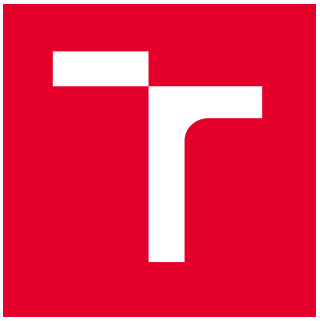


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

MODUL SÍŤOVÉ SONDY PRO ANALÝZU PRŮMYSLOVÝCH PROTOKOLŮ

NETWORK PROBE MODULE FOR INDUSTRIAL PROTOCOL ANALYSIS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Dominik Srovnal**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Petr Blažek**
SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Bc. Dominik Srovnal ***ID:*** 214091

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

NÁZEV TÉMATU:

Modul síťové sondy pro analýzu průmyslových protokolů

POKYNY PRO VYPRACOVÁNÍ:

Cílem diplomové práce je vytvoření rozšiřujícího modulu pro již realizovanou síťovou sondu (viz citace [1]), který bude sloužit pro analýzu komunikačního protokolu z průmyslového odvětví (IEC 61850, IEC 60870, C37.118, S7, EthernetIP). V rámci teoretické části práce se seznamte s fungováním síťové sondy, nastudujte implementované protokoly a zvolte jeden další protokol pro implementaci. V praktické části implementujte a otestujte vytvořený modul, aby komunikace odpovídala zvolenému protokolu. Dále proveďte testování na reálné nebo simulované komunikaci. Dílčím cílem práce bude realizace sady pravidel pro detekci zvolených útoků v modulu Suricaty (min. 3, kde útoky musí být zaměřeny na vybraný protokol nebo již implementované protokoly).
Výstupem diplomové práce bude realizovaný rozšiřující modulu pro analýzu zvoleného protokolu, včetně samotného podrobného testování modulu. Dílčím výstupem diplomové práce bude realizace sady pravidel pro detekční modul Suricaty.

DOPORUČENÁ LITERATURA:

[1] KLEČKA, Jan. Síťová sonda pro záznam a analýzu komunikace. Brno, 2020, 90 s. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Petr Blažek
[2] Ridho, M. Faqih. (2014). Analysis And Evaluation Snort, Bro, And Suricata As Intrusion Detection System Based On Linux Server. Skripsi thesis, Universitas Muhammadiyah Surakarta.

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

Vedoucí práce: Ing. Petr Blažek

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

ABSTRAKT

Průmyslové sítě jsou častým terčem útoků, na něž je třeba adekvátně reagovat. Proto je nutné předcházet těmto útokům již v počátku a to ochranou a prevencí vůči nim. Takovou ochranu poskytují systémy detekce a prevence průniku, jež na základě modulů jsou schopny zabraňovat nežádoucím průnikům. Jelikož jsou útoky čím dál sofistikovanější, je nutné, aby i tyto moduly byly neustále vyvíjeny a navrhovány nové, bezpečnější, opatření. Teoretická část popisuje průmyslové protokoly (IEC 61850, IEC 60870, Ethernet/IP a S7). Pratická část se soustředí na vytvoření modulu pro analýzu průmyslového protokolu S7. Dále jsou v práci popsány možné útoky na protokol S7 a navrženy pravidla pro detekci těchto útoků za pomocí modulu Suricaty.

KLÍČOVÁ SLOVA

Linux, protokol S7, průmyslové standardy, síťová sonda, systémy detekce a prevence průniku, suricata, modul

ABSTRACT

Industrial networks are often the target of attacks, which need to be adequately responded to. Therefore, it is necessary to prevent these attacks from the outset through protection and prevention. Such protection is provided by intrusion detection and prevention systems, which are capable of preventing unwanted intrusions, based on those modules. As attacks become more and more sophisticated, it is essential that these modules are continuously developed and got proposed in new, safer measures. The theoretical part describes industrial protocols (IEC 61850, IEC 60870, Ethernet/IP and S7). The practical part focuses on the creation of a module for the analysis of the industrial protocol S7. Furthermore, the paper describes possible attacks on the S7 protocol and proposes rules for detecting these attacks using the Suricata module.

KEYWORDS

Linux, protocol S7, industrial standards, network probe, intrusion detection and prevention systems, suricata, module

Vysázeno pomocí balíčku `thesis` verze 4.09; `https://latex.fekt.vut.cz/`

SROVNAL, Dominik. *Modul síťové sondy pro analýzu průmyslových protokolů*. Diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2024. Vedoucí práce: Ing. Petr Blažek

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. Dominik Srovnal
VUT ID autora:	214091
Typ práce:	Diplomová práce
Akademický rok:	2023/24
Téma závěrečné práce:	Modul síťové sondy pro analýzu průmyslových protokolů

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

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

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Petru Blažkovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů

Úvod

V současném rychle se vyvíjejícím technologickém prostředí je stále důležitější zajistit bezpečnost a efektivitu průmyslových komunikačních sítí. Je důležité nalézt způsoby a opatření, jak chránit přenos dat mezi zařízeními. Mezi takové způsoby patří analýza různých standardů a návrhů bezpečnostních opatření těchto standardů.

Teoretická část stručně uvede čtenáře do prostředí Linux a jeho různé distribuce, na nichž lze skvěle provádět manipulaci a bezpečnostní opatření. Objemnější a náročnější částí jsou systémy detekce a prevence průniku, díky nimž lze monitorovat a zachycovat nežádoucí pakety. Rozbor vybraných průmyslových protokolů a jejich standardů přiblíží čtenáři funkcionalitu a jejich zařazení v průmyslových sítích. Vybraným protokolem bude protokol S7, který bude detailně rozebrán, jelikož na něj navazuje praktická část.

Praktická část se zaměří na analýzu průmyslového protokolu, který bude vymezovat pracovní prostředí, samotnou síťovou sondu, její knihovnu a modul Suricata. Cíle práce jsou rozděleny do dvou dílčích částí, kde první dílčí částí je vytvoření modulu pro protokol S7 a druhou dílčí částí je vytvoření a otestvání sady pravidel pro tento protokol.

Vytvořený modul by měl být schopen zachytit pakety komunikace S7, následně celou komunikaci vyparsovat a na základě parsování vytvářet statistiky, které obsahují důležité informace o paketu, průběhu spojení, délky spojení a počtu zachycených paketů.

Sada pravidel pro protokol S7 bude vytvořena pro vybrané tři kybernetické útoky, jež mají narušit integritu samotných zařízení popřípadě jinak ohrozit průběh komunikace.

1 Linux

Linux je nejznámější a nejpoužívanější open source operační systém [18]. Původně byl Linux vyvinut pro osobní počítače na architektuře Intel x86, ale od té doby byl portován na více platforem než jakýkoli jiný operační systém. Linux je známý svou schopností běžet na široké škále zařízení od osobních počítačů, serverů až po superpočítače a vestavěné systémy. Linux je také základem operačního systému Android, který dominuje na trhu se smartphony. Vzhledem k tomu, že je Linux svobodně redistribuovatelný, může kdokoli vytvářet distribuce pro libovolný účel. Linux je také jedním z nejvýznamnějších příkladů spolupráce na free a open-source softwaru. Zdrojový kód může být používán, upravován a distribuován komerčně i nekomerčně kýmkoli podle podmínek jeho příslušných licencí [19]. Rozdíl mezi Linuxem a Unixem je minimální, jelikož Linux vychází z Unixu a mají podobné nástroje pro propojení se systémem, programovací nástroje, rozvržení souborového systému a další [18]. Linux, jak je již zmíněno, funguje na velké škále zařízení. Každý operační systém na bázi Linuxu zahrnuje tzv. Linux kernel (viz kapitola 1.1) který ovládá hardwarové zdroje. Velmi důležité pro Linux je schopnost být provozován jako operační systém nebo jako server. Operační systém zahrnuje pár základních komponentů jako jsou GNU nástroje. Tyto nástroje pak umožňují uživateli možnost spravovat prostředky poskytované kernelem, instalovat další software či nastavit zabezpečení. Jelikož je Linux open source, software z repozitáře není jednotný a může se lišit v závislosti na distribuci Linuxu viz kapitola 1.2. [20]

1.1 Linux Kernel

Linux Kernel je hlavní komponent operačního systému a prostředníkem mezi hardwarem a procesy v počitači. Kernel existuje uvnitř operačního systému (jako monolotické jádro) a kontroluje všechny důležité funkce hardwaru nezávisle na typu (telefon, počítač, server) [19, 21]. Kernel má 4 hlavní úkoly:

- Management paměti - mít povědomí o využití paměti (co, kde, jak?);
- Management procesů - rozhodování které procesy má využívat procesor (kdy a jak dlouho?);
- Správu ovladačů zařízení - být prostředníkem mezi hardwerem a procesy;
- Zpracování požadavků systému a jeho zabezpečení - přijímat požadavky na služby z procesů. [21]

Kernel, pokud je správně implementován, je zpravidla neviditelný uživateli. Běží na pozadí v prostoru zvaném „kernel space," kde alokuje paměť a kontroluje kde je co uloženo. Opak kernelu je pak „user space," což je prostor, který vidí uživatel (např. webový prohlížeč či správce souborů) [21].

1.2 Distribuce Linuxu

Linuxové distribuce, známé také jako distros, jsou různé operační systémy založené na Linuxovém jádru. Každá distribuce má svou jedinečnou kombinaci softwaru a správy balíčků, která vyhovuje různým potřebám uživatelů. Například Ubuntu je oblíbené pro svou přívětivost k uživatelům a rozsáhlou podporu, zatímco Fedora je známá svým zaměřením na nejnovější technologie. Debian je ceněn pro svou stabilitu a rozsáhlé repozitáře, zatímco Arch Linux je oblíbený mezi pokročilými uživateli pro jeho flexibilitu a filozofii KISS (Keep It Simple, Stupid).[19, 22]

Existují i specializované distribuce, jako je Kali Linux pro bezpečnostní testování nebo Raspbian pro Raspberry Pi. Linuxové distribuce často nabízejí široký výběr prostředí grafického uživatelského rozhraní (GUI), jako je GNOME, KDE Plasma, a Xfce, což umožňuje uživatelům přizpůsobit své pracovní prostředí. Většina Linuxových distribucí je dostupná zdarma a podporuje filozofii otevřeného zdrojového kódu, což znamená, že uživatelé mohou upravovat a sdílet software podle svých potřeb [19, 22]. Tabulka 1.1 obsahuje nejrozšířenější linuxové distribuce. Mimo jiné, k dalším aspektům Linuxových distribucí patří:

- Bezpečnost a soukromí: Linuxové distribuce jsou obecně považovány za bezpečnější než mnoho jiných operačních systémů, díky své otevřené povaze a pravidelným aktualizacím.
- Přizpůsobitelnost: Uživatelé mají možnost hluboce přizpůsobit své systémy, od vizuálního vzhledu až po způsob, jakým systém funguje. Tato flexibilita je vhodná pro pokročilé uživatele a vývojáře.
- Výběr pro různá zařízení: Některé distribuce jsou navrženy pro specifické typy zařízení nebo hardware. Například, Ubuntu Touch je určené pro mobilní zařízení, zatímco CentOS je oblíbené pro serverové použití.
- Životní cyklus a vydání: Různé distribuce nabízejí různé přístupy k vydávání aktualizací a nových verzí. Například, Ubuntu vydává nové verze každých šest měsíců, zatímco Debian vydává nové stabilní verze méně často, ale s důrazem na stabilitu a testování.
- Proprietární software: Některé distribuce, jako Ubuntu, nabízejí snadný přístup k proprietárnímu softwaru a ovladačům, což může být důležité pro uživatele, kteří potřebují kompatibilitu s určitým softwarovým nebo hardwarovým vybavením.
- Filozofie a cíle: Každá distribuce má svou vlastní filozofii a cíle, které ovlivňují její vývoj a uživatelskou zkušenost. [19, 22]

Distribuce	**Licence**	**Poslední vydání**
AlmaLinux	GPLv2 a další licence	AlmaLinux 9.2
Arch Linux	GNU GPL a další licence	Rolling release
Asahi Linux	Veřejná licence v2 nebo MIT	Asahi Linux Alpha Release
BlackArch Linux	Několik licencí	–
CentOS Linux	GNU GPL	CentOS Linux 8
CentOS Stream	GNU GPL	CentOS Stream 9
Clear Linux	GPL a další licence	Rolling release
CloudLinux	–	CloudLinux 8.8.1
Debian	BSD, GPL a další licence	Debian 12 (Bookworm)
Elementary OS	GNU GPL a další licence	Elementary OS 7 (Horus)
Fedora	GPL a další licence	Fedora 38
Gentoo	Svobodný software	Rolling release
Knoppix	Licence svobodného softwaru	Knoppix 9.2
Linux Mint	GPL	Linux Mint 21.1 (Vera)
Lubuntu	GNU GPL a další licence	Lubuntu 22.04
Mageia	GPL a další licence	Mageia 8
Manjaro	GPL a další licence	Manjaro 22.1 (Talos)
Navy Linux	GNU GPL	Navy Linux 8.6
Oracle Linux	GNU GPL a další licence	Oracle Linux 9
OpenSUSE	GNU GPL a další licence	OpenSUSE Leap 15.5
Peppermint OS	Licence svobodného softwaru	–
Red Hat Enterprise Linux	GPL	RHEL 9
Rocky Linux	BSD a další licence	Rocky Linux 9.2
Scientific Linux	GNU GPL a další licence	Scientific Linux 7
Slackware	GNU GPL	Slackware 15.0
SUSE Liberty Linux	–	SUSE Liberty Linux 9
SUSE Linux	–	SUSE Linux Server 15 SP5
Tizen	GPLv2, LGPL	Tizen 7.0
Ubuntu	GPL a další licence	Ubuntu 22.04 LTS
VzLinux	–	VzLinux 9
Zorin OS	Svobodný software	Zorin OS 16.2

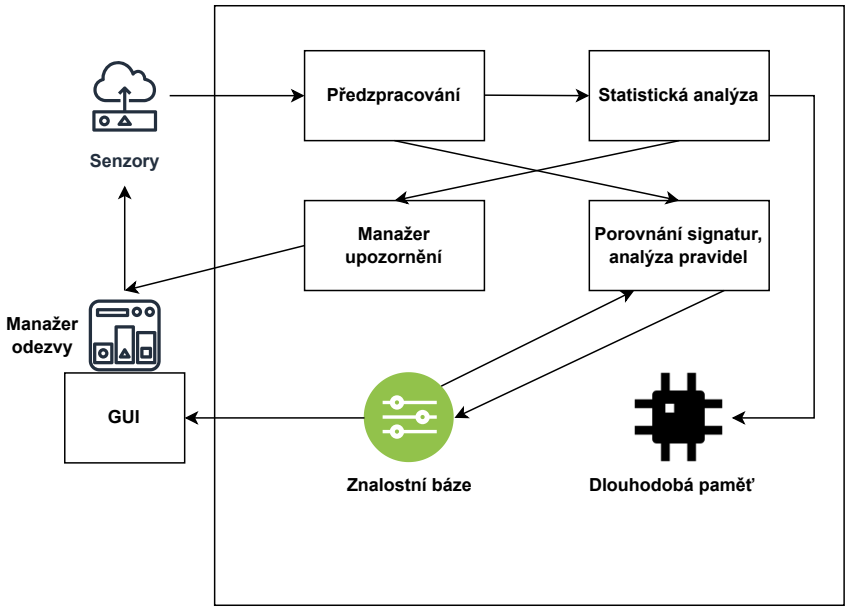
Tab. 1.1: Populární distribuce Linuxu (převzato a upraveno z: [22]

2 Systémy detekce a prevence průniku

Detekce průniku je monitorovací proces sítě, které na základě analýzy monitoringu vyhledává potenciální průniky, jimiž mohoou být pokusy o zneužití chyby či jakékoliv další incidenty ohrožující síť. Prevence průniku je proces detekce průniku a následného zastavení zjištěných incidentů zpravidla upuštěním od paketů či ukončením celé relace. Systémy detekce průniku (IDS) a systémy prevence průniku (IPS) jsou kritickými nástroji v síťové bezpečnosti a jsou implementovány do dnešních firewallů [12]. Alternativně lze IDS definovat jako

> *„(...) soubor nástrojů a metod, jejichž cílem je odhalit aktivitu, jejímž cílem je nebo může být průnik do systému, neoprávněný přístup k informacím, pokus o útok na dostupnost systému nebo služby apod. a upozornit na tuto nekalou či neoprávněnou aktivitu [13].“*

IPS, jak je zmíněno výše, je pak další úroveň zabezpečení [13]. Architektura IDS je znázorněna na obrázku 2.1



Obr. 2.1: Obecná architektura IDS (převzato a upraveno z:[13])

V IDS se používají zpravidla tři metodologie k detekci narušení:

- Detekce na základě podpisu;
- Detekce na základě výskytu anomálie;
- Analýza stavového protokolu.

IDS a IPS neustále sleduje tok sítě a identifikuje možné narušení a zhromažďuje o nich informace. Možné útoky aktivně zastavuje a případně je nahlašuje administrátorovi. IDS/IPS jsou nedílnou součástí veškerých firem a organizací [12]. IDS jsou pak rozdělovány dle funkcionality na:

- Systém detekce narušení založený na síti (NIDS);
- Systém detekce narušení založený na hostiteli (HIDS);
- Distribuovaný systém detekce narušení (DIDS). [17]

IDS je schopno detekovat hrozby na základě určitých vzorů, nevhodného chování a různých typů útoků. IDS je také schopno detekovat typy útoků, které se cíleně vyhýbají IDS s účelem nebýt odhalen např. v typech manipulace TCP/IP. IPS je schopno zabraňovat jistým typům útoku DDoS. [12]

Výhody a nevýhody IDS/IPS:

- Výhody:
  - Lepší zabezpečení IoT;
  - Efektivnější schopnost nalezení zranitelných míst;
  - Schopnost rychlé reakce na bezpečnostní incidenty;
  - Automatizování reakce a odpovědi na bezpečnostní incidenty;
  - Možnost logování, analýzy a tvorby reportů.
- Nevýhody:
  - Netriviální konfigurace IDS/IPS;
  - Špatná konfigurace může zapříčinit tzv. false reporty;
  - Problematika šifrování dat;
  - Špatná schopnost analýzy síťí v reálném čase. [12]

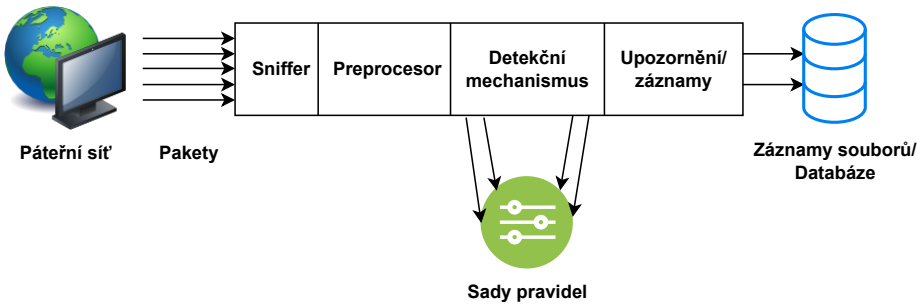
Existují open-source IDS, mezi něž se řadí Snort, Suricata, Zeek či Juniper, kde vybrané jsou přiblíženy v následujících kapitolách. [12, 14]

2.1 Snort

Snort je open source síťový systém pro detekci a prevenci průniků (NIDS/NIPS). Pracuje na jednom vlákně. Je to vysoce konfigurovatelný nástroj, který monitoruje síťový provoz a analyzuje ho na základě pravidel, aby odhalil pokusy o průnik a známé hrozby. SNORT může být použit v pasivním režimu pro detekci nebo v aktivním režimu pro blokování hrozeb. Jeho schopnost přizpůsobit se různým síťovým prostředím a rozmanitost podporovaných pravidel z něj činí jedno z nejpoužívanějších řešení pro síťovou bezpečnost. Architektura Snortu je znázorněna na obrázku 2.3. [17]

Snort je schopen provádět analýzu protokolů a obsahově hledat/propojovat, což může být použito pro detekci útoků a sond. Snort lze konfigurovat ve třech módech a to:

- Sniffer - čte pakety ze sítě a zobrazuje je na konzole;
- Packet logger (záznamník paketů) - zaznamenává pakety na disk;
- Network intrusion detection (síťová detekce průniku) - nejkomplexnější a nejlépe konfiguratelný, analyzuje síťový tok dat a snaží se najít shody s pravidly nastavenými uživatelem. [17]



Obr. 2.2: Architektura Snortu (převzato a upraveno z:[17])

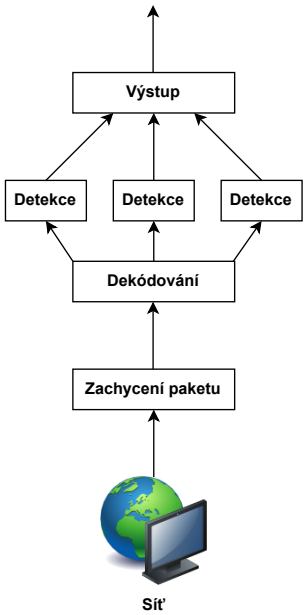
Snort v základu využívá Libpcap k zachycování paketů, kde následuje dekódování paketů pomocí dekóderu [14]. Dekóder může občasně nalézt falšenou pozitivitu a tím spustit upozornění bezdůvodně [16]. Normalizace paket je pak zpracována preprocesorem a konvertována do síťového provozu aby vytvořila takovou formu, kterou je schopen detekční mechanismus pochopit a pracovat s nimi [14]. Je velmi důležité správně nastavit preprocesor, jelikož při špatné konfiguraci bude detekční mechanismus zahlcen nedůležitými daty, tzv. junk data. Tímto způsobem je možné zvýšit výkonnost celého systému [16]. Detekční mechanismus pak aplikuje pravidla do síťového provozu k detekci potenciálního nežádoucího paketu. Detekční mechanismus analyzuje pakety na základě podpisu a určuje, jestli je provoz v pořádku či nežádoucí. V případě nalezení nežádoucího paketu se uplatňují opatření na základě předdefinovaných pravidel[14]. Struktura každého pravidla zahrnuje popis provozu, podpis který se hledal a popis reakce na zjištěnou hrozbu. V poslední fázi výstupní plug-iny definují jak se vypořádají s upozorněními, konkrétně se speficikací typu upozornění, který má být zaznamenán a v jakém adresáři by měl být zkontrolován [16]. Je nutné zmínit, že Snort v základu nepodporuje detekci na základě anomálií [14]. Dále, z bezpečnostních důvodů, je nutné ukládat IDS/IPS pravidla na vzdáleném hostovi a zajistit plán obnovy, které specifikuje konkrétní opatření k zajištění obnovy systému (na němž je Snort spuštěn) v případě, že by spadl a byl neschopen zajišťovat žádanou ochranu [16].

2.2 Suricata

Suricata je vysokovýkonný síťový systém pro detekci a prevenci průniků (IDS/IPS) a pro síťové monitorování bezpečnosti. Suricata se vyznačuje následujícími charakteristikami a schopnostmi:

- Vysoký výkon;
- Podpora různých síťových topologií;
- Flexibilní konfigurace;
- Otevřený vývoj a komunitní podpora.

S vývojem zákeřnějších a sofistikovanějších penetračních metod využívající čím dál větší výpočetní výkon je kladen důraz na technologické požadavky IDS/IPS [15, 16]. Suricata podporuje vícenásobné detekční mechanismy díky využití vícero vláken, tj. je schopna zpracovat více síťového provozu v porovnání se Snortem, který podporuje pouze jednovláknové detekční mechanismy viz 2.1. Co se týče detekce založené na podpisu, pak Suricata využívá stejný formát jako Snort v podobě nastavení pravidel a také má obdobné detekční algoritmy [14]. Obdobně jako Snort, Suricata využívá balíčky Libpcap a PF_RING, které značně zrychlují rychlost sběru paketů. Jak je zmíněno výše, Soricata je IDS/IPS, stejně jako Snort, a je založena na předdefinované sadě pravidel, jejichž přesnost určuje míru falešných chyb a falešných pozitiv při odhalování hrozeb. Podstata open source technologie Snortu umožnila vývoj vylepšených IDS/IPS vhodnějších pro větší výpočetní technologie a na architektuře Snortu vznika Suricata viz obrázek 2.3.



Obr. 2.3: Architektura Suricaty(převzato a upraveno z: [16])

To umožnilo Suricatě větší škálovatelnost avšak zachování i schopností Snortu, tj. využívat pouze jednovláknový mód, kde pakety mohou být zpracovávány po jednom. Detekční mechanismus Suricaty využívá vícero vláken, což umožňuje přesněji alokovat výpočetní výkon a tím zajistit vhodné rozmístění zpracovávaných operací.

3 Rozbor vybraných průmyslových protokolů

Průmyslové protokoly jsou zásadní pro komunikaci v řízení v automatizaci a energetických systémech. Mezi takové protokoly lze zařadit IEEE C37.118, IEC 61850, IEC 60870 a EtherNet/IP. IEEE C37.118 se zaměřuje na synchronizovaná fázorová měření v elektrických sítích. IEC 61850 a IEC 60870 se zabývají komunikací v energetických systémech, přičemž IEC 61850 je standardizovaný pro automatizaci rozvoden. IEC 60870 zahrnuje protokoly pro dálkové řízení a sledování elektrických sítí. EtherNet/IP, založený na standardním Ethernetu, se používá v průmyslové automatizaci pro řízení a sběr dat. [16]

3.1 Protokol IEEE C37.118

Standard IEEE C37.118 pokrývá měření synchronizovaných fázorů, jež se používá u elektrických systémů (sítí). Dále definuje formáty komunikačních dat používaných pro přenosy v reálném čase. Pro měření se používají jednotky meřící jednotlivé fázory, které jsou schopné komunikace a potažmo i ukládání dat z měření. Standard dále definuje specifické požadavky, jimiž jsou:

- Přesnost měření,
- synchronizace času,
- komunikační protokoly. [2]

Dle autorů článku o analýze aplikace standardu IEEE C37.118 a IEC 61850-90-5 má technologie synchronizovaných fázorů potenciál být kritickou součástí moderních elektrických systémů. [3]

3.1.1 Měřící jednotka fázorů

Koncept měření synchronizovaných fázorů vychází z předpokladu zjištění dat elektrických systémů jimiž jsou napětí a proud ke konkrétnímu času. Zde spadá první specifický požadavek, a to přesnost měření, což definuje, jak přesné musí být měření fázorů, které je klíčové pro spolehlivé monitorování a řízení elektrických sítí. Principem je schopnost využití přesných zařízení mezi něž se řadí i GPS, která je napojená na koordinovaný univerzální čas (UTC). Dále sem spadá druhý požadavek na synchronizaci času, což umožňuje srovnávat data z různých míst v síti. PMU (měřící jednotka fázorů) pak využívá reference, zjištěních pomocí GPS, a převádí je jako fázorovou reprezentaci v podobě signálu (sinusoidy). [2]

3.1.2 Komunikační protokol

Komunikační protokol IEEE C37.118 vychází ze standardu IEEE 1344, což je prvotní námět synchrofázorového měření, a je jeho vylepšenou verzí [3]. Do této podkapitoly spadá třetí požadavek (viz požadavky v kapitole 3.1), a to požadavek na protokoly pro přenos dat mezi PMU a kontrolními centry [2]. IEEE C37.118 definuje metody viz 3.1.1 a také formáty dat přenesených přes sítě [3]. Autoři článků [2, 3] popisují formáty dat rozdílně. Dle R. Khana jsou typy zpráv pouze čtyři:

- Data,
- konfigurace,
- header,
- command.

Autor K. E. Martin však doplňuje IEEE C37.118 o schopnost zjednodušení dekódování pomocí synchronizace a omezení velikosti slov. Analýza protokolu IEEE C37.118 poukazuje na velmi slabou až žádnou důvěrnost, slabou integritu a náchylnou dostupnost. Protokol je zranitelný na základě analýzy odolnosti vůči vybraným kybernetickým útokům, jimiž jsou: Vyzvídání, pokusy o ověření a přístup, tzv. Man in the middle, a DoS. [3]

3.1.3 Zhodnocení IEEE C37.118

Standard IEEE C37.118 je významný pro přesná fázorová měření v energetických systémech, poskytující specifikace pro měření, synchronizaci času a komunikaci. Přestože efektivně řeší potřeby v síťovém provozu a přesnosti, standard se potýká s omezeními v oblasti bezpečnostních funkcí, jako je absence šifrování. Tato slabina může omezovat jeho použitelnost v situacích, kde jsou vyšší bezpečnostní požadavky. Celkově je IEEE C37.118 uznáván pro jeho přínos k přesnému monitorování a řízení energetických systémů, ale vyžaduje další rozvoj v oblasti kybernetické bezpečnosti.

3.2 Standard IEC 61850

Protokol IEC 61850 je komunikačním protokolem s cílem zvýšení kvality komunikace mezi inteligentními elektronickými zařízeními (IED) [5]. Klíčovým aspektem protokolu IEC 61850 je adaptibilita pro budoucí požadavky. Obecně standardizuje komunikaci mezi IED v rozvodnách a podporuje efektivní integraci nezávisle na výrobci IED do jednotného systému. Umožňuje efektivní sběr dat, jejich konfiguraci a ovládání na dálku, což zjednodušuje řízení jednotlivých procesů. Českým ekvivalentem je norma ČSN EN 61850 [6]. Struktura standardu IEC 61850 je znázorněna v tabulce 3.1.

Část	Název
1	Úvod a přehled
2	Slovníček pojmů
3	Obecné požadavky
4	Systém a projektové řízení
5	Komunikační požadavky pro funkce a modely zařízení
6	Konfigurační popisovací jazyk pro elektrické rozvodny související s IED
7	Základní komunikační struktura pro rozvodny a přípojky
7.1	Zásady a modely
7.2	Abstraktní komunikační servisní rozhraní (ACSI)
7.3	Společné datové třídy (CDC)
7.4	Kompatibilní logické uzlové třídy a datové třídy
8	Specifické mapování komunikačních servisů (SCSM)
8.1	Mapování na MMS (ISO/IEC 9506 – Část 1 a Část 2) a ISO/IEC 8802-3
9	Mapování komunikačních servisů (SCSM)
9.1	Vzorkované hodnoty přes sériový jednosměrný multidrop Point-to-Point Link
9.2	Vzorkované hodnoty přes ISO/IEC 8802-3
10	Testování shody

Tab. 3.1: Struktura standardu IEC 61850 [4]

3.2.1 Komunikační systém

IEC 61850 je plně funkční komunikační systém, který byl navržen s několika cíli:

- Interoperabilita a integrace mezi komponenty elektrických systémů,
- modelování služeb a zařízení,
- popisování sebe sama,
- automatické odhalování objektů skrze meta-data,
- spolehlivost mechanismu skrze retranslaci,
- snížení ceny rozvoden skrze nahrazení přenosu drátového bezdrátovým
- podpora pro sdílení konfigurace v pomezí stroj-stroj. [3]

GOOSE (Generic Object Oriented Substation Event) je mimo další protokol pro IEC 61850 a jeho interoperabilita mezi IED od různých výrobců (např. ABB, VAMP a Siemens) je podložena studií [5]. Velmi důležitou vlastností tohoto komunikačního systému je schopnost přenášet data z IED do systémů SCADA [6] (blíže rozebráno v podkapitole 3.3.3). Analýza protokolu IEC 61850 poukazuje na silnou důvěrnost

a integritu z pohledu bezpečnosti. Nedostatkem je náchylnost protokolu na dostupnost. Na základě analýzy odolnosti vůči vybraným typům útoků, které byly použity i pro testování protokolu IEEE C37.118 (viz 3.1.2), je chráněn vůči všem vyjma DoS, kde je zranitelný. [3]

3.2.2 Zhodnocení IEC 61850

Standard IEC 61850 se vyznačuje vysokou mírou interoperability a integrace v energetických systémech, s důrazem na pokročilé bezpečnostní funkce, včetně šifrování. IEC 61850 je ceněn pro svou flexibilitu a schopnost integrace různých typů zařízení, což jej činí ideálním pro moderní a bezpečné řízení energetických sítí.

3.3 Standard IEC 60870

Standard IEC 60870 je součástí série mezinárodních standardů pro systémy dálkového ovládání a získávání dat (SCADA) v elektrických přenosových a distribučních systémech. SCADA systémy jsou blíže popsané v 3.3.3. Vychází z modelu master-slave. Tento standard IEC 60870-5 zahrnuje protokoly, které poskytují rámec pro komunikaci a výměnu dat mezi řídícím centrem a podřízenými stanicemi nebo mezi různými řídicími systémy. Protokol IEC 60870-5 se skládá ze sedmi dokumentů, kde nejdůležitější jsou:

- IEC 60870-5-1: Formáty přenosového rámce;
- IEC 60870-5-2: Procedury spojového přenosu;
- IEC 60870-5-3: Obecná struktura aplikačních dat;
- IEC 60870-5-4: Definice a kódování aplikačních informačních prvků;
- IEC 60870-5-5: Základní aplikační funkce. [7]

IEC komise 57 dále vytvořila komunikační standardy:

- IEC 60870-5-101: Společná norma pro základní úkoly dálkového ovládání;
- IEC 60870-5-102: Společná norma pro přenos integrovaných součtových hodnot v elektrizačních soustavách;
- IEC 60870-5-103: Společná norma pro informační rozhraní ochran;
- IEC 60870-5-104: Síťový přístup pro IEC 60870-5-101 používající normalizované transportní profily. [7]

IEC 60870-5-104 je nejdůležitějším komunikačním protokolem, stavěným na IEC 60870-5-101, aplikovaným v systémech SCADA [8]. Dle studií [8, 9] se navrhují IDS ve SCADA sítích pro IEC 60870-5-104. Nástroje na detekci průniku do sítě jsou schopné pozorovat a detekovat tok dat ve SCADA systémech. Principiálně

vyhledávají anomálie v chování a zlepšují tak kybernetickou bezpečnost protokolu IEC 60870-5-104.

3.3.1 Bezpečnostní slabiny protokolu IEC 60870-5-104

Dle studie *Intrusion Detection System* [8] se mohou potenciální slabiny a útoky z fyzické vrstvy projevit ve dvou druzích. Prvním je tzv. přenos zpráv v režimu prostého textu, kde je vysoká náchylnost na odposlouchávání a potenciálně i nechtěnou změnou přenesených dat. Způsob, jakým lze docílit zmíněné nežádoucí akce, je například útok man in the middle ke sběru dat, příkazů a dalších. Druhou slabinou je nedostatečný mechanismus určený pro autentizaci příkazů, jimiž jsou například ovládací či dotazcovací příkazy. Touto slabinou jsou pak útočníci schopní zajistit si neautorizovaný přístup do SCADA systému a narušit tak integritu informací (dat) či právě implantace man in the middle. Studie poukazuje na fakt, kterým je vážnost této slabiny a poskytnutí relativně snadného přístupu s rizikem velkého poškození systému.

3.3.2 Zhodnocení standardu IEC 60870

Standard se vyznačuje jako velmi schopný a propracovaný. Naráží však na různá úskalí v bezpečnosti. Navzdory těmto úskalím je hojně využívaný celosvětově. Existuje velké spektrum různých IDS které se snaží o zvýšení bezpečnosti přenosu dat do SCADA systémů a proto je velmi zajímavým a aktuálním pro řízení energetických sítí.

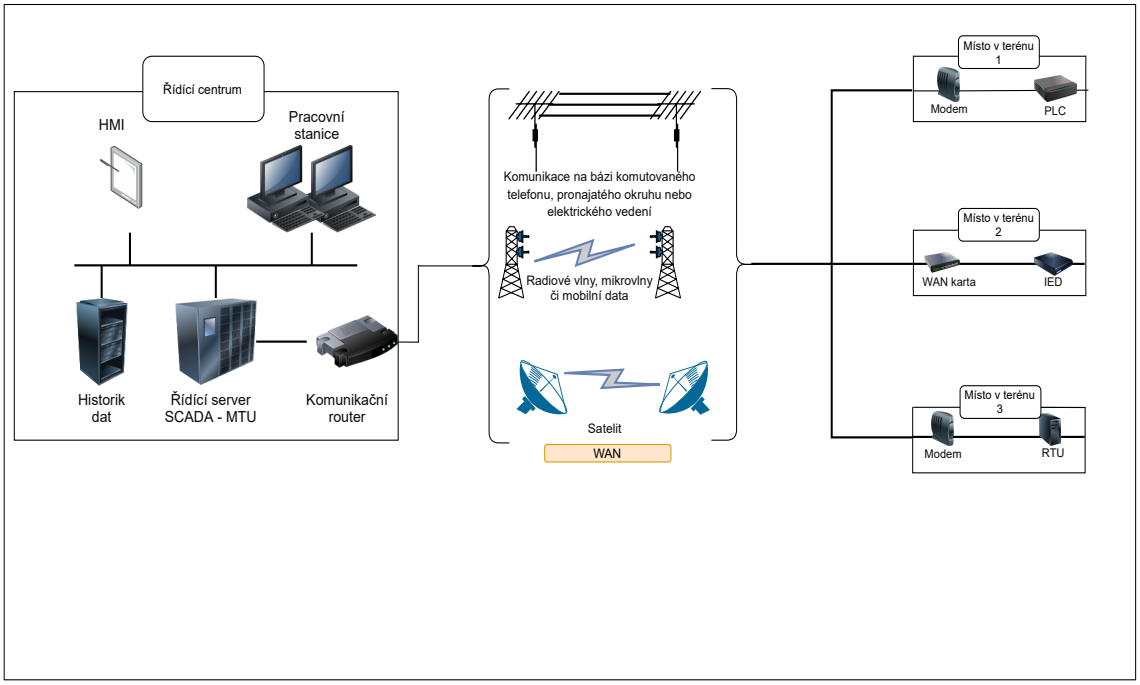
3.3.3 SCADA Systémy

Systémy SCADA se používají k řízení rozptýlených aktiv, kde je centralizovaný sběr dat stejně důležitý jako řízení. Tyto systémy se používají v distribučních systémech, jako jsou rozvody vody a systémy sběru odpadních vod, potrubí na ropu a zemní plyn, přenosové a distribuční systémy elektrické energie, železniční a jiné systémy veřejné dopravy.[1]

SCADA systémy integrují systémy sběru dat se systémy přenosu dat a softwarem HMI, aby poskytovaly centralizovaný monitorovací a řídicí systém pro četné procesní vstupy a výstupy. [1]

SCADA systémy jsou navrženy tak, aby shromažďovaly informace o terénu, přenášely je do centrálního počítačového zařízení a zobrazovaly informace operátorovi

graficky nebo textově, což operátorovi umožňuje monitorovat nebo ovládat celý systém z centrálního místa téměř v reálném čase. Na základě propracovanosti a nastavení jednotlivého systému může být řízení libovolného jednotlivého systému, operace nebo úkolu automatické nebo může být prováděno příkazy operátora.[1]



Obr. 3.1: Obecná konfigurace systému SCADA [1]

Obrázek 3.1 ukazuje součásti a obecnou konfiguraci systému SCADA. V řídicím centru je umístěn řídicí server a komunikační směrovače. Mezi další součásti řídicího centra patří HMI, inženýrské pracovní stanice a datový historik, které jsou všechny propojeny sítí LAN.[1]

Řídicí centrum shromažďuje a zaznamenává informace shromážděné v terénu, zobrazuje informace HMI a může generovat akce na základě zjištěných událostí.[1]

Řídicí centrum je také zodpovědné za centralizované alarmování, analýzy trendů a hlášení. Místo v terénu provádí místní ovládání akčních členů a monitoruje senzory. Pracoviště jsou často vybavena funkcí vzdáleného přístupu, která operátorům umožňuje provádět vzdálenou diagnostiku a opravy obvykle přes samostatný vytáčený modem nebo připojení WAN. Standardní a proprietární komunikační protokoly běžící přes sériovou a síťovou komunikaci se používají k přenosu informací mezi řídicím střediskem a provozními místy pomocí telemetrických technik, jako je telefonní linka, kabel, vlákno a rádiové frekvence a satelit. [1]

3.4 EtherNet/IP

EtherNet/IP je ethernetová komunikační síť, která uživatelům poskytuje nástroje pro nasazení standardní ethernetové technologie, jako je IEEE 802.3 v kombinaci se sadou TCP/IP Suite v aplikacích průmyslové automatizace a zároveň umožňuje připojení k internetu v podnikové konektivitě. [1]

EtherNet/IP nabízí různé možnosti topologie sítě, včetně hvězdicové nebo lineární se standardními zařízeními ethernetové infrastruktury nebo Device Level Ring (DLR) se speciálně aktivovanými zařízeními EtherNet/IP. [1]

Funkce QuickConnect umožňuje rychlou výměnu zařízení, zatímco síť běží. Shoda se standardy IEEE Ethernet poskytuje uživatelům výběr rychlostí síťového rozhraní, např. 10, 100 Mb/s a 1 Gb/s, a flexibilní síťovou architekturu kompatibilní s komerčně dostupnými možnostmi instalace Ethernetu, včetně metalických, optických a bezdrátových. Možnosti pro průmyslově hodnocená zařízení obsahující IP67 nebo lepší konektory s moduly a LED stavu sítě s označením zařízení umožňují snadné použití. [1]

EtherNet/IP pro své vyšší vrstvy využívá Společný průmyslový protokol (CIP). CIP sítě se řídí modelem OSI, který definuje rámec pro implementaci síťových protokolů v sedmi vrstvách: fyzická, datová linka, síťová, transportní, relační, prezentační a aplikační. Sítě, které se řídí tímto modelem, definují kompletní sadu síťových funkcí od fyzické implementace přes aplikační vrstvu nebo vrstvu uživatelského rozhraní. [1]

CIP zahrnuje komplexní sadu zpráv a služeb pro různé aplikace automatizace výroby, včetně řízení, bezpečnosti, zabezpečení, energie, synchronizace a pohybu, správy informací a sítě. Jako skutečně na médiích nezávislý protokol, který podporují stovky prodejců po celém světě, poskytuje CIP uživatelům jednotnou komunikační architekturu v celém výrobním podniku. [1] V tabulce 3.2 jsou vypsané běžné CIP objekty.

Kód objektu	**Popis**	**Potřebnost**
0x01	Objektová třída identity	Žádaný objekt
0x02	Objektová třída směrovače zpráv	Žádaný objekt
0x05	Objektová třída připojení	Žádaný objekt
0x04	Objektová třída sestavy	Volitelný objekt
0x0F	Objektová třída parametru	Volitelný objekt
0xNN	Objektová třída specifického odkazování sítě	Žádaný objekt
0x64	Objektová třída aplikace	Volitelný objekt

Tab. 3.2: CIP objekty [10]

Výměna dat v reálném čase na základě typu zařízení viz tabulky 3.3, 3.5 a 3.4. Implicitní zpráva je používána pro výměnu I/O dat. Je založena na UDP protokolu pro co nejrychlejší přenos a optimalizovaný výkon v závislosti na jeho nízkých režijních nákladech. Komunikační modul EtherNet/IP na PLC je původce a žádá data z cílových zařízení [11].

Funkce	**Typ paketu**	**Směr**	**Typ příkazu**
Konfigurační parametr	TCP	Cíl -> Původce	Unicast
Implicitní zpráva	UDP	Původce -> Cíl	Multicast
	TCP	Cíl -> Původce	Unicast
Explicitní zpráva	TCP	Cíl <-> Původce	Unicast

Tab. 3.3: PLC do I/O zařízení [11]

Funkce	**Typ paketu**	**Směr**	**Typ příkazu**
Explicitní zpráva	TCP	Cíl <-> Původce	Unicast

Tab. 3.4: HMI [11]

Explicitní zpráva je používána pro výměnu dat mezi zařízeními v síti EtherNet/IP, která nejsou cyklická a časově založená. Explicitní zprávy jsou založeny na TCP/IP, kde EtherNet/IP modul zasílá požadavky do cílového zařízení a cílové zařízení pak odpovídá na daný požadavek [11].

Funkce	**Typ paketu**	**Směr**	**Typ příkazu**
Implicitní zpráva	UDP	Cíl -> Původce	Multicast
	TCP	Původce -> Cíl	Unicast
Explicitní zpráva	TCP	Cíl <-> Původce	Unicast

Tab. 3.5: PLC do PLC [11]

4 Průmyslový protokol S7comm

S7comm, neboli Siemens S7 Communication Protocol, je průmyslový komunikační protokol používaný pro výměnu dat mezi řídicími systémy. [1]

4.1 S7comm

S7comm je proprietární protokol společnosti Siemens, který běží mezi programovatelnými logickými automaty (PLC).[1]

Používá se jako komunikační pro dorozumívání se s PLC, výměnu dat mezi PLC, přístup k PLC datům ze systémů SCADA (dohledové řízení a získávání dat) a pro diagnostické účely. [1]

OSI Vrstva	Protokol
7 Aplikační vrstva	S7comm
6 Prezentační vrstva	S7comm
5 Relační vrstva	S7comm
4 Transportní vrstva	ISO-on-TCP
3 Síťová vrstva	IP
2 Linková vrstva	Ethernet
1 Fyzická vrstva	Ethernet

Obr. 4.1: S7comm na referenčním modelu ISO/OSI [1]

4.1.1 Použití a rozdělení protokolu S7comm

Protokol S7comm se ve všech jeho verzích používá ke komunikaci Siemens PLC mezi sebou a TIA portálem. Všechny verze protokolů využívají TPKT (Transport paket) a transportního protokolu dle ISO8073 včetně portu 102/TCP. [1]

- S7Comm protokol
- Early S7CommPlus protokol
- S7CommPlus protokol

S7Comm protokol se využívá je komunikaci mezi PLC S7-200, S7-300 a S7-400. Tento protokol nezahrnuje žádný mechanismus proti útokům a je snadno zneužitelný. Early S7CommPlus protokol se používá při komunikaci mezi PLC S7-1200v3.0 a je

komplikovanější než S7Comm protokol, avšak je snadno prolomitelný. S7 protokol má následující výhody:

- Nezávislost na sběrnici (Profibus, Industrial Ethernet)
- Lze použít ve všech oblastech S7;
- Možnost přenosu až 64kB;
- Automatické potvrzení datových záznamů;
- Nízké zatížení procesoru a sběrnice při přenosu velkých objemů dat. [1]

4.1.2 Komunikace S7comm

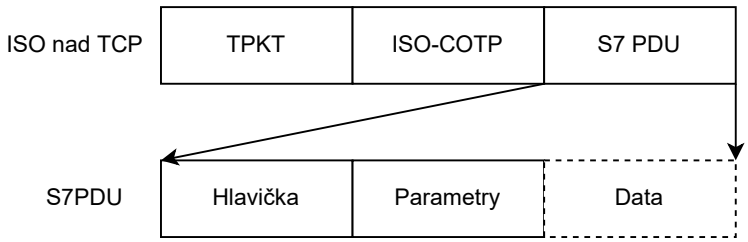
Při komunikaci se zařízeními S7 existuje celá rodina protokolů, které lze použít. Obecně je lze rozdělit na protokoly Profinet a protokoly S7 Comm. Protokoly S7 Comm mají mnohem jednodušší strukturu, ale také jsou mnohem méně zdokumentované.[1]

S7-PDU

Protokol S7 se skládá z bloků. Každý blok se jmenuje PDU (Protocol Data Unit), jeho maximální délka závisí na komunikačních procesorech (CP) a je vytvářena během spojení. Protokol S7 je funkčně nebo příkazově orientovaný, to znamená, že každý přenos obsahuje příkaz nebo odpověď. Pokud se velikost příkazu nevejde do PDU, musí být rozdělena mezi další následující PDU. [1]

Každý příkaz se skládá ze záhlaví, sady parametrů, data o parametrech a datovém bloku viz. obrázek 4.2, kde:

- Záhlaví (hlavička): obsahuje délkovou informaci, PDU reference a konstantu typu zprávy
- Parametry: Obsah a struktura parametrů se liší v závislosti na typu zprávy a funkci PDU
- Data: Volitelné pole k přenosu dat, pokud nějaká existují, např. hodnoty pamětí, kódy bloků, data firmwaru. [1]

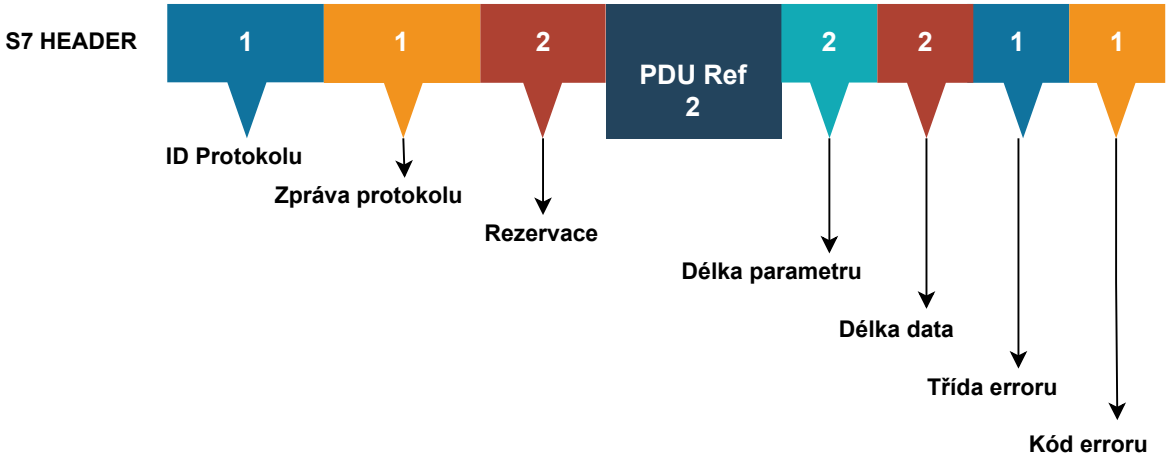


Obr. 4.2: S7-PDU [1]

Záhlaví S7 HEADER

Hlavička se sestává z 10-12 bajtů, potvrzovací zprávy obsahují dva extra errorové bajty viz. obr. 4.3. Mimo to je formát záhlaví stejný pro všechna PDU.

- ID protokolu: [1B] konstanta protokolu je vždy 0x32
- Zpráva protokolu: [1B] obecný typ zprávy
  - 0x01 - Požadavek: zaslaný tzv. masterem (nadřazeným) zařízením, jedná se např. o čtení/zápis paměti, čtení/zápis bloků, start/stop zařízení, nastavení komunikace
  - 0x02 - Ack: Potvrzení zaslané tzv. slave (podřízeným) zařízením bez datového pole
  - 0x03 - Ack-Data: Potvrzení s volitelným datovým polem, obsahuje zpětnou vazbu na požadavek
  - 0x07 - Uživatelská data: Rozšíření původního protokolu, pole parametru obsahuje ID požadavku/zpětné vazby - využívá se pro debugging (ladění programu), bezpečnostní funkce či nastavení času.
- Rezervace: [2B] vždy nastavena jako 0x0000
- PDU reference: [2B] generovaný masterem, navýšený s každým novým přenosem, používané pro propojení zpetné vazby a požadavků
  - Little-endian - druh endianity, jedná se o chování SIMATIC WinCC, Step7 a dalších Siemens programů
- Délka parametru: [2B] Délka pole parametru
  - Big-endian - druh endianity
- Datová délka: [2B] Délka datového pole
  - Big-endian - druh endianity
- Třída erroru: [1B] Vyskytuje se pouze v Ack-data, možné errory viz. tabulka 4.1.
- Kód erroru: [1B] Vyskytuje se pouze v Ack-data, výběr errorů viz. tabulka 4.2 [1]



Obr. 4.3: Header [1]

Kód erroru	Popis
0x00	Žádná chyba
0x81	Chyba vztahu aplikace
0x82	Chyba definice objektu
0x83	Chyba - k dispozici nejsou žádné zdroje
0x84	Chyba při zpracování služby
0x85	Chyba na zásobách
0x87	Chyba přístupu

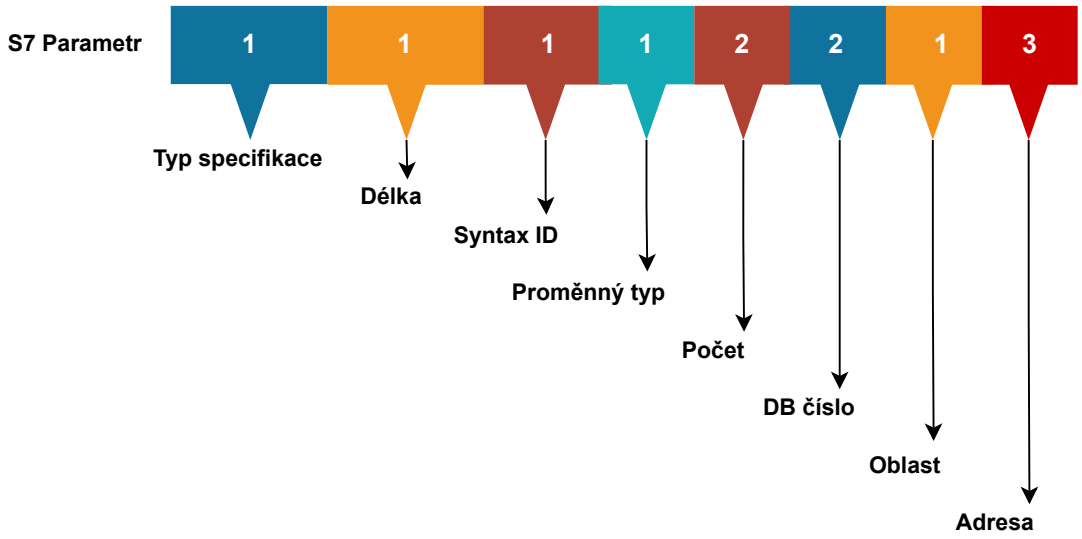
Tab. 4.1: Errory záhlaví [1]

Kód erroru	Popis
0x0000	Žádná chyba
0x0110	Neplatné číslo typu bloku
0x0112	Neplatný parametr
0x011A	Chyba zdroje PG
0x011B	Chyba zdroje PLC
0x011C	Chyba protokolu
0x011F	Uživatelská vyrovnávací paměť je příliš krátká
0x0141	Chyba požadavku
0x01C0	Neshoda verzí
0x01F0	Neimplementováno

Tab. 4.2: Errory parametru [1]

S7 Parametr

V následujícím obrázku 4.4 je vyobrazena struktura parametru S7. [1]



Obr. 4.4: Struktura parametru S7 [1]

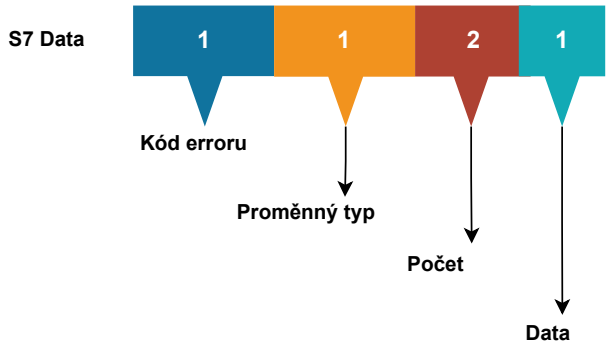
- Typ specifikace: [1B] Vyjadřuje hlavní typ struktury parametru, pro čtení/zápis se využívá hodnota 0x12 což znamená *Proměnná specifikace*
- Délka: [1B] Délka parametru
- Syntax ID: [1B] Vyjadřuje režim adresování a formát zbytku struktury, adresování libovolného typu využívá hodnotu 0x10
- Proměnný typ: [1B] Využívá se při určení typu a délky proměnné
- Počet: [2B] Vyjadřuje možnost zvolení celého pole podobných proměnných pomocí jediné struktury. Tyto proměnné musí mít stejný typ a musí být po sobě jdoucí v paměti a pole určuje velikost. Zpravidla je nastaveno na 1 pro čtení/zápis jedné proměnné.
- DB číslo: [2B] adresa databáze
- Oblast: [1B] Vybírá oblast paměti v adresované proměnné, výběr konstant viz tabulka 4.3
- Adresa: [3B] Obsahuje offset adresované proměnné ve vybrané oblasti paměti. Adresy jsou v podstatě přeloženy na bitové offsety a zakódovány na 3 bajty v síťovém pořadí bajtů (big endian)

Konstanty oblasti paměti	Popis
0x03	Informace o systému rodiny S200
0x05	Systémové příznaky rodiny S200
0x06	Analogové vstupy řady S200
0x07	Analogové výstupy rodiny S200
0x1C	Čítače S7 (C)
0x1D	Časovače S7 (T)
0x1E	IEC čítače (rodina 200)
0x1F	IEC časovače (rodina 200)
0x80	Přímý periferní přístup (P)
0x81	Vstupy (I)
0x82	Výstupy (Q)
0x83	Vlajky (M)
0x84	Datové bloky (DB)
0x85	bloky dat instance (DI)
0x86	Místní data (L)
0x87	Dosud neznámé (V)

Tab. 4.3: Oblasti paměti [1]

S7 Data

Níže je vyobrazena struktura S7 dat viz obr. 4.5. [1]



Obr. 4.5: Struktura dat S7 [1]

Kde jednotlivé části struktury značí:

- Kód erroru: [1B] vrací hodnotu operace, 0xff značí úspěch
- Proměnný typ a počet: [1B 2B] Stejné jako v parametru
- Data: toto pole obsahuje reálnou hodnotu adresované proměnné

4.1.3 Zabezpečení protokolu S7comm

Protokol S7 je díky jeho vlastnostem, konkrétně jeho proprietaritě a prvotním zapouzdření dat do COTP (Connection-Oriented Transport Protocol - transportní protokol orientovaný na připojení) protokolu, následném zapouzdření do TPKT protokolu a posléze umožnění PDU odeslání dat skrze TCP/IP. [1]

Model detekce vniknutí

Základními síťovými bezpečnostními prvky jsou výrobní firewally a detekční metody. Bezpečnostní prvky jsou spíše pasivní obranou načež detekční metody jsou obranou aktivní. Detekční metody, jak z názvu plyne, aktivně sledují výrobní síť a v případě zjištění jakékoliv abnormality o ní informuje.[1]

Ve své podstatě se dělí na dvě metody - metoda detekce zneužití a metoda detekce anomálií. První zmíněná metoda funguje na principu tzv. blacklistu, což je protokol, který má zabezpečení nastavené na zákazu konkrétních entit, které potenciálně mohou být nebezpečné. Charakteristickým znakem je zaměření na konkrétní hrozby, tzn. je třeba znalost takových hrozeb. Druhá metoda funguje na opačném principu a to na tzv. whitelistu, což je opakem blacklistu. Umožňuje přístup pouze vybraným entitám, které nejsou považované za hrozbu. Z toho vyplývá, že charakteristickým znakem whitelistu je zaměření na důvěryhodné přístupy. Mimo whitelist se metoda detekce abnormality věnuje chování komunikace v systému a opět v případě abnormálního komunikačního chování informuje o takové skutečnosti.[1]

4.1.4 S7comm Plus

Tato nová verze protokolu je vysoce šifrovaná, což znemožňuje analýzu paketů. Neexistuje žádná oficiální dokumentace protokolu a málo nebo žádné informace o něm online.[1]

Siemens S7-1200v4.0 a S7-1500 používají tento nový protokol S7CommPlus včetně paketů S7CommPlus Connection a paketů S7CommPlu Function. Všechny pakety používané protokolem S7CommPlus mají podobnou strukturu a má komplexní šifrování vůči opakovanému útoku. S7CommPlus se indentifikuje číslem protokolu 0x72. Existují tři verze S7CommPlus, a to S7CommPlusV1, S7CommPlusV2
a S7CommPlusV3, kde poslední verze se považuje za nejzabezpečenější vůči dvěma předchozím. Třetí verze se využívá výhradně u TIA portálu V13 a výše. [1]

S7comm Plus komunikace

K navázání spojení mezi TIA Portalem a PLC bylo použito třícestné handshake TCP spojení. Po připojení COTP (Connection Oriented Transport Protocol) odešle TIA

Portal požadavek na připojení S7CommPlus. První paket S7CommPlus Connection Response obsahuje ID objektu a pole hodnot, které generuje PLC. Při obdržení ID objektu a pole hodnot vypočítá TIA Portal ID relace a blok klíče. Poté se do PLC odešle druhý paket s požadavkem na připojení S7CommPlus obsahující ID relace a blok klíče. Pokud jsou ID relace a blok klíče správné, po ověření PLC bude odeslán paket s odpovědí, aby bylo spojení S7CommPlus dokončeno. Každý paket požadavku funkce S7CommPlus obsahuje část integrity. Část integrity vypočítá TIA Portal pomocí ID relace a pevné hodnoty pole jako vstupního parametru. Když PLC přijme paket S7CommPlus Function Request, bude ověřena integrita části. Paket S7CommPlus Function Response může být odeslán pouze v případě, že ověření bylo správné.[1]

4.1.5 Zabezpečení protokolu S7comm Plus

Nejbezpečnějším dnešním protokolem je S7CommPlusV3 který je používán pouze na nejnovějších TIA portálech a S7-1500 PLC. Podporuje různé operace, které jsou zpracované softwarem TIA portálu, jimiž jsou:

- Start/Stop aktuálně načteného ovládacího programu v paměti PLC;
- Stažení řídícího programu do PLC;
- Nahrání aktuálního řídícho programu z PLC do TIA portálu;
- Čtení hodnoty řídící proměnné;
- Upravení hodnoty řídící proměnné. [1]

Výše uvedené operace překládá software TIA portálu do S7CommPlus zpráv před tím, než jsou vysílané do PLC. PLC se následně chová dle zpráv, které obdrží, zpracuje řídící operace a provede zpětnou vazbu do TIA portálu. Zprávy jsou přenášeny v relaci, kde každá relace má ID které volí PLC. Relace začíná tzv. handshakem čtyř zpráv, které se používá k výběru kryptografických atributů relace včetně verze protokolu a klíčů. Po handshaku jsou všechny zprávy chráněny integritou využívající kryptografický ochranný mechanismus.[1]

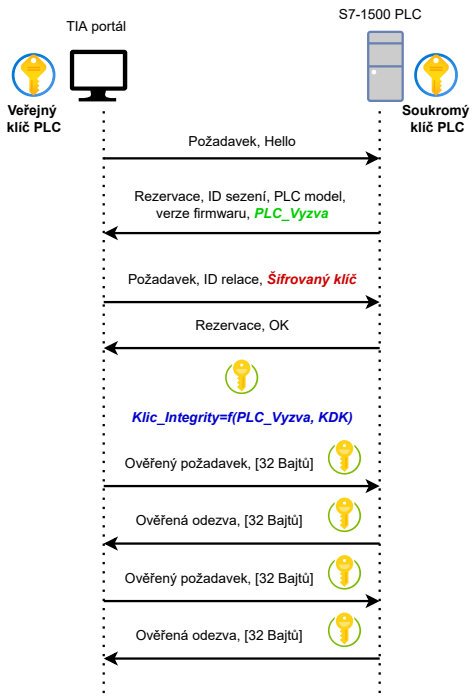
S7 mechanismus ochrany integrity

Siemens integroval kryptografickou ochranu v nejnovějším S7 proprietárním protokolu aby zajistil ochranu PLC před neautorizovanými přístupy. Tento nový mechanismus využívá dva hlavní moduly:

- Protokol výměny klíčů relace, které využívají obě strany (PLC a TIA portál) k zajištění tajného sdíleného klíče pro každé sezení (session).
- Fragmentová ochrana zpráv, která vypočítá hodnotu Message Authentication Code (MAC).[1]

Protokol výměny klíčů

Protokol výměny klíčů byl vylepšen nahrazením procesu generování klíče v předchozí verzi, tj. S7CommPlusV2, složitějším procesem pro novější verze S7CommPlusV3. Tento nový mechanismus zahrnuje novou techniku výměny klíčů na základě veřejného klíče s metodou šifrování založeném na kryptografii nad eliptickými křivkami jak je znázorněno na obr. 4.6.[1]



Obr. 4.6: Mechanismus vzniku relačního klíče S7 [1]

První požadavek je tzv. Hello zpráva, kterou TIA portál inicializuje novou relaci. Poté PLC odpoví a sdílí firmware verzi, model, ID relace a konkrétních 20 bajtů známých jako ***PLC_Vyzva***. Verze PLC firmwaru určuje eliptickou křivku při výměně veřejných klíčů. Poté, co TIA portál obdrží druhou zprávu z PLC, aktivuje derivační algoritmus k náhodnému vybrání KDK (key derivation key - klíčový derivační klíč) a generuje klíč relace z ***PLC_Vyzva*** a vybraného KDK. Následně TIA portál vysílá šifrovaný klíč pomocí kryptografické eliptické křivky do PLC skrze třetí zprávu. Třetí zpráva, mimo jiné, obsahuje dvě hlavní části:

- Datová struktura obsahující vybrané šifrované klíče s veřejným klíčem PLC
- Dva osmi bajtové klíčový otisk (přídavný klíč) ID veřejného klíče PLC a vybraného klíče. [1]

Posléze PLC zkontroluje třetí zprávu a v případě, že je kontrola úspěšná, vrátí informaci OK skrze čtvrtou zprávu a odsud veškeré následující zprávy v relaci jsou

chráněné integritou s derivačním klíčem relace.[1]

Fragmentová ochrana zpráv

Když TIA portál stáhne/nahraje řídící logický program do/z S7-1500 PLC, přiřazené S7CommPlus zprávy jsou fragmentované do vícero menších fragmentů odeslané skrce TCP/IP pakety. Všechny zprávy zaslané mezi oběma stranami jsou chráněny integritou HMAC-SHA256, což je kryptografický algoritmus hash klíčovaného klíče. Taková ochrana integrity je aplikována na úrovni fragmentace, tzn. nahrazuje hodnotu signálu MAC na konci každé zprávy a kryptografický obsah je umístěn v každém fragmentu mezi jeho záhlaví a data. Ačkoliv fragmentace S7 zpráv je pro útočníky výzvou, tak tenhle ochranný mechanismus byl prolomen. Útočníci mohli implementovat tzv. man-in-the-middle (prostředníka) přístup a úspěšně zvládli modifikovat síťový provoz na portu 102/TCP kvůli určitým vlastnostem v kalkulaci pro tuto ochranu integrity.[1]

Stahování S7CommPlus zpráv - objekty a atributy

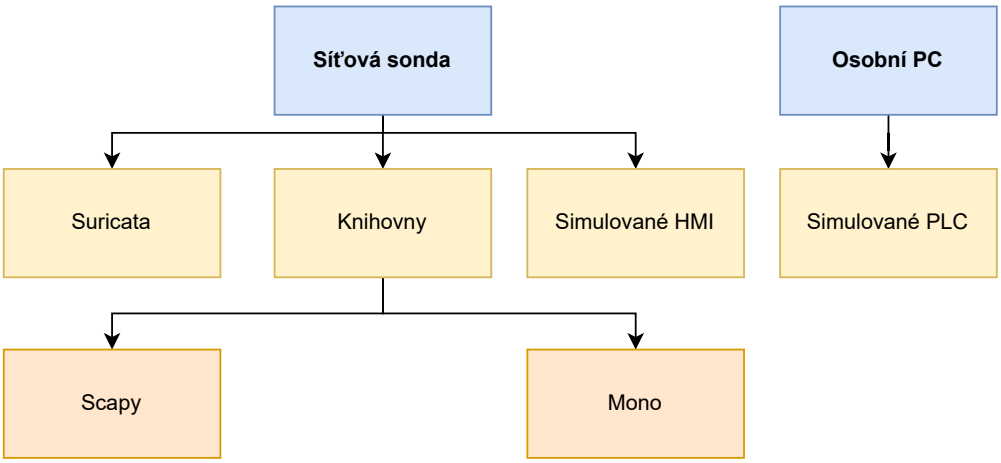
S7 je protokol na bázi odpovědi na požadavek. Každý požadavek zprávy se skládá ze záhlaví požadavku a sady požadavku. Záhlaví obsahuje funkční kód který identifikuje žádanou operaci (např. 0x31 pro stažení zprávy). Jedna S7CommPlus zpráva může obsahovat vícero objektů, kde každý může obsahovat vícero atributů. Všechny objekty a atributy mají unikátní identifikátor třídy. Nicméně požadavek 0x04 (Vytvoření objektu) staví nový objekt v paměti PLC s unikátním ID. Program po stažení zprávy vytvoří objekt třídy tzv. *ProgramCycleObjectBlock*. Tento objekt obsahuje vícero atributů, kde každý z nich má hodnoty určené ke specifickému účelu. Z pohledu zabezpečení jsou tyto atributy kritické data, která jsou vysílány přes S7CommPlusV3 protokol. Z toho vyplývá, že pokud útočník zachytí S7 pakety obsahující tyto atributy, může je jednotlivě modifikovat a v případě správné modifikace je schopen způsobit zdrojově-binární nesoudržnosti.[1]

5 Analýza průmyslového protokolu

Analýza vybraného průmyslového protokolu S7 je dělena do dvou dílčích částí, a to na modul, a sadu pravidel, která jsou vytvořena k detekci zvolených útoků (viz podkapitola 7.2), jež jsou zásadní pro analýzu zachycení uvedených útoků. Pro zpracování analýzy bylo zapotřebí zajistit pracoviště s vybranými zařízeními (viz podkapitoly 5.1 a 5.2), použití knihovny Scapy.

5.1 Vymezení pracoviště

Pracoviště se skládá ze dvou hlavních zařízení, jimiž je síťová sonda a osobní PC. V rámci síťové sondy jsou nainstalované prvky Suricata, simulované HMI a knihovny Scapy a Mono. Knihovna Scapy je určená pro zachytávání paketů a prvek simulované HMI využívá specifickou knihovnu Mono, aby bylo zajištěno spuštění tohoto prvku v prostředí Linux. Na osobním PC je nainstalovaný prvek simulovaného PLC viz obrázek 5.1.

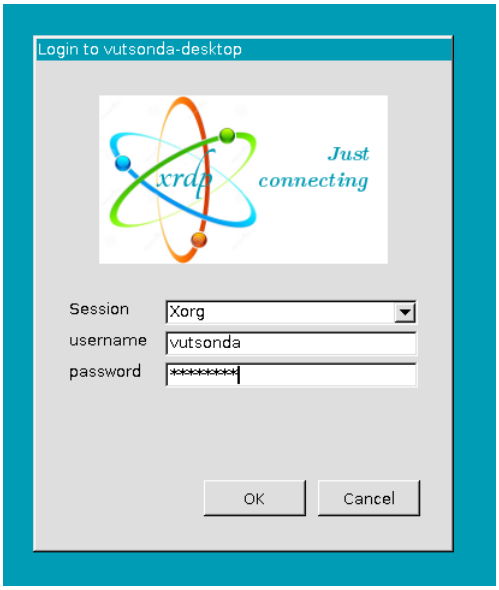


Obr. 5.1: Pracovní prostředí (vlastní zpracování)

Existují dvě verze pracoviště, a to pracoviště s reálným PLC a HMI, a již zmíněné pracoviště se simulovanými zařízeními. Pracoviště s reálnými prvky bylo zprvu využito a zprovozněno. Jelikož tato verze vyžadovala fyzickou přítomnost a s tím spojenou časovou náročnost v zázemí laboratoře (PLC a HMI nebylo vždy k dispozici), byla zvolena verze se simulovanými prvky, která předcházela zmíněnému problému.

5.2 Síťová sonda

Síťová sonda je určená k vyhodnocení událostí a sběr dat, tj. monitorování a analýze protokolů. Pro potřeby závěrečné práce je nutné zprovoznění síťové sondy. Přihlášení k síťové sondě probíhá skrze připojení VPN (existují i jiné způsoby). IP adresa sondy, ke které se přihlašovalo je 192.168.50.189. V přihlašovacím okně viz obrázek 5.2 se zadají přihlašovací údaje a umožní tak přístup do uživatelského prostředí.



Obr. 5.2: Přihlášení do sondy (vlastní zpracování)

V uživatelském prostředí (na bázi operačního systému Linux) lze přistoupit, popř. vkládat nové moduly, kam byl také vložen modul pro protokol S7, který se blíže rozebírá v kapitole 6. Bližší informace týkající se síťové sondy jsou uvedeny v diplomové práci Jana Klečky s názvem *Monitorovací sonda síťové komunikace* z roku 2020 [23].

5.2.1 Scapy

Scapy je knihovna určená pro manipulaci s počítačovými sítěmi, která je napsaná v jazyce Python. Pro potřeby diplomové práce umožňuje vytvářet, manipulovat, odesílat a přijímat síťové pakety. Dále umožňuje analýzu paketů, testování síťových zařízení a protokolů, vytváření vlastních síťových nástrojů a útoků, nebo diagnostiky a ladění sítí. Ze zmíněných využitelných možností bylo využito umožnění vytvoření a přidání vlastního protokolu, tj. rozšíření o daný protokol a následnou analýzu přenesených paketů.

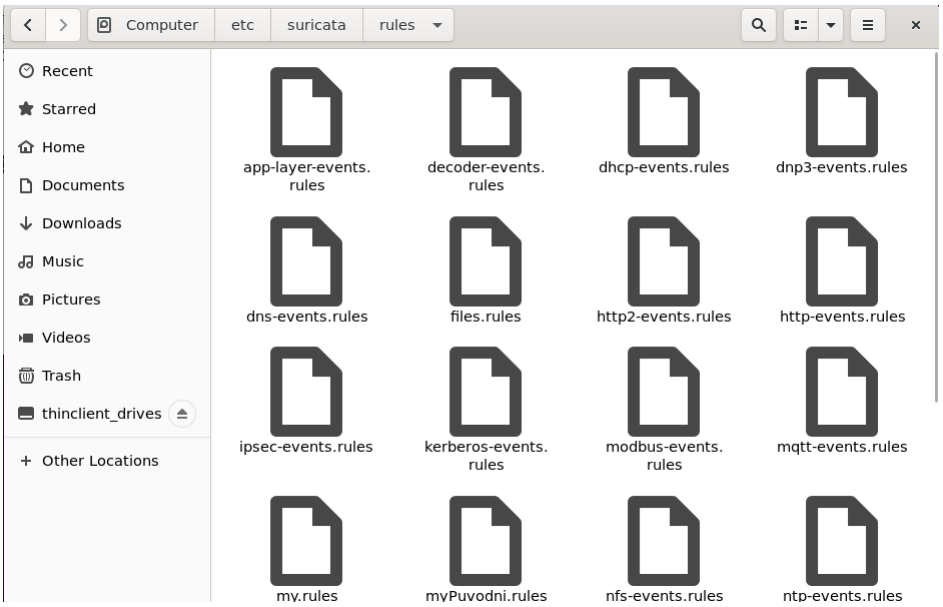
5.2.2 Suricata v síťové sondě

Suricata v síťové sondě byla již předinstalovaná a obsahovala již nějaké sady pravidel, k nimž se v kapitole 7 přidala další sada pravidel přímo pro detekci útoků S7. Suricata uchovává sady pravidel ve složce viz výpis 5.1

```
/etc/suricata/rules
```

Výpis 5.1: Lokace pravidel v Suricatě (vlastní zpracování)

V této složce viz obrázek 5.3 lze vytvořit novou sadu pravidel s příponou .rules.



Obr. 5.3: Složka pravidel Suricaty (vlastní zpracování)

V nadřazené složce složky rules je soubor *suricata.yaml*, který obsahuje nastavení Suricaty. V tomto souboru jsou definované dva druhy výstupů logu, konkrétně výstup *fast* a výstup *eve-log* viz výpis 5.2. Výstup *fast* je s příponou .log a je snáze čitelnější oproti výstupu *eve-log*, jež je ve formátu .json. Tato složka také obsahuje soubor *suricata-start.log*, který zaznamenává základní funkce Suricaty, avšak není podmíněn výstupu dle uvedeného výpisu.

```
outputs:
  - fast:
      enabled: yes
      filename: fast.log
      append: yes
  - eve-log:
      enabled: yes
      filetype: regular
      filename: eve.json
```

Výpis 5.2: Výstup logů na základě pravidel (vlastní zpracování)

Soubor *suricata.yaml* také obsahuje, mimo jiné, nastavení rozhraní, na kterém Suricata naslouchá a zpracovává pakety na základě výpisu 5.3, kde *interface* definuje název rozhraní (může být zadefinováno vícero rozhraní), *cluster-id* definuje identifikátor, do kterého jsou pakety přiřazeny, *cluster-type* definuje způsob, jakým jsou data rozdělovány mezi různé instance či jádra CPU pro pozorování a *defrag* povoluje či zakazuje defragmentaci paketů.

```
af-packet:
  - interface: enx00e04c3a593a
    cluster-id: 99
    cluster-type: cluster_flow
    defrag: yes
```

Výpis 5.3: Nastavení rozhraní Suricaty (vlastní zpracování)

Ve výpisu 5.4 je znázorněna konfigurace Suricaty ve smyslu použití typu sady pravidel. V původním nastavení byla použita sada pravidel *suricata.rules*.

```
default-rule-path: /etc/suricata/rules/

rule-files:
  - suricata.rules
```

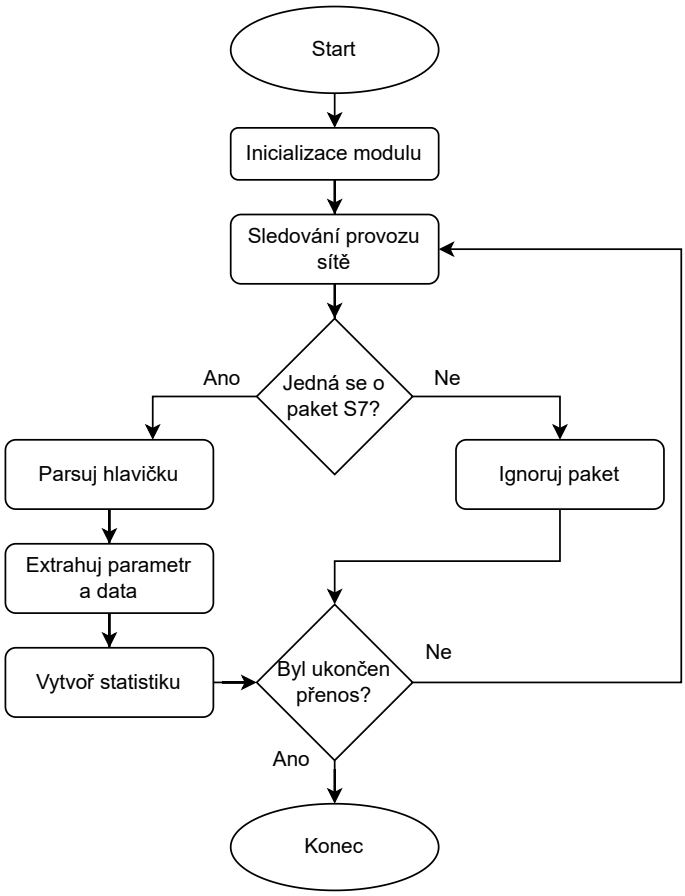
Výpis 5.4: Konfigurace pravidel (vlastní zpracování)

6 Modul pro protokol S7

Modul pro protokol S7 je dílčí částí závěrečné práce, jež bylo nutné vytvořit (implementovat) a následně otestovat s cílem zjistit, zdali je modul funkční a odpovídá komunikaci S7. Modul má za cíl parsovat protokol S7 a vytvářet statistiky, mezi něž se řadí:

- délka spojení,
- informace o paketu (číslo data bloku, typ zprávy, chybové hlášení, jaká data jsou přenášena),
- celkový počet zachycených paketů

Vývojový diagram na obrázku 6.1 znázorňuje proces zpracování dat skrze modul pro protokol S7.



Obr. 6.1: Vývojový diagram modulu (vlastní zpracování)

Proces začíná spuštěním modulu, kde je spuštěn s již zadefinovanými parametry viz podkapitola 6.1 a sleduje síťový provoz. Během sledování provozu vyhledává

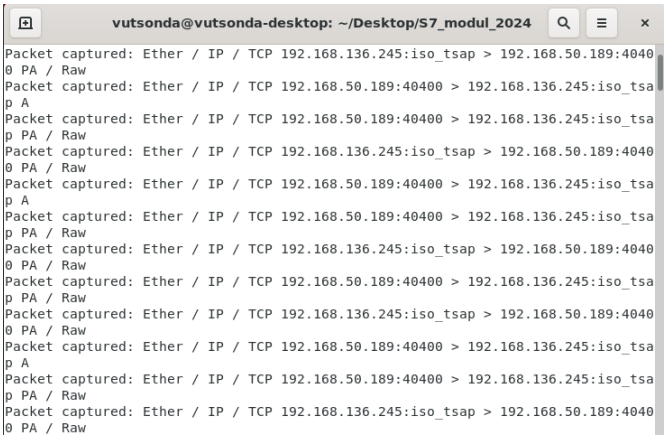
pakety k zachycení a v případě detekce paketu rozhoduje, zdali se jedná o paket protokolu S7. Pokud ano, modul pokračuje v procesu parsování hlavičky paketu, kde získává základní informace o paketu, jako je délka parametru a dat. Po úspěšném parsování hlavičky přechází modul k extrakci parametrů a dat z paketu. Zde získá čísla databloku, délku zprávy a typ zprávy, které jsou klíčové pro další analýzu. Na základě těchto získaných dat modul vytváří, které jsou poté uloženy pro další použití. Dalším krokem ověřuje, zda byl přenos dat dokončen. Pokud ano, proces končí. V případě, že přenos nebyl dokončen, modul se vrací k monitorování sítě a čeká na další pakety k analýze. Tento cyklus běží, dokud není přenos ukončen.

6.1 Vytvoření modulu

Vytvoření modulu pro zachytávání paketů protokolu S7 pomocí knihovny Scapy započalo vytvořením kódu (viz výpis 6.1), který byl navržen tak, aby sledoval síťový provoz a identifikoval pakety, které obsahují vrstvu TCP na portu 102, což je standardní port pro protokol S7.

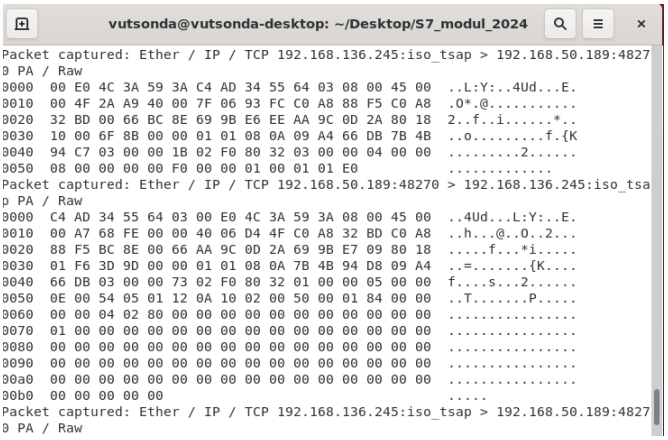
```
1  #Import potřebných modulů z knihovny Scapy
2  from scapy.all import sniff, TCP
3
4  #Funkce handle_packet která se zavolá při zachycení každého paketu
5  def handle_packet(packet):
6      if packet.haslayer(TCP):
7          print(f"Packet captured: {packet.summary()}")
8
9  #Funkce Sniff slouží k zachycení paketů na portu 102
10 sniff(filter="tcp port 102", prn=handle_packet, count=0)
```

Výpis 6.1: Zachytávání surového paketu (vlastní zpracování)



Obr. 6.2: Surové pakety (vlastní zpracování)

Výpis 6.1 ukazuje výstup našeho modulu při zachytávání surových paketů. Každý zachycený paket je zobrazen s informacemi o Ethernet vrstvě, IP adresách, TCP portech a dalších detailech. Je důležité zmínit, že Scapy nezná protokol S7 v základním nastavení a proto jsou některá data označená jako „raw," čili surové pakety. Obrázek 6.2 zobrazuje konzoli s výstupy zachycených surových paketů. Je zde vidět podrobné informace o každém paketu, což je klíčové pro analýzu síťového provozu a identifikaci potenciálních problémů nebo anomálií. Pomocí tohoto modulu je možné efektivně sledovat síťový provoz, identifikovat potenciální pakety protokolu S7 a získávat cenné statistiky. Obrázek 6.3 znázorňuje hexadecimální zobrazení payloadu



Obr. 6.3: Hexadecimální zobrazení payloadu (vlastní zpracování)

zachycených paketů, čehož se docílilo pomocí přidání funkce *hexdump* a její použití skrze *hexdump(packet)* do výpisu 6.1. Toto zobrazení poskytuje detailní pohled na obsah dat přenášených v jednotlivých paketech. V hexadecimálním výpisu lze vidět nejenom data, ale také strukturu přenášených informací na úrovni bajtů. Hexadecimální zobrazení poskytuje možnost analýzy dat a zobrazení hodnot každého bajtu, což na základě znalosti struktury paketu S7 protokolu umožňuje jeho indentifikaci. Vytvoření tohoto modulu je prvním krokem k implementaci komplexnějších funkcí pro zpracování a analýzu dat protokolu S7.

Po zjištění struktury surových dat je nutné oddělit bajty S7 protokolu od TPKT a COTP bajtů. TPKT a COTP bajty nejsou důležité pro potřeby analýzy protokolu S7, jelikož tyto bajty zajišťují pouze správný přenos dat na vrstvě TCP. Důvodem, proč se separují bajty S7 pouze od TPKT a COTP bajtů je, že předchozí vrstvy jsou pro Scapy knihovnu známé a nejsou součástí surových dat. Výpis 6.2 zahrnuje vybranou třídu, která parsuje z payloadu údaje parametru bajtu S7 protokolu. Nejdůležitějšími údaji parametru jsou *function*, která představuje typ zprávy (čtení či zápis) a *data_block*, který označuje číslo data bloku, jež je funkční. Další třídy jsou uvedeny v příloze A.

```
class S7COMM_Para(Packet):
    name = "S7COMM_Para"
    fields_desc = [
        ByteField("function", 0),
        ByteField("item", 0),
        ByteField("variable", None),
        ByteField("lfas", 0),
        ByteField("syntax_id", None),
        ByteField("transport_size", 0),
        ShortField("lenght", 0),
        ShortField("data_block", 0),
        ByteField("area", None),
        ThreeBytesField("address", None)
    ]
```

Výpis 6.2: Definování parametru protokolu S7 (vlastní zpracování)

Výpis 6.3 ukazuje, jakým způsobem jsou jednotlivé vrstvy protokolu S7 propojeny pro účely správného zpracování paketu. Ve výpisu je použitá funkce *bind_layers* k propojení dvou vrstev protokolu. V prvním řádku je znázorněno propojení základní vrstvy protokolu S7 s vrstvou *S7COMM_Header*, která obsahuje hlavičku S7 komunikace. V druhém řádku je propojení hlavičky s vrstvou *S7COMM_Para* za podmínky, že *rosctr* má hodnotu 1, což je specifikace typu komunikace. Třetí řádek propojuje vrstvu *S7COMM_Para* s vrstvou *S7COMM_Data*, jež obsahuje data přenášená v rámci S7 protokolu. Další propojení zahrnují specifické podmínky pro různé typy zpráv a propojení TCP vrstvy s protokolem S7. Pro účely testování byl zadefinován port 102, jelikož na něm standardně probíhá protokol S7. Tímto propojením vrstev bylo zajištěno správné rozpoznávání a zpracování parametru podle své struktury a obsahu. Toto propojení umožňuje dekódování a analýzu S7 komunikace v síti.

```
bind_layers(S7, S7COMM_Header)
bind_layers(S7COMM_Header, S7COMM_Para, rosctr = 1)
bind_layers(S7COMM_Para, S7COMM_Data)
bind_layers(S7COMM_Header, S7COMM_Ack, rosctr = 3)
bind_layers(S7COMM_Ack, S7COMM_Ack_Para)
bind_layers(S7COMM_Ack_Para, S7COMM_Ack_Data)
bind_layers(TCP, S7, dport=102)
bind_layers(TCP, S7, sport=102)
```

Výpis 6.3: Propojení vrstev protokolu S7 (vlastní zpracování)

Ve výpisu 6.4 je definovaná funkce *handle_packet(packet)*, která zpracovává každý zachycený paket. Funkce kontroluje, zda paket obsahuje TCP vrstvu skrze *packet.haslayer(TCP)*. Pokud ano, extrahuje se zdrojová IP adresa (*src*), cílová IP adresa (*dst*), zdrojový port (*sport*) a cílový port (*dport*) z IP a TCP vrstev paketu.

Následně se vytvoří klíč (*klic*), který je tvořen kombinací *src*, *dst*, *sport* a *dport*. Tento klíč umožňuje identifikovat konkrétní TCP spojení. Posléze probíhá kontrola, zda paket obsahuje *TCP flag S (SYN)* a neobsahuje *TCP flag A (ACK)*. Pokud ano, pak se jedná o začátek nového spojení, a uloží se aktuální čas do proměnné pod vytvořeným klíčem. Dále probíhá kontrola, zda paket obsahuje *TCP flag F (FIN)*, což znázorňuje ukončení spojení. Pokud paket obsahuje tento flag, a zároveň je klíč již v proměnné, pak je získán čas začátku spojení a vypočítá se délka spojení jako rozdíl mezi aktuálním časem a časem začátku spojení. Tato délka se vypíše a nakonec se klíč odstraní ze slovníku z důvodu ukončení spojení. Tento výpis tak umožňuje sledovat délku spojení S7 komunikace, což zajišťuje přístup k informacím o době trvání spojení mezi zařízeními v síti.

```
def handle_packet(packet):
    if packet.haslayer(TCP):
        src = packet[IP].src
        dst = packet[IP].dst
        sport = packet[TCP].sport
        dport = packet[TCP].dport
        klic = (src, dst, sport, dport)
        if 'S' in packet[TCP].flags and not 'A' in packet[TCP].
    flags:
            spojeni[klic] = datetime.now()
        if 'F' in packet[TCP].flags:
            if klic in spojeni:
                start = spojeni.get(klic, datetime.now())
                delka = datetime.now() - start
                print(f"Připojení z {src}:{sport} na {dst}:{dport}
    trvalo {delka.total_seconds()} sekund")
                del spojeni[klic]
            else:
                print(f"Navazuji spojení...")
```

Výpis 6.4: Kontrola délky spojení (vlastní zpracování)

Výpis 6.5 popisuje kontrolu, zda paket obsahuje vrstvu *S7Comm_Header*. V případě, že ano, získá hodnotu *rosctr* z hlavičky S7 komunikace, což určuje typ komunikace. Dále kontroluje, zda *rosctr* má hodnotu **1** a zda na **16.** bajtu od konce payloadu obsahuje hodnoty **f** a na **15.** bajtu od konce payloadu **0**. Pokud je tato podmínka platná, zavolá se funkce *setup1(packet)*, která se zaobírá nastavením paketu. Pokud *rosctr* je rovno **1**, avšak neodpovídá podmínkám pro *setup1*, pak zavolá funkci *rosctr1_packet(packet)*, jež se zabývá obecnějším zpracováním paketů s *rosctr* s hodnotou **1**. Obdobně je zpracované *rosctr* pro hodnotu **3**. Díky tomuto přístupu je možné rozlišování různých typů zpráv protokolu S7 a provádět tak zpracování na základě obsahu paketů.

```
    if packet.haslayer(S7COMM_Header):
        rosctr = packet[S7COMM_Header].rosctr
        if rosctr == 1 and str(bytes(packet.payload).hex()[-16]) ==
    "f" and str(bytes(packet.payload).hex()[-15]) == "0":
            setup1(packet)
        elif rosctr == 1:
            rosctr1_packet(packet)
        if rosctr == 3 and str(bytes(packet.payload).hex()[-16]) ==
    "f" and str(bytes(packet.payload).hex()[-15]) == "0":
            setup2(packet)
        elif rosctr == 3:
            rosctr2_packet(packet)
```

Výpis 6.5: Řazení průběhu zpracování (vlastní zpracování)

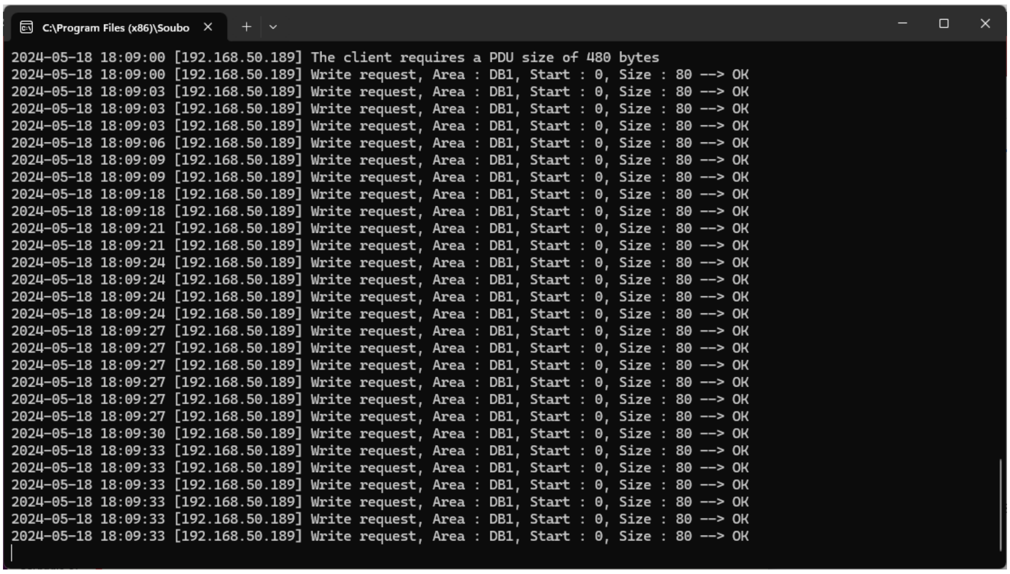
V příloze A v souboru *s7_sniff.py* jsou znázorněny funkce *setup1(packet)* a *setup2(packet)*, na které odkazují podmínky ve výpisu 6.5. Jedná se o ustanovení komunikace a výměnu parametrů (např. příjem a odeslání v jednu chvíli, což reprezentují maximální hodnoty volání *max_amq1_calling* a *max_amq1_called*). Funkce *rosctr1_packet(packet)* a *rosctr2_packet(packet)* jsou použité pro zpracování a analýzu paketů protokolu S7, jejíž hlavním úkolem je extrahovat a vypisovat definované informace z hlavičky, parametru a dat paketu. Funkce *rosctr1_packet(packet)* zpracovává paket v případě rovnosti *rosctr* **1** (zápis či čtení) a *rosctr2_packet(packet)* zpracovává paket v případě rovnosti *rosctr* **3** (potvrzování dat). Hlavička je stejná pro všechny funkce, konkrétně se jedná o identifikátor protokolu (*protocol_id*), typ zprávy (*rosctr*), identifikátor redundance (*redundancy_id*), délku PDUR (*pdur*), délku parametru (*param_length*) a délku dat (*data_length*). Pokud paket obsahuje vrstvu s parametry *S7COMM_Para* a délka parametrů je větší než **0**, pak funkce pokračuje v extrahování hodnot z této vrstvy v rámci funkce *rosctr1_packet(packet)*. V případě funkce *rosctr2_packet(packet)* se jedná o vrstvu s parametry

S7COMM_Para_ack. Obecně vzato, funkce jsou navrženy tak, aby umožnily ucelený pohled na obsah vybraných paketů protokolu S7 a vytvářely o nich statistiky.

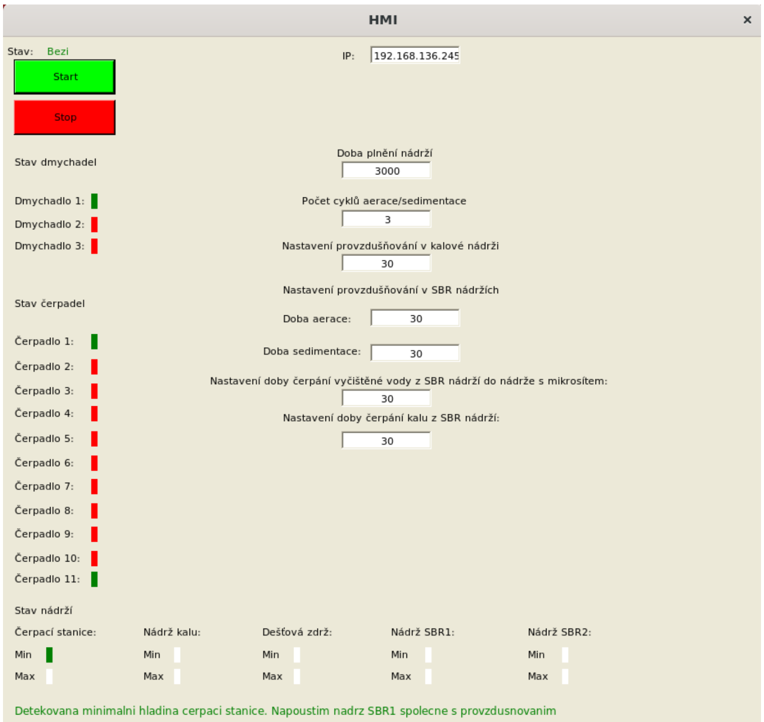
6.2 Testování modulu

Testování probíhalo na simulované komunikaci mezi HMI a PLC. Osobní počítač byl použit jako zařízení pro simulování PLC a samotná síťová sonda jako zařízení pro simulování HMI. Na obrázku 6.4 je znázorněno simulované PLC, které komunikuje s HMI na IP adrese **192.168.50.189**. Z obrázku je patrné, že se zapisuje do databloku číslo 1 (DB1) a velikost dat je **80**. Úspěšnost přenosu je znázorněna zkratkou **OK**. Celkově tento log ukazuje opakované úspěšné požadavky na zápis dat do konkrétního

datového bloku v paměti PLC. Všechny tyto údaje jsou uváděné ve statistikách, které jsou znázorněny níže.



Obr. 6.4: Simulované PLC (vlastní zpracování)



Obr. 6.5: Simulované HMI (vlastní zpracování)

Na obrázku 6.5 je znázorněno simulované HMI, které je aktivní a komunikuje s PLC na IP adrese **192.168.136.245**. PLC má nastaveny určité parametry, na

základě kterých se stavy HMI nastavují. Komunikace, která probíhá mezi těmito zařízeními, je zachytávaná vytvořeným modulem. Paket komunikace je následně zpracován pomocí výše zmíněných funkcí pomocí knihovny Scapy (viz podkapitola 6.1).

Soubor se statistikami se vytváří pomocí jednoduchého skriptu viz výpis 6.6.

```
sudo python s7_sniff.py >> statistiky.txt
```

Výpis 6.6: Statistiky (vlastní zpracování)

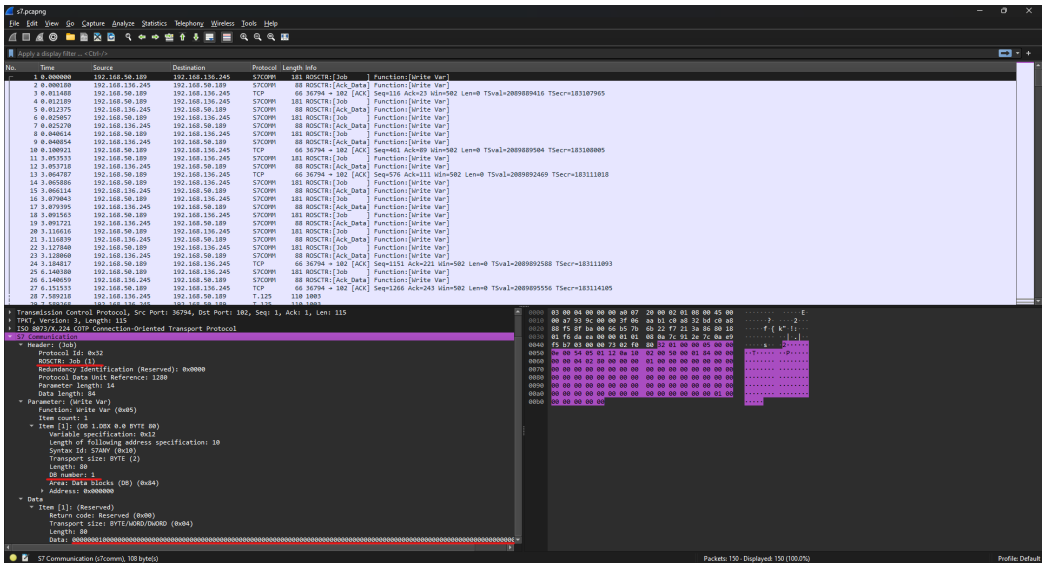
Pohled na statistiky je znázorněn na obrázku 6.6. Statistiky obsahují vyparsované informace (v reálném čase) každého S7 paketu, jako je hlavička, parametr a data. Dále umožňují výběr žádaných informací, jimž může být číslo databloku, chybové zprávy a délku spojení.

Obr. 6.6: Statistiky komunikace (vlastní zpracování)

Vyjma statistik vyobrazených na obrázku 6.6 se po ukončení modulu zobrazí přehled o všech zachycených paketech S7 a z toho je *rosctr1* (čtení nebo zápis) a *rosctr3* (potvrzení). Pokud se pakety z *rosctr1* rovnají počtu paketů z *rosctr3* (viz výpis 6.7, pak všechny pakety byly správně doručeny a potvrzeny, tj. komunikace je úspěšná.

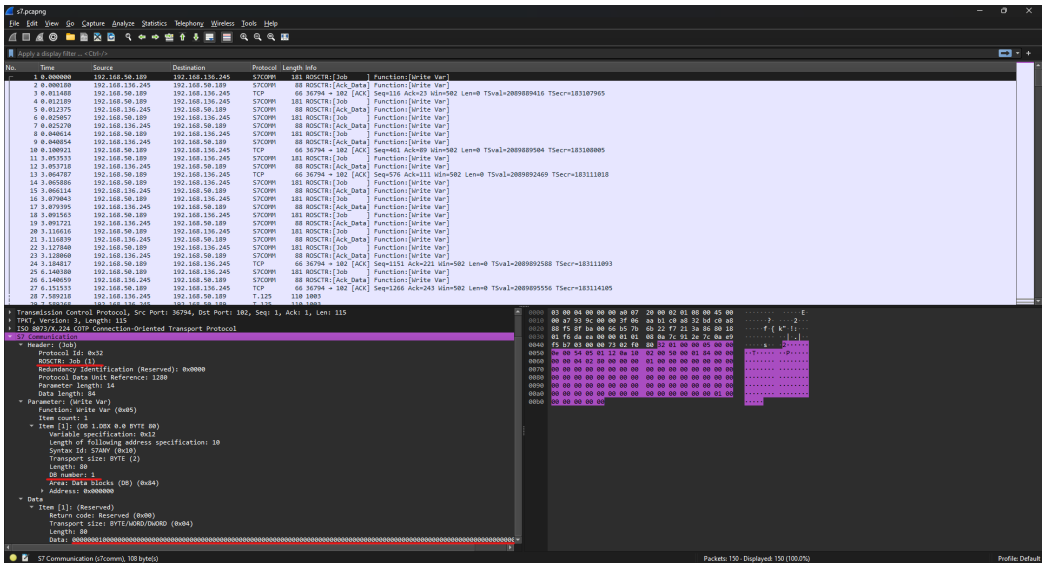
Obr. 6.7: Statistiky po ukončení komunikace (vlastní zpracování)

Na obrázku 6.8 lze vidět zápis dat pro **ROSCTR 1** v programu Wireshark. Tento obrázek potvrzuje správnost parsování vytvořeného modulu (viz obrázek 6.6). Pro kontrolu jsou zvýrazněná data ROSCTR, DB number (počet databloků) a samotná data. Obdobně na obrázku 6.9 lze vidět zápis dat pro **ROSCTR 3** v programu



Obr. 6.8: Znázornění ROSCTR 1 ve Wiresharku (vlastní zpracování

Wireshark. Tento obrázek také potvrzuje správnost parsování vytvořeného modulu (viz obrázek 6.6). Pro kontrolu jsou zvýrazněná data ROSCTR, error class (třída erroru) a error code (kód erroru). Jelikož je uveden kód erroru **0x00**, pak je zřejmé, že přenos je správný.



Obr. 6.9: Znázornění ROSCTR 3 ve Wiresharku (vlastní zpracování)

7 Sada pravidel pro protokol S7

Sada pravidel pro protokol S7 se dělí do dvou fází, a to vytvoření daných pravidel a jejich následné testování skrze vybrané typy útoků. Vytvoření pravidel je klíčové pro definování útoků, které má Suricata vyhledávat. V první fázi tvorby pravidel je nutné provést analýzu síťového provozu a identifikovat specifické charakteristiky protokolu S7, které mohou být využity k detekci potenciálních hrozeb, což je znázorněné v kapitole 6. Tato fáze zahrnuje psaní pravidel, která specifikují podmínky a akce, jež Suricata provede při detekci nežádoucího provozu. Druhá fáze se zaměřuje na testování těchto pravidel. To zahrnuje nasazení pravidel v kontrolovaném prostředí a simulaci různých typů útoků na protokol S7, aby se ověřila jejich účinnost a přesnost.

7.1 Vytvoření pravidel

Pravidla jsou klíčová pro nastavení Suricaty a její schopnosti detekovat potenciální útoky. Tato pravidla pro vybrané útoky (DoS, skenování portu a odposlech) jsou zadefinována ve výpisech 7.1, 7.2 a 7.3. V kontextu zmíněných pravidel je DoS (Denial of Service) typ kybernetického útoku, při kterém útočník zaplaví síťovou službu nebo server, jako je protokol S7 používaný v průmyslových systémech, nadměrným množstvím požadavků. Toto přetížení způsobuje, že služba nebo server nejsou schopny reagovat na legitimní požadavky uživatelů, což vede k jejich nedostupnosti. Sken portu, v kontextu pravidel, je technika, kterou útočníci používají k identifikaci otevřených portů, například portu 102 používaného pro protokol S7. Pravidlo detekující potenciální sken portu sleduje pokusy o přístup k tomuto portu bez ohledu na stav spojení, což umožňuje identifikovat možné útočníky, kteří se snaží zjistit, jaké služby běží na cílovém zařízení, a identifikovat zranitelnosti k dalším útokům. Dále odposlech, známý jako eavesdropping, je kybernetický útok, při kterém útočník tajně zachytává komunikaci mezi zařízeními, například při útoku na průmyslový protokol S7. Cílem je získat citlivé informace, jako jsou provozní data a řídicí příkazy.

První pravidlo (výpis 7.1 se zaměřuje na detekci potenciálního útoku typu DoS (Denial of Service) na protokol S7. Pravidlo generuje upozornění (alert) při detekci specifického vzoru v TCP paketech. Pravidlo se aplikuje na jakoukoli zdrojovou a cílovou IP adresu a port. Spojení musí být navázáno (established), což znamená, že pravidlo se aplikuje pouze na existující spojení. Pravidlo kontroluje obsah paketu, konkrétně hledá hexadecimální sekvenci **32 01 00** (případně 32 03 00 nebo 03 00 v případě, že se posílá i s hlavičkou ISO-TSAP) v prvních třech bajtech paketu. Aby se předešlo zahlcení systému falešnými pozitivními nálezy, je aplikováno omezení (threshold). Sleduje se počet událostí (**count 100**) od stejného zdroje (track by_src) v časovém okně **10** sekund. Pokud je detekováno více než **100** událostí, je spuštěno

upozornění. Identifikátor pravidla (sid) je **103**, což je unikátní číslo pro toto pravidlo. Klasifikace události je "pokus o DoS"(attempted-dos) s prioritou **2**, což označuje závažnost události (druhé nejvyšší).

```
alert tcp any any -> any any (msg:"Detekován potenciální DoS útok
    na S7"; flow:established; content:"|32 01 00|"; depth:3;
    threshold: type limit, track by_src, seconds 10, count 100; sid
    :103; classtype:attempted-dos; priority:2; rev:1;)
```

Výpis 7.1: Pravidlo pro DoS (vlastní zpracování)

Druhé pravidlo (výpis 7.2 také detekuje potenciální DoS útoky, ale je obecnější než první pravidlo. Opět se jedná o upozornění (alert) pro TCP pakety mezi jakoukoli zdrojovou a cílovou IP adresou a portem. V tomto případě není specifikováno, že spojení musí být navázáno. Podmínky pro omezení (threshold) jsou nastaveny tak, aby sledovaly jak počet událostí, tak časové okno (type both), přičemž je sledováno, kolikrát dojde k události od stejného zdroje (track by_src) během **10** sekund (seconds 10). Pokud je detekováno více než **100** událostí (count 100), je spuštěno upozornění. Unikátní identifikátor pravidla je **104**, klasifikace události je pokus o DoS (attempted-dos) a priorita je **2**. Pravidlo obsahuje metadata, která poskytují další informace: dopad události je označen jako červený (impact_flag red), což indikuje vysokou závažnost, služba je průmyslová (service industrial) a protokol je S7 (protocol s7).

```
alert tcp any any -> any any (msg:"Detekován potenciální DoS útok";
     threshold: type both, track by_src, count 100, seconds 10; sid
    :104; rev:1; classtype:attempted-dos; priority:2; metadata:
    impact_flag red, service industrial, protocol s7;)
```

Výpis 7.2: Pravidlo pro DoS (vlastní zpracování)

Třetí pravidlo (výpis 7.3 se zaměřuje na detekci potenciálního skenování portu **102**, který je běžně používán pro protokol S7. Pravidlo generuje upozornění (alert) pro jakékoli TCP pakety z jakékoli zdrojové IP adresy a portu směřující na cílový port **102**. Pravidlo je bezstavové (flow:stateless), což znamená, že se nezabývá stavem spojení, ale pouze kontroluje jednotlivé pakety. Identifikátor pravidla je **105**.

```
alert tcp any any -> any 102 (msg:"Potenciální sken portu S7"; flow
    :stateless; sid:105; rev:1;)
```

Výpis 7.3: Pravidlo pro sken portu (vlastní zpracování)

Čtvrté pravidlo (výpis 7.4 detekuje specifický typ komunikace, konkrétně čtení nebo zápis. Pravidlo funguje tak, že se aplikuje na TCP provoz (*alert tcp*) z jakéhokoliv zdrojového IP adresy a portu na jakoukoliv cílovou IP adresu s portem **102**. Pokud je pravidlo splněno, zobrazí se výstražná zpráva "Pozor komunikace je v otevřené podobě! - S7 Protocol". Toto pravidlo se vztahuje pouze na navázaná

TCP spojení (*flow:established*). Pravidlo hledá specifický obsah (hexadecimální řetězec "|32 01|") v datovém proudu od začátku TCP payloadu a prohledává první **2** bajty. Unikátní identifikátor tohoto pravidla je **sid:102**. Pravidlo může být upraveno podle dané komunikace. Pokud se S7 paket posílá navíc s COTP a TPKT hlavičkama, tak je nutné toto pravidlo tomuto případu přizpůsobit. Například zaměnit položku *content* na **03 00** nebo změnit hodnotu *offset*.

```
alert tcp any any -> any 102 (msg:"Pozor komunikace je v otevřené
    podobě! - S7 Protocol"; flow:established; content:"|32 01|";
    offset:0; depth:2; sid:102; rev:1;)
```

Výpis 7.4: Pravidlo pro detekci komunikace (vlastní zpracování)

Pro správnou funkci pravidel je nutné v prvé řadě provést nastavení modulu Suricaty v souboru *suricata.yaml*, viz podkapitola 5.2.2. Je nutné nastavit rozhraní (výpis 5.3 tohle nastavení již obsahuje), aby Suricata zachytávala správné pakety. Dále nastavení logování, jež je již uvedeno ve výpisu 5.2, a nastavení sady pravidel, jež oproti původní konfiguraci (výpis 5.4) jsou uvedeny dle výpisu 7.5.

```
default-rule-path: /etc/suricata/rules

rule-files:
    - S7-events.rules
```

Výpis 7.5: Konfigurace pravidel S7 (vlastní zpracování)

Pro aplikaci veškerých nastavení je nutné provést restart modulu Suricata. To se provede skrze příkaz dle výpisu 7.6.

```
sudo systemctl restart suricata.service
```

Výpis 7.6: Příkaz pro restart modulu Suricata (vlastní zpracování)

7.2 Testování pravidel

Testování pravidel je prováděno z důvodu ověření funkčnosti definovaných pravidel uvedených v podkapitole 7.1. Pravidla se testují vybranými kybernetickými útoky.

7.2.1 Odposlech

V rámci komunikace se pracuje výhradně s otevřenými daty, a tudíž je snadné je odposlechnout skrze nějaký sniffer. Suricata reaguje právě na každý jeden paket. Pro zabezpečení komunikace protokolu S7 je klíčové implementovat šifrování, například TLS/SSL, aby se znemožnilo čtení zachycených dat. Segmentace sítě, pomocí VLANs a firewallů, izoluje průmyslovou síť od podnikové a veřejné sítě. Silná autentizace a autorizace, včetně vícefaktorové autentizace (MFA), zvyšují bezpečnost

přístupu. Monitorování sítě a detekce anomálií pomáhají odhalit podezřelou aktivitu.

Výpis 7.7 upozorňuje na nešifrovanou komunikaci pomocí protokolu S7. K události došlo 18. května 2024 ve 21:37:11.521796. Výstraha s vysokou prioritou [**1:102:1**] hlásí, že komunikace je v otevřené podobě, což znamená, že není šifrována. Komunikace probíhá přes TCP mezi IP adresami **192.168.136.245** (port **49373**) a **192.168.50.189** (port **102**). Toto upozornění signalizuje potenciální bezpečnostní riziko, protože nezabezpečená komunikace může být snadno odposlechnuta.

```
05/18/2024-21:37:11.521796  [**] [1:102:1] Pozor komunikace je v
    otevřené podobě! - S7 Protocol [**] [Classification: (null)] [
    Priority: (null)] {TCP} 192.168.136.245:49373->
    192.168.50.189:102
```

Výpis 7.7: Simulace odposlechu komunikace (vlastní zpracování)

7.2.2 DoS útok

Kód ve výpisu 7.8 je napsán v Pythonu a využívá knihovnu Scapy k simulaci DoS (Denial of Service) útoku. Na začátku kódu jsou importovány potřebné knihovny: *scapy.all*, která umožňuje manipulaci s pakety v síti, a *time*, která slouží pro zpoždění mezi odesíláním paketů. Kód začíná vytvořením TCP paketu s konkrétními daty. Hexadecimální obsah **320100** je převeden na bajty a uložen do proměnné data. Následně je vytvořen IP paket směřující na IP adresu **192.168.50.189** s cílovým portem **102**, přičemž tento paket obsahuje TCP segment s danými daty. Dále je definována funkce *send_packets*, která umožňuje odesílání paketů rychle za sebou. Tato funkce přijímá dva argumenty: počet paketů (count) a zpoždění mezi nimi (delay). Uvnitř funkce je cyklus, který odesílá specifikovaný počet paketů s daným zpožděním mezi jednotlivými odesláními. Pro simulaci útoku je nastavena proměnná *packet_count*, která určuje počet paketů, jež budou odeslány (v tomto případě **105**), a proměnná *delay_between_packets*, která určuje zpoždění mezi jednotlivými odeslanými pakety (**10** milisekund). Nakonec je volána funkce *send_packets* s parametry *packet_count* a *delay_between_packets*, což spustí proces odesílání paketů. Tento kód tak simuluje DoS útok na zařízení s IP adresou **192.168.50.189** na portu **102**, zasíláním **105** TCP paketů s konkrétním obsahem v intervalu **10** milisekund mezi jednotlivými pakety. Cílem tohoto útoku je přetížit cílový port nebo službu a způsobit, že bude nedostupná pro legitimní uživatele.

```
from scapy.all import *
import time

# Vytvoření TCP paketu s specifickými daty
data = bytes.fromhex('320100')
```

```
packet = IP(dst="192.168.50.189")/TCP(dport=102)/Raw(load=data)

# Funkce pro odesílání paketů rychle za sebou
def send_packets(count, delay):
    for _ in range(count):
        send(packet)
        time.sleep(delay)

# Počet paketů a zpoždění mezi nimi (v sekundách)
packet_count = 105
delay_between_packets = 0.01  # 10 milisekund

# Odeslání paketů
send_packets(packet_count, delay_between_packets)
```

Výpis 7.8: Simulace DoS útoku (vlastní zpracování)

Ve výpisu 7.9 je výstup z logu systému Suricata, který zobrazuje detekci DoS útoků.

První záznam ukazuje, že 13. května 2024 v 16:01:56 byl detekován potenciální DoS útok. Tento záznam obsahuje identifikátor události [**1:103:1**], kde **1** označuje generátor události, **103** je identifikátor pravidla (SID) a **1** je revize pravidla. Událost je klasifikována jako pokus o DoS útok s prioritou **2**, což znamená vysokou závažnost. Detekovaný útok využívá protokol TCP a pochází z IP adresy **192.168.136.245** na portu **1382**, přičemž cílí na IP adresu **192.168.50.189** na portu **102**. Druhý záznam ukazuje, že 13. května 2024 v 16:26:12 byl detekován další potenciální DoS útok, specificky zaměřený na protokol S7. Tento záznam má identifikátor události [**1:104:1**], a liší se od prvního záznamu pouze identifikátorem pravidla. Událost je opět klasifikována jako pokus o DoS útok s prioritou **2**. Útok využívá protokol TCP a pochází z IP adresy **192.168.136.245** na portu **2251**, přičemž cílí na stejnou IP adresu **192.168.50.189** na portu **102**. Tyto záznamy znázorňují, jak systém Suricata detekuje a loguje pokusy o DoS útoky, poskytuje detaily o zdrojové a cílové IP adrese, portech, protokolu a klasifikaci události, což je klíčové pro monitorování a reakci na bezpečnostní incidenty v síti.

```
05/13/2024-16:01:56.614786  [**] [1:104:1] Detekován potenciální
   DoS útok [**] [Classification: Attempted Denial of Service] [
   Priority: 2] {TCP} 192.168.136.245:1382 -> 192.168.50.189:102
05/13/2024-16:26:12.451734  [**] [1:103:1] Detekován potenciální
   DoS útok na S7 [**] [Classification: Attempted Denial of Service
   ] [Priority: 2] {TCP} 192.168.136.245:2251 -> 192.168.50.189:102
```

Výpis 7.9: Reakce Suricaty na DoS útok (vlastní zpracování)

7.2.3 Skenování portu

Příkaz ve výpisu 7.10 je použit k provedení skenu specifického portu na vzdáleném zařízení pomocí nástroje nmap. Volba -sV říká nmapu, aby zjistil verzi služby běžící na otevřeném portu. Parametr -p **102** určuje, že skenovaný bude pouze port číslo **102**. IP adresa **192.168.50.189** je cílová adresa zařízení, které má být skenováno. Tento příkaz je užitečný pro získání informací o konkrétní službě běžící na daném portu na cílovém zařízení.

```
1 nmap -sV -p 102 192.168.50.189
```

Výpis 7.10: Simulace skenu portu (vlastní zpracování)

Záznam ve výpisu 7.11 ukazuje výstup detekce potenciálního skenování portů. K detekci došlo 18. května 2024 ve 23:31:04. Událost je označena jako "Potenciální sken portu"a má identifikátory [**1:105:1**]. Klasifikace události není uvedena a priorita je nastavena na **3**. Protokol použitý při skenování je TCP. Zdrojem skenování je zařízení s IP adresou **192.168.136.245** a portem **48672**, zatímco cílové zařízení má IP adresu **192.168.50.189** a port **102**.

```
1 05/18/2024-23:31:04.462189  [**] [1:105:1] Potenciální sken portu
    S7 [**] [Classification: (null)] [Priority: 3] {TCP}
    192.168.136.245:48672-> 192.168.50.189:102
```

Výpis 7.11: Reakce Suricaty na skenu portu (vlastní zpracování)

Závěr

Zabezpečení komunikačních sítí, obzvláště přenos paketů, je čím dál náchylnější a proto je nutné se v neustále vyvýjejícím prostředí aktivně pohybovat a navrhovat nové, bezpečnější způsoby ochrany přenosu dat. Teoretická část se zabývá rešerší prostředí Linux, Linux Kernel a distribucí Linuxu, což je provedeno za účelem přiblížení čtenáře operačního systému, ve kterém byla prováděna manipulace se síťovou sondou v praktické části.

Dílčí teoretickou částí je přiblížení systémů detekce a prevence průniku z důvodu pochopení problematiky a studia již známých přístupů zabezpečení.

Nejrozsáhlejší dílčí teoretická část pak zahrnuje nastínění do protokolů a standardů, které byly možnými kandidáty pro návrh modelu pro diplomovou práci. Mezi ně se řadí protokol IEEE C37.118, standard IEC 61850, S7, standard IEC 60870 a EtherNet/IP. Jelikož autor má zkušenosti s průmyslovým protokolem S7, byla na něj zpracována kapitola samostatně a z ní vychází pro praktickou část.

Praktická část zahrnuje dvě hlavní části práce. První část popisuje vytvoření modulu pro analýzu protokolu S7. V této části je detailně popsán postup vytváření daného modulu s popisem jednotlivých funkcí, ze kterých se skládá. Dále je popsáno samotné testování modulu, které probíhalo na simulované komunikaci. Modul vytváří statistiky, které jsou důležité pro další analýzu paketu S7. Statistiky zobrazují zejména informace o paketu, délku spojení, komunikující strany a celkový počet zachycených paketů. Modul byl následně porovnán s pakety zachycenými program Wireshark.

Druhá část praktické části se věnovala vytváření sady pravidel na detekci zvolených útoku na protokol S7. Byla vytvořena 3 pravidla, která byla následně implementována do modulu Suricaty a následně otestována v kontrolovaném prostředí. Mezi zvolené útoky patří DoS, odposlech a skenování portů.

Modul je navržen univerzálně, je možné jej implementovat s minimálními úpravami na jakékoliv zařízení podporující programovací jazyk Python. Je možné modul rozšířit o více protokolů, například z průmyslového odvětví a udělat tak z něj velký modul pro detekci všech známých průmyslových protokolů.

Literatura

[1] SROVNAL, Dominik. *Analýza protokolu S7 a vytvoření virtualizovaného průmyslového scénáře* [online]. Brno, 2022 [cit. 2023-12-09]. Dostupné z: `https://www.vut.cz/studenti/zav-prace/detail/141333`. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce Ondřej Pospíšil.

[2] K. E. Martin et al., *Exploring the IEEE Standard C37.118–2005 Synchrophasors for Power Systems* in IEEE Transactions on Power Delivery, vol. 23, no. 4, pp. 1805-1811, Oct. 2008, doi: 10.1109/TPWRD.2007.916092.

[3] R. Khan, K. McLaughlin, D. Laverty and S. Sezer, *Analysis of IEEE C37.118 and IEC 61850-90-5 synchrophasor communication frameworks*, 2016 IEEE Power and Energy Society General Meeting (PESGM), Boston, MA, USA, 2016, pp. 1-5, doi: 10.1109/PESGM.2016.7741343.

[4] R. E. Mackiewicz, "Overview of IEC 61850 and Benefits,"2006 IEEE PES Power Systems Conference and Exposition, Atlanta, GA, USA, 2006, pp. 623-630, doi: 10.1109/PSCE.2006.296392.

[5] S. -J. Chen, Y. -H. Wang, C. -H. Lin, T. -S. Zhan, R. -F. Chang and Y. -C. Chang, "Using Multi-vendor IEDs for IEC 61850 Interoperability and HMI-SCADA Applications,"2012 International Symposium on Computer, Consumer and Control, Taichung, Taiwan, 2012, pp. 745-748, doi: 10.1109/IS3C.2012.193.

[6] VLADYKA, Pavel a Bruno FORGUE. Časopis Automa IEC 61850: soubor norem pro komunikaci v energetice s velkým potenciálem výhod. In: www.automa.cz [online]. 3.2010 [cit.09.12.2023]. Dostupné z: https://www.automa.cz/cz/casopis-clanky/iec-61850-soubor-norem-pro-komunikaci-v-energetice-s-velkym-potencialem-vyhod-2010_03_40771_5154/

[7] RUDZINSKI, Yvan a Pavel VLADYKA. Komunikační protokoly pro dálkové ovládání IEC/ISO 60870-5 [online]. Automa, 2010 [cit.10.12.2023]. Dostupné z:https://automa.cz/Aton/FileRepository/pdf_articles/40552.pdf

[8] Y. Yang, K. McLaughlin, T. Littler, S. Sezer, B. Pranggono and H. F. Wang, "Intrusion Detection System for IEC 60870-5-104 based SCADA networks,"2013 IEEE Power & Energy Society General Meeting, Vancouver, BC, Canada, 2013, pp. 1-5, doi: 10.1109/PESMG.2013.6672100.

[9] Y. Yang, K. McLaughlin, S. Sezer, Y. B. Yuan and W. Huang, "Stateful intrusion detection for IEC 60870-5-104 SCADA security,"2014 IEEE PES General

Meeting | Conference & Exposition, National Harbor, MD, USA, 2014, pp. 1-5, doi: 10.1109/PESGM.2014.6939218.

[10] EtherNet/IP Protocol Overview - Real Time Automation. In: Real Time Automation, Inc. [online].[b.r.] [cit.10.12.2023]. Dostupné z: https://www.rtautomation.com/technologies/ethernetip/

[11] SCHNEIDER ELECTRIC INDUSTRIES SAS . Principles of EtherNet/IP Communication [online]. 2011 [cit.10.12.2023]. Dostupné z: https://scadahacker.com/library/Documents/ICS_Protocols/Schneider

[12] JUNIPER NETWORKS. What is IDS and IPS? | Juniper Networks. In: www.juniper.net [online]. 2023. Dostupné z:https://www.juniper.net/us/en/research-topics/what-is-ids-ips.html

[13] KOKRMENT, Lukáš a Jan PAVLOVIČ. Systémy pro detekci a prevenci průniků [online]. [b.r.] [cit.10.12.2023]. Dostupné z:http://147.229.242.132/images/dokumenty/Programovani-TSW-1975-2014/2005/2005-13.pdf

[14] WALEED, Abdul, Abdul Fareed JAMALI a Ammar MASOOD. Which open-source IDS? Snort, Suricata or Zeek. Computer Networks. 2022, s. 109116. DOI:https://doi.org/10.1016/j.comnet.2022.109116

[15] VIGLIONE, Mark. Suricata: What is it and how can we use it | Infosec. In: resources.infosecinstitute.com [online]. 4.3.2022 [cit.10.12.2023]. Dostupné z:https://resources.infosecinstitute.com/topics/network-security-101/suricata-what-is-it-and-how-can-we-use-it/

[16] FELKOKIN, Roman. Intrusion Detection and Prevention Systems: Overview of Snort and Suricata. In: Research Gate [online]. 6.1.2015 [cit.10.12.2023]. Dostupné z:https://www.researchgate.net/profile/Roman-Fekolkin-2/publication/297171228_Intrusion_Detection_and_Prevention_Systems_Overview_of_S Detection-and-Prevention-Systems-Overview-of-Snort-and-Suricata.pdf

[17] CASWELL, Brian, Jay BEALE a Andrew BAKER. Snort Intrusion Detection and Prevention Toolkit. Burlington: Elsevier, 2007. ISBN 9780080549279.

[18] OPENSOURCE. What is Linux? In: Opensource.com [online]. 2019 [cit. 10.12.2023]. Dostupné z:https://opensource.com/resources/linux

[19] SIEVER, Ellen et al. Linux in a Nutshell. „O'Reilly Media, Inc.", 2005. ISBN 9780596529499.

[20] RED HAT. What is Linux? In: Redhat.com [online]. 3.1.2023 [cit.10.12.2023]. Dostupné z:https://www.redhat.com/en/topics/linux/what-is-linux

[21] RED HAT. What is the Linux kernel? In: www.redhat.com [online]. 27.2.2019 [cit.10.12.2023]. Dostupné z:https://www.redhat.com/en/topics/linux/what-is-the-linux-kernel

[22] Stackscale. 17 popular Linux distributions and OS. In: Stackscale [online]. 22.12.2020 [cit.11.12.2023]. Dostupné z: https://www.stackscale.com/blog/popular-linux-distributions/

[23] KLEČKA, Jan. Síťová sonda pro záznam a analýzu komunikace. Brno, 2020, 90 s. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Petr Blažek. Dostupné také z: `https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=226254`

Seznam symbolů a zkratek

KISS	Keep It Simple, Stupid
GUI	Graphical user interface
IDS	Intrusion Detection System
IPS	Intrusion Prevention Systems
NIDS	Network Intrusion Detection System
NIPS	Network Intrusion Prevention System
HIDS	Host-based Intrusion Detection System
DIDS	Distributed Intrusion Detection System
DDos	Distributed Denial of Service
Dos	Denial of Service
IoT	Internet of Things
GPS	Global Positioning System
UTC	Coordinated Universal Time
PMU	Phasor Measurement Unit
IED	Intelligent Electronic Device
HMI	Human-Machine Interface
WAN	Wide Area Network
LAN	Local Area Network
CIP	Common Industrial Protocol
PLC	Programmable Logic Controller
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
TPKT	Transport Packet
CP	Control Protocol

PDU	Protocol Data Unit
COTP	Connection-Oriented Transport Protocol
VPN	Virtual Private Network
IP	Internet Protocol
CPU	Central Processing Unit
DB	DataBlock
TLS	Transport Layer Security
SSL	Secure Sockets Layer
MFA	Multi-Factor Authentication
SID	Security Identifier

A Obsah elektronické přílohy

V elektronické příloze ve složce modul lze najít vytvořený modul sloužící pro zachytávání S7 paketů a vytváření statistik. Dále ve složce suricata lze nalézt vytvořená pravidla pro detekci útoků na protokol S7 a demonstrativní útok typu DoS.

```
/ ..... kořenový adresář přiloženého archivu
├── modul ..... Vytvořený modul pro analýzu S7
│   └── s7_sniff.py
└── suricata ..... Obsahuje pravidla pro Suricatu a DoS útok.
    ├── pravidla
    │   └── pravidla.txt
    └── utok
        └── dos_utok.py
```