
TLS	Transport Layer Security
VPN	Virtual Private Network
WireGUIard	Grafické uživatelské rozhraní pro zjednodušení implementace Wireguard tunelu

Seznam příloh

A Zdrojové kódy

Vytvořené zdrojové kódy v rámci zachování obchodního tajemství společnosti EASYCON Solution s.r.o. dle § 504 Zákona č. 89/2012 Sb. nejsou zveřejněny.

B Testování Modbus TCP komunikace

B.1 Ukázka simultánního připojení klientů

```
2024-04-12 20:22:51.231961: Client 172.16.27.160:42862 disconnected.
2024-04-12 20:22:51.792641: Client 172.16.27.160:42920 connected.
2024-04-12 20:22:51.989021: Client 172.16.27.160:42876 disconnected.
2024-04-12 20:22:52.001774: Client 172.16.27.160:42928 connected.
2024-04-12 20:22:52.035689: Client 172.16.27.160:42876 disconnected.
2024-04-12 20:22:52.001774: Client 172.16.27.160:42928 connected.
2024-04-12 20:22:52.035689: Client 172.16.27.160:42944 connected.
2024-04-12 20:22:52.117732: Client 172.16.27.160:42878 disconnected.
2024-04-12 20:22:52.237177: Client 172.16.27.160:42950 connected.
2024-04-12 20:22:52.328903: Client 172.16.27.160:42888 disconnected.
2024-04-12 20:22:52.994093: Client 172.16.27.160:41944 connected.
2024-04-12 20:22:53.062874: Client 172.16.27.160:42900 disconnected.
2024-04-12 20:22:53.123178: Client 172.16.27.160:41952 connected.
2024-04-12 20:22:53.334160: Client 172.16.27.160:41962 connected.
2024-04-12 20:22:53.625856: Client 172.16.27.160:42902 disconnected.
2024-04-12 20:22:53.633249: Client 172.16.27.160:42908 disconnected.
2024-04-12 20:22:53.893501: Client 172.16.27.160:42914 disconnected.
2024-04-12 20:22:54.068619: Client 172.16.27.160:41970 connected.
2024-04-12 20:22:54.631805: Client 172.16.27.160:41976 connected.
2024-04-12 20:22:54.638310: Client 172.16.27.160:41982 connected.
2024-04-12 20:22:54.800422: Client 172.16.27.160:41992 connected.
2024-04-12 20:22:54.801860: Client 172.16.27.160:42920 disconnected.
2024-04-12 20:22:55.003349: Client 172.16.27.160:42928 disconnected.
2024-04-12 20:22:55.038532: Client 172.16.27.160:42944 disconnected.
2024-04-12 20:22:55.239814: Client 172.16.27.160:42950 disconnected.
2024-04-12 20:22:55.799817: Client 172.16.27.160:42008 connected.
2024-04-12 20:22:56.090408: Client 172.16.27.160:41944 disconnected.
2024-04-12 20:22:56.097362: Client 172.16.27.160:42018 connected.
2024-04-12 20:22:56.098755: Client 172.16.27.160:42022 connected.
2024-04-12 20:22:56.125563: Client 172.16.27.160:41952 disconnected.
2024-04-12 20:22:56.244929: Client 172.16.27.160:42034 connected.
2024-04-12 20:22:56.336674: Client 172.16.27.160:41962 disconnected.
2024-04-12 20:22:57.002041: Client 172.16.27.160:42044 connected.
2024-04-12 20:22:57.070713: Client 172.16.27.160:41970 disconnected.
2024-04-12 20:22:57.145469: Client 172.16.27.160:42058 connected.
2024-04-12 20:22:57.342087: Client 172.16.27.160:42064 connected.
2024-04-12 20:22:57.633667: Client 172.16.27.160:41976 disconnected.
2024-04-12 20:22:57.641130: Client 172.16.27.160:41982 disconnected.
2024-04-12 20:22:57.797453: Client 172.16.27.160:41992 disconnected.
2024-04-12 20:22:58.075893: Client 172.16.27.160:42066 connected.
```

Obr. B.1: Znázornění připojení více klientů na Modbus TCP server

B.2 Ukázka sekundového čtení dat

```
 Odeslán požadavek na čtení dat pro 20 registrů/veličin v čase 2024-04-18 20:16:42.000854
Registr 40000 = 51065 [kW/V/°C/Pa]
Registr 40001 = 50832 [kW/V/°C/Pa]
Registr 40002 = 52046 [kW/V/°C/Pa]
Registr 40003 = 50516 [kW/V/°C/Pa]
Registr 40004 = 51818 [kW/V/°C/Pa]
Registr 40005 = 52452 [kW/V/°C/Pa]
Registr 40006 = 52406 [kW/V/°C/Pa]
Registr 40007 = 50523 [kW/V/°C/Pa]
Registr 40008 = 52225 [kW/V/°C/Pa]
Registr 40009 = 50740 [kW/V/°C/Pa]
Registr 40010 = 51376 [kW/V/°C/Pa]
Registr 40011 = 51489 [kW/V/°C/Pa]
Registr 40012 = 52021 [kW/V/°C/Pa]
Registr 40013 = 51137 [kW/V/°C/Pa]
Registr 40014 = 51932 [kW/V/°C/Pa]
Registr 40015 = 51089 [kW/V/°C/Pa]
Registr 40016 = 50596 [kW/V/°C/Pa]
Registr 40017 = 50388 [kW/V/°C/Pa]
Registr 40018 = 51410 [kW/V/°C/Pa]
Registr 40019 = 52254 [kW/V/°C/Pa]
 Doba potřebná pro čtení a zpracování dat: 0.105913 s
 Odeslán požadavek na čtení dat pro 20 registrů/veličin v čase 2024-04-18 20:16:43.108454
Registr 40000 = 51064 [kW/V/°C/Pa]
Registr 40001 = 50830 [kW/V/°C/Pa]
Registr 40002 = 52049 [kW/V/°C/Pa]
Registr 40003 = 50521 [kW/V/°C/Pa]
Registr 40004 = 51822 [kW/V/°C/Pa]
Registr 40005 = 52457 [kW/V/°C/Pa]
Registr 40006 = 52404 [kW/V/°C/Pa]
Registr 40007 = 50526 [kW/V/°C/Pa]
Registr 40008 = 52227 [kW/V/°C/Pa]
Registr 40009 = 50738 [kW/V/°C/Pa]
Registr 40010 = 51376 [kW/V/°C/Pa]
Registr 40011 = 51492 [kW/V/°C/Pa]
Registr 40012 = 52024 [kW/V/°C/Pa]
Registr 40013 = 51140 [kW/V/°C/Pa]
Registr 40014 = 51936 [kW/V/°C/Pa]
Registr 40015 = 51088 [kW/V/°C/Pa]
Registr 40016 = 50599 [kW/V/°C/Pa]
Registr 40017 = 50393 [kW/V/°C/Pa]
Registr 40018 = 51411 [kW/V/°C/Pa]
Registr 40019 = 52253 [kW/V/°C/Pa]
```

Obr. B.2: Ukázka sekundového čtení při 25 hodinovém testu

C MQTT komunikace

C.1 Přijímaní MQTT dat odběratelem

```
-----2024-04-20 10:13:24.297554-----
Byla přijata zpráva `Velicina 0  = 24` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 1  = 2` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 2  = 82` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 3  = 98` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 4  = 24` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 5  = 46` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 6  = 33` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 7  = 39` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 8  = 36` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 9  = 46` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 10  = 43` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 11  = 25` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 12  = 44` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 13  = 41` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 14  = 34` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 15  = 23` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 16  = 24` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 17  = 89` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 18  = 14` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 19  = 22` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
-----2024-04-20 10:13:25.297670-----
Byla přijata zpráva `Velicina 0  = 4` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 1  = 23` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 2  = 39` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 3  = 60` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 4  = 9` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 5  = 64` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 6  = 46` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 7  = 64` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 8  = 100` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 9  = 22` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 10  = 46` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 11  = 58` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 12  = 44` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 13  = 66` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 14  = 95` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 15  = 97` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 16  = 44` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 17  = 68` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 18  = 41` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 19  = 20` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
```

Obr. C.1: Ukázka implementace MQTT odběratele

C.2 Změna odesílacího rozhraní z MQTT na HTTPS

```
-----2024-04-20 10:29:21.008637-----
Byla přijata zpráva `Velicina 0  = 94` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 1  = 52` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 2  = 25` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 3  = 34` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 4  = 18` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 5  = 13` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 6  = 99` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 7  = 22` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 8  = 15` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 9  = 59` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 10  = 32` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 11  = 66` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 12  = 9` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 13  = 44` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 14  = 58` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 15  = 8` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 16  = 33` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 17  = 51` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 18  = 48` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 19  = 30` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
-----2024-04-20 10:29:22.007097-----
Byla přijata zpráva `Velicina 0  = 15` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 0  = 15
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 1  = 57
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 2  = 66
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 3  = 78
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 4  = 84
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 5  = 68
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 6  = 39
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 7  = 11
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 8  = 57
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 9  = 85
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 10  = 32
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 11  = 52
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 12  = 3
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 13  = 50
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 14  = 53
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 15  = 46
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 16  = 92
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 17  = 84
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 18  = 70
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 19  = 20
172.16.27.160 - - [20/Apr/2024 10:29:22] "POST /data HTTP/1.1" 200 -
```

Obr. C.2: Ukázka změny způsobu odesílání dat z MQTT na HTTPS

C.3 Změna odesílacího rozhraní z HTTPS na MQTT

```
172.16.27.160 - - [20/Apr/2024 10:33:27] "POST /data HTTP/1.1" 200 -
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 0  = 71
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 1  = 3
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 2  = 22
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 3  = 78
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 4  = 59
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 5  = 25
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 6  = 80
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 7  = 60
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 8  = 38
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 9  = 4
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 10  = 31
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 11  = 85
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 12  = 97
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 13  = 24
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 14  = 78
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 15  = 90
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 16  = 13
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 17  = 61
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 18  = 68
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 19  = 65
172.16.27.160 - - [20/Apr/2024 10:33:28] "POST /data HTTP/1.1" 200 -
-----2024-04-20 10:33:44.314636-----
Byla přijata zpráva `Velicina 0  = 76` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 1  = 86` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 2  = 70` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 3  = 82` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 4  = 48` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 5  = 41` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 6  = 100` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 7  = 68` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 8  = 83` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 9  = 48` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 10  = 20` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 11  = 39` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 12  = 47` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 13  = 33` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 14  = 6` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 15  = 36` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 16  = 25` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 17  = 85` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 18  = 19` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
Byla přijata zpráva `Velicina 19  = 44` s tématem `Easycon_Veliciny` určená pro EasyconSubscriber1234
```

Obr. C.3: Ukázka změny způsobu odesílání dat z HTTPS na MQTT

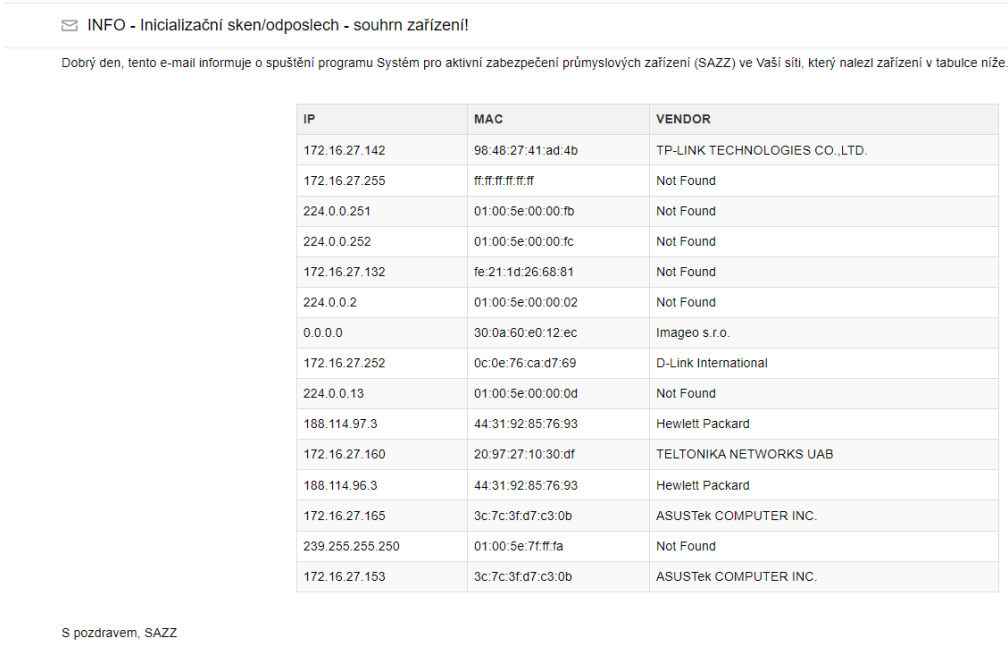
C.4 Zpětné přijetí dat za uplynulý časový okamžik

```
#################################################################
 Zpětně přijatá data za období 2024-04-20 17:08:02.609694 - 2024-04-20 17:08:14.642503
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 0 - 2024-04-20 17:08:02.609694 = 41
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 1 - 2024-04-20 17:08:02.611802 = 15
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 2 - 2024-04-20 17:08:02.614374 = 20
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 3 - 2024-04-20 17:08:02.616645 = 94
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 4 - 2024-04-20 17:08:02.618932 = 75
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 5 - 2024-04-20 17:08:02.621225 = 90
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 6 - 2024-04-20 17:08:02.623527 = 99
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 7 - 2024-04-20 17:08:02.626128 = 70
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 8 - 2024-04-20 17:08:02.628436 = 69
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 9 - 2024-04-20 17:08:02.630742 = 71
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 10 - 2024-04-20 17:08:02.633076 = 68
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 11 - 2024-04-20 17:08:02.638174 = 45
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 12 - 2024-04-20 17:08:02.640298 = 100
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 13 - 2024-04-20 17:08:02.642382 = 19
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 14 - 2024-04-20 17:08:02.645790 = 51
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 15 - 2024-04-20 17:08:02.647708 = 75
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 16 - 2024-04-20 17:08:02.651030 = 78
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 17 - 2024-04-20 17:08:02.653569 = 46
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 18 - 2024-04-20 17:08:02.655490 = 65
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 19 - 2024-04-20 17:08:02.657751 = 27
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 0 - 2024-04-20 17:08:03.610696 = 37
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 1 - 2024-04-20 17:08:03.612771 = 46
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 2 - 2024-04-20 17:08:03.615336 = 20
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 3 - 2024-04-20 17:08:03.617616 = 32
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 4 - 2024-04-20 17:08:03.619944 = 5
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 5 - 2024-04-20 17:08:03.622253 = 10
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 6 - 2024-04-20 17:08:03.624668 = 42
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 7 - 2024-04-20 17:08:03.627023 = 21
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 8 - 2024-04-20 17:08:03.629367 = 92
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 9 - 2024-04-20 17:08:03.631667 = 23
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 10 - 2024-04-20 17:08:03.634036 = 71
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 11 - 2024-04-20 17:08:03.636513 = 50
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 12 - 2024-04-20 17:08:03.638848 = 10
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 13 - 2024-04-20 17:08:03.641284 = 51
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 14 - 2024-04-20 17:08:03.643432 = 17
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 15 - 2024-04-20 17:08:03.646014 = 53
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 16 - 2024-04-20 17:08:03.648368 = 90
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 17 - 2024-04-20 17:08:03.650722 = 87
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 18 - 2024-04-20 17:08:03.653063 = 83
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 19 - 2024-04-20 17:08:03.655501 = 41
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 0 - 2024-04-20 17:08:04.611784 = 65
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 1 - 2024-04-20 17:08:04.613803 = 14
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 2 - 2024-04-20 17:08:04.616239 = 65
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 3 - 2024-04-20 17:08:04.618631 = 3
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 4 - 2024-04-20 17:08:04.620962 = 48
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 5 - 2024-04-20 17:08:04.623282 = 52
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 6 - 2024-04-20 17:08:04.625701 = 24
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 7 - 2024-04-20 17:08:04.628105 = 55
Přijatá HTTPS data od EasyconPublisher1234 : Velicina 8 - 2024-04-20 17:08:04.630261 = 32
```

Obr. C.4: Ukázka zpětně přijatých dat přes HTTPS při předchozím 12s výpadku komunikace

D Systém pro aktivní zabezpečení průmyslových zařízení

D.1 Reportování vytvořeného otisku sítě



INFO - Inicializační sken/odposlech - souhrn zařízení!

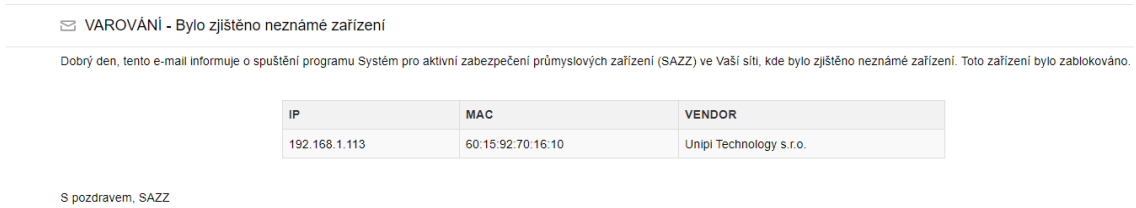
Dobrý den, tento e-mail informuje o spuštění programu Systém pro aktivní zabezpečení průmyslových zařízení (SAZZ) ve Vaší síti, který nalezl zařízení v tabulce níže.

IP	MAC	VENDOR
172.16.27.142	98:48:27:41:ad:4b	TP-LINK TECHNOLOGIES CO.,LTD.
172.16.27.255	ff:ff:ff:ff:ff:ff	Not Found
224.0.0.251	01:00:5e:00:00:fb	Not Found
224.0.0.252	01:00:5e:00:00:fc	Not Found
172.16.27.132	fe:21:1d:26:68:81	Not Found
224.0.0.2	01:00:5e:00:00:02	Not Found
0.0.0.0	30:0a:60:e0:12:ec	Imageo s.r.o.
172.16.27.252	0c:0e:76:ca:d7:69	D-Link International
224.0.0.13	01:00:5e:00:00:0d	Not Found
188.114.97.3	44:31:92:85:76:93	Hewlett Packard
172.16.27.160	20:97:27:10:30:df	TELTONIKA NETWORKS UAB
188.114.96.3	44:31:92:85:76:93	Hewlett Packard
172.16.27.165	3c:7c:3f:d7:c3:0b	ASUSTek COMPUTER INC.
239.255.255.250	01:00:5e:7f:ff:fa	Not Found
172.16.27.153	3c:7c:3f:d7:c3:0b	ASUSTek COMPUTER INC.

S pozdravem, SAZZ

Obr. D.1: Znázornění reportu nalezených zařízení při iniciálním skenu sítí

D.2 Reportování neznámého zařízení



VAROVÁNÍ - Bylo zjištěno neznámé zařízení

Dobrý den, tento e-mail informuje o spuštění programu Systém pro aktivní zabezpečení průmyslových zařízení (SAZZ) ve Vaší síti, kde bylo zjištěno neznámé zařízení. Toto zařízení bylo zablokováno.

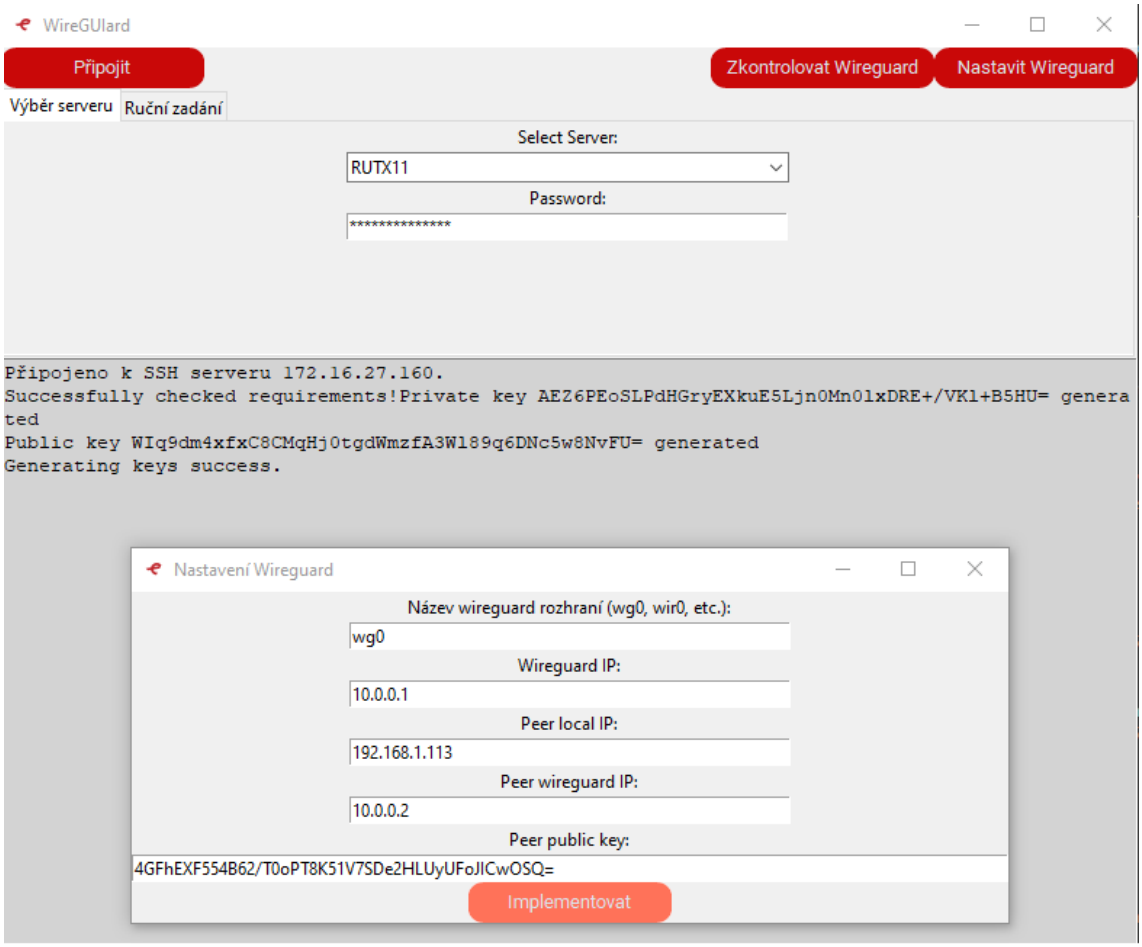
IP	MAC	VENDOR
192.168.1.113	60:15:92:70:16:10	Unipi Technology s.r.o.

S pozdravem, SAZZ

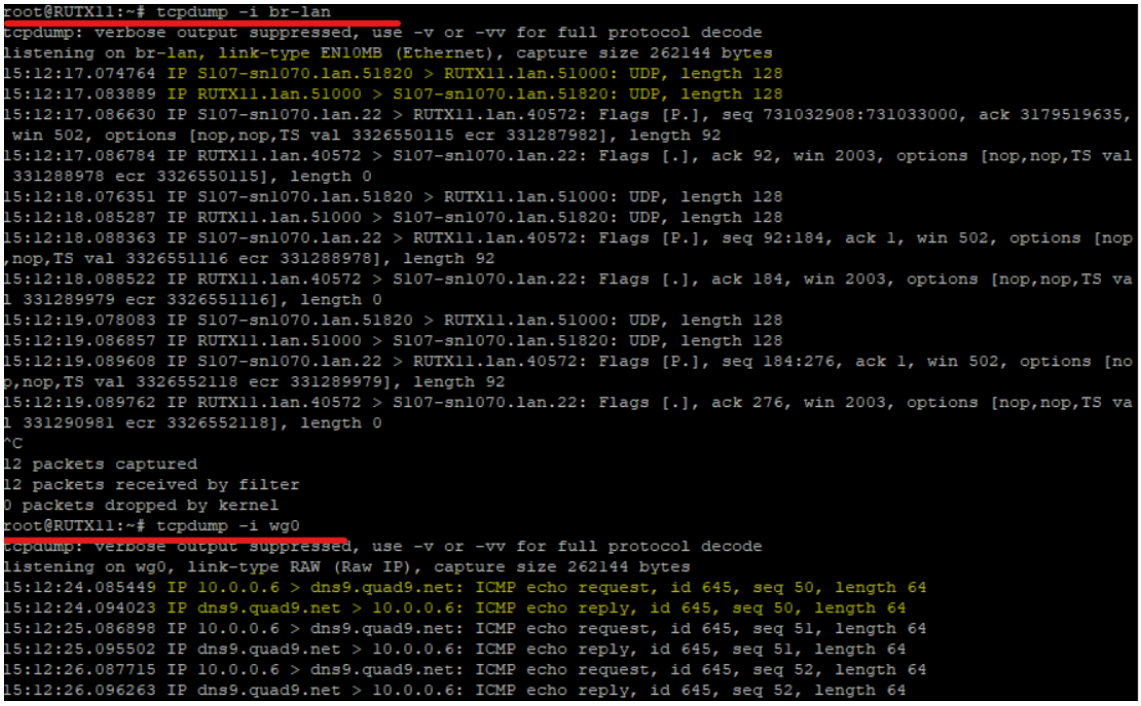
Obr. D.2: Znázornění reportu nalezeného neznámého zařízení

E WireGUIard

E.1 Grafické uživatelské rozhraní Wireguard konfigurátoru



Obr. E.1: WireGUIard – Znázornění průběhu konfigurace



Obr. E.2: WireGUIard – Ukázka komunikace zachycené mimo a uvnitř Wireguard tunelu