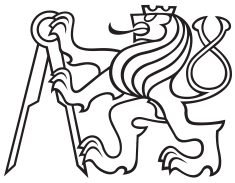
Bakalářská práce



České
vysoké
učení technické
v Praze

F2

Fakulta strojní
Ústav přístrojové a řídící techniky

Průmyslový ethernet pro distribuované řídicí systémy v duchu Industry 4.0

Lukáš Podmela

Vedoucí: Mgr. Ing. Jakub Jura, Ph.D
Obor: -
Studijní program: Teoretický základ strojního inženýrství
Květen 2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Podmela** Jméno: **Lukáš** Osobní číslo: **499085**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav přístrojové a řídicí techniky**

Studijní program: **Teoretický základ strojního inženýrství**

Studijní obor: **bez oboru**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Průmyslový ethernet pro distribuované řídicí systémy v duchu Industry 4.0

Název bakalářské práce anglicky:

Industrial Ethernet for I 4.0 DCS

Pokyny pro vypracování:

1) Vytvořit přehled v současnosti v průmyslu využívaných konceptů distribuovaného řízení (HMS, MAS, IoT etc.)
2) Provést rešerši vědeckých článků popisujících jednotlivé aplikace
3) Provést rešerši vědeckých článků popisujících aplikace kooperace a koordinace autonomních jednotek
4) Teoreticky se seznámit a prakticky vyzkoušet využití inteligentního průmyslového switche XC208
5) Naprogramovat sestavit jednoduchou úlohu pro přenos průmyslových dat, kde bude možné demonstrovat analýzu odposlechnuté komunikace

Seznam doporučené literatury:

[1] Janeček, J. Distribuované systémy. Praha: Vydavatelství ČVUT - FEL, vydání druhé, 1994.
[2] K. P. Keyur, "Internet of things-iot: Definition, characteristics, architecture, enabling technologies, appliction future challanges," p. 6122, 2016. Jura, J.:
[3] Holonický výrobní systém Diplomová práce. Praha: ČVUT FS, Ústav přístrojové a řídicí techniky, 2004,
[4] XC 208 manuály:
https://cache.industry.siemens.com/dl/files/837/109762837/att_969706/v1/PH_SCALANCE-XB-200-XC-200-XF-200BA-XP-200-XR-300WG-WBM_76.pd
https://cache.industry.siemens.com/dl/files/149/109743149/att_1133069/v1/BA_SCALANCE-XC-200_76.pdf

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Mgr. Jakub Jura, Ph.D. U12110.3

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **26.04.2024** Termín odevzdání bakalářské práce: **31.05.2024**

Platnost zadání bakalářské práce: _____

Ing. Mgr. Jakub Jura, Ph.D.	prof. Ing. Tomáš Vyhlídal, Ph.D.	doc. Ing. Miroslav Španiel, CSc.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

26.4.2024
Datum převzetí zadání

Podpis studenta

Poděkování

Chtěl bych poděkovat vedoucímu práce doktoru Jakubu Jurovi za kvalitní vedení a rady při zpracování této práce.

Dále bych chtěl poděkovat své rodině a nejbližším přátelům, kteří mě podporovali při mém studiu.

V poslední řadě bych chtěl poděkovat za tři roky kvalitního studia na Fakultě strojní ČVUT v Praze.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškerou použitou literaturu.

V Praze, 31. května 2024

Abstrakt

Cílem této práce je vytvořit úlohu pro možnosti analýzy odposlechnuté komunikace probíhající na průmyslovém ethernetovém switchi Scalance XC208. V práci je nejdříve rešerše na téma výrobních systémů multiagentních a holonických. Zároveň je v práci rešerše na základy síťové komunikace. V praktické části je vytvořena úloha na odposlech portů switche za pomoci SNMP protokolu v TIA portalu. V poslední části je uvedena ukázka analýzy OPC UA komunikace v programu Wireshark.

Klíčová slova: výrobní systémy, síťová komunikace, ethernet, Scalance XC208, SNMP, PLC, SCADA

Vedoucí: Mgr. Ing. Jakub Jura, Ph.D
12110 - Department of Instrumentation and Control Engineering,
Technická 4,
Praha 6

Abstract

The aim of this thesis is to create a task for analysis of intercepted communication on ethernet switch Scalance XC208. In the thesis, there is research on topic multiagent manufacturing systems and holonic manufacturing systems . Also there is research on basics of network communication. The practical part contains TIA portal code using SNMP protocol for realising the task of interception of communication. In the last part of thesis, there is sample analysis of OPC UA in Wireshark.

Keywords: manufacturing systems, network communication, ethernet, Scalance XC208, SNMP, PLC, SCADA

Title translation: Industrial Ethernet for I 4.0 DCS

Obsah

Obrázky

Tabulky

Část I

Teoretická část práce

Kapitola 1

Úvod

Práce se zabývá průmyslovou komunikací jakožto nutným předpokladem pro vyspělé systémy řízení v duchu iniciativy průmyslu 4.0.

V praktické části práce jsem naprogramoval úlohu pro možnosti odposlechu síťové komunikace v rámci jednotlivých portů průmyslového ethernetového switche XC208. Úloha je nahrána do PLC S7-1200, které následně komunikuje se switchem za pomoci síťového protokolu SNMP. PLC je ovládáno SCADA systémem ve kterém je vytvořena vizualizace celé úlohy. Komunikace mezi SCADA systémem a PLC probíhá za pomoci nezašifrovaného komunikačního protokolu OPC UA. Na závěr praktické části zanalyzuji proběhlou OPC UA komunikaci pomocí programu Wireshark a zároveň identifikuji jednotlivé struktury proběhlé OPC UA komunikace.

V teoretické části práce nejdříve popisuji různé koncepty pokročilých řídicích systémů jako jsou holonické výrobní systémy nebo multiagentní výrobní systémy a zároveň s tím spjaté principy jejich koordinace a kooperace. následně se práce věnuje popisu komunikačních struktur.

Průmysl 4.0 si mimo jiné klade za cíl změny napříč společností. Tento cíl je v plánu dosáhnout za pomoci automatizace, a ta je spjata primárně s průmyslovou výrobou. Z těchto důvodů je potřeba stále více myslet na aplikaci a vylepšování výrobních systémů. Díky této revoluci se tak například dostávají do popředí myšlenky vycházející od filozofa Arthura Koestlera ohledně tzv. holonů.

Kapitola 2

Úvod do průmyslu 4.0

Technologický pokrok nutí firmy přizpůsobovat se změnám a dokázat si udržet pozici na trhu. S rostoucí konkurencí v produktivitě a kvalitě práce je potřeba se zaměřit na vylepšení manažerských a výrobních procesů. Nedílnou součástí průmyslu 4.0 je digitalizace. K tomu nám mohou dopomoci technologie jako jsou internet věcí (Internet of Things), big data, Cloud computing, digitální dvojče nebo také aditivní výroba[14]. Tyto technologie jsou součástí širšího konceptu, nazývaný jako Průmysl 4.0 nebo také Čtvrtá průmyslová revoluce. Průmysl 4.0 je vnímán jako další krok průmyslové revoluce, která může výrazně změnit vnímání průmyslu, a změnit přístup v komunikaci mezi lidmi a stroji stejně jako v interakci mezi dodavateli, výrobci a zákazníky. Myšlenka Průmyslu 4.0 je založena na konceptu integrování virtuálních a fyzických systémů za pomoci Cyber-physical systems (CPS). Efektivním využitím IoT, cloud computing, AI a big data a jejich následná integrace do automatizačního procesu výrazně zlepší průmysl nejen po stránce provozní, ale také po ekonomickém a enviromentálním aspektu. [15]

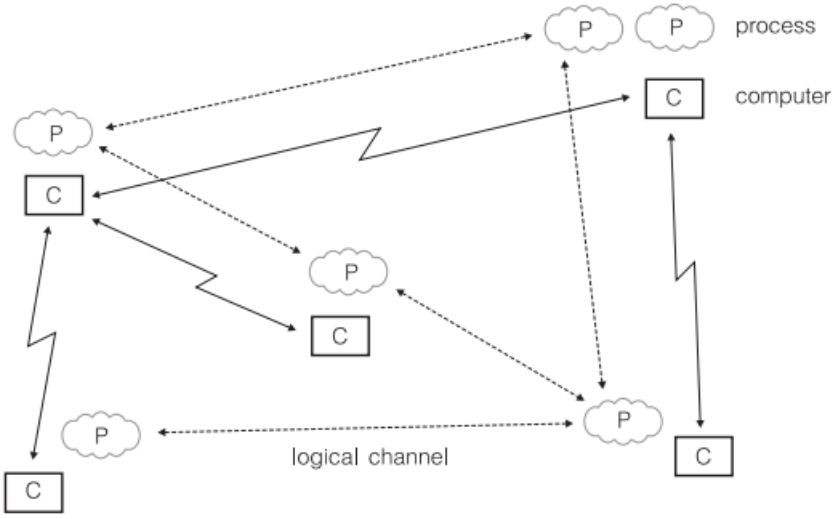
2.1 Distribuovaný systém

Z definice plyne, že distribuovaný systém je takový systém, který je složen z několika autonomních výpočetních prvků, které se uživateli jeví jako jeden koherentní systém. Tato definice odkazuje na dvě charakteristické vlastnosti distribuovaných systémů. První z nich je ta, že výpočetní elementy jsou na sobě nezávislé. Výpočetním elementem může být jak hardwarové zařízení, tak softwarový proces. Druhou důležitou vlastností je představa uživatele, že

pracuje s jednotným systémem. To znamená, že autonomní výpočetní elementy musí mezi sebou spolupracovat. Způsob, jakým zařídit tuto spolupráci je klíčové pro distribuované systémy. [16]

Jako distribuovaný systém označujeme:

- systém, který obsahuje více než jeden počítač nebo procesor[1]
- program, krerý je rozdělený na dílčí celky, ty si vzájemně předávají data [1]
- data, která jsou počítána na různých procesorech/počítačích [1]



Obrázek 2.1: Architektura distribuovaného systému[1]

Na obrázku 1.1 vidíme architekturu distribuovaného systému, kde je program rozložený na jednotlivé počítače, které jsou propojeny dvoubodovými spoji. Komunikační spoje mezi procesy nekopírují propojení počítačů. V některých případech se totiž počítače navíc podílejí na zprostředkování komunikace mezi ostatními počítačovými procesy, a to i přesto, že samy vykonávají své vlastní výpočty. [1]

2.2 Umělá inteligence (AI)

Existuje velké množství definic, kterými lze popsat umělou inteligenci, jednou z nich je například tato: "Umělá inteligence je schopnost strojů napodobovat lidské schopnosti, jako je uvažování, učení se, plánování nebo kreativita."[17]. Mezi úkoly, které je schopné AI vykonávat řadíme, například rozpoznávání řeči, strojové vidění, překládání jazyka. Aplikace AI zahrnují také například webové vyhledávání, systém doporučení daného obsahu aj. Umělou inteligenci lze dělit na tzv. Úzkou umělou inteligenci (NAI), kdy se jedná o systémy, které jsou schopny řešit jednu vymezenou úlohu. Příkladem může být virtuální asistent Google/Apple. Druhým typem je Obecná umělá inteligence (AGI), která je schopna řešit zadané úlohy na stejné nebo vyšší úrovni, než by je řešil člověk. Toto je schopna zvládnout bez předchozího učení vymezených úloh. V průmyslu se můžeme s využitím AI setkat u výrobních systémů, jako tomu je u multiagentních systémů, kde se využívají aplikace strojového vidění nebo strojového učení. AI může ze získaných dat přebírat a vizualizovat zásadní informace pro lidskou obsluhu, predikovat budoucí selhání nebo například předpokládat, jak moc velký lidský zásah bude potřeba do řešení daného problému.[18]

2.2.1 IoT (Internet of things)

IoT, z překladu internet věcí, lze popsat, jako miliony objektů, které mohou vnímat, komunikovat a sdílet informace, kdy je vše zároveň vzájemně propojeno přes soukromé nebo veřejné IP sítě. U těchto vzájemně propojených objektů jsou zároveň pravidelně sbírána a analyzována data. Na základě znalostí těchto dat jsou následně iniciovány akce za účelem plánování, rozhodování a managementu. Takto lze pohlížet na internet věcí. IoT je běžně definováno jako síť fyzických objektů. Internet není pouze sítí počítačů, ale sítí zařízení všech typů a velikostí, jako jsou vozidla, chytré telefony, hračky, kamery, zařízení v domovech, přičemž vše je připojené k síti, vzájemně komunikuje a sdílí informace, které jsou založené na stanovených protokolech. Tyto znalosti mohou být využity například v trasování, pozicování, bezpečnosti atd. Technologie využívané v rámci IoT mohou být například RFID (Radio Frequency Identification, v překladu rozpoznávání základě radiových frekvencí) nebo IP protokol[19].

2.2.2 Big data

Žijeme v éře velkého objemu dat, kdy čím dál více roste jejich velikost a data se stávají čím dál více rozmanitá. Tento nárůst a rozmanitost způsobuje, že se data stávají náročnějšími na zpracovávání, kontrolu a analýzu. Data mohou být generována z různých zdrojů, například z již zmíněného IoT, emailů, vědeckých dat, dokumentů, webových stránek a dalších. K tomu, abychom byli schopni zpracovávat tato data, se často využívá umělá inteligence (AI) jako třeba machine learning (ML) a deep learning (DL)[20]. Termín big data bývá často nadužíván a proto je důležité rozeznat, co to jsou big data. Tři nejzákladnější charakteristiky k definování termínu Big data je jejich objem, rozmanitost a rychlost. Rozmanitostí je myšlen fakt, že data jsou z vícero kategorií (nezpracovaná data, strukturovaná, semi-strukturovaná a nestrukturovaná) Objem označuje velmi velké množství generovaných dat, obvykle v petabytech nebo vyšších jednotkách. Rychlost se zabývá přílivem dat přicházejících z různých zdrojů.[21]

Díky velkému objemu dat tak vzniká spousta možností, jak je využít. V průmyslu se tato data využívají k lepší analýze potřeb a požadavkům klientů. Díky tomu lze přinést zcela nové produkty a inovace. S tím se pojí i možné zvýšení produktivity a snížení nákladů, tím, že budeme schopni predikovat prodej. Big Data najdou uplatnění také například v otázce životního prostředí, zdravotní péče, veřejného sektoru atd.[21]

2.2.3 Cloud computing

Jedna z definic říká, že Cloud je rozsáhlé úložiště jednoduše použitelných a přístupných virtualizovaných zdrojů, jako je například hardware, vývojové platformy a služby. Tyto zdroje mohou být dynamicky rekonfigurovány k upravení proměnné zátěže, což také umožňuje optimální využití zdrojů. Tento model je založen na vzdáleném internetovém přístupu, například skrze webový prohlížeč. Základním principem Cloud computingu je propůjčení výpočetního výkonu serverů uživateli.[22]

Cloud computing se vyznačuje především těmito body:

- Multitenancy - neboli sdílení zdrojů počítače mezi uživateli[23]
- Aktuálnost - poskytovatel musí zaručit, že software je vždy aktualizovaný tak, aby se uživatel do tohoto procesu nevměšoval[23]
- Přístup přes internet - jak již bylo zmíněno, důležitou vlastností je, aby se uživatel mohl připojit skrze internet a využít tak sofwaru odkudkoliv[23]
- Škálovatelnost, elasticita - uživatel má možnost operativně měnit výpočetní zdroje[23]
- Pay per use utility model - cenový tarif, nastavený různými kritérii, jako je například množství spotřebovaných dat a doba po kterou uživatel službu platí (např. jednorázový, měsíční, roční poplatek)[23]
- Virtualizace - virtualizací se označuje technologie, která nám vytváří virtuální reprezentaci fyzických komponent, jako jsou servery, úložiště, sítě. Software virtualizace přenáší funkce těchto hardwarových zařízení do virtuálního prostředí pro ovládání více virtuálních strojů současně. To pomáhá řídit infrastrukturu více efektivně.[23] Příkladem takového softwaru může být open-source program Docker[24].

Stěžejní výhodou Cloud computingu je jeho vysoká škálovatelnost a spolehlivost. Tyto vlastnosti je ale potřeba podporovat, tedy například při výpadku aplikace je nutné, aby byl uživatel přesměrován na jinou instanci bez ztráty rozpracovaných dat. Nevýhodou je nutnost internetového připojení, bez něho není možné získat přístup k datům. Další nevýhodou může být závislost na poskytovateli. Firmy vlastnící cloud ztrácí možnost rozhodovat.[23]

2.2.4 Digitální dvojče

S konceptem digitálního dvojčete poprvé přišel v roce 2002 Michael Grieves v rámci průmyslové prezentace. Digitální dvojče v originální podobě je popisováno jako digitální počítačový model fyzického systému, vytvořený jako entita sebe samého. [25] Toto bylo v roce 2012 doplněno o definici od E. Glaessegena : *"Digitální dvojče je integrovaný multifyzikální, vícestupňová pravděpodobnostní simulace komplexního produktu, kde se využívá nejlepších dostupných fyzikálních modelů, aktualizace senzorů atd. pro napodobení života příslušného dvojčete."[26]*

Díky konceptu digitálního dvojčete je možné v průmyslu odhalit různé chyby a nedostatky ještě předtím, než se daný stroj nebo zařízení uvede do provozu. Tento model nalezl velké využití ve výrobních závodech, kde je možné díky tomuto přístupu zkrátit dobu zprovoznění linek, případně ještě zvýšit jejich efektivitu [25]

Kapitola 3

Holonický výrobní systém

Impulsem pro vývoj technologie holonických výrobních systémů (anglicky Holonic manufacturing systems), které mají za úkol realizovat systémy inteligentního řízení a průmyslové výroby, byla nastupující složitost výrobních systémů. Zároveň bylo potřeba udržet stávající kvalitu a produktivitu, přičemž hlavní důraz byl kladen na flexibilitu. Počátek konceptů HMS se datuje k roku 1990.[27] [28]

Fungování holonického systému je založeno na relativně autonomních řídících systémech, které nazýváme holony. Ty jsou fyzicky propojeny s regulovanými systémy. Díky tomu jsme schopni zajistit celý proces výroby pomocí malých distribuovaných jednotek. Celý koncept je zamýšlen tak, aby zde nebyla v ideálním případě žádná centrální jednotka, která by proces regulovala. Každý holon má omezený přístup k celkovým informacím o struktuře, schopnostech a cílech výrobního systému jako celku. I přesto jsou holony schopny fungovat samostatně a spolupracovat mezi sebou výměnou informací, což je nezbytný předpoklad pro dosažení požadovaných cílů.[4]

Holony provádějí potřebné regulační procesy lokálně a samostatně po většinu provozní doby. Výjimkou jsou kritické situace, jako je přetížení zařízení, výpadek subsystému nebo změna výrobního plánu. V těchto situacích holony informují ostatní holony, prostřednictvím zpráv, a spouští složité procesy spolupráce, aby se situace změnila účelným způsobem, např. k prevenci důsledků poruchy, rozložení zátěže nebo proaktivní rekonfiguraci výrobního systému.[4]

3.1 Holony

Arthur Koestler v roce 1968 zavedl termín "holon", který popisuje základní organizační jednotku biologických a společenských systémů. Slovo holon vzniklo kombinací řeckých slov holos, tedy celek a slova on, znamenající část. Koestler zmiňuje, že holon, který si můžeme představit například jako nějaký stroj, je autonomní a samostatná jednotka, která má jistou úroveň svobody v rozhodování. Může tedy řešit problémy bez konzultace s nadřízenými, resp. bez přímých pokynů. Zároveň jsou však holony řízeny několika autoritami současně, díky poskytnutým informacím o problému. [28] [29]

Tyto předpoklady podporují princip decentralizovaného řízení, zároveň tento přístup přispívá ke stabilitě řízení tohoto systému. Koestler tak přichází s pojmem Hierarchie holonu, tzv. Holarchie, ta je řízena informacemi od nadřízených jednotek, informacemi od podřízených jednotek a zároveň informacemi z vnějšího prostředí. [28]

Pro využití v oblasti průmyslového řízení holony naleznou využití v dopravě, transformaci a ukládání informací a fyzických objektů, které lze řadit do hierarchistických struktur. Holony, které jsou používané pro řízení v reálném čase, jsou propojeny s fyzickým zařízením výrobního systému. Z čehož vyplývá, že může pozorovat a také fyzicky ovlivňovat výrobní proces. To si lze představit jako vypnutí elektrického proudu při poruše, spuštění pohonu dopravníku aj. Základní vlastností holonu je jeho rekurzivita, kdy daná funkce je opakovaně volána dokud není zastavena podmínkou, která je předem stanovena.[4]

Obecnější úvahy vedou k myšlence holonického výrobního závodu. V tomto závodě jsou všechny zde prováděné operace založeny na principu holonu. Začínaje objednávkou produktu, přes plánování, výrobu až po doručení klientovi. Každá složka, která existuje buď fyzicky a nebo virtuálně (stroj, objednávka,..), je vnímána jako autonomní holon. Komunita holonů nám zajistí automatický běh výrobního procesu, přičemž není potřeba centrálního řízení. [4]

3.1.1 Schéma holonu

Holon tedy vnímáme jako autonomní prvek schopný kooperace ve výrobním systému.

Obrázek 3.1: Schéma holonu [2]

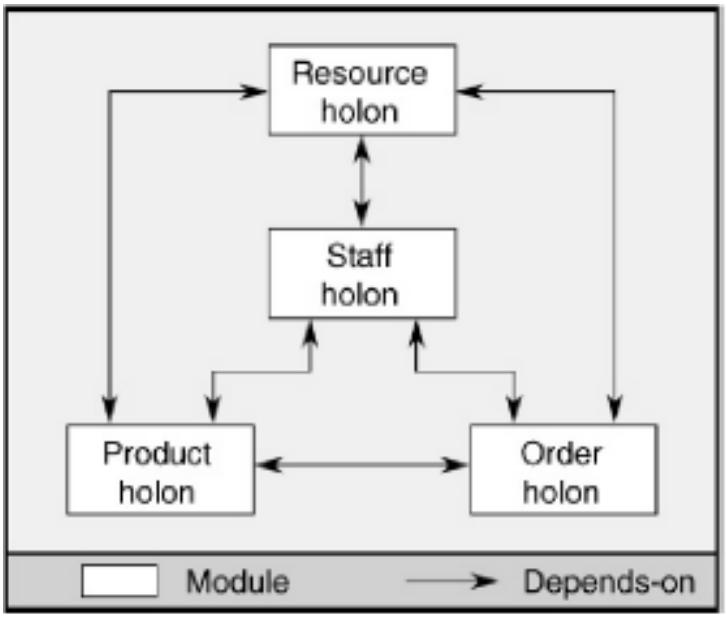
Na Obrázku 2.1 lze vidět schéma holonu s jeho jednotlivými částmi.

- Část fyzického vykonávání činnosti není nezbytná pro určité spektrum holonu. Příkladem může být rozvrhovací nebo plánovací holon.[28]
- Oblast zpracování pohybu je složeno ze samotné fyzické jednotky. Ta má za úkol vykonávat výrobní operace. Část fyzického řízení stojí za samotným řízení pohybu (řídící jednotka u PLC)[28]
- Oblast zpracovávání informací je poté složena z vnitřního rozhraní, které se stará o komunikaci s ostatními holony. Dále z části rozhodování, kdy jde pravděpodobně o software s AI a části uživatelského rozhraní, které je určeno obsluze k sledování holonu a následnému udávání příkazů. [28]

Nejpoužívanějším modelem je holon skládající se ze subsystémů nižší úrovně založené na funkčních blocích a obalu tvořící softwarový agent, kdy se berou v potaz 3 úrovně komunikace. Komunikace probíhá v samotném holonu, kdy mezi sebou interagují funkční bloky a softwarový agent. Komunikovat mezi sebou mohou také na úrovni softwarových agentů nebo na úrovni funkčních bloků. Tento způsob je definice holonů je pak obdobný definici agentům (viz kapitola Multiagentní systém). Při standardizaci komunikace mezi holonu je pak možné tyto holony zařadit do komunity agentů v podniku.[30]

3.1.2 Referenční architektura PROSA

Referenční architektura PROSA popisuje holarchii jako takovou, nepopisuje schéma jednotlivých holonů. Zkratka PROSA vychází z jednotlivých složek holonů, kteří tuto architekturu reprezentují. Jde o Product (Produktový) holon, Resource (Zdrojový) holon, Order (Objednávací) holon a Staff (Personální) holon. Na obrázku 3.2 lze vidět modulový pohled na architekturu PROSA, kde v jednotlivých modulech jsou konkrétní typy holonů, spojující šipky značí jednotlivé závislosti. [31]



Obrázek 3.2: PROSA architektura [3]

Resource holon

Takový holon zodpovídá za využívání a aplikaci jednotlivých zdrojů pro konkrétní oblast využití. Například v rámci logistiky to konkrétně znamená, že všechny dopravní prostředky a stroje na manipulaci s materiálem bude tento holon představovat. Každý zdrojový holon se skládá jak z fyzické části, tak z části softwarové, která ovládá daný zdroj.[3] Resource holon se stará o:

- Zobrazování reality - Zobrazování aktuálního a budoucího stavu zdroje.[3]
- Předávání informací ostatním holonům - Procesní informace, místní topologie, různé limitující informace (maximální kapacita) atd.[3]
- Udržování a spravování harmonogramu - Každý holon spravuje harmonogram, kde jsou zapisovány všechny budoucí operace/ úkony. Zdrojový holon má pravomoc v organizování jednotlivých operací.[3]
- Virtuální provedení operace - To slouží pro Order holony, kteří mohou Source holon požádat o virtuální výstup daného úkonu.[3]
- Správa zdrojů - Kontrola a monitorování reálných zdrojů.[3]

Product holon

Hlavní úkony, o které se stará product holon:

- Udržování procesních znalostí - Takový holon má povědomí o probíhajícím procesu, jako je například jeho plánování, parametry a nároky na kvalitu.[3]
- Určování provozních možností - Order holon je informován o možnosech dalších operací.[3]
- Poskytování provozních informací - Předávání procesních parametrů resource holonu před započetím jeho operace. [3]

Order holon

Order holon odpovídá úkonu nebo zadání, který musí být vykonán . Je zodpovědný za zpracování alokace zdrojů (přidělování omezeného množství zdrojů).[3]

Order holon je zodpovědný za:

- Zobrazování reality - Například informace o aktuálním stavu objednávky (umístění objednávky, jaká objednávka je aktuálně zpracovávána, zdroje využívané k zpracování objednávky,..[3]
- Vyhledávání a výběr řešení - Hledá možná řešení pro vykonání úkolu, to komunikuje s product holonem, aby mohl tyto operace virtuálně vykonat a zjistil dostupnost zdrojů. Na základě dostupných dat vybere nejlepší vyhledané řešení. Ostatní holony o tomto záměru informuje a rezervuje si potřebné zdroje k provedení operace. [3]

Staff holon

Staff holon je vnímán jako tzv. asistenční holon, kdy zmíněným základním typům holonů může asistovat a pomáhat jim se svými znalostmi v určitých aspektech jejich rozhodování. Základní holony jsou ovšem stále zodpovědny za své finální rozhodnutí. Díky staff holonům jsme schopni zavést do systému jistou centralizaci. Tyto holony získávají své znalosti z různých algoritmů, AI metodám, či přístupem člověka. [3]

Interakce mezi holony

Komunikace product holonu s order holonem je založena na tom, jak úspěšně vykonat svůj úkol za použití určitých zdrojů. Na základě výsledku virtuálního vykonání úkonu, order holon předá informaci o vykonání a dalších možných zdrojů. Na základě této informace product holon předá zprávu order holonu o dalších možných operacích.[3]

Product holon s resource holonem sdílí provozní informace. Při vytváření seznamu možných budoucích operací pro order holon, product holon nejdříve zjistí od resource holonu informace o dostupných zdrojích, aby věděl, jaké operace je možné vykonat. Resource holon od product holonu také je seznámen o technických předpokladech pro správné zpracování objednávky.[3]

Resource holon s order holonem spolu interagují pro operace spojené s rezervací jednotlivých zdrojů. Resource holon předává order holonu výsledky virtuálně vykonaných operací a při požadavku od order holonu, rezervuje potřebné zdroje. Jakmile je operace spuštěna, resource holon dává také informace o průběhu a výsledku operací. [3]

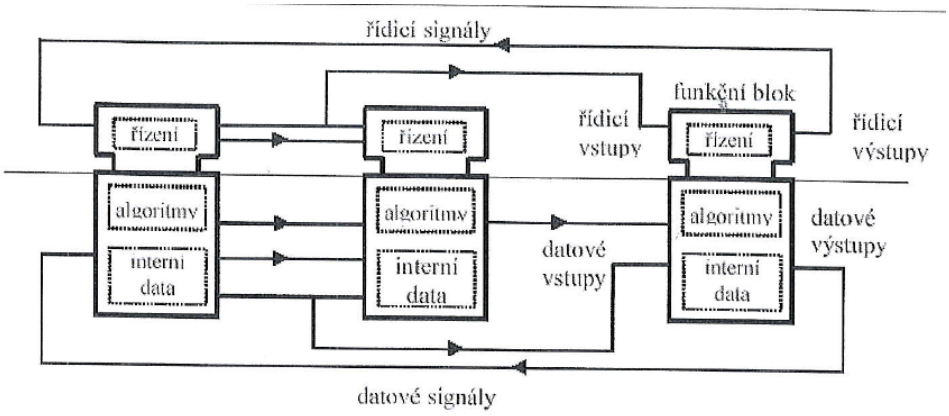
3.2 Standardizace HMS

Výzkum zabývající se holonickými systémy a uplatňování jeho myšlenek se soustředil kolem mezinárodního konsorcia HMS. Pro implementování vize holonického výrobního podniku je zásadní zpracování informací na více úrovních:

- úroveň řízení v reálném čase, která je úzce spjata s fyzickou stránkou výrobního zařízení[4]
- úroveň plánování a rozvrhování výroby na vícero úrovních - od jednotlivých výrobních úseků po celý závod[4]
- úroveň dodavatelského-odběratelského řetězce, zajišťující přesun materiálů, produktů, služeb,...[4]

3.2.1 IEC 61499

Řízení v reálném čase holonickými systémy je založeno na standardu IEC 61499, pracující s funkčními bloky (function blocks), který se zaměřuje na komponentní přístup. Standard rozšiřuje již zavedenou architekturu standardem IEC 1131-3 pro programovací jazyky PLC. Neznatelnější změnou je oddělení datových a řídících signálů, jak lze vidět na obr 3.3 [32][4]



Obrázek 3.3: Funkční bloky[4]

Signály, které přijdou na řídící vstupy zjistí spolu s informací stavu (interní data) chování bloku, přičemž zvolí správný algoritmus, ten z datových vstupů vypočte hodnoty datových výstupů. [4] Standard může být vnímán jako abstrakce systémových komponent, které jsou implementované a kontrolované softwarem FBs. Řídící aplikace je vytvořena spojením vícero FBs. Model řízený FBs dává inteligenci a autonomii zdrojům systému, což jim umožňuje rozhodovací schopnosti. Výhodou standardu IEC 61499 je, že jako modelovací jazyk je přímo spustitelný a proto přímo připravený pro simulace. Simulační model může být bezproblémově nahrazen hardware rozhraním reálných senzorů a akčních členů. Díky funkčním blokům je zajištěna výborná modularita, která nám umožňuje znovu použít softwarové komponenty. Pro standard IEC 61499 je důležitá distribuovanost systému (v případě, že FBs umístíme fyzicky na jiné zařízení, funkční propojení nebude ovlivněno). [32]

Kapitola 4

Multiagentní výrobní systém

Jako multiagentní systém vnímáme ve výrobě takový systém, ve kterém působí agenti, kteří mají známky určité elementární inteligence, díky které jsou schopni vyřešit zadaný problém. U MAS dochází k interakci určitých agentů mezi sebou a nebo v prostředí, ve kterém kooperují. Multiagentní systémy s holonickými spolu sdílí velké množství principů. Primární rozdíl mezi agenty a holony je takový, že zatímco u agentů je základem zpracování dat, tak holony se váží fyzicky k výrobnímu procesu. [33]

Pro MAS je stěžejní, že každý agent má pouze určitý úhel pohledu na danou problematiku, tedy má informaci pouze o určitém segmentu problému a nemá kompletní informaci. Z toho důvodu je zaveden takzvaný Middle agent (střední agent), který spravuje seznam služeb nabízených jednotlivými agenty. Agent, který požaduje určité služby nejdříve kontaktuje Middle agenta, ten ho nasměruje k agentu s požadovanými informacemi. Middle agenty lze dělit na Facilitátora a Mediátora. Facilitátor se chová jako zprostředkovatel mezi agentem posílající žádost a agentem poskytujícím službu. Implementace mediátora je založena na tom, že jednotlivé subjekty spolu komunikují napřímo a tím se snižuje zátěž na mediátora. Důvodem, proč využívat MAS je pro její efektivitu, nízké náklady na činnost, flexibilita a spolehlivost pro řešení komplexních úkonů.[34]

Multiagentní systémy se v mnohém shodují se systémy holonickými. Principy, které sdílí jsou především v oblasti komunikačních protokolů, koordinačních a komunikačních strategií, oblast rozhodování a rozvrhování. Základním prvkem u obou systémů je autonomnost a kooperativnost jejich jednotek [4]

MAS jsou standardizovány organizací FIPA, která přichází s definicemi mechanismů externí komunikace mezi agenty v rámci nichž je definován například standard jazyka ACL.[4]

Možné dělení MAS:

- Z pohledu vedení - V takovém MAS je jeden agent Vůdce, který stanovuje cíle a úkoly ostatním agentům. MAS může být s vůdcem nebo bez vůdce.[35]
- Z pohledu rozhodovací funkce - V případě lineárního MAS, rozhodnutí, které učiní agent je proporcionální k vnímaným parametrům. Opačným typem je nelineární MAS [36]
- Z pohledu heterogenity - Homogenní (všichni agenti mají stejné charakteristiky a funkcionality) nebo heterogenní.[37]
- Z pohledu parametrů dohody - V nějakých aplikací se musí dohodnout na parametrech. Základní dělení je na základě jednoho, dvou či více parametrů. Parametry může být například rychlost, pozice, ...[38]
- Z pohledu mobility - Agenti mohou být statičtí či dynamičtí. Statičtí jsou umístěny na jedné pozici v prostředí, zatímce dynamičtí se pohybují po vymezeném prostoru [39]

4.1 Agenti

V rámci různých zdrojů a literatur nalezneme mnoho definic pro definování agentů podle toho, v jaké technické oblasti se s nimi můžeme setkat. Jelikož se s MAS lze setkat v různých odvětvích, rozeberu v této části jednu z více obecnějších:

"Agent: Subjekt, který je umístěn v prostředí snímá různé parametry, které jsou využity v rámci rozhodování, na základě konkrétních cílů subjektu. Subjekt na základě těchto rozhodnutí provede potřebnou akci v tomto prostředí."[5]

- Subjekt - Subjektem je v této definici zamýšlen agent. Softwarový, hardwarový nebo kombinace.[5]
- Prostředí - Odkazuje na místo, kde je agent umístěn. Prostředí má tyto klíčové vlastnosti:
  - Přístupnost - Jak přesně je agent schopen vnímat a sbírat data z prostředí pro své rozhodování.[5]
  - Předvídatelnost - Míra předvídatelnosti výsledků jednotlivých činností. V prostředí, které je předvídatelné, agent dokáže přesně znát status nadcházející činnosti[5]
  - Dynamičnost - Ta souvisí se změnami, které se stanou v prostředí bez spojitosti s činností agenta. Prostředí, které se mění pouze s činností agenta se nazývá statické.[5]
  - Kontinuita - Prostředí může být buď spojité nebo diskrétní. Spojité prostředí se dá popsat pomocí spojité funkce - agent pohybující se ve fyzickém prostředí. V rámci diskrétního prostředí agent vstupuje do předem určených stavů.[5]
- Parametry - To odkazuje na různé typy dat, které může agent přijímat. Například pozice, rychlost, vzdálenost,...[5]
- Akce - Jednotliví agenti vykonávají různé akce, které ovlivňují prostředí. Např. robot pohne předmětem z bodu A do bodu B. [5]

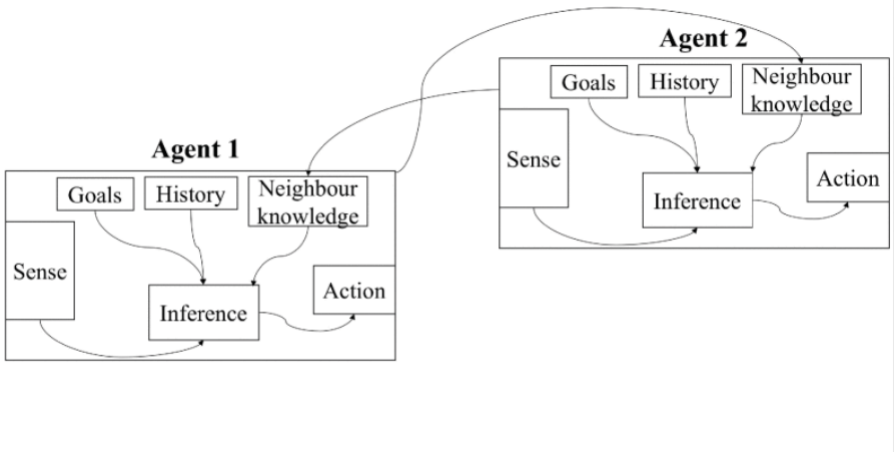
Pro jednotlivé agenty je klíčové dosažení požadovaných cílů. Pro dosažení svých cílů agent vnímá své prostředí a na základě toho vykonává své akce. Agent může využít také znalostí ostatních agentů.[5]

Díky těmto vlastnostem jsou MAS jsou schopny vykonávat komplexní úkony:

- Socializace - Agenti sdílí data mezi sebou, to pomáhá celkovému výkonu. [5]
- Autonomnost - Každý agent vykonává nezávisle svá rozhodnutí. [5]
- Proaktivita - Agenti využívají různých zdrojů dat (okolí, senzory, hisotrie) pro predikování situací pro vylepšení jejich celkové efektivity.[5]

4.1.1 Architektura agenta

Architektura se vyznačuje vícevrstevnatostí, v případě agentních systémů jde o tělo/obal. Obal agenta zodpovídá za oblast plánování. Tato část se také stará o sociální interakce. Obal tvoří komunikační vrstva a model sociálního chování. Pokud jde o tělo agenta, to nepřijímá žádné informace o komunitě. Na obrázku 4.1 vidíme strukturu agenta v prostředí. Znalosti sousedícího agenta, historie předchozích vykonaných akcí společně s cíli a vnímáním prostředí se zpracuje v Inference (z angličtiny závěr), na základě toho agent vykoná akci. [5]



Obrázek 4.1: Struktura agenta[5]

Kapitola 5

Kooperace a koordinace multiagentních (autonomních) systémů

Pro dosahování efektivních výsledků v autonomních systémech je zapotřebí metod kooperace a koordinace, které zajistí harmonický a koherentní průběh jednotlivých agentů. I přes nutnost určitého omezení chování agenta (to spočívá v nutnosti zapojení se do celkového systému), přináší koordinace a kooperace s ostatními agenty výhody, které by pro něj jinak nebyly přístupné.[40] Mezi hlavní důvody, proč systém musí být koordinovaný patří:

- Agent musí brát v úvahu závislosti mezi jednotlivými úkony agentů. V případě, že každý agent má daný svůj úkon, ve kterém však existuje vzájemná závislost s úkonem jiného agenta, je důležité, aby na to jednotliví členové brali zřetel. Díky tomu jsme schopni v multiagentních systémech zabránit kolizím nebo zrychlit jednotlivé akce. Může se jednat například o případ, kdy pro vykonání činnosti jednoho agenta je zapotřebí, aby jiný agent dokončil svůj úkon. Agenti také mohou vykonávat své činnosti paralelně, poté je nezbytná jejich koordinace.[41]
- Koordinovanost MAS je důležitá, když jednotliví agenti potřebují navzájem sdílet výpočetní zdroje, různé kompetence nebo informace pro dokončení svých úkolů[41]

Existuje mnoho způsobů, jakými se dá nahlížet na koordinaci MAS. Jako nejzásadnější rozdělení je vnímáno takové, které bere v potaz metodu kooperujících agentů proti metodě kompetitivních (konfliktních) agentů. Další důležité dělení je takové, které rozlišuje koordinaci pro uzavřený systém a koordinaci

pro otevřený systém, ten vnímáme jako takový systém, do kterého mohou autonomně vstoupit a zároveň ze kterého mohou odejít různí heterogenní agenti. V takovém rozsáhlém systému se stírají rozdíly mezi kooperativními a kompetitivními agenty.[41][42]

5.1 Metoda kooperace

Základním principem pro řešení úloh v distribuovaném systému je rozdělit problém na menší části, které mohou být vzájemně logicky propojeny. Řešení těchto částí se následně distribuuje mezi jednotlivé autonomní jednotky dle jejich aktuálního stavu, který závisí mimo jiné například na požadovaných zdrojích nebo jejich výrobních kapacit. Celý mechanismus je postaven na zajištění vzájemné soudržnosti bez nutnosti centrálního řízení. Toto se řídí za pomoci zasílání zpráv mezi jednotlivými prvky. Hlavním předpokladem pro tuto metodu je, že různé autonomní jednotky (agenti) jsou vůči sobě benevolentní, tedy jsou schopni vzájemné kooperace a toho využívají pro dosažení společných cílů. Dle N.Jennings [43] si lze tuto metodu představit jako metodu závazků a z toho vyplývajících sociálních konvencí. Závazky jsou zde vnímány jako jisté činnosti, které agent musí vykonat pro vykonání konkrétního úkonu, zatímco konvence představují podmínky, dle kterých si agent vybírá konkrétní závazky. Závazky jsou pro agenty důležité z hlediska plánování dalších činností a zároveň pro predikci činností ostatních agentů.[41]

5.1.1 Contract Net Protocol

Protokol používaný pro realizaci kooperace se zabývá zprávami, které si mezi sebou jednotliví agenti pošlou. Jedná se o vysokoúrovňový protokol. Samotné rozdělení rolí, tedy který agent bude plnit roli manažerskou a který řešitelskou protokol nezajišťuje. To vychází z úkolů jednotlivých agentů, kdy jeden agent může vykonávat obě role zároveň v závislosti na jednotlivých úkolech a jejich vzájemnému propojení. Protokol funguje na principu posloupnostech vyjednávání, kdy agent s manažerskou rolí je informován o nečinných agentech. Nečinný agent dostane informaci o nepřidělených úkolech. Na základě svých kompetencí vybere úkony pro něj vhodné a zašle nabídku manažerskému agentu. Ten ze získaných nabídek vybere nejvhodnější a ostatní odmítne. Agent, který řeší daný úkol, se na základě vlastní analýzy rozhodne pro vykonání úkolu sám nebo s dopomocí dalších agentů. V tom případě by takový agent přijal dvojí roli, jak manažerskou, tak řešitelskou.[44][45]

5.2 Kompetitivní metoda

Softwarové jednotky spolu potřebují interagovat i z jiných důvodů, než pro účely vzájemné spolupráce, díky které jsou schopny vytvořit jeden efektivní systém. Lze si to představit na příkladu internetové komerce, kdy jednotliví agenti zastupují vlastní zájmy, tedy nespolupracují spolu pro dosažení společných cílů. Zájmy jednotlivých agentů mohou jít do konfliktu s ostatními jednotkami a proto je důležité, aby byl systém více odolnější vůči částečným poruchám jednotlivých komponent. Kooperace mezi jednotlivými agenty, kteří nemají společný cíl, se nazývá kompetitivnost. Interakce mezi agenty probíhá pomocí tzv. vyjednávání, kdy výsledkem je kontrakt, tedy dohoda. Celý tento proces je založen na protokolu, který stanovuje jednotlivým agentům pravidla pro vzájemnou výměnu strukturovaných zpráv v podobě pevného síťového rámce. Pro tuto metodu se také využívá Contract Net Protocolu, který je však upraven tak, aby byl schopen pracovat s agenty, kteří nemají společný cíl.[41]

Klíčovým bodem výzkumů je zdokonalování procesu vyjednávání mezi kompetitivními agenty tak, aby došlo k co nejlepším vlastnostem celkového systému. Jelikož agenti sledují pouze své vlastní cíle a plánují nejlepší strategie pro jejich naplnění. Pro řešení tohoto problému se často vychází z teorie her. Teorie her je pojem označující část aplikované matematiky zabývající se konfliktními situaci, ve kterých je zapotřebí rozhodování. Pro teorii her jsou typické matematické modely snažící se analyzovat konfliktní situace. Tyto modely zároveň nabízí co nejlepší strategie pro jednotlivé strany. Jedna z variant, vycházející z článku týmu okolo Pablo Gómez Esteban, vychází z předpokladu, že po nějakém čase, co spolu kompetitivní agenti kooperují, jejich přístup se může vyvinout kooperativně vzhledem k ostatním agentům a obráceně. Takové jednání závisí však na jejich cílech, které potřebuje splnit. Toto je modelováno pomocí ARA metody (Adversarial Risk Analysis, v překladu Analýza rizik protivníka).[46]

Jiný pohled pracuje s jazykem pro komunikaci mezi agenty (Agent Communication Language, ACL), díky kterému jsou jednotliví agenti schopni mezi sebou komunikovat a zároveň pochopit chování ostatních agentů. Tento model je vydaný nadací FIPA.[41]

Kapitola 6

Aplikace autonomních systémů

Využití autonomních systémů spadá do více sekcí technických oborů, velmi rozsáhle zasahuje do oblasti počítačových sítí, do robotiky, nalezne uplatnění také v rámci městské infrastruktury nebo například v modelování komplexních systémů[5].

6.1 MAS v rámci počítačových sítí

Se stále narůstajícím vývojem nových počítačových technologií, které jsou čím dál tím více propojovány s internetovou sítí, počítačová síť se zesložiťuje a je tak komplexnější. Pro překonání této složitosti se ve velké míře využívá právě multiagentních systémů[5].

6.1.1 Cloud computing

Služby cloud computingu jsou obvykle poskytovány skrze internet. Cloud computing využívá jako jednu ze svých hlavních technologií virtualizací, tedy jak již bylo zmíněno, fyzické zařízení je v rámci virtualizace sdíleno mezi více uživateli, jako tzv Virtuální zařízení. Z toho plyne komplexnost, kdy hlavními potřebami je správa cloudových zdrojů, komunikace a také účtování využití výpočetních zdrojů jednotlivými uživateli. Za pomoci MAS je v cloudu navržen

rámec pro provádění souboru nezávislých úkolů, tzv. Bag-of-Tasks (BoT). Skupina agentů tak sbírá informace o dostupných zdrojích a poskytovatelech cloudu pro každý jeden úkol v BoT. Agenti přiřadí každému zákazníkovi odpovídajícího poskytovatele cloudu a následně probíhá vyjednávání o zdrojích nabídnutých zákazníkovi a o ceně, která by měla být zaplacena poskytovateli. Pro studování reálné efektivity MAS je využíváno simulací, které sledují přiřazování jednotlivých úkolů, čas po kterou je úkol prováděn a také cena z pohledu koncového uživatele.[47][48]

V rámci cloud computingu je MAS využíván pro zajištění bezpečnosti, automatický managment služeb, nebo také monitorování dostupných zdrojů.[5]

6.1.2 Internetová bezpečnost

Vzhledem k autonomitě MAS je ideální řešení využití agentů v internetové bezpečnosti, jelikož je schopen se učit a detekovat nové hrozby. Systém na detekování internetových hrozeb je navržen v [49]. Systém se skládá z celkem 5 agentů, kdy každý má svojí specifickou roli. Jedná se o agenta sběratelského, ten se sbírá data z internetového protokolu SNMP a routovacích tabulek. Tato data zašle detekujícímu agentovi. Ten využívá nástroj pro detekci poškozených či jinak neobvyklých paketů, vzešlé výsledky následně zašle rozhodovacímu agentovi, který rozhodne, zda se jedná o škodlivá data a případně vydá rozhodnutí, které zmírní jejich celkový dopad na systém. Toto rozhodnutí dodá mobilnímu agentovi, který zašle data reakčnímu agentovi. Ten učiní odpovídající rozhodnutí.[49]

MAS lze využít také v rámci ochrany vůči tabnabbing útokům, což jsou takové útoky, které jsou užívané v rámci webových služeb. Tento útok je veden za účelem vylákání přihlašovacích údajů (hesel) z uživatele. V tomto případě agenti monitorují otevřené záložky a kontrolují celkem 5 zvolených ukazatelů pro ověření pravosti stránek, například URL adresu, obrázky nebo text. Simulace které porovnávaly agentní metodu s metodami bez agentními, ukázaly větší přesnost agentních metod.[50]

6.1.3 Další využití MAS v rámci počítačových sítí

Mimo jiných metod lze MAS využít také v rámci sociálních sítí, které jsou velmi komplexní například z pohledu velkého množství uživatelů přicházejících a odcházejících ze sítě, zajišťování spojení mezi uživateli, atd. Spojení MAS se sociálními sítěmi lze využít například pro sběr dat o chování uživatele na síti. Na základě těchto nasbíraných dat je vytvořen profil uživatele, ten je vložen do predikčního systému, který je schopen předpovídat budoucí chování uživatele (like, sdílení, oblíbená témata,...). Dále lze MAS využít například v rámci trasování v rámci sítě, kdy MAS zajišťuje nalézaní cesty pro jednotlivé pakety od počáteční destinace do cílové, MAS pak sledují různé metriky, jako například takto sledování počtu tzv. hopů (jeden úsek cesty mezi odesílatelem a adresátem).[5]

6.2 MAS v rámci robotiky

Problematika v rámci robotiky se dá dle [51] popsat v rámci dvou klíčových problémů: kooperace a koordinace mezi roboty a plánování jejich trajektorie. Využití agentů v rámci robotiky se pak dá rozdělit do dvou kategorií. V jedné kategorii máme softwarové agenty starající se o rozhodující činnosti, plánování cesty, plnění jednotlivých úkolů, komunikaci a v té druhé hardwarové agenty odpovídající fyzickým součástem robota. Postup MAS v rámci robotiky je pak uveden na následujícím příkladě. Hardwarový agent využije senzoru, například kamery, aby zachytil obrázek. Agent starající se o komunikaci zašle tento snímek agentovi obstarávající zpracování obrázku. Po zpracování snímku nalezne překážky v okolí a zjistí svoji polohu. Tato informace je zaslaná rozhodovacímu agentovi, který určí nejkratší možnou cestu s co nejméně překážkami, kterým bude potřeba se vyhnout[51].

6.3 MAS pro modelování komplexních systémů

Díky flexibilitě, škálovatelnosti a autonomnosti multiagentních systémů je možné modelovat komplexní systémy za nižších cenových a zdrojových nákladů. Agent Based Modeling (ABM, modelování založené na multiagentním systému) využívá metologie založené na pravidlech (rule-based methodology, cílem této metodologie je nalézt v datech takové pravidelnosti, které lze vyjádřit pomocí IF-THEN podmínek). [52]

Výhody ABM oproti jiným metodám jsou:

- Možnost kombinování a nasazování na jiné výpočetní modely [52]
- Flexibilita jednotlivých předpokladů [52]
- Flexibilita získaných znalostí, ty se mohou měnit na základě pozorování prostředí [52]
- Možnost paralelního vykonávání zvyšuje efektivitu [52]
- Sledování havarijního chování[52]

Metody ABM je využíváno například v rámci [52] pro modelování zásobovacího řetězce. Zásadním v tomto řetězci je, že každý prvek je modelovaný samostatně, má své vlastní podmínky chování a je schopný definovat své interakce s ostatními subjekty. V této metodě se opět využívá rozdělení agentů do dvou stěžejních skupin, plánovací agenti a fyzičtí agenti. V tomto řetězci dodavatel a zákazník využívá 6 plánovacích agentů. Agent spravující poptávku zákazníka, agent řídící plánování zdrojů, který komunikuje s výrobci a objednává potřebné produkty. Agent analyzující poptávku na základě aktuálních a historických dat. Vrchní plánovací agent, který se stará o plánování celkové produkce. Agent plánující produkci, který pracuje s daty plánovacího agenta, tyto data transformuje na dílčí kousky. A na konec plánovací agent směrující výrobu na jednotlivé agenty. V rámci fyzických agentů pak operují 3 druhy agentů starající se o vykonávání fyzických úkolů, jako je výroba, přijímání a skladování materiálů atd. [52]

ABM lze také využít například pro chytrá města, kdy je schopné kontrolovat dopravní zácpy, znečištění, atd. [5]

6.4 MAS ve městech

V posledních letech vzniká poptávka po spravování měst za pomoci MAS, tento koncept je jinak nazývaný také jako smart cities. V rámci [53] Khayyat a Awasthi implementují metodu založenou na multiagentním systému pro organizaci distribuce nákladů. Metoda využívá 6 různých agentů. RFIDG agent využívá RFID tagů (identifikace založená na radiových frekvencích) pro správu dodávky zdrojů. Zákazník následně zašle požadavek nákupu zboží na retailer (maloobchodní) agenta nebo dodavatelského agenta. Dle obdrženého

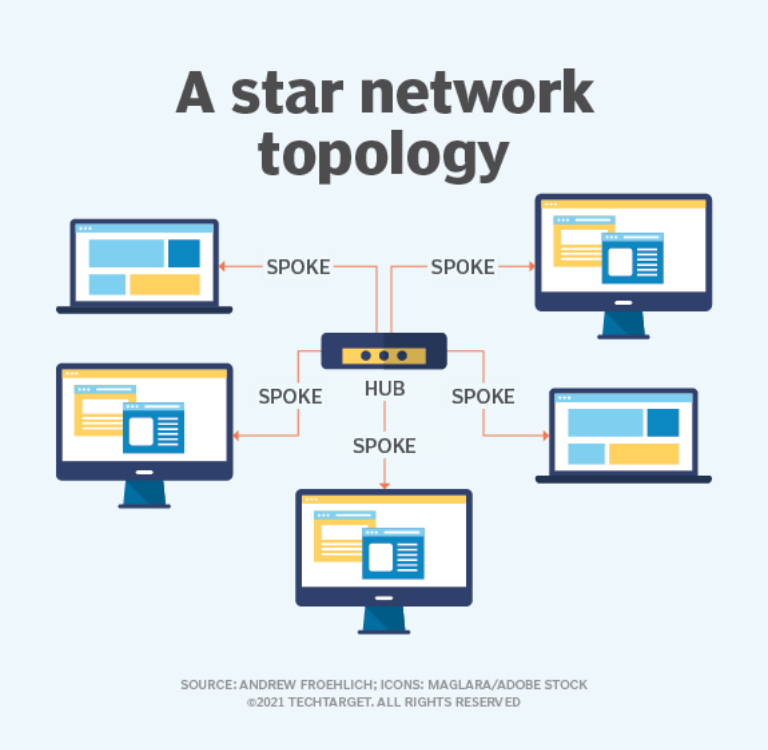
požadavku následně agenti prohledávají databázi s požadovaným zbožím. Následně je zboží zasláno dodavatelskému agentovi, který se stará o zaslání zboží zákazníkovi. Síťový agent určí optimální cestu dodavatelskému agentovi, ve které ideálně není dopravní zácpa. Administrativní agent následně upozorní obě strany dohody na pravidla a podmínky.[53]

Další využití MAS lze nalézt v kontrolování a spravování přepravního systému nebo pro správu budov, jako jsou například vyhřívací systémy. MAS můžeme také nalézt v inteligentních dodavatelských sítích, kdy se například agenti starají o balancování vyrobené a poptávané energie.[5]

Kapitola 7

Síťová komunikace

Má praktická část, zabývající se programováním úlohy pro umožnění odposlechu síťové komunikace, využívá prvků počítačové sítě, síťových protokolů a různých dalších počítačových principů. Počítačovou síť definujeme jako skupinu nebo systém navzájem propojených zařízení. Počítačová síť se skládá ze tří typů fyzických komponent. Prvním typem jsou koncová zařízení, kterými jsou myšleny přístroje na konci samotného řetězce, jako jsou servery, stolní počítače, tiskárny, ale také chytré mobilní telefony, bezpečnostní kamery nebo třeba inteligentní osvětlení. Druhým typem jsou síťová zařízení zprostředkovávající spojení s koncovými zařízeními. Těmito zařízeními mohou být například switche nebo routery. Poslední komponentou jsou kabely předávající elektrický signál, optická vlákna, radiové vlny atd. Důvod, proč je počítačová síť výhodná, lze vysvětlit na jejích klíčových vlastnostech. Pomocí ní jsme schopni zajistit sdílený přístup k výpočetním zdrojům, programům nebo datovým souborům. V případě, že bychom chtěli z tiskárny vytisknout dokument na našem počítači, a nevyužili jsme počítačové sítě, napojili bychom se kabelem z počítače přímo do tiskárny a bez potíží by se požadavek splnil. Problém by nastal v případě, kdybychom chtěli k tiskárně připojit další zařízení. V tom okamžiku by bylo nutné přepojit kabel z počítače na jiné zařízení, anebo pořídit další tiskárnu s každým dalším zařízením. [54]

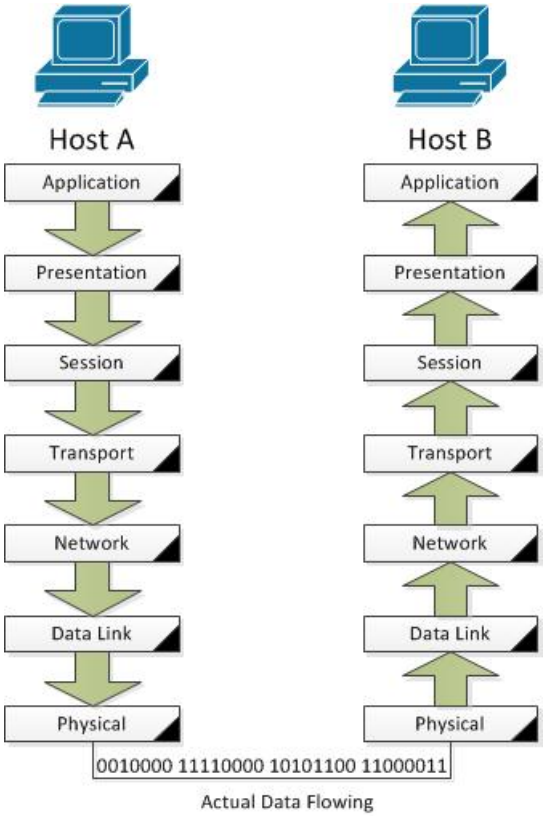


Obrázek 7.1: Zapojení do hvězdicové topologie[6]

Pro tento případ je tedy velice výhodné využít aktivních prvků počítačové sítě, jako jsou huby nebo switche, díky kterým budou moci jednotliví členové mezi sebou komunikovat a sdílet informace. Na obrázku 7.1 je uveden příklad zapojení počítačové sítě do hvězdicové topologie přes hub, kdy jednotlivá zařízení mohou mezi sebou komunikovat a sdílet výpočetní zdroje. Úkolem počítačové sítě je tedy realizace komunikace mezi jednotlivými zařízeními. Samotná komunikace se řídí síťovými protokoly, díky kterým jsme schopni usměrňovat komunikaci mezi jednotlivými prvky. [54]

7.1 Referenční model ISO/OSI

Open System Interconnection model (OSI) je teoretickým referenčním modelem mezinárodní organizace pro standardizaci (ISO), který vznikl za účelem vytvoření základních pravidel pro potřeby komunikace v počítačových a telekomunikačních sítích. V OSI modelu je komunikace mezi systémy rozdělena na sedm odlišných, myšlených vrstev, jak lze vidět na obrázku 7.2, od vrchní vrstvy to jsou: aplikační, prezentační, relační, transportní, síťová, spojová a fyzická. Každá vrstva OSI modelu má jasně definovaný rozsah funkcí a metod, jak jednotlivá vstva komunikuje se sousedícími vrstvami. Každá z těchto vrstev je schopna poskytovat rozsah svých funkcionalit do vrstev nad ní a být obsluhována vrstvou pod ní. Pokud bereme v úvahu komunikaci mezi vrstvami v rámci Hosta A (obrázek 7.2), například komunikace mezi vrstvou transportní a síťovou, tato komunikace se potom řídí pravidly rozhraní. V případě komunikace transportní vrstvy Hosta A a Hosta B se komunikace poté řídí protokoly.[55]



Obrázek 7.2: OSI Model [7]

7.1.1 Tok dat skrz OSI model

Pro to, aby data posílaná skrz počítačovou síť z jednoho zařízení na druhé byla pro člověka čitelná, musejí data poslaná ze zařízení projít všech sedm vrstev od vrchu. Od aplikační vrstvy po fyzickou. Tato data příjme fyzická vrstva přijímacího zařízení a putuje vzhůru vrstvami až do aplikační vrstvy příjemce, jak je uvedeno na obrázku 7.2.[55]

7.1.2 Aplikační vrstva

Tato vrstva je nejblíže koncovému uživateli. Je to jediná vrstva, která interaguje s daty poskytnutými uživatelem. Aplikační vrstva poskytuje uživateli síťové služby. V této vrstvě uživatel přímo interaguje se softwarem (systémem), který zajišťuje komunikaci mezi klientem a serverem. Tato vrstva se zabývá pouze takovými aplikacemi, pro které je vhodné zavést standardizovaný postup, například mechanismus přenosu e-mailu, webového prohlížeče, sdílení složek, zpracovávání zpráv nebo přístup k databázím. Do této vrstvy tedy nepatří rozhraní klienta, jehož design je na samotném výrobci. Nejběžnějšími protokoly v aplikační vrstvě jsou HTTP, FTP, DNS[56].

SNMP protokol

SNMP protokol (anglicky Simple Network Management protocol), který dále využívám v praktické části práce, byl vyvinut pro potřeby správy síťových zařízení, jako je například monitorování či upravování a konfigurování jednotlivých nastavení/statutu zařízení. SNMP je původně vyvinut pro switche a routery (viz 7.8), lze jej ovšem využít také pro tiskárny, IP kamery a další zařízení. SNMP protokol je založen na agentním systému, kdy si celý proces můžeme popsat na základě tří složek. Hlavní složkou je spravované zařízení, například ethernetový switch, agenti vykonávající úkony na spravovaném zařízení a nakonec správcovský počítač, ze kterého celý tento proces monitorujeme. Základním pojmem v rámci SNMP je OID (zkratka pro angl. Object Identifier, identifikátor objektu), který má hierarchickou strukturu oddělující jednotlivé uzly. Všechno, co lze v rámci zařízení monitorovat nebo upravovat, má svůj vlastní OID identifikátor. Některé OID jsou určené pouze pro čtení informací, u jiných lze také upravovat parametry. Typickým příkladem parametru, který lze pouze monitorovat může být teplotní senzor zařízení. OID identifikátor má formát String, kdy může mít například tuto formu:

'1.3.6.1.2', kdy tečky oddělují uzly, význam jednotlivých uzlů v rámci tohoto OID stromu by byl poté následující[57]:

- 1 : ISO - standard ze kterého se vychází
- 1.3 : Organizace spravující tento standard (ISO/IEC 6523)
- 1.3.6 : DoD
- 1.3.6.1 : Internet
- 1.3.6.1.2 : Management zařízení

Tyto OID identifikátory jsme schopni přeložit do textového významu díky MIB složce (Management information base), která se stará o tuto stromovou strukturu, jak byla naznačena výše. V rámci SNMP protokolu existují 3 verze, SNMP 1, SNMP 2c a SNMP3. Největším rozdílem v rámci verzí těchto protokolů je zabezpečení přenosu dat. Pro verze 1 a 2c pro získání těchto dat není zapotřebí zadávat žádné uživatelské jméno, stačí pouze community string, což představuje jakési heslo pro přístup k informacím. Data z těchto verzí protokolů nejsou zašifrovaná. V rámci SNMP3 systém již vyžaduje uživatelské jméno a heslo. Tato data jsou navíc šifrovaná[58].

V rámci SNMP jsou definovány funkce: GetRequest, GetNextRequest, GetBulkRequest, SetRequest, Traps. Get složí jako požadavek agentovi pro podání informace o proměnných v rámci spravovaného zařízením a jako odpověď se vrátí aktuální hodnota. Zatímco odpověď u GetNext navazuje na hodnotu předchozí proměnné. GetBulk navíc vnáší požadavek na více iterací požadavku GetNext. Funkce Set je používána pro změnu proměnné hodnoty v rámci zařízení[58].

7.1.3 Prezentační vrstva

Úkolem této vrstvy je zajištění formátování a překlad dat do takového tvaru, který je specifikovaný v aplikační vrstvě pro zašifrování odcházejících dat do nižší vrstvy OSI a samozřejmě také rozšifrování příchozích dat z nižších vrstev. To znamená, že tato vrstva překládá data z nejvyšší vrstvy tak, aby byla data čitelná pro vrstvy nižší a naopak. Z těchto důvodů jsou odchozí data zašifrována do specifického formátu, který je dán prezentační vrstvou. Pro rozšifrování dat se postupuje reverzním způsobem. Prezentační vrstva zahrnuje konverzi protokolů, zašifrování/rozšifrování dat, kompresi/dekompresi dat. Vrstva se nezabývá významem jednotlivých dat[55].

TLS protokol

TLS, z angličtiny Transport layer security (bezpečnost transportní vrstvy), je kryptografický protokol, který zajišťuje bezpečnou komunikaci v rámci internetu. Protokol TLS nahradil jeho předchůdce SSL. V rámci protokolu HTTPS se využívá protokolu TLS pro zašifrování dat. TLS protokol je založen na dvou fázích, handshake protokolu pro ověření autenticity a následně proběhne samotná šifrovací fáze[59].

7.1.4 Relační vrstva

Tato vrstva se stará o mechanismy pro vytváření, spravování a ukončování relací mezi koncovým uživatelem a procesy aplikací. Relací je myšlena doba, po kterou probíhá komunikace mezi těmito uzly. Komunikační relace je založena na požadavcích a zpětných vazbách mezi aplikacemi. Tato vrstva také rozhoduje o tom, zda se bude jednat o spojení duplexní nebo poloduplexní. U duplexního spojení oba uzly přijímají a zároveň vysílají data. U poloduplexní se tyto úkony střídají, kdy jeden uzel data přijímá, druhý vysílá a naopak. Příkladem komunikačního protokolu je X225 nebo ISO 8327. Tento protokol se stará o to, aby v případě ztráty spojení se pokusil o navázání tohoto spojení. V případě, že toto spojení není delší dobu používáno, tento protokol ukončí spojení a znovu ho obnoví. Základní jednotkou relační vrstvy je relační paket vkládající se do transportního paketu[56].

7.1.5 Transportní vrstva

Transportní vrstva se stará o rozdělení jednotlivých odeslaných dat do paketů které následně převezme síťová vrstva a pošle je příjemci. V opačném případě se stará o sestavení packetů dohromady. Tento protokol je zodpovědný za spolehlivost, za spojovanou komunikaci (anglicky connection-oriented communication). Spojovanou komunikací se rozumí komunikace, se kterou je navázána relace (viz relační vrstva) před tím, než proběhne samotný přenos dat[60].

Úkoly transportní vrstvy

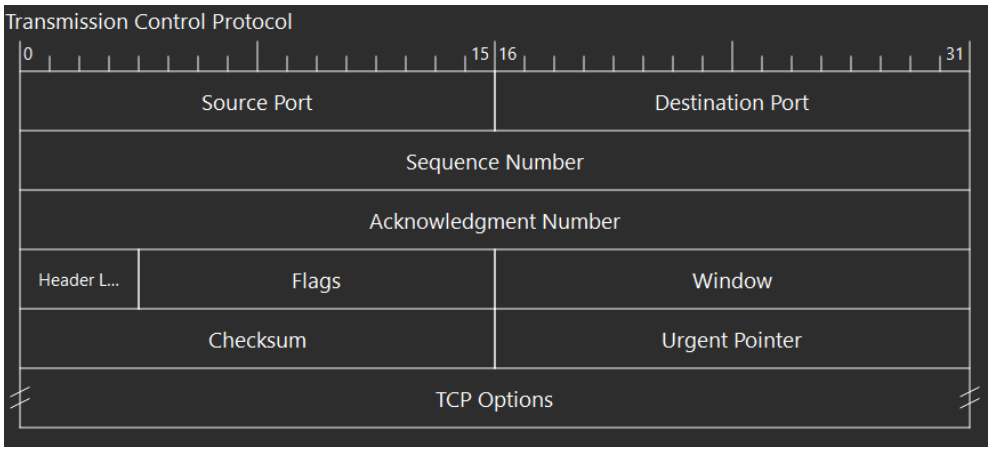
- Proces doručování: Pro správné doručení segmentů dat, transportní vrstva vyžaduje síťový port, síťový port je 16 bitová adresa, nesoucí čísla od 0 do 65535. Známé porty jsou od 0 do 1023. Pro HTTP protokol je to například 443, pro SMTP 25 nebo pro SNMP 161. Registrované porty od 1024 do 49151, kdy registrace podléhá ICANN. Porty v rozsahu od 49152 do 65535 nejsou pevně přiděleny a jsou definovány jako dynamické[60].
- Multiplexování a demultiplexování: K procesu multiplexování dochází v okamžiku, kdy jsou data obdržená z více probíhajících procesů od odesílatele. Procesy jsou rozlišovány dle síťových portů. Tyto analogové či digitální datové toky jsou následně zkombinovány do jednoho signálu pro vyšší efektivitu. Demultiplexování je reverzní proces, kdy u příjemce datový tok ze síťové vrstvy distribuuje do více procesů[60].
- End-to-end (konec-konec) spojení mezi systémy: Jedná se o takové propojení, kdy veškeré činnosti a operace se vykonávají v koncových bodech komunikační sítě nebo co možná nejblíže zařízení, které chceme obsluhovat[60].
- Transportní vrstva je zodpovědná za kontrolu toku dat. Jedná se o proces spravování míry toku dat mezi dvěma uzly k zabránění přesycení pomalého příjemce rychlým odesílatelem. Jelikož síťová vrstva stanovuje maximální velikost packetu (MTU), který může být poslán během jedné transakce síťové vrstvy, je potřeba rozmělnit velká data z protokolů nebo dlouhá proudění dat do menších celků nazývaných segmenty. Obvyklou hodnotou MTU je 1500 bytů, kterou nalezneme například u dat posílaných technologií Ethernet[60].

TCP protokol

Transmission control protokol je jedním z hlavních protokolů pro fungování internetu. TCP dává do popředí spolehlivost vůči latenci, pořadí jednotlivých úkonů je důležité. Jedná se o protokol založený na spojové komunikaci. Z čehož vyplývá, že je nutné navázat spojení před samotným odesláním. Za tímto účelem využívá trojcestný handshaking. Příjemce i odesílatel si během navazování spojení zvolí náhodně zvolí své pořadové číslo. Od tohoto čísla zvětšeného o 1 následně čísluje odesílané byty. První datagram odeslaný jednotlivými stranami má příznak SYN. Navázání spojení probíhá ve 3 krocích. Zatímco IP se stará o technologii posílání packetů, TCP zajišťuje výměnu dat mezi zařízeními v síti. Pro zajištění, že jednotlivé packety se dostanou k příjemci neporušeny, TCP/IP model, fragmentuje jednotlivá data. Například pokud se uživatel chce připojit na webovou stránku, daný server kdekoliv na světě tento požadavek zpracuje a uživateli zašle zpátky HTML stránku. Daný server k tomuto úkonu využije HTTP protokol (Aplikační vrstva). HTTP požádá TCP o navázání spojení a pošle HTML soubor. TCP tato HTML data fragmentuje na malé packety a pošle do mezisíťové vrstvy, která se následně postará o trasování packetu. TCP uživatele následně počká na dokončení přenosu dat a jakmile získá všechny packety, potvrdí přenos[61].

Na obrázku 7.3 je TCP datagram, který obsahuje zvrchu tato pole:

- Zdrojový a cílový port, 16 bitová pole
- Pořadové číslo, 32 bitové pole, ukazuje, kolik dat bylo posláno během doby trvání TCP.
- Potvrzovací číslo, 32 bitové pole, využíváno příjemcem pro požadavek dalšího segmentu.
- Offsetové pole, 4 bitové pole, určuje velikost TCP hlavičky
- Po offetovém poli následuje rezerované pole, které není vyobrazeno na obrázku 5.9. Tomu vždy náleží 0.
- Příznaky, 9 bitové pole, někdy také kontrolní bity. Ty jsou využívány pro zajištění a ukončení spojení, posílání dat. Jedná se například o SYN, URG nebo ACK
- Okénko, 16 bitové pole, specifikující kolik bitů příjme příjemce. Příjemce tak může odesílateli říct, může přijmout více dat, než aktuálně přijímá.
- Kontrolní součet, 16 bitové pole, kontrolující TCP hlavičku
- Urgent pointer, 16 bitové pole, určující kde končí urgent data v případě, že byl URG bit nastaven.
- TCP volby, toto pole je volitelné v rozsahu 0 až 320 bitů.[62]



Obrázek 7.3: TCP Datagram

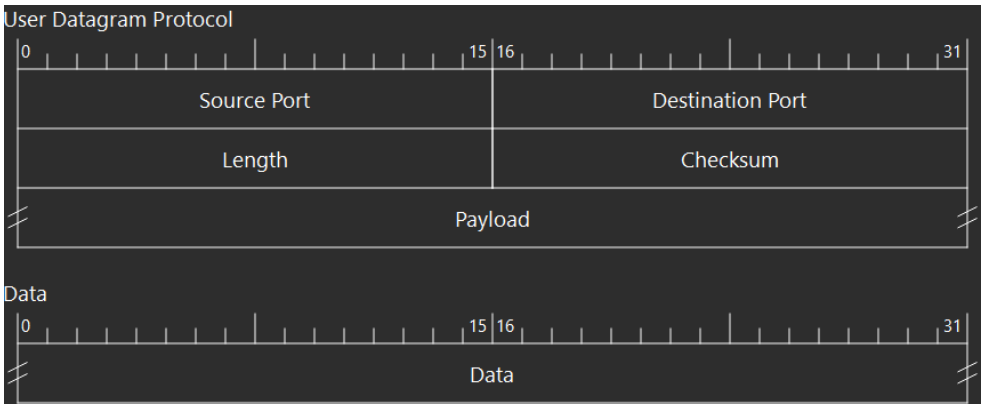
Vlastnosti TCP:

- Číslování segmentů: TCP sleduje posílání/přijímání segmentů a přiřazuje jim na základě toho specifická čísla.
- Spojová komunikace
- Duplexní spojení - při tomto spojení lze posílat data oběma směry současně.
- Kontrola toku dat - to má za úkol limitovat míru toku dat pro zajištění spolehlivosti.
- Kontrola chybovosti - pro bezpečné posílání dat, TCP má kontrolní mechanismy kontrolující jednotlivé segmenty.
- Kontrola přetížení[61]

UDP Protokol

Hlavním rozdílem UDP oproti TCP je, že nám nedává záruky doručení a komunikace je vedená nespojovaně. Není proto potřeba před odesíláním dat navázat spojení. UDP jako takové je vhodné pro aplikace, kde vyžadujeme nízkou latenci nebo tam, kde tolerujeme komunikační ztráty. UDP protokol nezajišťuje kontrolu chyb. Místo zpracovávání zpožděných packetů dovoluje UDP jejich zahození. Toho je využíváno při audio nebo video komunikacích[63].

Hlavička UDP je 8 bytová, kdy oproti TCP hlavičce, 20-60 bytové přenáší výrazně méně informací. Těchto 8 bytů přenáší nezbytné informace, zbytek náleží datům, jak lze vidět na obrázku 7.4. Zdrojový port o velikosti 2 bytů zde slouží pro identifikaci adresy portu odesílatele. Jelikož v rámci UDP protokolu nemusí odesílatel očekávat reakci od příjemce, zdrojový port nemusí být použit. Pokud není, je nastaven na nulu. Cílový port je také o velikosti 2 bytů, avšak slouží k identifikaci portu příjemce. Pole délky specifikuje velikost hlavičky v bytech, kdy minimální hodnota je 8 bytů. Kontrolní součet o velikosti 2 bytů pokrývá hlavičku i data. Slouží pro kontrolu chyb. Pro IPv4 je toto pole volitelné, povinné pro většinu případů IPv6. Pokud toto pole není využito, je vyplňeno nulami[64].



Obrázek 7.4: UDP Datagram

Výhody UDP:

- Rychlost - jelikož se UDP nestará o zajištění spojení a o spolehlivost dat, je výrazně rychlejší než TCP
- Nízká latence
- Přenos - UDP podporuje přenos více příjemcům, toho se dá využít například u video hovorů, on-line her
- Menší velikost dat oproti TCP
- Jednoduchost - Oproti TCP, UDP protokol má jednodušší provedení. Díky tomu je snazší ho implementovat.[63]

Nevýhody tohoto protokolu částečně vycházejí z jeho výhod. Nespolehlivost protokolu může vést ke ztrátě nebo duplicitě dat. Protokol nekontroluje přetížení, může tedy posílat data v takové míře, že způsobí přetížení sítě. Zahltit může i příjemce, jelikož nedisponuje kontrolou toku dat. UDP protokol je náchylný k DOS útokům. Kvůli těmto nevýhodám má UDP protokol menší šířku využití a není vhodný pro většinu aplikací, jako například posílání emailu, kde vyžadujeme spolehlivost[63].

7.1.6 Síťová vrstva

Účelem síťové vrstvy je přenos dat z jednoho zdroje do druhého umístěných v odlišných sítích. Data z vyšších vrstev OSI modelu odpouzdří do datagramů (které také nazýváme packety) s hlavičkou síťové vrstvy[65].

Síťová vrstva má na starosti routing (trasování) jednotlivých packetů. Posílá data skrz napříč propojenými zařízeními (routery). Funkcí jednotlivých zařízení a softwaru je řídit příchozí packety z více zdrojů, zjistit jejich cílový bod a následně poslat data nejkratší cestou k dalšímu bodu tak, aby nakonec data dorazila k příjemci[65].

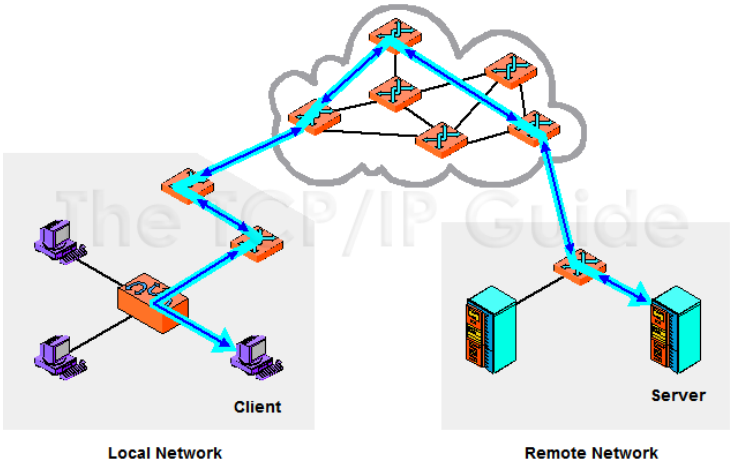
Síťová vrstva nabízí jak nespojovanou (connectionless communication), tak spojovanou (connection-oriented) komunikaci pro doručování packetů napříč počítačovou sítí. Nespojovaná komunikace je nejběžnější v síťové vrstvě. Při tomto druhu komunikace se datové packety přemísťují od odesílatele k příjemci bez toho, aniž by příjemce musel odeslat potvrzení k této činnosti[65].

Každé zařízení komunikující v rámci počítačové sítě má přiřazenou vlastní logickou adresu. Například v rámci internetu stanovuje IP protokol každému zařízení připojenému k síti IP adresu. Samotné adresování je tvořeno také v rámci spojové vrstvy, to se však vztahuje k místním fyzickým zařízením. Oproti tomu logická adresa je nezávislá na konkrétním hardwaru a musí být unikátní napříč celou sítí.[55]

Síťová vrstva musí posílat data do spojové vrstvy. Některé technologie spojové vrstvy mají však omezenou kapacitu velikosti přijímaných dat. Pokud packet, který odesíláme ze síťové vrstvy je příliš velký, síťová vrstva musí packet rozdělit na příslušný počet dílů a poslat jednotlivé díly do spojové vrstvy. Jakmile data dorazí k přijemci, musí síťová vrstva příjemce jednotlivé díly packetů spojit zpátky do jednoho.[65]

IP protokol

Základním principem Internetového protokolu je posílání datagramů z jednoho zařízení do druhého skrz internetovou síť, který lze také vidět na obrázku 7.5, kde např. klient odesílá datagramy do serveru skrz navzájem propojenou síť[8].



Obrázek 7.5: Základní princip IP[8]

Protokol je navržen pro přenos dat na libovolném typu sítě, která umí pracovat s TCP/IP protokolem. Umí tedy pracovat s protokoly nižší úrovně, např Ethernet. Velké bloky dat IP fragmentuje dle parametrů fyzické sítě na menší části, ty následně dokáže spojit dohromady. V rámci jedné sítě (LAN, WLAN, WAN) lze IP datagram poslat přímo. Ve většině případů však IP datagram musí být poslat nepřímo pomocí trasování (routing), to je zajištěno routery. IP protokol plní tyto funkce také za pomoci protokolu ICMP a TCP/IP trasovacích protokolů[8].

Hlavní charakteristikou IP je:

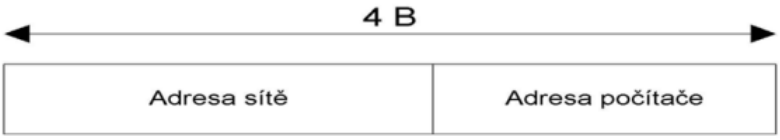
- Univerzální adresování
- Nespojovaná komunikace
- Nespolehlivost komunikace - IP nesleduje datagramy, které již vyslal
- Bez zpětné vazby - neposílá potvrzení o doručení dat[8]

IPv4. Čtvrtá verze IP používá 32 bitovou IP adresu, tato verze je stále nejpoužívanější napříč internetem. Unikátní adresa, která má za úkol jednoznačně identifikovat síťové rozhraní (Unicast). V případě, že má systém více síťových rozhraní, pak obvykle každé má svoji IP adresu. Zápis IP adresy se obvykle provádí do desítkové soustavy, např. 77.75.79.53. Lze se setkat také s binárním zápisem nebo hexadecimálním, ten je však většinově využíván pro zápis MAC adres.[66]

Nevýhody IPv4:

- Vyčerpání adresního prostoru [66]
- Není možné strukturovat IP adresu dle skutečné topologie sítě [66]
- Složitá konfigurace [66]
- Nevyžaduje mechanismus ověření proběhlé komunikace [66]

Na obrázku 7.6 lze vidět, že IP adresa je složena z části adresy lokální sítě a z části adresy zařízení v síti.



Obrázek 7.6: Složení IP adresy[9]

Síťová maska. Pro určení adresy sítě se využívá tzv. síťové masky, kdy nám určuje bity, které jsou určeny pro adresu sítě. Z toho principu se musí jednat také o 32 bitovou adresu. Pokud bychom síťovou masku rozepsali do binární soustavy, tak pro adresu sítě připadají jedničky, ostatní bity jsou nulové. Maska sítě pro adresu typu C má tvar 255.255.255.0. [9]

IPv6. IPv6 je sada protokolů, který řeší problémy aktuální verze IPv4 jako jsou vyčerpání adres, zabezpečení, automatická konfigurace atd. IPv6 adresy využívají 128 bitovou IP adresu, což umožňuje rozdělení adres odrážejících topologii internetu. Podoba IPv6 adresy se zapisuje v hexadecimálním zápisu, např. 2a02:2078:0105:1337:0000:0000:0000:0002.[66][9]

7.1.7 Spojová vrstva

Spojová vrstva je taková vrstva, kde spousta bezdrátových nebo drátem spojených technologií LAN (WLAN) využívá svých funkcí. Technologie jako jsou Ethernet, FDDI a Wi-Fi (803.11) jsou občas označovány jako technolgiemi spojové vrstvy[67]

Jednotlivé funkce spojové vrstvy tedy jsou:

- LLC: Řízení logického spoje se týká funkcí, které jsou požadované pro zajištění a řízení logických spojení mezi místními zařízeními v síti. Lze tedy říci, že LLC se stará o řízení toku dat, tak, aby odlišné technologie mohli mezi sebou bezproblémově komunikovat. Úkolem LLC je zároveň multiplexování, které se stará o to, aby se v jedné síti dalo využívat více síťových protokolů (IP,...). Nejběžněji využívaným protokolem v rámci LAN síťových technologií je IEEE 802.2 LLC protokol.
- MAC: MAC se stará o postupy, které zařízení používá k řízení přístupu síťových medií. Jelikož více sítí využívá sdíleného média (jako je síťový kabel), tak je nutné stanovit pravidla pro spravování media pro vyhnutí se konfliktům.
- Rámcování dat: Spojová vrstva je zodpovědná za zapouzdření dat z vyšších vrstvev do framů (rámec), které jsou odeslány do fyzické vrstvy.
- Adresování: Spojová vrstva je nejnižší vrstvou OSI modelu, která se stará o adresování. Ta označuje jednotlivá zařízení informacemi o jejich cílovém uzlu. Každé zařízení má svoji unikátní adresu, tzv. MAC adresu, která je využívána spojovou vrstvou pro zajištění toku dat ke správnému příjemci.
- Detekce a vypořádávání se s chybami: Spojová vrstva spravuje chyby, které se objeví v nižších vrstvě. Jedná se například o kontrolu cyklické redundance[67].

7.1.8 Fyzická vrstva

Základní funkcí fyzické vrstvy je skutečný přenos dat. Mohlo by se zdát, že funkcí fyzické vrstvy je pouze o síťovém hardwaru nebo že fyzická vrstva je o síťových kartách a kabelech. Fyzická vrstva ovšem definuje mnoho síťových funkcí, tedy není to pouze o hardwaru a softwaru. Zároveň však nelze definovat,

že hardware má pouze spojitost s fyzickou vrstvou, kdy například ethernetová síťová karta spadá jak pod fyzickou, tak pod spojovou vrstvu. Fyzická vrstva je tedy tou nejnižší vrstvou, která se stará o posílání binárních dat do sítě. Nejjednodušší zařízení, jako jsou opakovače nebo huby pracují na této vrstvě. Zařízení, jako je například ethernetový switch pracuje na vyšší vrstvě, jelikož pracuje na složitějším principu a více sleduje data, než z pohledu jedniček a nul.[55]

Funkce fyzické vrstvy:

- Definování hardwarové specifikace: Do této kategorie spadají specifikace provozu jednotlivých kabelů, konektorů, síťových karet atd. Funkce hardwarových zařízení spadají obecně do fyzické vrstvy.[55]
- Fyzická vrstva je zodpovědná za kódování a signalizaci, kdy mění poslaná data z bitů na signály, které mohou být poslány dále do sítě.[55]
- Přenos dat a příjem dat. Po zakódování jednotlivých dat, fyzická vrstva pošle data a reverzně také příjme bez rozdílu jestli jde o bezdrátové technologie nebo technologie vedené drátem.[55]
- Tato vrstva také stanovuje jednotlivé topologie sítí.[55]

7.2 OPC UA komunikace

Zkratka OPC UA komunikace, kterou využívám v rámci své praktické části, se skládá z anglického Open Platform Communication (otevřená komunikační platforma) a Unified Architecture (sjednocená architektura). Jedná se o tzv. machine-machine komunikační protokol, který se stará o komunikaci mezi jednotlivými zařízeními. Je založen na client-server modelu, kdy typicky v tomto vztahu server disponuje daty, která po něm žádá klient. Protokol OPC UA vychází z jeho předchůdce OPC založeného na technologii COM/DCOM. OPC UA je již však rozšířen mezi více OS hlavně díky využití protokolů TCP/IP a není tak vázaný na konkrétní OS. Hlavní záběr OPC UA je zamýšlen do průmyslové výroby (především PLC) a je proto výhodné jej využít pro sběr dat v rámci průmyslové sítě. Kdy například jedno PLC může fungovat jako server a druhé jako klient. Toto je omezeno softwarem jednotlivých PLC, kdy některé verze PLC mohou fungovat pouze jako server. OPC UA využívá binárního protokolu opc.tcp://Server a webového protokolu http://Server. O bezpečnost dat posílaných pomocí tohoto protokolu se starají procesy spojené s šifrováním, autorizací a autentizací elektronických podpisů.[68]

7.3 Ethernet

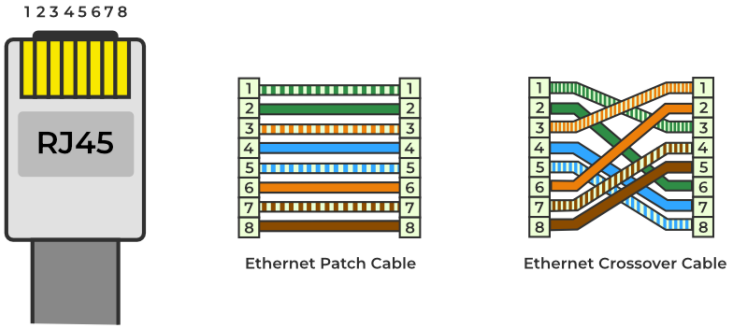
Ethernetem označujeme komunikační standard používaný zejména v LAN a MAN sítích. Technologie ethernetu byly vyvíjeny v mezi lety 1973 a 1974, komerčně představen byl až v raných osmdesátých letech v rámci standardu IEEE 802.3 ("The Ethernet, A Local Area Network. Data Link Layer and Physical Layer Specifications"), postupně nahradil LAN technologie jako Token Ring, ARCNET, FDDI. Ze standardu tedy vyplývá, že v rámci OSI modelu, Ethernet operuje v prvních dvou vrstvách, fyzické vrstvě a spojové vrstvě. PDU v rámci ethernetu je rámec (frame). Výhoda ethernetu spočívá v nízkých nákladech a velkých přenosových rychlostech, oproti využití technologie Wi-Fi. V rámci technologie ethernet je možné mezi sebou propojit až 1024 zařízení. Pro zabránění kolizím v rámci sdíleného přenosového media se využívá protokolu CSMA/CD.[69]

7.3.1 Přenosová média

V počátcích ethernetu se pro přenos dat využívalo koaxiálních kabelů, přenosové rychlosti u koaxiálních kabelů jsou 10 Mbit/s, což je pro dnešní dobu nedostačující. K celosvětovému rozšíření ethernetu přispěla technologie 10Base2[70].

Kroucená dvojlinka

Jak již z názvu vypovídá, kabel je tvořený barevně rozlišenými páry vodičů, které jsou pravidelně zakrouceny a zároveň jsou do sebe zakrouceny i samotné páry. Kroucením vodičů dosahujeme lepších elektrických vlastností. Minimalizujeme přeslechy mezi páry a zároveň omezujeme vyzařování elektromagnetického záření do okolí a jeho příjem z okolí (elektromagnetická indukce). Signál se vodiči vede symetricky. Pro počítačovou síť se běžně využívá 4 párů vodičů, které jsou zakončeny osmi pinovou koncovkou RJ45, jak lze vidět na obrázku 7.7. Kroucená dvojlinka se řadí do kategorií Cat 1 až Cat 8, kdy kabely vyšších kategorií (pro průmyslové užití) využívají tzv. stínění (STP) oproti kabelům z těch nižších (UTP). Stínění slouží k ochraně kabelu před vnějšími elektrickými poli, k tomu se využívá fóliový štít. Kroucené dvojlinky přenáší data v poloduplexně nebo plně duplexně. Nejrozšířenější rychlostí v rámci ethernetového rozhraní je aktuálně 10Gbit/s. [70][71]



Obrázek 7.7: RJ-45[10]

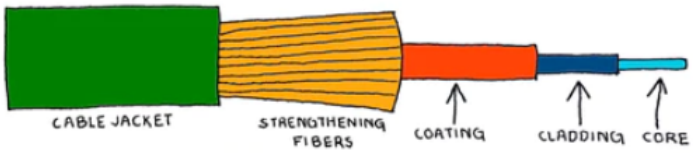
- Cat 6 - šířka pásma 250 MHz, přenosové rychlosti 1Gbit/s, maximální povolená délka kabelu 55 metrů[72]
- Cat 6a - šířka pásma 500MHz, přenosové rychlosti 10Gbit/s, max. 100m[72]
- Cat 7 - šířka pásma 600MHz, přenosové rychlosti 10Gbit/s, stíněný, max. 100m [72]
- Cat 8 - šířka pásma 2000MHz, přenosové rychlosti 40/20 Gbit/s, stíněný, max. 30m[72]

Optické vlákno

Zatímco kroucená dvojlinka se využívá pro menší vzdálenosti, kabely z optického vlákna jsou využívány pro vedení na větší vzdálenosti. Pro vzdálenosti do 2km se využívá mnohovidových vláken, pro větší vzdálenosti jednovidových. [73]

Pro posílání dat napříč kabelem z optického kabelu se nevyužívá elektrických signálů, jako tomu je u kroucené dvojlinky, ale využívá světelných pulsů. Důvodem, proč je výhodné využít světlo pro vedení internetového provozu je rychlost a délka jeho vedení, kdy můžeme aktuálně jednovidovým vláknem dosáhnout vzdáleností vedení až 100km, díky jeho nízkému tlumení (míra klesání signálu). U kroucených dvojlinek s přibývající délkou klesá síla elektrického signálu. Další výhodou optického vlákna je, že nám nevzniká elektromagnetická indukce, tedy ani žádné ztráty tím způsobené.[73]

Vícevidové vlákno. Na obrázku 7.8 vidíme řez vláknem, zprava lze vidět skleněné jádro, což je část, kterou se šíří světlo. U vícevidového vlákna je jádro 10x větší, než u jednovidového (50-60 mikronu). Následuje opláštění (sklo), které je u obou druhů vláken okolo 125 mikronů velké. Zajišťuje, aby se světlo šířilo jádrem. Paprsky světla jsou vysílány pod různým úhlem (index lomu) a odráží se od stěn jádra. Díky tomu paprsek se nešíří rychlostí světla, ale asi o 31 procent pomaleji. Ostatní části kabelu slouží pro zvýšení jeho odolnosti. Víceividová vlákna mají větší tlumení než vlákna jednovidová, přenos proto probíhá na vzdálenosti do 600m. Přenosové rychlosti se u vícevidového vlákna pohybují do 10Gbit/s.[74]



Obrázek 7.8: Řez vláknem[11]

Jednovidové vlákno. Velikost jádra jednovidového vlákna se pohybuje okolo 5-9 mikronů. Vzhledem k jeho velikosti se výrazně zvýší úhel odrazu v jádře, to vede k prodloužení dráhy paprsku a ten se šíří vláknem rovnoběžně (neodráží se od rozhraní). Díky tomu jsme schopni dosáhnout menšího tlumení a proto se jednovidová vlákna využívají pro přenos na velké vzdálenosti a to až 100km. Důvodem, proč se využívá jak jednovidového, tak vícevidového vlákna, je cena. Šířka pásma u jednovidových vláken je teoreticky neomezena. Nejvyšší přenosové rychlosti u komerčních jednovidových vláken jsou 100Gbit/s.[73]

7.4 Switch(přepínač)

Switch je víceportové zařízení v počítačové síti využívané jako centrální bod pro hvězdicovou topologii. Je to zařízení spojové vrstvy OSI modelu, pro posílání dat tedy pracuje s MAC adresy. Některé switche dokáží posílat data také v rámci síťové vrstvy OSI modelu, do nich se přidává trasovací funkce. Tyto switche se někdy nazývají jako Layer 3 switche. Switch je vnímán jako nástupce hubů, narozdíl od nich však rozesílá datové rámce pouze na zařízení, jehož MAC adresa je uvedena v hlavičce. Switch operuje v plně duplexním režimu. [9]

Switch funguje na principu přepojování zpráv, kdy rámec z jednoho rozhraní příjme, ten uloží do vyrovnávající paměti, prozkoumá hlavičku rámce, detekuje a opraví chyby a poté pošle rámec do odpovídajícího rozhraní na základě MAC adresy. Vysílání do cílového rozhraní probíhá dříve, než do switche dorazí celý rámec a to po přijetí prvních 64 bytů kdy ví, že nevznikla žádná kolize na daném segmentu za účelem minimalizace zpoždění.[9]

V případě, že switch nezná MAC adresu příjemce, pošle datový rámec do všech rozhraní (vyjma odesílatele), v tuto chvíli se chová jako hub. Jakmile odpovídající zařízení odpoví, tuto adresu si switch následně uloží a při dalším posílání rámců již zašle odpovídajícímu zařízení.[9]

V případě, že jsou v síti vytvořeny smyčky za účelem redundance, mohou data od jednoho odesílatele procházet různými rozhraními (i stejným switchem vícekrát) chaoticky. Proto je vytvořen protkol SPT (Spanning tree protokol), který stanoví které trasy se nebudou využívat, tím zmizí ze sítě smyčky.[9]

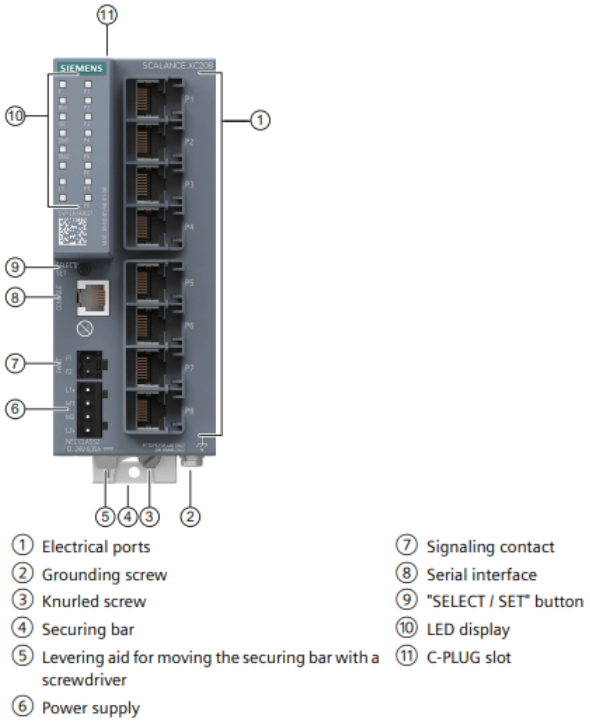
Část II

Praktická část práce

Kapitola 8

Odposlech komunikace v rámci switche XC208

V rámci praktické části mám za úkol naprogramovat úlohu, pomocí které bychom byli schopni odposlechnout komunikaci probíhající na switchi Scalance XC208. Na obrázku 8.1 můžeme vidět obrázek z dokumentace switche, který disponuje 8 ethernetovými porty, LED displayem pro zobrazování stavů jednotlivých portů, případně chybových stavů switche. Mimo napájecích portů a resetovacího tlačítka switche se na něm také nachází C-Plug sloužící k ukládání informací o nastavení zařízení a sériové rozhraní, které lze využít pro přístup k rozhraní příkazového řádku. Pro účely praktické části využiji pouze technologií spojenými s ehternetovými porty.



Obrázek 8.1: Scalance XC208[12]

Součástí switche Scalance XC208 je také IP adresa a webové rozhraní (WBM), pomocí kterého jsme schopni provést konfiguraci switche. V rámci tohoto ethernetového switche lze mimo jiné provést rozdělení sítě do různých VLAN, nastavit síťovou redundanci nebo v rámci webového rozhraní provést vyhledávání a konfiguraci PROFINET zařízení.

Pro účely praktické práce odposlechu jednotlivých portů využiji SNMP protokol.

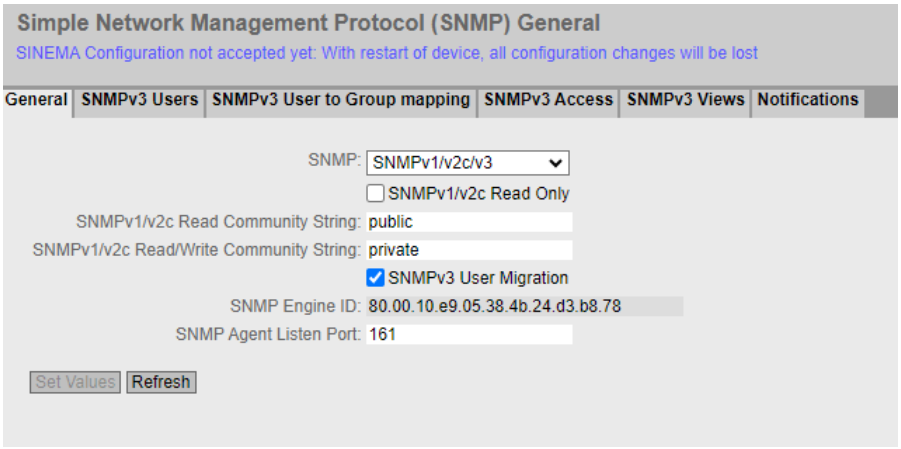
Jako klienta pro OPC UA komunikaci využiji SCADA software. Zkratka SCADA se skládá z anglických slov Supervisory Control And Data Acquisition, tedy v překladu systém pro kontrolu a sběr dat. SCADA tedy nemá řídicí funkci, ale spíše monitorovací[75].

V rámci SCADY (klient) bude probíhat komunikace pomocí OPC UA protokolu s PLC 1200 (server) ve kterém bude naprogramována úloha za pomocí SNMP knihovny, kdy PLC bude vznášet požadavky na agenty switche. Agenti podle typu požadavku následně adekvátně odpoví.

Mimo odposlechu síťové komunikace využiji SNMP protokol také k demonstraci funkcionalit jako je například monitorování teploty zařízení.

8.1 Nastavení SNMP v rámci WBM

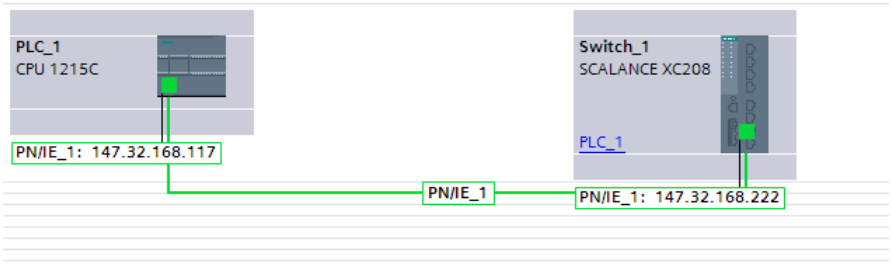
Pro pracování s SNMP je nejdříve potřeba zkontrolovat nastavení v rámci WBM, aby následná komunikace pomocí PLC proběhla v pořádku. Důležité informace z obrázku 8.2 jsou hodnoty Community string, jelikož pro účely programu v TIA portálu využívám SNMP verze v2c. Community string pro GET je public, pro SET private. Dále lze z obrázku vyčíst využití UDP portu 161.



Obrázek 8.2: Nastavení SNMP v WBM

8.2 Nastavení projektu v TIA portalu

Pro práci s PLC a switchem je využit TIA portal V18. Po přidání jednotlivých zařízení do TIA portalu je potřeba propojit zařízení pomocí PN/IE spojení (viz obrázek 8.3) stejně tak, jak jsou propojena ve skutečnosti. Následně do projektu nahraji SNMP knihovnu se kterou nadále budu pracovat.



Obrázek 8.3: Spojení PLC a Switche v TIA portalu

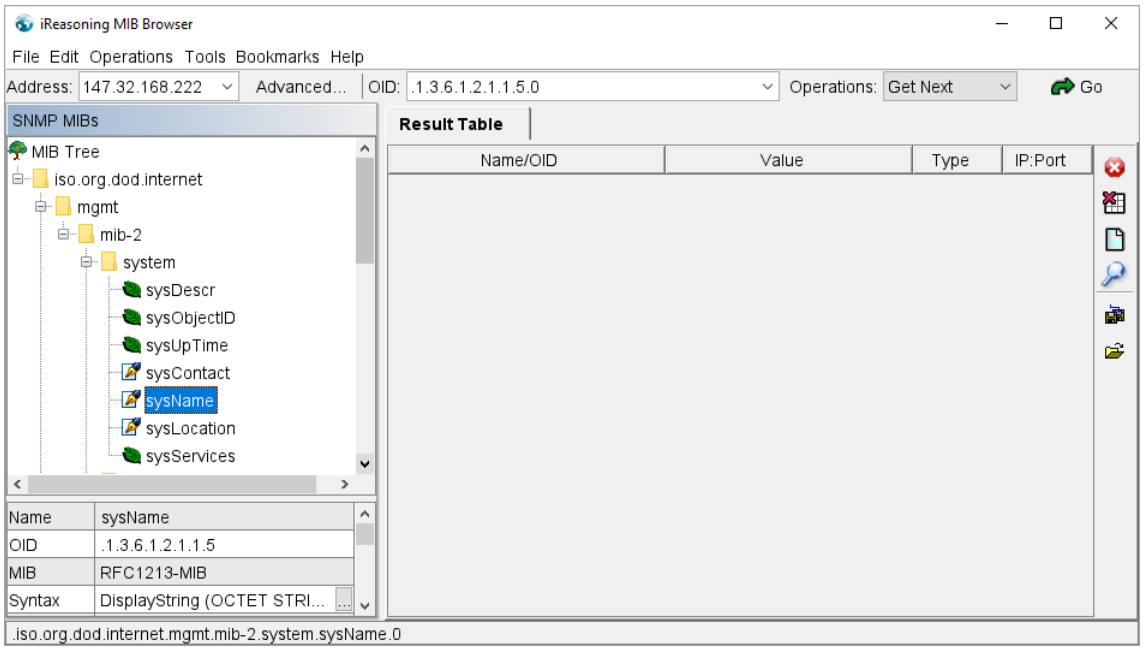
8.3 MIB Browser

Pro vyhledávání jednotlivých OID identifikátorů pro komunikaci s agentem switche je zapotřebí využít MIB browser. Pro tento účel jsem využil volně dostupný software iReasoning MIB Browser. Pro vyhledávání v tomto prohlížeči je nutné znát IP adresu switche a mít staženou jeho MIB složku. Ta lze stáhnout z WBM switche, viz obrázek 8.4.

LoginWelcomeMessage	Login Welcome Message	Load	Save	Delete
MIB	SCALANCE X200 MSPS MIB		Save	
RunningCLI	'show running-config all' CLI settings		Save	

Obrázek 8.4: WBM MIB složka

Následně lze vyhledávat ve stromové struktuře MIB Browseru potřebné OID (obrázek 8.5).



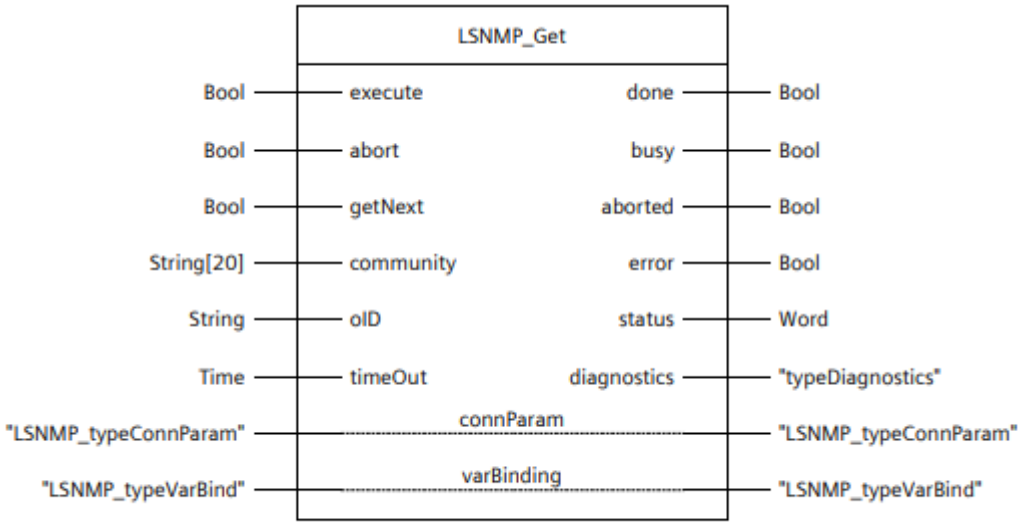
Obrázek 8.5: WBM Browser

8.4 Aplikace SNMP

8.4.1 Aplikace SNMP pro monitoring teploty/ typu zařízení

Z důvodu jednodušší aplikace nejdříve popíšu postup programu pro monitoring teploty zařízení/typu zařízení. V další sekci se následně budu věnovat nastavení switche pro odposlech jednotlivých portů.

Pro účely monitorování využiji funkčního bloku SNMP GET, viz obrázek 8.6.



Obrázek 8.6: Funkční blok GET[13]

Pro aplikaci knihovny je nejdříve zapotřebí definovat proměnné, se kterými budu nadále pracovat (obrázek 8.7). U všech dalších aplikací knihoven bude vypadat seznam proměnných obdobně, jen se bude lišit v závislosti na konkrétním funkčním bloku.

Static								
FB_CALL_GET	"LSNMP_Get"			☑	☑	☑	☐	Requests information for a specified OID (
POM	Struct		Non-retain	☑	☑	☑	☐	
IN	Struct		Non-retain	☑	☑	☑	☐	
execute	Bool	false	Non-retain	☑	☑	☑	☐	
abort	Bool	false	Non-retain	☑	☑	☑	☐	
getNext	Bool	false	Non-retain	☑	☑	☑	☐	
community	String[20]	''	Non-retain	☑	☑	☑	☐	
oID	String	''	Non-retain	☑	☑	☑	☐	
timeOut	Time	T#0ms	Non-retain	☑	☑	☑	☐	
InOut	Struct		Non-ret...	☑	☑	☑	☐	
connParam	"LSNMP_typeConnP...		Non-retain	☑	☑	☑	☐	Describes the connection parameters for
varBinding	"LSNMP_typeVarBin...		Non-retain	☑	☑	☑	☑	Describes a tag binding to be sent or rec
Out	Struct		Non-retain	☑	☑	☑	☐	
done	Bool	false	Non-retain	☑	☑	☑	☐	
busy	Bool	false	Non-retain	☑	☑	☑	☐	
aborted	Bool	false	Non-retain	☑	☑	☑	☐	
error	Bool	false	Non-retain	☑	☑	☑	☐	
status	Word	16#0	Non-retain	☑	☑	☑	☐	
diagnostics	"typeDiagnostics"		Non-retain	☑	☑	☑	☐	Provides a structure for detailed diagnost
TempVal	Int	0	Non-retain	☑	☑	☑	☐	

Obrázek 8.7: Definované proměnné

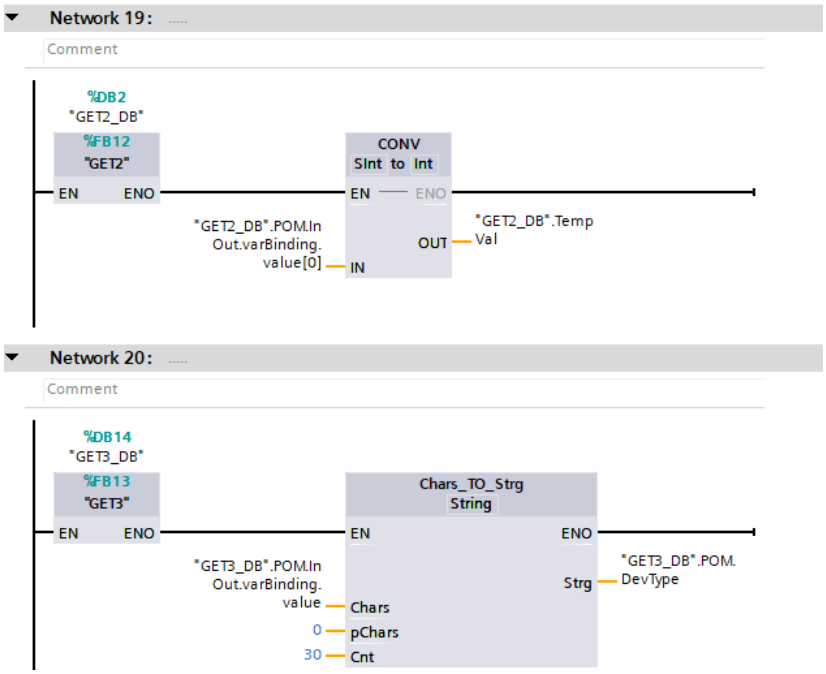
V první části kódu 8.1 nastavuji parametry připojení, kdy connID a local port (rozsah 1 až 4095) musí být unikátní čísla pro jednotlivá využití SNMP protokolu (v případě že přichází více požadavků na SNMP agenta). IP adresa switche, se kterým komunikujeme. Na řádku 18 nastavuji vstup GetNext na

FALSE pro požadavek tagu pro níže zvolené OID. Použité OID na řádku 24 je pro teplotu switche. Na konec na řádku 29 vyvolám funkční blok s přiřazenými parametry.

Kód 8.1: Nastavení parametrů pro funkční blok

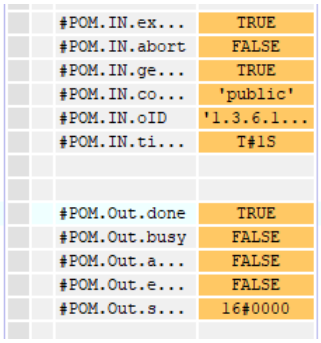
```
1  REGION TypeConn
2      #POM.InOut.connParam.localPort := 2020;
3      #POM.InOut.connParam.connID := 16#70;
4      #POM.InOut.connParam.hwIdentifier := 64;
5      //IP
6      #POM.InOut.connParam.ipAddress.ADDR[1] := 147;
7      #POM.InOut.connParam.ipAddress.ADDR[2] := 32;
8      #POM.InOut.connParam.ipAddress.ADDR[3] := 168;
9      #POM.InOut.connParam.ipAddress.ADDR[4] := 222;
10 END_REGION
11
12 REGION IN
13
14     #POM.IN.execute := "Clock_1Hz";//Tik 1S
15
16
17
18     #POM.IN.getNext := FALSE;
19     IF #POM.Out.done THEN
20         #POM.IN.getNext := TRUE;
21     END_IF;
22
23     #POM.IN.community := 'public';
24     #POM.IN.oID := '1.3.6.1.4.1.4329.20.1.1.1.1.79.7.1.16.0';
25     #POM.IN.timeOut := T#1s;
26
27 END_REGION
28
29 #FB_CALL_GET(execute := #POM.IN.execute,
30                              abort := #POM.IN.abort,
31                              getNext := #POM.IN.getNext,
32                              community := #POM.IN.community,
33                              oID := #POM.IN.oID,
34                              timeOut := #POM.IN.timeOut,
35                              connParam := #POM.InOut.connParam,
36                              varBinding := #POM.InOut.
   varBinding,
37                              done => #POM.Out.done,
38                              busy => #POM.Out.busy,
39                              aborted => #POM.Out.aborted,
40                              error => #POM.Out.error,
41                              status => #POM.Out.status,
42                              diagnostics => #POM.Out.
   diagnostics);
```

Vytvořený blok GET2 a GET3 (viz obrázek 8.8) následně vložím do Main nekonečné smyčky, výstupní hodnoty těchto bloků ukládám do proměnných, které následně využiji pro zobrazení v rámci SCADA. V případě teploty ještě probíhá konverze z hexadecimálního výstupu na integer. Pro zobrazení typu zařízení je konverze na String.



Obrázek 8.8: Bloky GET2 a GET3

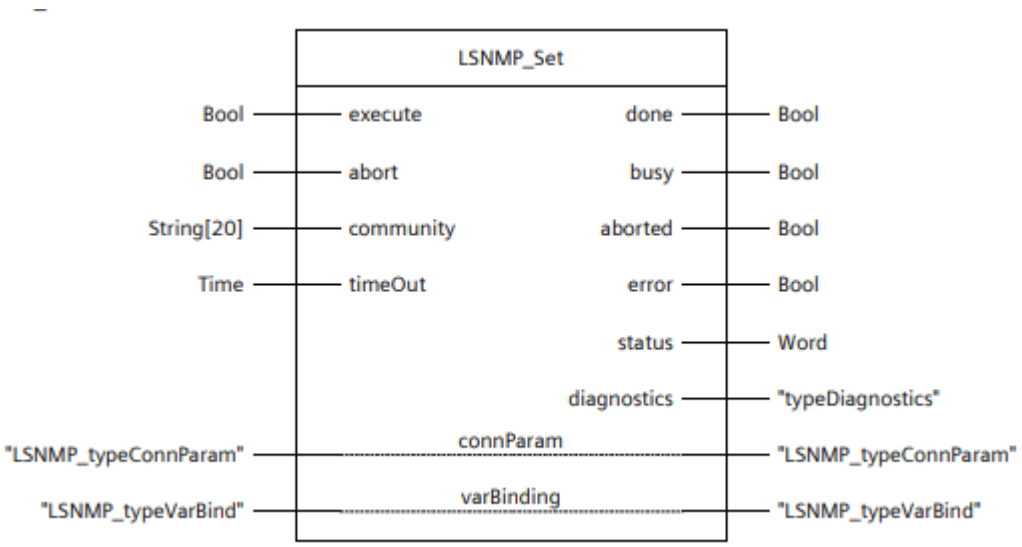
Při monitorování průběhu kódu nám funkční blok GET vrací stavovou hodnotu v hexadecimálním kódu jako 0000, což značí úspěšně provedení činnosti.



Obrázek 8.9: Monitorování kódu

8.4.2 Aplikace SNMP pro odposlech portů switche

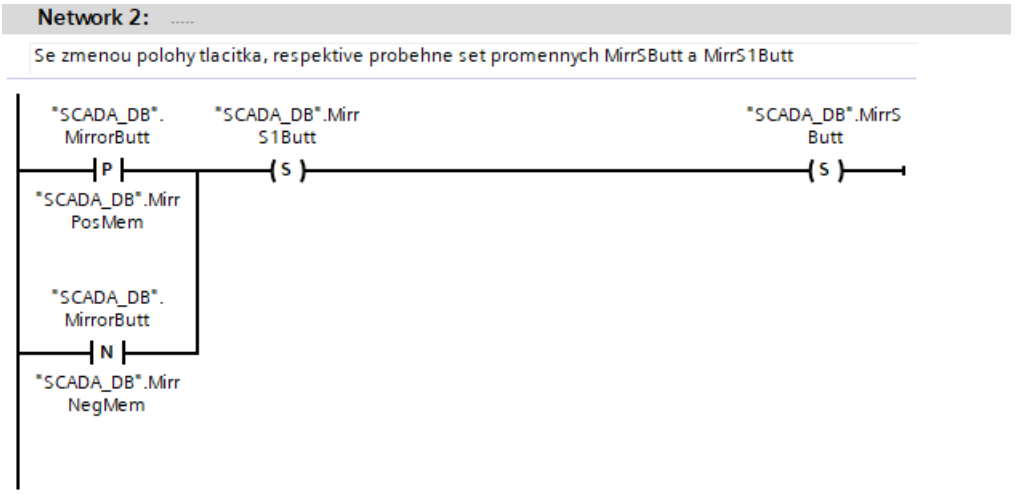
Princip, jakým probíhá odposlech portů switche je takový, že na zvolený port (v našem případě Port 8, na tom máme připojené zařízení ze kterého budeme monitorovat odposlechnutou komunikaci) zrcadlíme proběhnutou komunikaci z ostatních námi vybraných portů. Nejdříve proto musíme povolit zrcadlení v rámci switche, následně nastavíme port, na který proběhlou komunikaci budeme zrcadlit a na konec vybereme jednotlivé porty které potřebujeme monitorovat. Pro změnu nastavení v rámci switche využijeme funkční blok SET, viz obrázek 8.9.



Obrázek 8.10: Funkční blok SET[13]

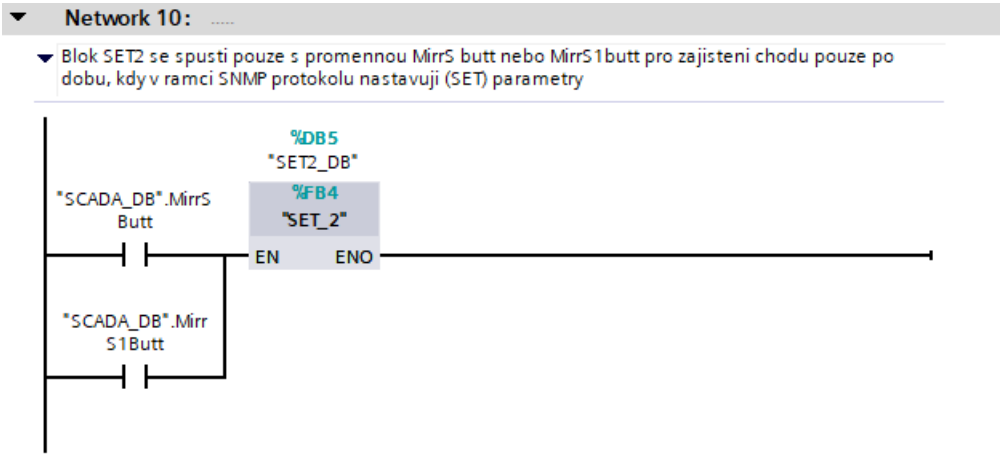
Pro zajištění, aby se nastavované parametry (OID a jeho požadovaná hodnota) nastavily správně, v kódu je využito také funkčního bloku GET pro ověření hodnot. Bez tohoto ověření se poté v důsledku stávalo, že se nepropsaly veškeré požadované hodnoty.

Jelikož je princip kódu pro ostatní tlačítka obdobný, budu většinu popisovat pouze na tlačítku pro zapnutí monitorování. Na obrázku 8.10 je vidět Network 2 na kterém je využito bloků, které se spustí v případě, že tlačítko (které je uloženo pod proměnnou MirrorButt) změní svůj stav z 1 na 0 nebo obráceně. To následně setuje proměnné, které jsou využity v dalším Network. Tímto krokem zajistíme spuštění SNMP Set bloku pouze se změnou polohy tlačítka.



Obrázek 8.11: Network 2

Tyto proměnné následně spouští blok SET2, v rámci kterého po vykonání určitých činností se znovu vyresetují.



Obrázek 8.12: Network 10

Nastavení parametrů připojení je obdobné jako u SNMP GET, v rámci SET je však zapotřebí nastavit typ vstupní hodnoty (kód 8.2), budeme zadávat integer a tomu přísluší dle dokumentace 02. Délka vstupní hodnoty je 1, jelikož v následujícím případě budeme nastavovat pouze 1 nebo 2 (disable nebo enable).

Kód 8.2: Nastavení parametrů pro funkční blok

```
1
2 REGION varBinding
3     #POM0.InOut.varBinding.type := 16#02;
4     #POM0.InOut.varBinding.length := 1;
5
6 END_REGION
```

V rámci kódu 8.3 v případě, že je spuštěné tlačítko proběhne kód, který postupně nastaví zrcadlení portů. Následně se ověří pomocí bloku GET na řádcích 12-15 (v kódu pojmenováno jako CALL), jestli jsou hodnoty skutečně propsané do switche. Na konec resetuje proměnnou MirrSButt a kód pokračuje na nastavení zrcadlení na port 8.

Kód 8.3: Kód pro tlačítko na aktivování zrcadlení

```
1
2 IF "SCADA_DB".MirrorButt
3     THEN
4         IF "SCADA_DB".MirrSButt
5           THEN
6
7             #POM0.InOut.varBinding.oID :=
    '1.3.6.1.4.1.4329.20.1.1.1.1.1.6.1.0';
8             #POM0.InOut.varBinding.value[0] := 16#2;
9
10            #FB_SET_0(execute := #POM0.IN.execute,...);
11
12            IF "CALL_DB".POM.InOut.varBinding.value[0] = 16#2
    AND "CALL_DB".POM.InOut.varBinding.oID =
    '1.3.6.1.4.1.4329.20.1.1.1.1.1.6.1.0'
13            THEN
14                "SCADA_DB".MirrSButt := FALSE;
15            END_IF;
16        ELSE
17            #POM0.InOut.varBinding.oID :=
    '1.3.6.1.4.1.4329.20.1.1.1.1.1.6.2.0';
18            #POM0.InOut.varBinding.value[0] := 16#8;
19
20            #FB_SET_0(execute := #POM0.IN.execute,...);
21
22            IF "CALL_DB".POM.InOut.varBinding.value[0] = 16#8
    AND "CALL_DB".POM.InOut.varBinding.oID =
    '1.3.6.1.4.1.4329.20.1.1.1.1.1.6.2.0'
23            THEN
24                "SCADA_DB".MirrS1Butt := FALSE;
25            END_IF;
26        END_IF;
```

V případě nesplnění podmínky zapnutého tlačítka pro odposlech proběhne kód, který zakáže odposlech. S tím zároveň proběhne postupné zkontrolování ostatních Network pro jednotlivé porty, které by zrcadlily svůj provoz do zmíněného portu 8. Pokud by byly zaplé, proběhne jejich postupné vypnutí, viz kód 8.4

Kód 8.4: Deaktivování portů

```
1
2  IF "CALL_DB".POM.InOut.varBinding.value[0] = 16#1 AND "CALL_DB
      ".POM.InOut.varBinding.oID =
      '1.3.6.1.4.1.4329.20.1.1.1.1.1.6.1.0'
3          THEN
4              "SCADA_DB".MirrSButt := FALSE;
5              "SCADA_DB".MirrS1Butt := FALSE;
6              "SCADA_DB".Port1 := FALSE;
7              "SCADA_DB".Port1S := TRUE;
8              "SCADA_DB".Port1S1 := TRUE;
9          END_IF;
```

Ověřování pomocí GET bloků současně probíhá pomocí IF/ELSIF podmínek a konkrétních nastavených OID (kód 8.5)

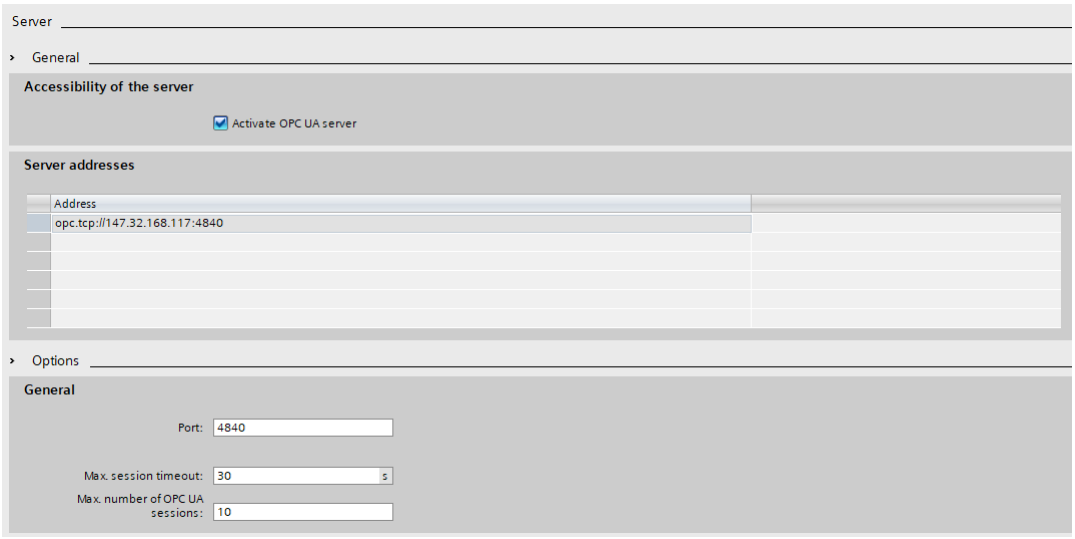
Kód 8.5: Ověřování OID pomocí GET bloku

```
1
2  IF "SCADA_DB".MirrSButt //overovani jednotlivych OID
3      THEN
4
5          #POM.IN.community := 'public';
6          #POM.IN.oID := '1.3.6.1.4.1.4329.20.1.1.1.1.1.6.1.0';
7          #POM.IN.timeOut := T#1s;
8
9      ELSIF "SCADA_DB".MirrS1Butt AND NOT "SCADA_DB".MirrSButt
10     THEN
11
12         #POM.IN.community := 'public';
13         #POM.IN.oID := '1.3.6.1.4.1.4329.20.1.1.1.1.1.6.2.0';
14         #POM.IN.timeOut := T#1s;
```

Ostatní bloky pro jednotlivá tlačítka následně probíhají na podobném principu.

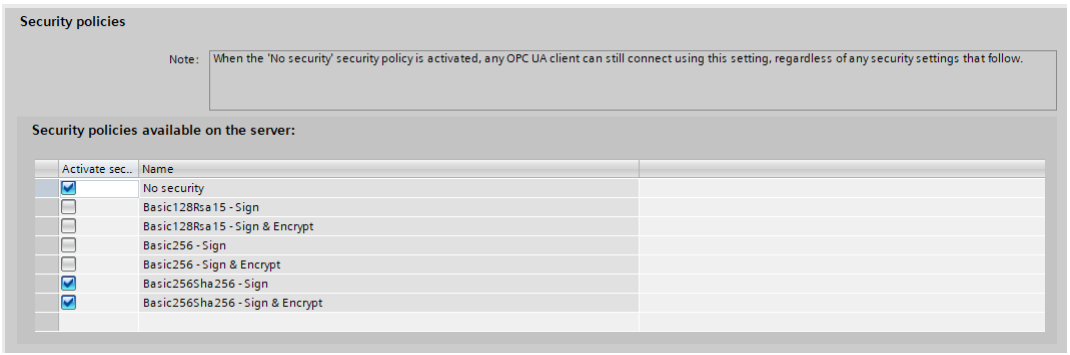
8.5 OPC UA komunikace / SCADA

Pro potřeby ovládání požadovaných dat v rámci SCADA je zapotřebí nastavit PLC jako OPC UA server. To nastavíme v Device configuration v sekci OPC UA server, viz obrázek 8.12, kde zároveň vidíme i nastavený port 4840.



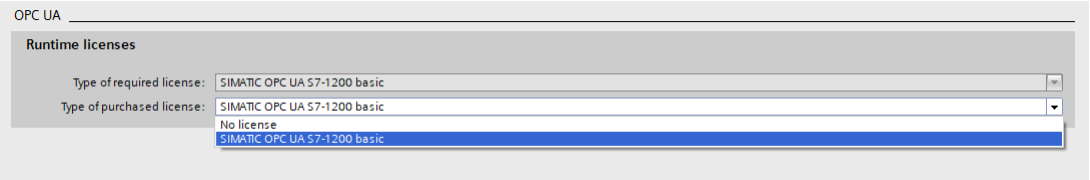
Obrázek 8.13: PLC nastavení OPC UA server

Pro budoucí potřebu odposlechu dat nastavíme server tak, aby neprobíhalo žádné šifrování dat (obrázek 8.13).



Obrázek 8.14: OPC UA Server zabezpečení

Pro funkčnost celého nastavení v PLC je ještě potřeba nastavit pořízenou licenci v rámci nastavení OPC UA serveru (obrázek 8.14).



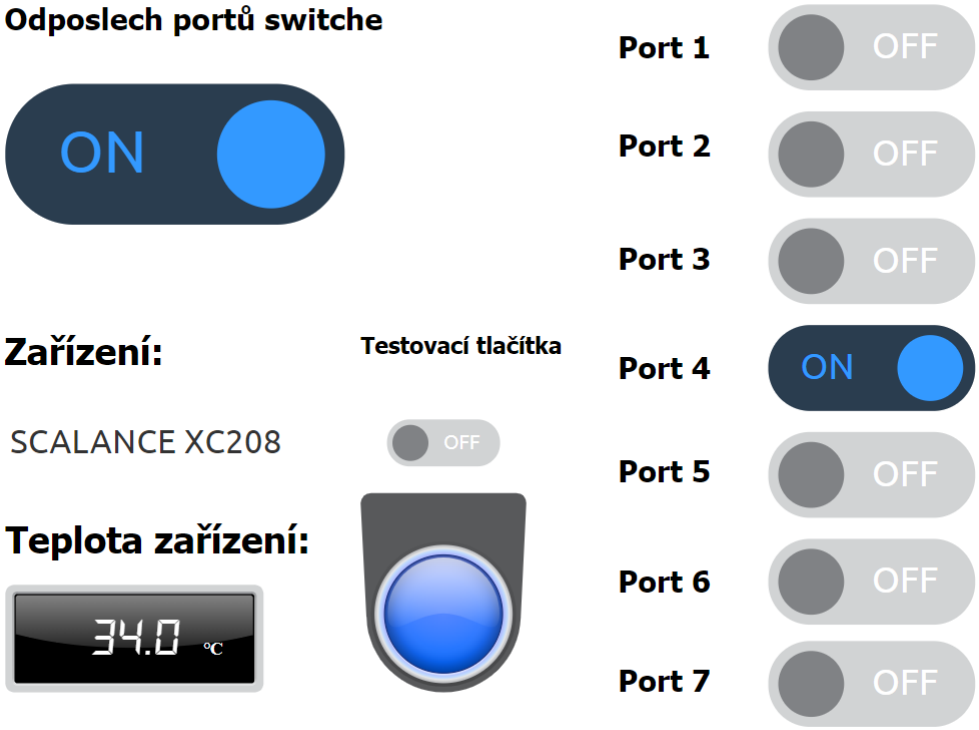
Obrázek 8.15: OPC UA licence

Na závěr vytvořím Server Interface s daty, které budu chtít posílat v rámci OPC UA komunikace.

Switch_port	Interface	---	
MirrorButt	BOOL	RD/WR	"SCADA_DB"."MirrorButt"
Port1	BOOL	RD/WR	"SCADA_DB"."Port1"
Port2	BOOL	RD/WR	"SCADA_DB"."Port2"
Port3	BOOL	RD/WR	"SCADA_DB"."Port3"
Port4	BOOL	RD/WR	"SCADA_DB"."Port4"
Port5	BOOL	RD/WR	"SCADA_DB"."Port5"
Port6	BOOL	RD/WR	"SCADA_DB"."Port6"
Port7	BOOL	RD/WR	"SCADA_DB"."Port7"
TempVal	INT	RD/WR	"GET2_DB"."TempVal"
DevType	STRING	RD/WR	"GET3_DB"."POM"."DevType"
DI0	BOOL	RD/WR	"DI0"
DQ0	BOOL	RD/WR	"DQ0"
<Add new>			

Obrázek 8.16: Server Interface

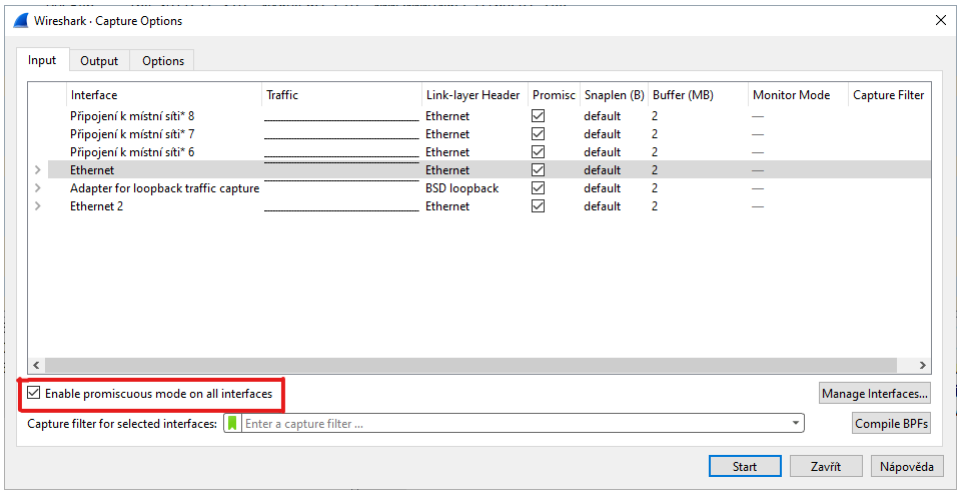
Jako OPC UA klient použiji SCADA software myDesigner, který běžným způsobem nastavím pro OPC UA komunikaci s mým PLC. Následně do něho nahraji požadované proměnné v podobně tagů, nakonec vytvořím pohled ze kterého se celý proces bude spouštět. Zároveň do výsledného projektu zasadím i testovací tlačítka s PLC tagy pro digitální vstup a výstup pro ověření OPC UA komunikace. Na obrázku 8.16 je výsledný pohled. Pomocí testovacích tlačítek je ověřeno, že komunikace mezi klientem a server probíhá. Zároveň v tomto pohledu máme zapnutý odposlech komunikace, kdy se nám na port 8 zrcadlí port 4. Aktuální teplota switche Scalance XC208 je 34 stupňů Celsia.



Obrázek 8.17: Výsledný pohled

8.6 Analýza odposlechnuté komunikace

Pro analýzu odposlechnuté komunikace využiji programu Wireshark, který slouží těmto účelům. Po výběru síťového rozhraní je zapotřebí zkontrolovat nastavení v sekci Capture Options. Zde povolíme promiscuous (obrázek 8.18), který nám umožní v programu zachytit komunikaci, která je určená také jiným síťovým kartám.

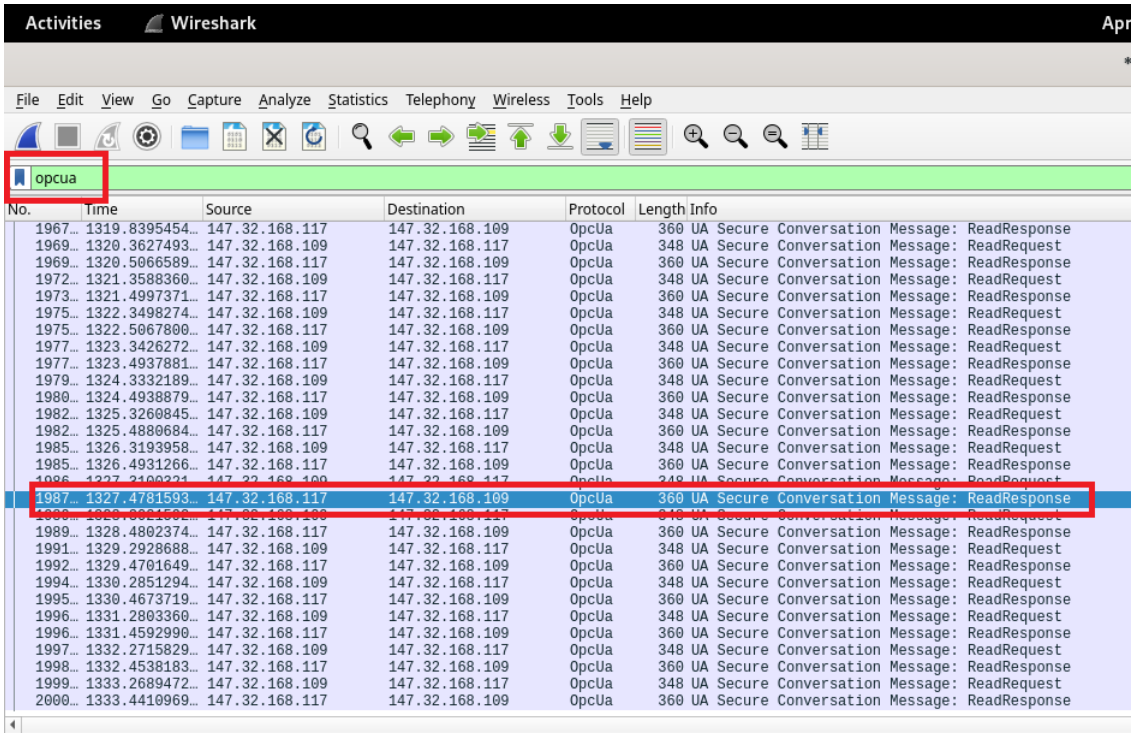


Obrázek 8.18: Nastavení promiscuous mode

Analýzu odposlechnuté komunikace demonstruji na OPC UA komunikačním protokolu, kdy na základě předchozí práce vím, že PLC (server), které je připojené na portu 4 switche komunikuje s OPC UA klientem v tomto případě počítač s IP adresou 147.32.168.109. Tuto komunikaci budu monitorovat ze zařízení připojeného na port 8. Proto můžu ponechat nastavení odposlechu portů tak, jak je uvedeno na obr. 8.17, tedy zapnutý odposlech portu 4. OPC UA komunikaci dle nastavení serveru mám nezašifrovanou. Díky tomu bude možné vidět tok nezašifrovaných dat.

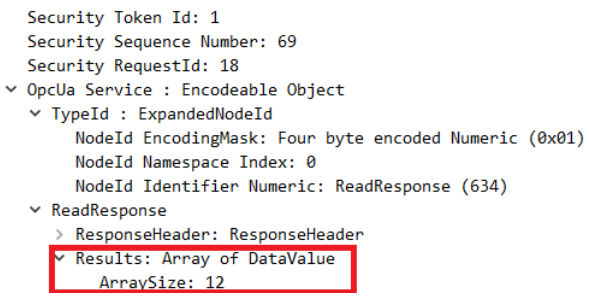
Ve wiresharku zapnu zachytávání paketů a nastavím filtr na opcua (obrázek 8.19). Díky tomu vidím, všechny pakety které jsou posílány v rámci tohoto protokolu. Ze znalosti komunikace v rámci switche vím, že na zařízení na portu 8 (ze kterého monitorujeme port 4) neprobíhá žádná komunikace pomocí OPC UA. Zároveň také vím, že OPC UA na portu 4 probíhá pouze se zařízením s IP adresou definovanou výše. Na základě výše uvedeného si mohu dovolit nepřidávat žádné další filtry, v opačném případě by bylo zapotřebí ještě přidat filtry například na základě IP adresy zdroje/cíle, případně MAC adresy pro zachycení mnou požadované komunikace. Na obrázku 8.19 zároveň je označen

paket pro další analýzu, z tohoto základního pohledu lze vyčíst například zdrojová a cílová IP adresa, protokol a délka paketu.



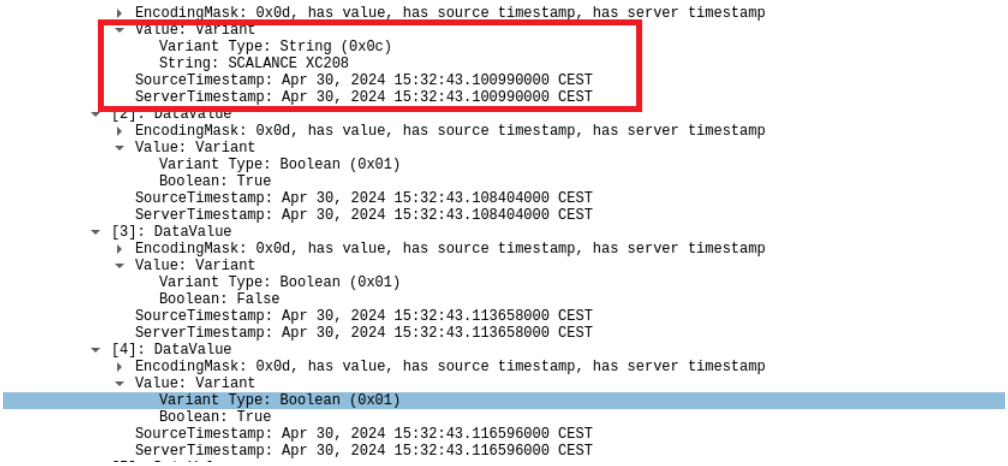
Obrázek 8.19: Nastavení filtru ve Wiresharku

Při hlubším prozkoumáním tohoto paketu následně v sekci OPC UA protokolu vidím, že v rámci paketu je dvanácti prvkové datové pole (obrázek 8.20). 12 prvků odpovídá tagům, které posílám z PLC do OPC UA klienta (2 I/O tagy pro testovací tlačítka, 8 tagů pro odposlech portů a na konec 1 tag pro monitorování teploty a 1 tag pro monitorování názvu zařízení)



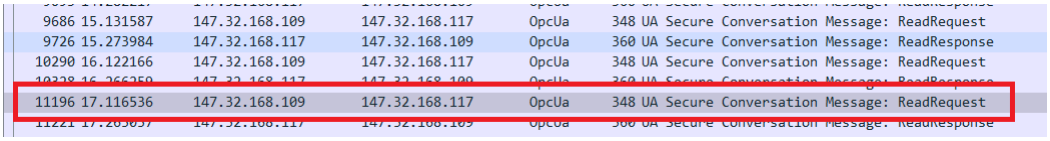
Obrázek 8.20: Analýza paketu

Po bližším nahlédnutí do jednotlivých prvků tohoto pole lze například vidět hodnotu druhého prvku pole (obrázek 8.21), kdy se jedná o string s hodnotou SCALANCE XC208. Tato hodnota odpovídá posílaným datům v rámci OPC UA komunikace. Zároveň vidíme, že odpovídají i ostatní hodnoty tohoto pole.



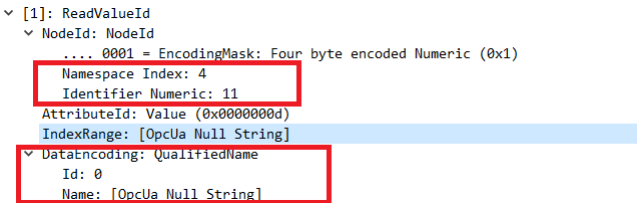
Obrázek 8.21: Analýza paketu

Pro případ, že má více prvků daného pole stejnou hodnotu, jako je tomu například u jednotlivých proměnných ethernetových portů, kdy prvek může nabývat booleovských hodnot True nebo False, je zapotřebí dané prvky blíže identifikovat, abychom si byli skutečně jistí s kterou proměnnou pracujeme. V prostředí Wireshark tak vybereme packet s OPC UA Read Request, jak je uvedeno na obrázku 8.22



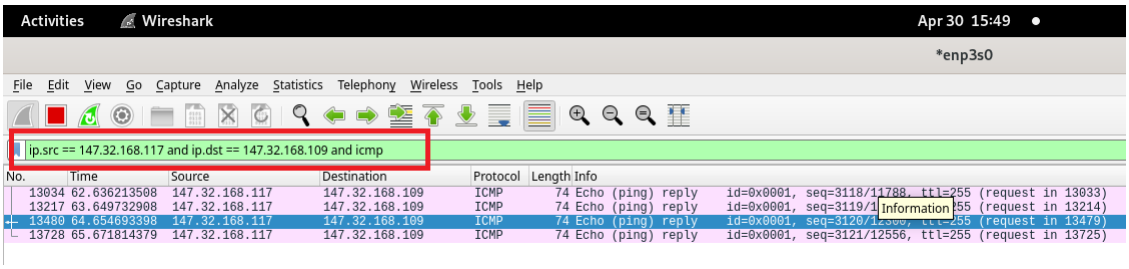
Obrázek 8.22: Read Request

Následně když nahlédneme do datového pole, opět vidíme 12 prvků. Pro ukázku znovu zanalyzuji druhý prvek, jak lze vidět na obrázku 8.23. Z uvedených dat lze vyčíst hodnota přiděleného tagu v rámci SCADY, tedy 4:11. Zároveň této proměnné není přidělené jméno, jak můžeme vyčíst ze spodního rámečku. Pro jednoznačné určení proměnné je hodnota tagu dostačující.



Obrázek 8.23: Read Request

Kdybychom potřebovali ze zařízení na portu 8 monitorovat ping z příkazového řádku počítače s IP adresou 147.32.168.109 na PLC s adresou 147.32.168.117, nastavení odposlechu zůstane stejné, pouze bude potřeba upravit nastavení filtru ve Wiresharku. Filtr na obrázku 8.22 je nastaven tak, aby se zobrazovaly pouze pakety s odpovědí od PLC. Z obrázku vidíme, že odpověď přišla 4 krát a ping proběhl úspěšně.



Obrázek 8.24: Ping PLC

Kapitola 9

Závěr

Tuto práci jsem začal rešerší na téma výrobních systémů a zejména jejich využití v průmyslové praxi. Probrány byly hlavně výrobní systémy holonické a multiagentní, kdy tyto systémy mezi sebou sdílí mnoho společných principů, hlavním rozdílem však mezi systémy je, že holonické systémy jsou přímo spjaty s fyzickým zařízením zatímco multiagentní nikoliv. Multiagentní systémy mají důležitou roli například v rámci počítačových sítí, kdy je jich využíváno pro různé potřeby tohoto odvětví, jako je například u SNMP protokolu, který jsem využil pro svojí praktickou část. Využití holonických systémů lze nalézt například v oblasti logistického řetězce. Důraz v bakalářské práci byl také kladen na kooperaci a koordinaci autonomních systémů. Vzhledem k využívání prvků síťové komunikace v praktické části jsem v rámci rešerše také rozebral referenční ISO/OSI model a jeho jednotlivé, který je využíván pro popis fungování sítě. Popsány byly některé důležité protokoly, jako například TCP/IP.

V praktické části se věnuji naprogramování úlohy pro odposlech komunikace pomocí průmyslového ethernetového switche Scalance XC208, kdy pro tento účel byl využit SNMP protokol. Za pomoci PLC S7-1200 a ovládání přes Scada systém pomocí OPC UA komunikace se mi podařilo vytvořit plně funkční úlohu pro odposlech jednotlivých portů switche. Mnou vytvořené řešení je funkční pro odposlech skrze poslední port (port 8), možné vylepšení této úlohy vidím v naprogramování řešení pro možnost odposlechu z libovolně zvoleného portu. Při spuštění programu po delší době nečinnosti switche (rámec dnů) také výjimečně dochází k zablokování odposlechu skrz port, který byl původně zvolený pro opačnou činnost. Toto je však možné změnit v rámci WBM switche. Řešení odposlechu portů switche lze částečně vykonat také v rámci WBM switche nebo v TIA portalu při spuštění switche do stavu "Go

Online"při spojení pomocí SNMP protokolu. Tato uvedená řešení nejsou z mého pohledu uživatelsky příznivá.

V poslední části praktické části dojde k analýze probíhající OPC UA komunikace mezi Scadou(OPC UA klient) a PLC S7-1200 (server), kdy tuto komunikaci při nastavování serveru zvolím jako nezašifrovanou. Pomocí programu Wireshark lze následně vidět hodnoty posílaných paketů, které zjevně odpovídají realitě.

Přílohy

Příloha A

Literatura

[1] J. Janeček, "Distribuované systémy," Praha, 2000. [Online]. Available: https://dsn.felk.cvut.cz/wiki/_media/vyuka/x36dsv/prednasky/x36dsv_skripta_2000.pdf

[2] J. Christensen, "Holonic manufacturing systems: Initial architecture and standards directions," 12 1994.

[3] H. Van Brussel and P. Valckenaers, "Design of holonic manufacturing systems," *Journal of Machine Engineering*, vol. 17, no. 3, pp. 5–23, 2017.

[4] P. Vrba, "Holonické výrobní systémy," 12 2000.

[5] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-agent systems: A survey," *IEEE Access*, vol. 6, pp. 28 573–28 593, 2018.

[6] A. Froehlich, "What is a star network?" https://www.techtarget.com/searchnetworking/definition/star-network, 8 2021, (Accessed on 05/06/2024).

[7] "Osi model tutorial," https://www.subnetting.net/school/docs/osi-model, (Accessed on 05/06/2024).

[8] C. M. Kozierok, "Ip overview and key operational characteristics," http://www.tcpipguide.com/free/t_IPOverviewandKeyOperationalCharacteristics.htm, 9 2005, (Accessed on 05/06/2024).

[9] M. Klement, "Technologie počítačových sítí," https://www.pdf-info.upol.cz/klement/vyuka/TPS@_2023_prednasky.pdf, 2023, (Accessed on 05/06/2024).

[10] "What is an rj45 connector?" https://www.geeksforgeeks.org/what-is-an-rj45-connector/, 2 2024, (Accessed on 05/06/2024).

[11] "Basic components of a fiber optic cable," https://www.truecable.com/blogs/cable-academy/basic-components-of-a-fiber-optic-cable, (Accessed on 05/06/2024).

[12] "Scalance xc-200," https://cache.industry.siemens.com/dl/files/149/109743149/att_1133069/v1/BA_SCALANCE-XC-200_76.pdf, (Accessed on 05/06/2024).

[13] "Libraries for communication for simatic controllers," https://support.industry.siemens.com/cs/document/109780503/libraries-for-communication-for-simatic-controllers?dti=0&lc=en-CZ, 4 2024, (Accessed on 05/06/2024).

[14] L. Schwab, S. Gold, and G. Reiner, "Exploring financial sustainability of smes during periods of production growth: A simulation study," *International Journal of Production Economics*, vol. 212, pp. 8–18, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925527318305012

[15] D. D. E. S. Zhanybek Suleiman, Sabit Shaikholla and A. Turkyilmaz, "Industry 4.0: Clustering of concepts and characteristics," *Cogent Engineering*, vol. 9, no. 1, p. 2034264, 2022. [Online]. Available: https://doi.org/10.1080/23311916.2022.2034264

[16] M. van Steen and A. S. Tanenbaum, *Distributed Systems*, 3rd ed. distributed-systems.net, 2017.

[17] "Umělá inteligence: definice a využití | zpravodajství | evropský parlament," https://www.europarl.europa.eu/pdfs/news/expert/2020/9/story/20200827STO85804/20200827STO85804_cs.pdf, 11 2023, (Accessed on 05/29/2024).

[18] A. Sabharwal and B. Selman, "S. russell, p. norvig, artificial intelligence: A modern approach, third edition." *Artif. Intell.*, vol. 175, pp. 935–937, 04 2011.

[19] K. P. Keyur, "Internet of things-iot: Definition, characteristics, architecture, enabling technologies, appliction future challanges," p. 6122, 5 2016.

[20] A. Katal, M. Wazid, and R. H. Goudar, "Big data: Issues, challenges, tools and good practices," in *2013 Sixth International Conference on Contemporary Computing (IC3)*, 2013, pp. 404–409.

[21] M. Al-Mekhlal and A. Khwaja, "A synthesis of big data definition and characteristics," 08 2019, pp. 314–322.

[22] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, p. 50–55, dec 2009. [Online]. Available: https://doi.org/10.1145/1496091.1496100

[23] E. Knorr and G. Gruman, "What cloud computing really means," *InfoWorld*, vol. 7, no. 20-20, pp. 1–17, 2008.

[24] Docker Inc., "Docker," 2024, accessed: 2024-05-30. [Online]. Available: https://www.docker.com/

[25] E. Glaessgen and D. Stargel, "The digital twin paradigm for future nasa and u.s. air force vehicles," 04 2012.

[26] E. Glaessgen, "The digital twin paradigm for future nasa and u.s. air force vehicles," 4 2012.

[27] H. Suda, "Future factory system in japan," *Journal of Advanced Automation Technology*, vol. 1, 1989.

[28] F. Koblasa and J. Vavruska, "AdaptivnÍ vÝrobnÍ systÉmy," 11 2017.

[29] A. Koestler, *The Ghost in the Machine*. Macmillan, 1967.

[30] P. Doležal, "Navrhování distribuovaných řídicích systémů," Praha, 2020.

[31] H. Van Brussel, J. Wyns, P. Valckenaers, L. Bongaerts, and P. Peeters, "Reference architecture for holonic manufacturing systems: Prosa," *Computers in Industry*, vol. 37, no. 3, pp. 255–274, 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016636159800102X

[32] K. Kruger and A. Basson, "Multi-agent systems vs iec 61499 for holonic resource control in reconfigurable systems," *Procedia CIRP*, vol. 7, pp. 503–508, 2013, forty Sixth CIRP Conference on Manufacturing Systems 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827113002928

[33] K. P. Sycara, "Multiagent systems," *AI Magazine*, vol. 19, no. 2, p. 79, June 1998. [Online]. Available: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/1370

[34] F. M. Al-Shrouf, "Facilitator agent design pattern of procurement business systems," in *2008 32nd Annual IEEE International Computer Software and Applications Conference*, 2008, pp. 505–510.

[35] J. Fu and J. Wang, "Adaptive coordinated tracking of multi-agent systems with quantized information," *Systems & Control Letters*, vol. 74, pp. 115–125, 2014.

[36] Y. Zhao, G. Wen, Z. Duan, X. Xu, and G. Chen, "A new observer-type consensus protocol for linear multi-agent dynamical systems," *Asian journal of control*, vol. 15, no. 2, pp. 571–582, 2013.

[37] Y. Kim and E. T. Matson, "A realistic decision making for task allocation in heterogeneous multi-agent systems," *Procedia Computer Science*, vol. 94, pp. 386–391, 2016.

[38] G. Miao and Q. Ma, "Group consensus of the first-order multi-agent systems with nonlinear input constraints," *Neurocomputing*, vol. 161, pp. 113–119, 2015.

[39] M. Wooldridge, *An introduction to multiagent systems*. John wiley & sons, 2009.

[40] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Autonomous Agents and Multiagent Systems*, G. Sukthankar and J. A. Rodriguez-Aguilar, Eds. Cham: Springer International Publishing, 2017, pp. 66–83.

[41] N. Fornara, "Interaction and communication among autonomous agents in multiagent systems," 2003.

[42] A. Netrvalová. (19) Úvod do problematiky multiagentních systémů. [Online]. Available: https://www.kiv.zcu.cz/netrvalo/phd/MAS.pdf

[43] J. Doran, S. Franklin, N. Jennings, and T. Norman, "On cooperation in multi-agent systems," *The Knowledge Engineering Review*, vol. 12, 01 2000.

[44] L. Chen, Q. Xue-song, Y. Yang, Z. Gao, and Z. Qu, "The contract net based task allocation algorithm for wireless sensor network," in *2012 IEEE Symposium on Computers and Communications (ISCC)*, 2012, pp. 000 600–000 604.

[45] FOUNDATION FOR INTELLIGENT PHYSICAL AGENTS, "FIPA Contract Net Interaction Protocol Specification," [Online], 03 2002. [Online]. Available: http://www.fipa.org/specs/fipa00029/SC00029H.pdf

[46] P. Gomez Esteban, S. Liu, D. Rios, and J. Gonzalez Ortega, "Competition and cooperation in a community of autonomous agents," *Autonomous Robots*, vol. 44, 03 2020.

[47] J. Bajo, F. De la Prieta, J. M. Corchado, and S. Rodríguez, "A low-level resource allocation in an agent-based cloud computing platform," *Applied Soft Computing*, vol. 48, pp. 716–728, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S156849461630268X

[48] J. O. Gutierrez-Garcia and K. M. Sim, "Agent-based cloud bag-of-tasks execution," *Journal of Systems and Software*, vol. 104, pp. 17–31, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016412121500045X

[49] L. Mechtri, F. D. Tolba, and S. Ghanemi, "Masid: Multi-agent system for intrusion detection in manet," in *2012 Ninth International Conference on Information Technology-New Generations*. IEEE, 2012, pp. 65–70.

[50] S. Sarika and V. Paul, "Agenttab: An agent based approach to detect tabnabbing attack," *Procedia Computer Science*, vol. 46, pp. 574–581, 2015.

[51] C. G. Cena, P. F. Cardenas, R. S. Pazmino, L. Puglisi, and R. A. Santonja, "A cooperative multi-agent robotics system: Design and modelling," *Expert Systems with Applications*, vol. 40, no. 12, pp. 4737–4748, 2013.

[52] D. Helbing, *Agent-Based Modeling*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 25–70. [Online]. Available: https://doi.org/10.1007/978-3-642-24004-1_2

[53] M. Khayyat and A. Awasthi, "An intelligent multi-agent based model for collaborative logistics systems," *Transportation Research Procedia*, vol. 12, pp. 325–338, 2016, tenth International Conference on City Logistics 17-19 June 2015, Tenerife, Spain. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352146516000703

[54] L. L. Peterson and B. S. Davie, *Computer networks: a systems approach*. Elsevier, 2007.

[55] M. M. Alani, *OSI Model*. Cham: Springer International Publishing, 2014, pp. 5–17. [Online]. Available: https://doi.org/10.1007/978-3-319-05152-9_2

[56] J. Day and H. Zimmermann, "The osi reference model," *Proceedings of the IEEE*, vol. 71, no. 12, pp. 1334–1340, 1983.

[57] D. Harrington, R. Presuhn, and B. Wijnen, "An architecture for describing snmp management frameworks," Tech. Rep., 1998.

[58] J. Schönwälder and V. Marinov, "On the impact of security protocols on the performance of snmp," *IEEE Transactions on Network and Service Management*, vol. 8, no. 1, pp. 52–64, 2011.

[59] H. Krawczyk, K. G. Paterson, and H. Wee, "On the security of the tls protocol: A systematic analysis," in *Advances in Cryptology – CRYPTO 2013*, R. Canetti and J. A. Garay, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 429–448.

[60] S. Iren, P. D. Amer, and P. T. Conrad, "The transport layer: tutorial and survey," *ACM Comput. Surv.*, vol. 31, no. 4, p. 360–404, dec 1999. [Online]. Available: https://doi.org/10.1145/344588.344609

[61] L. Parziale, W. Liu, C. Matthews, N. Rosselot, C. Davis, J. Forrester, D. T. Britt *et al.*, "Tcp/ip tutorial and technical overview," 2006.

[62] W. M. Eddy, "Transmission control protocol (tcp)," RFC 9293, August 2022.

[63] S. Kumar and S. Rai, "Survey on transport layer protocols: Tcp & udp," *International Journal of Computer Applications*, vol. 46, no. 7, pp. 20–25, 2012.

[64] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach*, 5th ed. Boston, MA: Pearson Education, 2010.

[65] J. Kurose and K. Ross, "Computer networks: A top down approach featuring the internet," *Peorsoim Addison Wesley*, 2017.

[66] A. N. A. Ali, "Comparison study between ipv4 & ipv6," *International Journal of Computer Science Issues (IJCSI)*, vol. 9, no. 3, p. 314, 2012.

[67] S. Topics, "Data link layer in osi model," 2023, accessed: 2024-05-29. [Online]. Available: https://www.scaler.com/topics/data-link-layer-in-osi-model/

[68] "Unified architecture - opc foundation," https://opcfoundation.org/about/opc-technologies/opc-ua/, (Accessed on 05/06/2024).

[69] J. Sommer, S. Gunreben, F. Feller, M. Kohn, A. Mifdaoui, D. Sass, and J. Scharf, "Ethernet – a survey on its fields of application," *IEEE Communications Surveys Tutorials*, vol. 12, no. 2, pp. 263–284, 2010.

[70] Ethernet cables - comparison between cat5, cat5e, cat6, cat7 cables. Online. [Online]. Available: https://web.archive.org/web/20190311163252/http://discountcablesusa.com/ethernet-cables100.html

[71] "Ccna: Network media types," https://www.ciscopress.com/articles/article.asp?p=31276, 3 2003, (Accessed on 05/06/2024).

[72] J. Buhagiar, *CompTIA Network+ Review Guide: Exam N10-007*. John Wiley & Sons, 2018.

[73] E. Conway, *Optical Fiber Communications: Principles and Practice*. Scientific e-Resources, 2019.

[74] W. Chen, L. Yuan, B. Zhang, Q. Yu, Z. Lian, Y. Pi, C. Shan, and P. P. Shum, "Applications and development of multi-core optical fibers," *Photonics*, vol. 11, no. 3, 2024. [Online]. Available: https://www.mdpi.com/2304-6732/11/3/270

[75] A. Daneels and W. Salter, "What is scada?" 1999.

Příloha B

Seznam použitých zkratek

- AI - Artificial intelligence / umělá inteligence
- NAI - Narrow Artificial intelligence / úzká umělá inteligence
- AGI - Artificial General intelligence / obecná umělá inteligence
- CPS - Cyber physical system
- IoT - Internet of things / internet věcí
- IP - Internet protocol / internetový protokol
- ML - Machine learning / strojové učení
- DL - Deep learning / hluboké učení
- HMS - Holonic manufacturing system / holonický výrobní systém
- PLC - Programmable logic controller / programovatelný logický automat
- PROSA - Product-Resource-Order-Staff-Architecture
- IEC - Internatonal Electrotechnical Commision - mezinárodní elektrotechnická komise
- FB - Function block / funkční blok
- MAS - Multiagent system / multiagentní (výrobní) systém
- FIPA - Foundation for Intelligent Physical Agents / organizace pro inteligentní fyzické agenty
- ARA - Adversarial Risk Analysis / analýza rizik protivníka

- ACL - Agent Communication Language / jazyk pro komunikaci mezi agenty
- BoT - Bag of Tasks / soubor nezávislých úkolů
- SNMP - Simple Network Management Protocol
- URL - Uniform Resource Locator / jednotný lokátor zdroje
- ABM - Agent Based Modelling / modelování založené na multiagentním systému
- RFID - Radio Frequency Identification / identifikace na rádiové frekvenci
- ACMS - Adaptive-Cognitive Manufacturing System / adaptivně-kognitivní výrobní systém
- IoP - Internet of People / internet lidí
- IoS - Internet of Services / internet služeb
- HMI - Human Machine Interface / rozhraní člověk-stroj
- CDT - Cognitive Digital Twin / kognitivní digitální dvojče
- PAN - Personal Area Network / osobní síť
- LAN - Local Areal Network / místní síť
- MAN - Metropolitan Area Network / metropolitní síť
- WAN - Wide Area Network / rozlehlá síť
- OSI - Open System Interconnection
- ISO - International Organization for Standardization / mezinárodní organizace pro standardizaci
- HTTP - Hypertext Transfer Protocol
- FTP - File Transfer Protocol / protokol pro přenos souborů
- DNS - Domain Name System / systém doménových jmen
- TLD - Top Level Domain / doména nejvyššího řádu
- ISP - Internet Service Provider / poskytovatel internetového připojení
- OID - Object Identifier / identifikátor objektu
- DoD - United States Department of Defense / Ministerstvo obrany Spojených států amerických
- MIB - Management Information Base

- TLS - Transport Layer Security / bezpečnost transportní vrstvy
- ICANN - Internet Corporation for Assigned Names and Numbers
- TCP - Transmission Control Protocol
- UDP - User Datagram Protocol
- MTU - Maximum transmission unit / maximální velikost paketu
- CLNS - Connectionless-mode Network Service / nespojovaná síťová služba
- WLAN - Wireless Local Area Network / bezdrátová lokální síť
- FDDI - Fiber distributed data interface
- MAC - Media Access control
- LLC - Logical Link Control - řízení linkového spoje
- PDU - Protocol Data Unit
- ICMP - Internet Controll Message Protocol
- HTML - Hypertext Markup Language
- OPC - Open Platform Communication / otevřená komunikační platforma
- UA - Unified Architecture / sjednocená architektura
- COM/DCOM - Distributed Component Object Model
- CSMA/CD - Carrier Sense Multiple Access with Collision Detection
- Cat - Category / kategorie
- STP - Spanning Tree Protocol / protokol kostry grafu

Příloha C

Obsah přiloženého CD

- Diplomová práce v elektronické podobě