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

IMPLEMENTACE KOMUNIKAČNÍHO SCÉNÁŘE VYUŽÍVAJÍCÍHO DATA DISTRIBUTION SERVICE A HODNOCENÍ BEZPEČNOSTI

COMMUNICATION MODEL USING DATA DISTRIBUTION SERVICE AND COMMUNICATION SECURITY ASSESSMENT

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — **Bc. Martin Frollo**

AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jiří Pokorný**

SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

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

Student: Bc. Martin Frollo ***ID:*** 177256
Ročník: 2 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Implementace komunikačního scénáře využívajícího Data distribution service a hodnocení bezpečnosti

POKYNY PRO VYPRACOVÁNÍ:

Bude provedena analýza dostupných implementací DDS (Data Distribution Service; např. DDS Community, Vortex DDS, Connext DDS) z pohledu bezpečnosti, funkcionalit a implementace. Bude zvolen model komunikačního scénáře a následně bude tento scénář implementován ve vybrané distribuci pomocí dostupného HW (např. Raspberry Pi) – bude možné využít mj. i virtualizačních technik pro vytvoření komplexnějšího modelu. Komunikační model bude otestován z pohledu funkčnosti a základních komunikačních parametrů. Dále bude provedeno hodnocení bezpečnosti a zranitelností pomocí dostupných SW (např. Kali Linux či Wireshark), mj. bude také provedeno testování závislosti komunikačních parametrů na probíhajícím útoku typu DoS. V neposlední řadě budou navrženy mitigační opatření pro nalezená rizika a bezpečnostní nedostatky.

DOPORUČENÁ LITERATURA:

[1] SCHLESSELMAN, Joseph M.; PARDO-CASTELLOTE, Gerardo; FARABAUGH, Bert. OMG data-distribution service (DDS): architectural update. In: Military Communications Conference, 2004. MILCOM 2004. 2004 IEEE. IEEE, 2004. p. 961-967.

[2] CALVO, Isidro, et al. Towards a OMG DDS communication backbone for factory automation applications. In: Emerging Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on. IEEE, 2011. p. 1-4.

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 16.5.2019

Vedoucí práce: Ing. Jiří Pokorný
Konzultant:

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

ABSTRAKT

Diplomová práca sa zameriava na problematiku middleware vrstvy v distribuovaných systémoch. Uvádza typicky používané komunikačné protokoly a štandardy, ktoré na tejto vrstve pracujú. Jej užšie zameranie je z hľadiska teoretickej aj praktickej časti na špecifikáciu OMG DDS. Priblížené sú jej základné funkcionality a architektúra spolu s jednotlivými prvkami. V práci je taktiež popísaný RTPS protokol. Na záver teoretickej časti sú popísané existujúce implementácie štandardu OMG DDS. V praktickej časti je využitá OpenDDS 3.13 implementácia tejto špecifikácie. Otestované je jej nasadenie na rôznych platformách. V ďalšej časti je overená vlastnosť zabezpečenia prenosu dát pomocou beta verzie OMG DDS Security, ktorú táto implementácia obsahuje. Porovnaný je zabezpečený a nezabezpečený prenos. V distribuovaných systémoch je dôležité aj oneskorenie prenosu systému. Pre analýzu rôznych vplyvov na výsledné oneskorenie prenosu je v praktickej časti vytvorený DDS systém, ktorý meria oneskorenie pri rôznych nastaveniach QoS a zabezpečení. Prevedené a analyzované sú merania pri rôznych vlastnostiach prenosu. Z výsledkov meraní sú zrejme vplyvy výkonu zariadenia pri zvyšovaní veľkosti odosielaných vzoriek. Zaznamené sú rozdiely oneskorenia medzi spoľahlivým a nespoľahlivým a zabezpečeným a nezabezpečeným prenosom. Uskutočnené sú aj merania medzi 2 fyzickými zariadeniami s útočníkom a bez neho. Útok je typu MITM a zachytáva RTPS prenos ktorý mu nie je určený.

KĽÚČOVÉ SLOVÁ

DDS, IoT, middleware, MITM, OMG, oneskorenie, QoS, RTPS

ABSTRACT

The diploma thesis is focusing on middleware layer in distributed systems. It introduces typically used communication protocols and standards operating on this layer. In theoretical part it brings closer look at OMG DDS specification. This part contains fundamental functionalities of this specification along with its architecture blocks. Thesis also describes the RTPS protocol functionality. Existing implementations of OMG DDS standard are described at the end of theoretical part. OpenDDS 3.13 implementation is used in practical part of thesis. It is deployed and tested on various platforms. Next part verifies option of securing RTPS data stream using beta version of OMG DDS Security, which OpenDDS 3.13 implementation contains. Secured and unsecured data flows are being compared. Latency of data stream is also important, especially in distributed systems. DDS system, which measures latency of RTPS stream is created in practical part. Latency of this DDS system can be measured in various configurations. Difference of devices' performance used in measurements can be clearly seen in latency results where the size of data samples is increasing. Differences of measured latency are also recognizable between reliable and unreliable and secure and unsecure RTPS stream. Part of measurements is made between 2 physical devices with and without an attacker. Type of attack is MITM and it captures RTPS flow, which does not belong to attacking machine.

KEYWORDS

DDS, IoT, latency, middleware, MITM, OMG, QoS, RTPS

FROLLO, Martin. *Implementace komunikačního scénáře využívajícího DDS a hodnocení bezpečnosti*. Brno, Rok, 75 s. Diplomová práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedúci práce: Ing. Jiří Pokorný,

VYHLÁSENIE

Vyhlasujem, že som svoju diplomovú prácu na tému „ Implementace komunikačního scénáře využívajícího DDS a hodnocení bezpečnosti“ vypracoval samostatne pod vedením vedúceho diplomovej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej diplomovej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto diplomovej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora

POĎAKOVANIE

Rád by som poďakoval vedúcemu diplomovej práce pánovi Ing. Jiřímu Pokornému za odborné vedenie, konzultáce, trpezlivosť a podnetné návrhy k práci.

Brno ....

podpis autora

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam výpisov

Úvod

Táto práca sa venuje možnostiam pre zaistenie komunikácie vzdialených procesov v distribuovaných systémoch. Konkrétne sa zameriava na prostriedky ktoré v prostrediach akým je napríklad internet umožňujú zariadeniam nadviazať spojenie a vymieňať si medzi sebou užitočné informácie v reálnom čase. Jedná sa o protokoly a špecifikácie takzvanej middleware vrstvy.

Počiatočná časť tejto práce spočíva v stručnom oboznámení s pojmom internet vecí – Internet of Things, ktorý so spomínanou middleware vrstvou úzko spolupracuje. Po oboznámení s týmto termínom sú v práci priblížené hlavné účely middleware vrstvy. Po nich nasleduje základný popis vlastností najbežnejších protokolov a štandardov pre túto vrstvu v distribuovaných systémoch, ktoré sa v súčastnsoti aplikujú. Sú tu spomenuté ich hlavné spoločné aj odlišné črty.

Po úvodnom zoznámení s danou problematikou je zvyšná časť práce zameraná na štandard OMG DDS. V teoretickej časti práce je opísaná architektúra tohto štandardu spolu s jednotlivými entitami, ktoré ju tvoria. Práca sa zaoberá aj významom jednotlivých entít v celej architektúre, ich vzájomnom prepojení a špecifických funkcionalitách ktoré v nej majú. V ďalšej pasáži teoretickej časti je popísaný RTPS protokol, ktorý bol štandardizovaný ako interoperatibilný komunikačný protokol v DDS systémoch. Teoretická časť ďalej popisuje možnosti zaistenia kvality služby v DDS systéme. Popísané sú niektoré z QoS atribútov definovaných v štandarde OMG DDS. Spomentá je aj realizácia zabezpečenia DDS systému podľa štandardu OMG DDS Security. V časti zameranej na zabezpečenie sú popísané základné funkčné bloky zabezpečeného DDS systému. V závere teoretickej časti sa nachádza prehľad niektorých dostupných DDS implementácií.

Praktická časť je zameraná na zoznámenie sa s konkrétnou DDS implementáciou. Ide o implementáciu OpenDDS 3.13, na ktorej sú otestované a analyzované niektoré vlastnosti štandardu OMG DDS spomínané v teoretickej časti. Testovanie a analýza komunikácie je uskutočnená vo viacerých scenároch na rôznych fyzických a virtualizovaných zariadeniach s rozličnými operačnými systémami. V scenároch sú využité aplikácie nachádzajúce sa v balíku OpenDDS 3.13 implementácie. Z hľadiska bezpečnosti je overený rozdiel v správach so zabezpečením a bez neho.

Po dohode s vedúcim práce je v ďalšej pasáži praktickej časti popísané vytvorenie DDS systému umožňujúceho simulovať prenos pri rôznych nastaveniach a merať jeho oneskorenie. Simulácie s meraním oneskorenia sú uskutočnené pre budúce porovnanie s inými DDS implementáciami. Prostredníctvom implementovaného DDS systému je možné voliť veľkosť vzoriek, rýchlosť ich odosielania, zabezpečenie a spoľahlivosť prenosu. Simulácie sú prevedené na 1 zariadení alebo medzi 2 zariadeniami. V závere praktickej časti sú porovnané výsledky z jednotlivých meraní.

1 Teoretická časť

Táto časť je zameraná na teoretické poznatky o pojme internet vecí. Od pasáží, v ktorých je popísaný význam tohto pojmu, je následne teoretická časť zameraná na princípy a pojmy spojené s prenosom dát prostredníctvom DDS systému. Na poznatky o vlastnostiach tohto systému naväzuje následne praktická časť práce.

1.1 Internet vecí

Skratka IoT (internet vecí – Internet of Things) v sebe zahŕňa široké spektrum systémov rôznych veľkostí. Tvoria ich zariadenia s limitovanými parametrami z pohľadu použitého hardvéru a potrebnej energie pre ich funkčnosť. Za ich ďalšiu črtu sa može považovať čo najnižšia potrebná váha a veľkosť. Tieto vlastnosti ich oddeľujú od zariadení, akými sú serverové systémy, desktopové počítače, laptopy, smartfóny a iné. Typickými príkladmi zariadení, ktoré tvoria prostredie IoT, môžu byť senzory, aktuátory a rôzne iné inteligentné zariadenia. Tieto zariadenia sú medzi sebou prepojené a vymieňajú si informácie v reálnom čase. Na základe výsledku ich následného spracovania može byť vyvolaná určitá akcia [1, 2, 3].

Počet IoT zariadení pripojených k internetu neustále rastie. Štatistiky od spoločnosti Gartner pre rok 2018 udávajú 11,196 miliardy IoT entít a predpovedajú nárast ich počtu na 20,415 miliardy v roku 2020. Pre tieto zariadenia je potrebné zaistenie ich vzájomnej konektivity s určitými požiadavkami na parametre prenosu a bezpečnosť. To vedie k vzniku komunikačných protokolov a štandardov od rôznych organizácii a pracovných skupín. Protokoly, pomocou ktoých si IoT zariadenia vymieňajú dáta, sú spájané s pojmom middleware [3, 4].

1.2 Middleware

V distribuovaných systémoch, akým je internet, middleware v zásade slúži na to, aby mohli spolu komunikovať ľudia, programy alebo samostatné zariadenia medzi sebou. V prostredí IoT sú entity vyvíjané na rôznych platformách, ktoré fungujú na hardvéri od rozličných výrobcov. Zmyslom middleware vrstvy je zabezpečiť medzi týmito entitami možnosť komunikácie a spracovania dát za účelom sprostredkovania určitých služieb. Vrstva tiež definuje QoS (Quality of Service) pre dané IoT aplikácie spolu s požiadavkami na bezpečnosť prenosu v reálnom čase [1, 5].

1.2.1 Príklady protokolov middleware vrstvy

K protokolom z prostredia IoT definujúcich spôsob middleware implementácie patria napríklad [1, 5]:

- **AMQP** - Advanced Message Queuing Protocol
- **CoAP** - Constrained Application Protocol
- **DDS** - Data Distribution Service
- **MQTT** - Message Queuing Telemetry Transport

AMQP

AMQP je M2M (Machine to Machine) protokol, ktorý vyvíjal John O'Hara, je založený na 2 spôsoboch výmeny správ pri komunikácii. Využité sú možnosti typu požiadavka/odpoveď (request/response) a publikovanie/odoberanie (publish/subscribe). Transportným protokolom je štandardne TCP (Transmission Control Protocol). Pre výmenu správ je potrebná konektivita so serverom, inak nazývaným broker. Ponúka taktiež možnosť nastavenia QoS [3, 6].

CoAP

CoAP protokol pochádzajúci od pracovnej skupiny IETF CoRE (Internet Engineering Task Force Constrained RESTful Environments). Jeho schopnosťou je preklad do HTTP (Hypertext Transfer Protocol) protokolu pre prepojenie s webom. Podporuje model požiadavka/odpoveď a modifkovaný model publikovanie/odoberanie. Využíva transportný protokol UDP (User Datagram Protocol) a tiež obsahuje možnosť pridania QoS [3, 6].

DDS

DDS špecifikácia bola štandardizovaná spoločnosťou OMG (Object Managment Group). Zameriava sa na výkonnostné požiadavky distribuovaných systémov pracujúcich v reálnom čase. Pre komunikáciu v nich je využívaný DCPS (Data-Centric Publish-Subscribe) systém. Jeho hlavnou myšlienkou je rozširovanie užitočných informácii k zariadeniam, ktoré sa o ne zaujímajú bez potreby centrálneho riadiaceho prvku. Namiesto neho je v DCPS využívaný GDS (Global Data Space). Zariadenia spadajúce do tohto priestoru v ňom môžu prispievať užitočnými dátami alebo ich odtiaľ získavať. Ďalej štandard obsahuje možnosť zaistenia QoS [7, 8].

MQTT

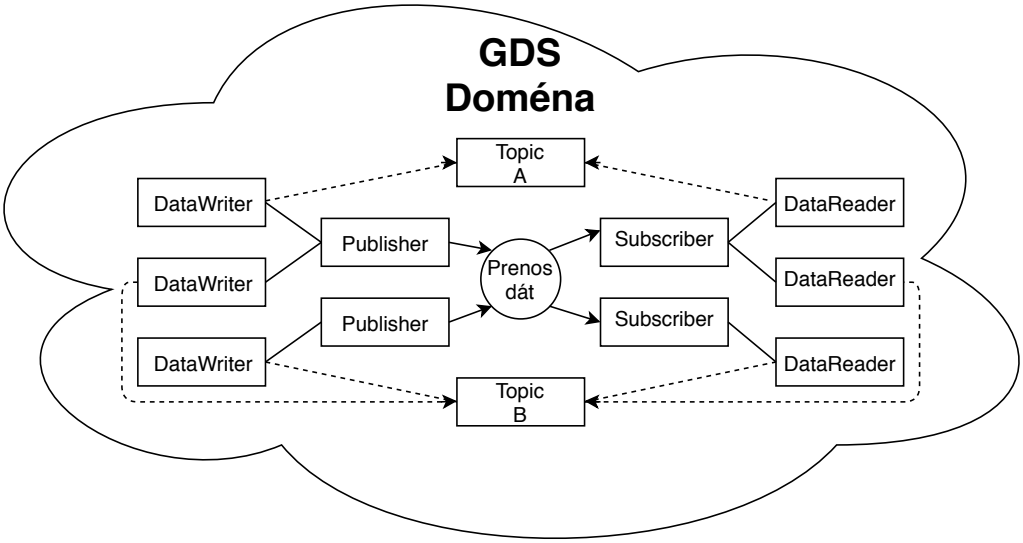
MQTT protokol začal byť v roku 1999 vyvíjaný spoločnosťou IBM a následne pridaný ako štandard organizáciou OASIS (Organization for the Advancement of

Structured Information Standards). Výmenu správ zaisťuje publish/subscribe model. Transportným protokolom je TCP. Komunikácia je riadená centrálnym prvkom. Protokol je navrhnutý ideálne pre zariadenia s limitovanými schopnosťami [9].

1.3 DCPS architektúra

Základom DDS systému v špecifikácii OMG DDS je jeho dátový model. Ten definuje GDS spolu s ďalšími prvkami DCPS architektúry, ktorej diagram je načrtnutý na obr. 1.1. Jej jednotlivé prvky rozširujú bázovú triedu DomainEntity okrem triedy s názvom DomainParticipiant. Ich význam bude priblížený v ďalších kapitolách. Triedy sa delia na [7, 10]:

- DomainParticipant
- DataReader
- DataWriter
- Publisher
- Subscriber
- Topic



Obr. 1.1: Diagram entít DCPS architektúry [11].

Global Data Space

Priestor, do ktorého vstupujú inštancie triedy Publisher a Subscriber tvorí jadro systému. Každé pridané dáta do GDS od Publisher entity sú propagované k entitám Subscriber, ktoré o ne majú záujem. Hlavným dôvodom použitia GDS v DDS je zabránenie situáciám, kedy v dôsledku zlyhania 1 entity nie je umožnené komunikovať ostatným účastníkom systému. Ako bolo už spomenuté, GDS neobsahuje žiadny centrálny riadiaci prvok a entity Publisher a Subscriber doňho môžu vstupovať dynamicky. Inštancie GDS sa nazývajú domény a ich identifikátorom je prirodzené číslo [7, 12, 13].

DomainParticipant

Túto triedu je možné si predstaviť ako vstupný bod do GDS domény. Hierarchicky slúži ako kontainer pre ostatné inštancie DDS datového modelu. Inými slovami sa dá popísať ako akási „továreň“ pre ostatné objekty, ktoré sa podieľajú na zapisovaní a čítaní prenášaných dát. Reprezentuje aplikáciu v doméne a virtuálne ju môže oddeľovať od ostatných aplikácii, s ktorými je fyzicky na 1 zariadení. Ďalej poskytuje možnosti pre ignorovanie iných objektov typu DomainParticipiant a Topic [7, 11].

DataReader

Úlohou objektov tohto typu je vytvárať rozhranie pre príjem a predanie dát asociovanému Subscriber objektu. Každý objekt typu DataReader je priradený k práve jednému objektu typu Subscriber. Podmienky splnenia výmeny užitočných informácií medzi objektami typu DataWriter a DataReader sú [11, 14]:

- Priradenie k objektu Topic s rovnakým menom.
- Participovanie v rovnakej doméne.
- Kompatibilita v QoS požiadavkach.
- Spoločný využívaný transportný protokol.

DataWriter

Objekty typu DataWriter využívajú aplikácie, ktoré chcú pridávať vlastné dáta do GDS domény. Tento objekt slúži ako rozhranie pre odoslanie dát do GDS domény od asociovaného objektu typu Publisher. Každý objekt typu DataWriter je priradený k práve jednému objektu typu Publisher [7, 11].

Publisher

Úlohou Publisher objektu je distribúcia dát všetkým relevantným objektom typu Subscriber, s ktorými zdieľa rovnakú doménu. Štandard OMG DDS oddeľuje jeho

funkciu striktne na publikovanie dát. To znamená že takýto objekt nemôže odoberať užitočné dáta v GDS doméne [7, 11].

Subscriber

Zodpovednosťou objekov typu Subscriber je prijatie publikovaných dát a ich sprístupnenie pre koncovú aplikáciu. Rovnako ako pre Publisher objekty, platí aj pre Subscriber objekty striktné oddelenie funkcionality v DDS systéme. Subscriber objekty nemôžu publikovať užitočné dáta, môžu ich len prijímať. Aj tieto objekty môžu pre koncovú aplikáciu zaisťovať QoS [7].

Topic

Objekty typu Topic sú dátovými jednotkami prenášanými medzi aplikáciami participujúcimi v rovnakom DDS systéme. Skladajú sa z dátového typu, unikátneho mena a QoS politiky. Topic každého datového typu v sebe môže špecifikovať 0 alebo viac kľúčov (keys). V terminológii DCPS sa ďalej uvádzajú inštancie (instances) konkrétneho dátového typu objektu Topic. Každá inštancia má svoj jedinečný kľúč. Jednotlivé publikované správy rôznych inštancií sa nazývajú vzorky (samples) [7, 11, 12].

1.4 RTPS protokol

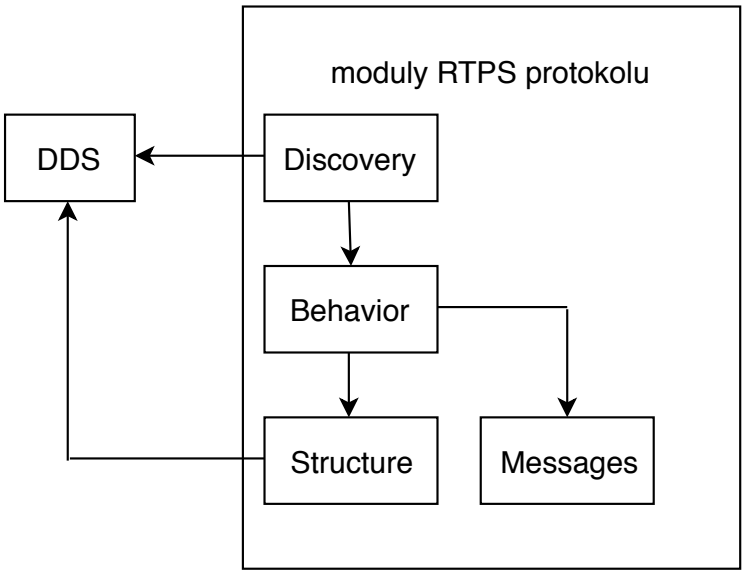
RTPS (Real-Time Publish-Subscribe) protokol bol vyvinutý spoločnosťou RTI (Real-Time Innovations) v roku 2002. Vznikol ako spojovací protokol pre DDS systémy. Bol navrhnutý pre fungovanie nad UDP protokolom. Jedná sa o PS (Publish-Subscribe) protokol, ktorého úlohou je prenášať dáta od entít Publisher k Subscriber entitám. Medzi cieľové funkcionality RTPS protokolu patria [15, 16, 17]:

- Plug and play konektivita, jej význam je umožniť aplikáciam byť automaticky objavovanými v sieti s možnosťou odpojenia a opätovného pripojenia bez nutnosti rekonfigurácie.
- Výkonnostné a QoS nastavenia pre spoľahlivé PS komunikácie pre aplikácie bežiace v reálnom čase v IP (Internet Protocol) sietiach.
- Konfigurovateľnosť z pohľadu požiadaviek na spoľahlivosť a včasné prenesenie dát.
- Škálovateľnosť pre umožnenie systémom rásť do väčších sietí.
- Rozšíriteľnosť protokolu o nové funkcie bez porušenia spätnej kompatibility.
- Odolnosť voči znemožneniu komunikácie v sieti následkom zlyhania jej 1 bodu.
- Typová bezpečnosť pre stabilnejšie aplikácie s menším počtom programovacích chýb.

1.4.1 Platform Independent Model

RTPS PIM (Platform Independent Model) obsahuje nasledujúce 4 moduly, ktoré sú zobrazené na obr. 1.2 [17]:

- **Structure** - definuje koncové body komunikácie.
- **Messages** - definuje množinu správ, ktorú si môžu koncové body posielať.
- **Behavior** - definuje množiny povolených výmen správ.
- **Discovery** - definuje ako sú objavované a konfigurované koncové body.

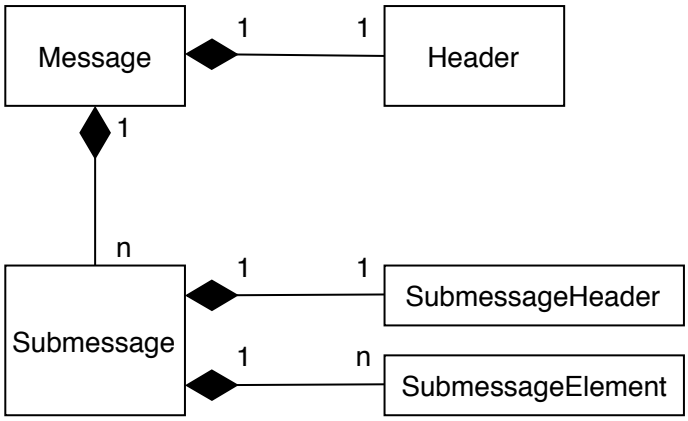


Obr. 1.2: RTPS PIM model [17].

1.4.2 RTPS správa

PIM Messages modul popisuje štruktúru a obsah správ vymieňaných medzi koncovými bodmi používajúcimi RTPS protokol. RTPS správa, ktorej štruktúra je načrtnutá na obr. 1.3, sa skladá z 2 hlavných častí. začiatok každej RTPS správy tvorí hlavička s fixnou dĺžkou. Za ňou nasleduje sekvencia podspráv (submessage), ktorých počet môže byť v každej správe odlišný. Hlavička RTPS správy je tvorená nasledujúcimi poliami [17]:

- **Protocol** - identifikuje dátovú jednotku ako RTPS správu.
- **Version** - identifikuje verziu RTPS protokolu.
- **VendorId** - identifikuje autora konkrétnej implementácie RTPS protokolu.
- **GuidPrefix** - definuje východiskový prefix pre všetky GUID (Globally Unique Identifier) nachádzajúce sa v danej správe.



Obr. 1.3: Štruktúra RTPS správy [17].

1.4.3 RTPS podspráva

Každá RTPS podspráva sa skladá z časti SubmessageHeader nasledovanou časťami SubmessageElement. Časť SubmessageHeader obsahuje okrem iných pole SubmessageId, ktoré identifikuje typ podsprávy. Jednotlivé typy RTPS podspráv sú [17]:

- **AckNack** - odosiela entita na prijímacej strane pre pozitívne alebo negatívne informovanie zdrojovej entity o prijatých sekvenčných číslach správ.
- **Data** - prenáša zmeny v dátach alebo stave životného cyklu zdrojových entitít k ich prijímateľom.
- **DataFrag** - významovo ekvivalentná k RTPS podspráve Data, využíva sa pri fragmentácii dátových jednotiek v sieti.
- **NackFrag** - odosiela entita na prijímacej strane pre informovanie zdrojovej entity o chýbajúcich sekvenčných číslach prijatých fragmentov.
- **Gap** - zdrojová entita notifikuje prijímateľov o irelevantnosti množiny sekvenčných čísel predchádzajúcich odoslaných správ.
- **Heartbeat** - zdrojová entita notifikuje prijímateľov o zmenách v jej poskytovaných dátových vzorkách pomocou sekvenčných čísel alebo žiada potvrdenie o ich prijatí.
- **HearbeatFrag** - podspráva s rovnakým významom ako Heartbeat, ktorá je využitá pri fragmentácii správ.
- **InfoDestination** - poskytuje informácie o cieľovej destinácii nasledujúcich prenášaných podspráv.
- **InfoReply** - explicitne udáva informácie alternatívnych cieľov odpovedí na prijaté podsprávy.
- **InfoSource** - indikuje spoločnosť a protokol použitý pri zapuzdrení podspráv.
- **InfoTimestamp** - časové razítko platné pre nasledujúce podsprávy v správe.
- **Pad** - uvádza použitie výplne.

1.5 Kvalita služby

Štandard DDS ponúka možnosť nastavenia pravidiel pre zaistenie kvality služby, ktoré vplývajú na charakter správania sa daných DDS systémov. Nastavením množiny týchto pravidiel sa dá docieliť napríklad určitá spoľahlivosť doručenia dát. Jednotlivé objekty, z ktorých sa skladá politika QoS v DDS systémoch sú potomkami objektu QosPolicy. Tieto objekty sú definované štruktúrami v ktorých sú špecifikované východiskové hodnoty pravidiel. Množina týchto pravidiel je aplikovaná na všetky entity DCPS architektúry. Počet pravidiel v tejto množine je pre každú z entít DCPS architektúry rôzny. Niektoré z týchto prvkov sú aplikované len pre 1 typ entity, iné môžu byť nastavené pre viac typov. Kompletný obsah množiny spolu s vysvetlením jej atribútov je popísaný v DDS špecifikácii [7, 11, 18].

Spôsob, akým sú vyjednávané podmienky zaistenia kvality služby v DDS, je založený na modeli požiadavky a ponuky. Entita typu DataWriter má priradenú množinu svojich QoS atribútov, ktorú ponúka entite typu DataReader. Entita typu DataReader má taktiež nastavenú množinu QoS možností, ktorú požaduje pre vytvorenie spojenia. Ak entita typu DataWriter nedokáže uspokojiť QoS požiadavky entity typu DataReader, tak medzi nimi nenastane zhoda a zostavenie spojenia je zamietnuté z dôvodu inkompatibility QoS. Pre upresnenie, pre nadviazanie spojenia nesmie byť požiadavka na kvalitu služby entity typu DataReader pre daný Topic striktnejšia ako ponuka od entity typu DataWriter. V opačnom prípade to neplatí. Ak entita typu DataReader požaduje menej striktnú úroveň daného QoS atribútu ako jej ponúka entita typu DataWriter, je spojenie zostavené s úrovňou požadovanou entitou typu DataReader [7, 19].

1.5.1 QoS atribúty

V nasledujúcom zozname sú uvedené niektoré QoS atribúty, ktoré špecifikuje štandard DDS. K jednotlivým atribútom je v ďalších častiach podkapitoly uvedený popis ich účelu. Príklady špecifikovaných QoS atribútov sú [7]:

- DURABILITY
- DEADLINE
- OWNERSHIP
- LIVELINESS
- PARTITION
- RELIABILITY
- HISTORY

DURABILITY

Atribút určuje, či bude práve odoslaná dátová vzorka uložená pre odoslanie entitám, ktoré začnú daný Topic odoberať neskôr v čase. Štandard DDS definuje 4 hodnoty, z ktorých sú povinne implementované hodnoty VOLATILE a TRANSIENT_LOCAL. Hodnota VOLATILE znamená neukladanie vzoriek pre entity, ktoré začnú daný Topic odoberať v budúcnosti. Hodnota TRANSIENT_LOCAL ponecháva vzorky dostupné kým ich publikujúca entita nezanikne [7, 11, 18, 19].

DEADLINE

Pomocou tohto atribútu je definovaná perióda, počas ktorej je očakávané odoslanie novej vzorky na strane entity typu DataWriter a prijatie na strane entity typu DataReader. Aplikácia môže indikovať situácie, kedy je nová vzorka odoslaná alebo prijatá po uplynutí tejto periódy [7, 11, 19].

OWNERSHIP

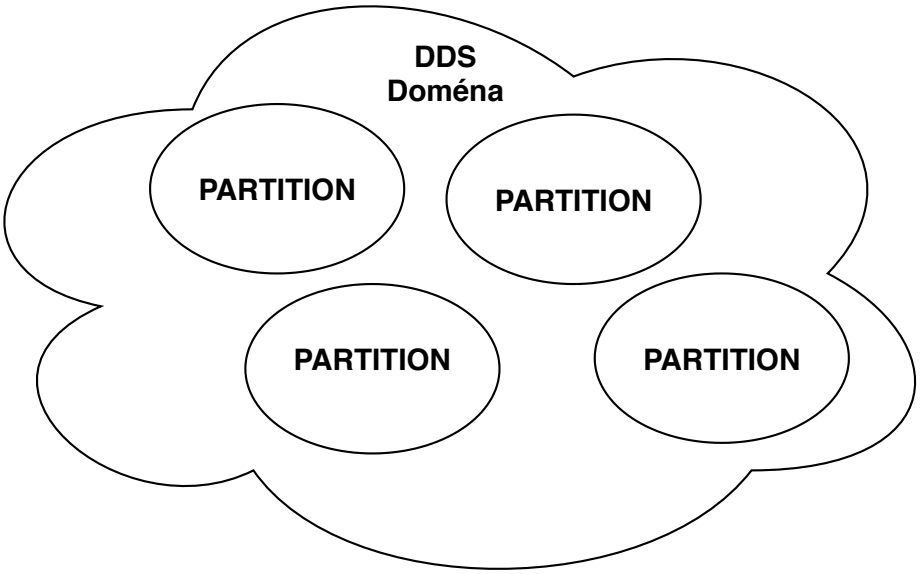
Atribút rozhoduje, či môže viacero entít typu DataWriter prispievať do DDS domény rovnaké inštancie (entita Topic + identifikátor). Hodnoty definované pre tento atribút sú SHARED a EXCLUSIVE. Hodnota SHARED indikuje, že inštanciu môže do DDS domény prispievať viacero DataWriter entít, pri hodnote EXCLUSIVE to môže byť 1 DataWriter [7, 11, 19].

LIVELINESS

Atribút slúži na kontrolu aktívnosti entít participujúcich v danej DDS doméne. Špecifikácia DDS udáva 3 hodnoty tohto atribútu, ktorými sú AUTOMATIC_LIVELINESS_QOS, MANUAL_BY_PARTICIPANT_LIVELINESS_QOS a MANUAL_BY_TOPIC_LIVELINESS_QOS. Ďalej je špecifikovaná perióda, počas ktorej je očakávaná správa pre overenie aktívnosti participujúcej entity. Pri hodnote AUTOMATIC_LIVELINESS_QOS notifikujú o aktivite entít všetky typy podspráv, ktoré odošlú. Pri zvyšných hodnotách atribútu je pre notifikáciu aktivity požadovaný periodický prenos Heartbeat podspráv [7, 11].

PARTITION

Atribút logicky rozdeľuje DDS doménu na menšie časti. Takéto rozdelenie DDS domény je znázornené na obr. 1.4. Identifikátor PARTITION je typu *string*. Pre nadviazanie spojenia je potrebný zhodný názov danej PARTITION. Tento QoS atribút je priradený len entitám typu Publisher a Subscriber [7, 11, 19].



Obr. 1.4: Delenie DDS domény na logické časti [20].

RELIABILITY

Atribút určuje úroveň spoľahlivosti prenosu. Sú definované 2 hodnoty tohto atribútu. Hodnota BEST_EFFORT_RELIABILITY_QOS nezaručuje doručenie všetkých dátových vzoriek. Pri hodnote RELIABLE_RELIABILITY_QOS je očakávaný spoľahlivý prenos dátových vzoriek [7, 11, 18, 19].

HISTORY

Atribút rozdhoduje, ako si DataWriter a DataReader entity držia dátové vzorky. Nastavením konkrétnej hodnoty atribútu sa kontroluje, či majú byť uložené aj hodnoty prijatých vzoriek alebo tých na odoslanie, ktoré nie sú najnovšie. Atribút môže mať hodnotu KEEP_LAST, pri ktorej entity uchovávajú vzorky do určitej hĺbky. Táto hĺbka udáva počet najnovších uchovávaných vzoriek a jej východisková hodnota je 1. Ďalšou hodnotou atribútu je KEEP_ALL, ktorá sa snaží uchovať všetky vzorky [7, 11, 19].

Východiskové hodnoty QoS atribútov

Východiskové hodnoty QoS atribútov spomenutých v predchádzajúcich častiach podkapitoly sú pre entity typu DataWriter uvedené v tab. 1.1 a pre DataReader v tab. 1.2.

Tab. 1.1: Východiskové hodnoty niektorých QoS atribútov entity DataWriter [11].

QoS atribút	východisková hodnota
DURABILITY	VOLATILE_DURABILITY_QOS
OWNERSHIP	SHARED_OWNERSHIP_QOS
LIVELINESS	AUTOMATIC_LIVELINESS_QOS
RELIABILITY	RELIABLE_RELIABILITY_QOS
HISTORY	KEEP_LAST_HISTORY_QOS

Tab. 1.2: Východiskové hodnoty niektorých QoS atribútov entity DataReader [11].

QoS atribút	východisková hodnota
DURABILITY	VOLATILE_DURABILITY_QOS
OWNERSHIP	SHARED_OWNERSHIP_QOS
LIVELINESS	AUTOMATIC_LIVELINESS_QOS
RELIABILITY	BEST_EFFORT_RELIABILITY_QOS
HISTORY	KEEP_LAST_HISTORY_QOS

1.6 Bezpečnosť v DDS

Štandard OMG DDS 1.4 v sebe nešpecifikuje oblasť zabezpečenia komunikácie. V dnešnej dobe je však v sieťovej komunikácii kladený vysoký dôraz na jej zabezpečenie. V dôsledku toho vydala organizácia OMG v júli roku 2018 štandard zameraný na bezpečnosť s názvom About DDS Security 1.1. Táto špecifikácia sa zameriava na možné typy hrozieb v súvislosti s používaním DDS systémov bez zabezpečenia a ich riešeniami. Popisuje bezpečnostný model a 5 SPI (Service Plugin Interface), ktoré sú v DDS vyžadované pre plnenie rôznych úloh z oblasti zabezpečenia komunikácie. SPI, ktoré zaisťujú bezpečnosť v DDS spolu s ich významom sú [21]:

- **Authentication Service Plugin** - zabezpečuje overovanie identity. Obsahuje aj funkcie pre vzájomnú autentifikáciu entít typu participant v DDS.
- **AccessControl Service Plugin** - reguluje práva aplikácii v DDS systémoch. Napríklad do akých domén sa môžu pripojiť, aké operácie v nich môžu robiť.
- **Cryptographic Service Plugin** - implementuje alebo tvorí rozhranie s knižnicami ktoré prevádzajú rôzne kryptografické operácie.
- **Logging Service Plugin** - slúži pre zaznamenávanie udalostí spojených so zabezpečovacou časťou DDS.
- **Data Tagging Service Plugin** - popisuje značkovanie dátových vzoriek.

Špecifikácia popisuje aj 4 najrelevantnejšie typy útokov, ktorým majú jednotlivé funkcie daných SPI zabrániť. Typy týchto útokov sú [21]:

- Neautorizovaný odber v rámci DDS domény.

- Neautorizované prispievanie v rámci DDS domény.
- Falšovanie a preposielanie datových jednotiek.
- Neautorizovaný prístup k prenášaným dátam.

1.6.1 Bezpečnostný model

Bezpečnostný model popisuje možnosti zabezpečovania DDS systémov a objektov DCPS architektúry. Reguluje operácie, ktoré môžu objekty v danej DDS doméne vykonávať. V konečnom dôsledku ide o zabezpečenie jedinečnej DDS domény. Následne je to riadenie prístupu pre zápis a čítanie informácii vo vnúri konkrétnej domény. Ponúkané možnosti zabezpečenia tohto modelu v DDS systémoch sú [21]:

- Dôveryhodnosť prijatých datových vzoriek.
- Úplnosť datových vzoriek a správ, ktoré obsahujú.
- Autentifikácia entít participujúcich v DDS doméne.
- Autorizácia entít participujúcich v DDS doméne.
- Autentifikácia pôvodu správ.
- Autentifikácia pôvodu dát.

Postup prístupu aplikácie do zabezpečenej DDS domény sa dá zhrnúť do niekoľkých základných krokov. V prvom rade sa musí každý proces pri vstupe autentifikovať. Po overení svojej identity je ďalším krokom aplikácie autorizácia svojich práv v danej DDS doméne. Inými slovami sa jedná o uplatnenie nároku na špecifické akcie akými sú napríklad zápis a čítanie dát v rámci Topic inštancií vyskytujúcich sa v DDS doméne alebo vytváranie nových inštancií pomocou ich identifikátorov. Ďalším právom aplikácie v doméne môže byť právo na vytvorenie Topic inštancie nového datového typu. Riadenie prístupu je následne podporované nastavením kryptografických úkonov pre podporu dôveryhodnosti a úplnosti prenášaných dát [21].

1.6.2 Autentifikácia

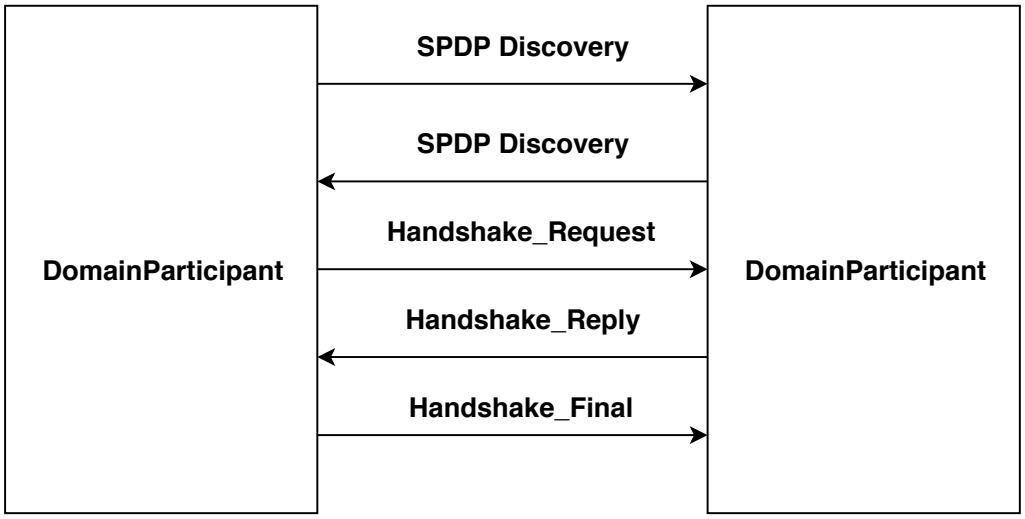
V špecifikácii DDS Security je popísaná vzájomná autentifikácia 2 objavených DomainParticipant entít pomocou digitálneho podpisu vytvoreného RSA (Rivest–Shamir–Adleman) alebo ECDSA (Elliptic Curve Digital Signature Algorithm) algoritmom. Prostredníctvom DH (Diffie-Hellman) algoritmu je získaný tajný kľúč, pomocou ktorého je zabezpečený prenos kľúča použitého pri šifrovaní užitočných dát. Pre povolenie rozšírenia PKI-DH (Public Key Infrastructure-Diffie-Hellman) autentifikácie musia byť k DomainParticipant entite priradené [21]:

- Digitálny certifikát certifikačnej autority.
- Digitálny certifikát identity a privátny kľúč entity DomainParticipant.

Po vzájomnom objavení DomainParticipant entít prostredníctvom SPDP (Simple Participant Discovery Protocol) protokolu sa pri povolenom rozšírení autentifikácie

uskutoční výmena ďaších 3 správ. Názvy vymieňaných správ sú Handshake_Request, Handshake_Reply a Handshake_Final. Priebeh výmeny správ pri autentifikácii je graficky znázornený na obr. 1.5. Počas výmeny týchto správ si 2 DomainParticipant entity medzi sebou uskutočnia [21, 22]:

- Výmenu certifikátov a dokumentov o oprávnení.
- Overenie digitálnych podpisov.
- Ustanovenie tajného kľúča prostredníctvom DH algoritmu.



Obr. 1.5: Výmena správ pri autentifikácii [22].

1.6.3 Kontrola prístupu

Po úspešnej autentifikácii DomainParticipant entity dochádza k overeniu jej oprávnení. Rozšírenie pre kontrolu prístupu spracováva 2 súbory. Ich názvy sú Governance a Permissions. Sú to súbory vo formáte XML (Extensible Markup Language) podpísané certifikačnou autoritou. Súbor Governance obsahuje pravidlá platiace pre danú DDS doménu. Súbor Permissions obsahuje práva pre DomainParticipant entitu v rámci danej DDS domény. Úkony, ktoré rozšírenie pre kontrolu prístupu vykonáva, sú [21, 22, 23]:

- Vytváranie konfigurácie pravidiel pre DDS doménu.
- Kontrola práv lokálnej DomainParticipant entity.
- Kontrola práv vzdialených DomainParticipant entít.

Governance dokument

Governance dokument špecifikuje konfiguráciu pravidiel zabezpečenia v rámci celej DDS domény. Úroveň zabezpečenia môže byť bez zabezpečenia, podpísanie alebo

zašifrovanie podspráv. Konfigurácia v sebe zahŕňa [21]:

- Spôsob zabezpečenia správ týkajúcich sa objavovania.
- Spôsob zabezpečenia celej RTPS správy.
- Spôsob zabezpečenia podspráv overujúcich aktívnosť entít.
- Možnosť neautentifikovanej DomainParticipant entity vstúpiť do domény a odoberať nezabezpečené podsprávy.

Okrem pravidiel pre danú doménu je v tomto dokumente špecifikované aj zaobchádzanie s Topic entitami. Možnosti nastavenia zabezpečenia pre konkrétnu Topic entitu sú [21]:

- Možnosť nastavenia prístupu k odoberaniu a prispievaniu Topic inštancií všetkým alebo len autorizovaným DomainParticipant entitám.
- Spôsob zabezpečenia na úrovni podspráv.
- Spôsob zabezpečenia užitočných dát aplikácie.

Permissions dokument

Permissions dokument definuje práva DomainParticipant entity v rámci danej DDS domény. Obsah dokumentu je tvorený sekvenciou pravidiel povolenia alebo zamietnutia určitej činnosti. V danom pravidle je vnorená DDS doména, na ktorú sa pravidlo aplikuje. Ďalej sú v ňom vnorené mená Topic inštancií, ktoré môžu byť danými DomainParticipant entitami publikované alebo odoberané. Taktiež môžu byť v pravidlách vnorené názvy PARTITION oblastí, na ktoré sa pravidlo vzťahuje [21].

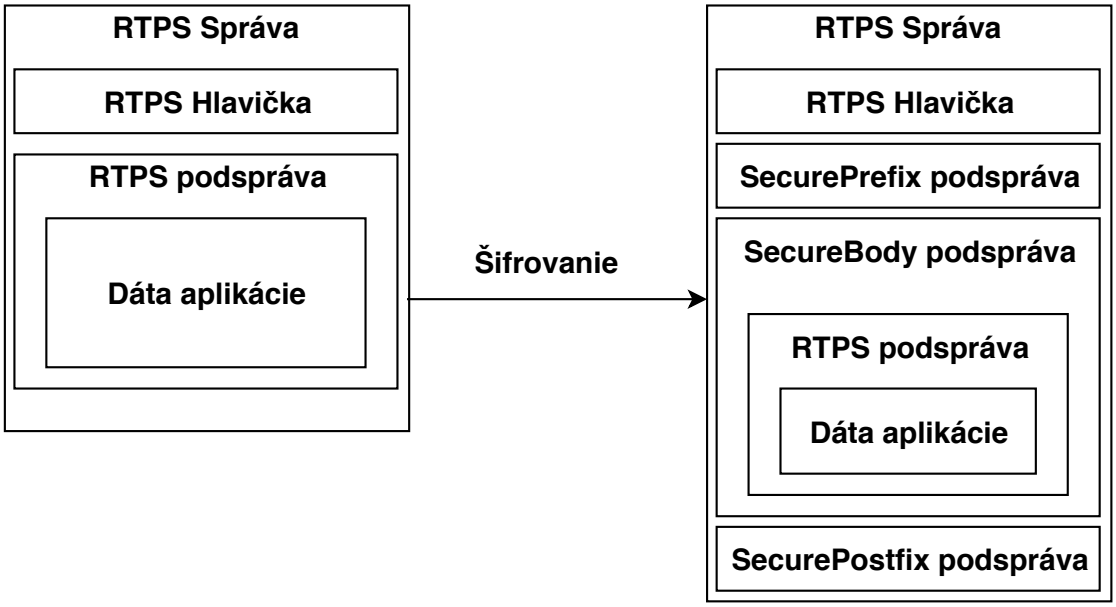
1.6.4 Šifrovanie

Šifrovanie RTPS prenosu je uskutočňované v Cryptographic SPI. Na šifrovanie RTPS prenosu je použitý AES (Advanced Encryption Standard) algoritmus. Špecifikácia DDS Security určuje 2 možnosti veľkostí kľúčov používaných pre šifrovanie prenosu. Uvedené sú v nej veľkosti kľúča 128 a 256 bitov. Úlohou tohto SPI je aj generovanie kľúčov a ich výmena. Prenos kľúča pre AES algoritmus je zabezpečený tajným kľúčom DH algoritmu ustanoveným pri autentifikácii DomainParticipant entít [21, 22].

V spojitosti so zabezpečením RTPS prenosu rozširuje špecifikácia DDS security množinu RTPS podspráv zo špecifikácie DDS. Príklad zloženia správy zabezpečeného RTPS prenosu je znázornený na obr. 1.6. RTPS podsprávy definované v špecifikácii DDS Security spolu s ich obecným významom sú [21]:

- **SecureBodySubMsg** - zapuzdruje 1 alebo viac originálnych RTPS podspráv do zabezpečenej formy.
- **SecurePrefixSubMsg** - slúži ako záhlavie pre SecureBodySubMsg podsprávu. Nesie informáciu o použitom zabezpečení podsprávy pred ktorou sa nachádza.

- **SecurePostfixSubMsg** - slúži ako zápätie, ktoré autentifikuje predchádzajúcu SecureBodySubMsg podsprávu. Nachádza sa hneď za SecureBodySubMsg podpsrávou, ktorá jej predchádza a pred ktorou sa hneď nachádza príslušná SecurePrefixSubMsg podspráva.
- **SecureRTPSPrefixSubMsg** - tvorí záhlavie celej zabezpečenej RTPS správy.
- **SecureRTPSPostfixSubMsg** - tvorí zápätie celej zabezpečenej RTPS správy.



Obr. 1.6: Zapuzdrenie nezabezpečenej a zabezpečenej RTPS podsprávy [21].

1.7 DDS implementácie

Existuje niekoľko spoločností, ktoré majú medzi svojími produktami DDS implementáciu. Jednotlivé DDS implementácie sa odlišujú napríklad podľa toho, či sú voľne dostupné alebo sa za ich používanie platí. Prípadnie či podporujú zabezpečenie a do akej miery ho implementujú. V nasledujúcom zozname sú uvedené niektoré spoločnosti spolu s názvami ich DDS implementácii [24]:

- **ADLINK Technology Ltd** - Vortex OpenSplice, DDS Community.
- **Object Computing, Inc.** - OpenDDS.
- **Real-Time Innovations** - RTI Connext.

Vortex OpenSplice, DDS Community

Medzi hlavné rozdiely medzi týmito implemetáciami od spoločnosti ADLINK Technology Ltd patrí to, že Vortex OpenSplice má platenú licencicu a DDS Community

voľnú licenciu. Z toho vyplývajú aj rozdiely medzi nimi. Vortex OpenSplice okrem iného obsahuje väčšie množstvo nástrojov. Značným rozdielom je však podpora zabezpečenia, ktorú DDS Community momentálne neobsahuje. Odlišnosťou Vortex OpenSplice od iných DDS implmentácii je aj podpora 3 veľkostí kľúčov pre AES algoritmus, ktoré sú 128 bitov, 192 bitov a 256 bitov [25, 26].

OpenDDS

DDS implementácia OpenDDS je od spoločnosti Object Computing, Inc. Táto implementácia je dostupná pod voľnou licenciou. Vo verzii 3.13 obsahuje beta implementáciu špecifikácie DDS Security, ktorá je vyvíjaná do vyspelejšej podoby. Beta implementácia napríklad nepodporuje šifrovanie na úrovni celej RTPS správy a podporuje dĺžku kľúča 256 bitov pre AES algoritmus [22, 27].

RTI Connext

Spoločnosť RTI (Real-Time Innovations) ponúka na trh DDS implementáciu pod platenou licenciou s názvom RTI Connext. Pre túto DDS implementáciu je deklarovaná implementácia celej DDS Security špecifikácie. List s technickými informáciami produktu uvádza podporované dĺžky kľúča AES algoritmu 128 bitov a 256 bitov. Produkt má aj ďalšie nástroje napríklad na administráciu a monitorovanie [28, 29].

2 Spracovanie témy a výsledky

Nasledujúce oddiely popisujú praktickú časť tejto práce. Na jej začiatku je zvolená OpenDDS implementácia nasadzovaná na rôzne platformy a analyzovaný je RTPS prenos. V ďalších častiach je priblížený princíp merania oneskorenia RTPS prenosu OpenDDS implementácie, za ktorým nasleduje popis implementácie DDS systému využívaného pri meraniach. V poslednej časti sú priblížené typy simulácií prenosu spolu s diskusiou výsledkov jednotlivých meraní.

2.1 Nasadenie OpenDDS implementácie

Táto sekcia praktickej časti práce sa zaoberá použitím OMG DDS implementácie OpenDDS 3.13 pre otestovanie a analýzu niekoľkých komunikačných scenárov, ktoré obsahuje. Najnovšia verzia tejto implementácie bola stiahnutá z oficiálnej stránky vývojárskej spoločnosti OCI (Object Computing, Inc). V tejto sekcii sú taktiež uvedené zariadenia, na ktorých bola OpenDDS implementácia nasadená. Okrem iného táto sekcia obsahuje SW (Software) komponenty potrebné pri kompilácii.

2.1.1 Použité zariadenia

Kompilácia implementácie OpenDDS 3.13 dostupnej na oficiálnych stránkach bola uskutočnená na 3 rôznych zariadeniach a 3 operačných systémoch. Na zariadeniach s operačným systémom Ubuntu 18.04 a Windows 10 figuruje aj Security časť OpenDDS 3.13 implementácie. Jednotlivé použité zariadenia spolu s ich HW (Hardware) parametrami a nainštalovanými operačnými systémami sú uvedené v tab. 2.1.

Tab. 2.1: Použité zariadenia s HW parametrami a operačným systémom.

Použité zariadenie	Typ procesoru	Veľkosť operačnej pamäte [GB]	Operačný systém
Dell Optiplex 755	Intel® Core™ 2 Duo E4600	6	Windows 10 Ubuntu 18.04
Dell Latitude E7440	Intel® Core™ i7-4600U	16	Windows 10 Ubuntu 18.04
Raspberry Pi 3 B	Broadcom Quad-Core BCM2837	1	Raspbian 9

2.1.2 Kompilácia

V práci bola OpenDDS implementácia nasadená na 3 rôzne operačné systémy. Pre úspešnú kompiláciu bolo potrebné nainštalovať na ne rôzne SW komponenty. Ich zoznam spolu s konkrétnymi verziami použitími v tejto práci je pre každý operačný systém uvedený v nasledujúcich častiach tejto kapitoly. Ďalšie verzie týchto komponentov otestované vývojármi je možné nájsť na [30].

Kompilácia na zariadeniach s operačným systémom Windows 10

Na operačnom systéme Windows 10 bolo pred uskutočnením samotnej kompilácie OpenDDS 3.13 potrebné nainštalovať nasledujúci SW:

- Visual Studio 2017 15.6.7
- ActivePerl 5.24.3
- Microsoft Visual C++ 2017 Redistributable 14.15.26706 cl 19.13.26132

Voliteľnou časťou pri kompilácii OpenDDS 3.13 implementácie je beta verzia špecifikácie OMG DDS Security. Požiadavky na kompiláciu tejto časti sú, aby operačný systém obsahoval nasledujúce SW komponenty:

- OpenSSL 1.1.1a
- Xerces-C++ 3.2.2

Kompilácia na zariadeniach s operačným systémom Ubuntu 18.04 a Raspbian 9

Pre nasadenie OpenDDS 3.13 implementácie na operačný Raspbian 9 bola využitá takzvaná „krížová kompilácia“ (cross-compile). Tým je myslené uskutočnenie kompilácie pre cieľovú distribúciu na hostiteľskej (Ubuntu 18.04). Samotná kompilácia bola uskutočnená so Security časťou. Použité SW komponenty spolu s ich konkrétnymi použitými verziami boli:

- gcc 8.2.0
- GNU Make 4.1
- Perl v5.24.1
- OpenSSL 1.1.1a
- Xerces-C++ 3.2.2

Implementácia OpenDDS 3.13, ktorá bola skompilovaná pre zariadenia s operačným systémom Ubuntu 18.04 obsahovala aj Security časť. SW komponenty využité pri kompilácii boli:

- gcc 7.3.0
- GNU Make 4.1
- Perl v5.26.1
- OpenSSL 1.1.1a
- Xerces-C++ 3.2.2

2.1.3 Messenger aplikácia

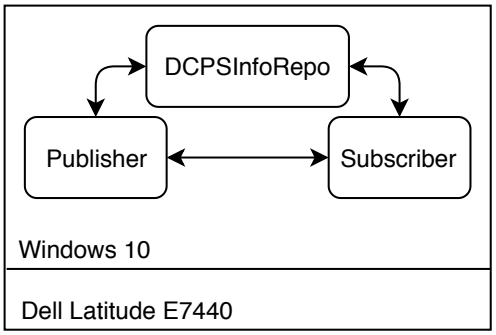
Po kompilácii bola pre testovanie OpenDDS 3.13 využitá Messenger aplikácia, ktorá sa nachádza medzi aplikáciami zakomponovanými v balíku tejto implementácie. Aplikácia sa v balíku nachádza v 2 variantách. Odlišnosťou týchto variánt je okrem iného spôsob objavovania Publisher a Subscriber entít, kedy 1 z variánt využíva proces DCPSInfoRepo. Cez tento proces nie je smerovaný samotný prenos dát medzi participujúcimi entitami. Ďalšou odlišnosťou je, že 1 z variánt obsahuje možnosť otestovania Security časti. Rozdielna je aj implementácia Publisher entity v rozdielnom počte generovaných vzoriek Topic inštancie, ktoré odosiela. Topic inštancia v tejto aplikácii prenáša 3 premenné dátového typu *string* a 2 typu *long*. Celkovo boli v testovacích scenároch využité nasledujúce 3 procesy tejto aplikácie:

- DCPSInfoRepo
- Publisher
- Subscriber

V 1 scenári, kedy bola táto aplikácia testovaná po prvotnej kompilácii bez Security časti, bola použitá varianta s procesom DCPSInfoRepo. Vo všetkých ostatných scenároch bola použitá varianta bez procesu DCPSInfoRepo a vzájomné objavovanie entít bolo uskutočňované prostredníctvom RTPS protokolu. Dokopy boli s Messenger aplikáciou uskutočnené 3 rôzne scenáre. Pri scenároch s 3 HW zariadeniami bol na ich fyzické prepojenie použitý prepínač Edimax es-3305p.

Scenár komunikácie na 1 zariadení bez Security časti

Simulácia tohto scenára prebehla na operačnom systéme Windows 10 zariadenia Dell Latitude E7440. Bloková schéma tohto scenára je načrtnutá na obr. 2.1. V príkazovom riadku Developer Command Prompt for VisualStudio 2017 bol v adresári aplikácie spustený Perl skript run_test.pl, ktorý spustil procesy Publisher, Subscriber a DCPSInfoRepo. Do konzoly bol vypísaný obsah prenesených dátových vzoriek Topic inštancie prijatých procesom Subscriber.



Obr. 2.1: Bloková schéma scenára komunikácie na 1 zariadení bez Security časti.

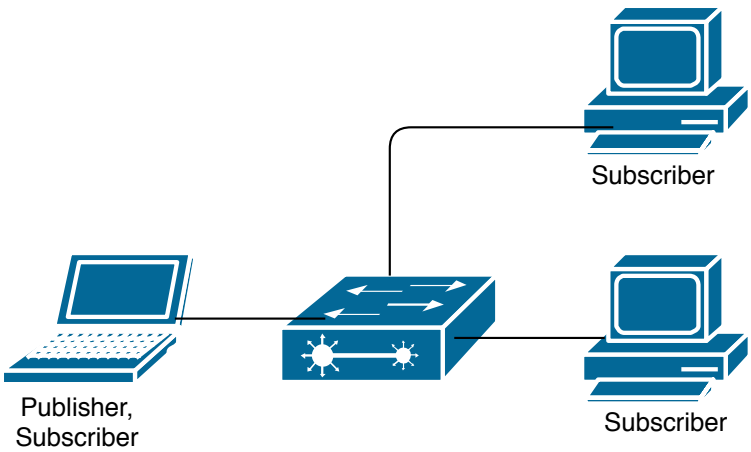
Scenár komunikácie medzi 4 zariadeniami bez Security časti

V tomto scenári figurovali 3 fyzické zariadenia a použité boli 3 typy operačných systémov. Na zariadení Dell Latitude E7440 bol okrem oprečného systému Windows 10 použitý aj virtualizovaný operačný systém Ubuntu 18.04. Virtualizácia bola uskutočnená pomocou softvéru VMware Workstation 15 Player. Zariadenia použité spolu s operačnými systémami, konkrétnymi spustenými procesmi a ich IP adresou v testovacej sieti sú uvedené v tab. 2.2. Adresa siete, v ktorej sa zariadenia pri testovaní nachádzali, bola 192.168.1.0/24.

Tab. 2.2: Scenár komunikácie 4 zariadení s aplikáciou Messenger.

Použité zariadenie	Operačný systém	Spustený proces	IP adresa
Dell Optiplex 755	Ubuntu 18.04	Subscriber	192.168.1.1/24
Dell Latitude E7440	Windows 10	Publisher	192.168.1.3/24
	Ubuntu 18.04	Subscriber	192.168.1.2/24
Raspberry Pi 3 B	Raspbian 9	Subscriber	192.168.1.10/24

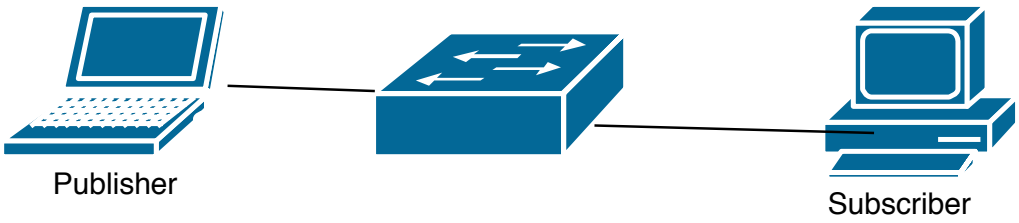
V tejto simulácii nefiguroval proces DCPSInfoRepo. Na objavovanie participujúcich aplikácií v doméne bol využitý RTPS protokol. Topológia testovacej siete je znázornená na obr. 2.2. Fyzické prepojenie HW zariadení bolo uskutočnené pomocou prepínača Edimax es-3305p. Na 3 zariadeniach bol spustený proces Subscriber a na 1 zariadení proces Publisher. Po nadviazaní spojenia medzi entitami vypísali procesy Subscriber do konzoly obsah vzoriek prijatých od procesu Publisher.



Obr. 2.2: Topológia scenára s 3 Subscriber procesmi a 1 Publisher procesom.

Scenár komunikácie medzi 2 zariadeniami so Security časťou

Tento scenár poslúžil na overenie funkčnosti OMG DDS Security časti v implementácii OpenDDS 3.13. Pri zachytávaní výmeny RTPS správ medzi procesom Publisher a Subsriber bol použitý program Wireshark. Proces Publisher bol spustený na zariadení Dell Latitude E7440 s operačným systémom Windows 10. Na zariadení Dell Optiplex 755 s operačným systémom Ubuntu 18.04 bol následne spustený Subscriber proces. Topológia scenára je znázornená na obr. 2.3. Pre porovnanie zachytenej komunikácie prebehla výmena správ medzi zariadeniami v tomto scenári aj bez Security časti. Zachytené RTPS správy sú analyzované v kap. 2.1.4.



Obr. 2.3: Topológia scenára komunikácie medzi 2 zariadeniami so Security časťou.

2.1.4 Analýza RTPS protokolu programom Wireshark

Táto časť sa zaoberá rozdielmi medzi zabezpečenou a nezabezpečenou výmenou RTPS správ. Konkrétne sú porovnávané zachytené RTPS toky zo scenára komunikácie medzi 2 zariadeniami so Security časťou popísaného v kap. 2.1.3. Rozpoznané rozdiely sú medzi RTPS podsprávami typu Data, ktoré prenášali vlastné dáta Messenger aplikácie. Pri zachytenej komunikácii bez zabezpečenia boli dáta v týchto podsprávach prenášané vo forme nezašifrovaného textu. Inými slovami bez zabezpečenia sa v podsprávach typu Data vyskytoval text v čitateľnej podobe. Obsah zachytenej podsprávy typu Data bez zabezpečenia je priblížený vo výpise 2.1.

V prípade zachyteného zabezpečeného prenosu dát bola podspráva typu Data, ktorá niesla vlastné dáta aplikácie, zašifrovaná a zabalená v podspráve SecureBodySubMsg. Pred podsprávou SecureBodySubMsg sa nachádzala podspráva SecurePrefixSubMsg. Za podsprávou SecureBodySubMsg sa nachádzala podspráva SecurePostfixSubMsg. Zachytené podsprávy týchto typov sa nachádzajú vo výpise 2.2.

Výpis 2.1: Nezašifrovaná podspráva typu Data.

```
submessageId: DATA (0x15)
    Flags: 0x05, Data present, Endianness bit
    octetsToNextHeader: 0
    0000 0000 0000 0000 = Extra flags: 0x0000
    Octets to inline QoS: 16
    readerEntityId: ENTITYID_UNKNOWN (0x00000000)
    writerEntityId: 0x00000002
    writerSeqNumber: 11
    serializedData
        encapsulation kind: CDR_LE (0x0001)
        encapsulation options: 0x0000
        serializedData: 0f00000043....
        serializedData: ....Comic Book Guy.Í....Review.Í
                        c.......Worst. Movie. Ever......
```

Výpis 2.2: Zašifrovaná podspráva typu Data.

```
submessageId: SEC_PREFIX (0x31)
    Flags: 0x00
    octetsToNextHeader: 20
    Secure Data Header
submessageId: SEC_BODY (0x30)
    Flags: 0x00
    octetsToNextHeader: 140
    Secured payload
        Secure Data Length: 136
        Secure Data: 190f9453b230ae...
submessageId: SEC_POSTFIX (0x32)
    Flags: 0x00
    octetsToNextHeader: 20
    Secure Data Tag
```

2.2 Záťažové testy s meraním oneskorenia

Cieľom tejto sekcie bolo dosiahnúť výsledky jednosmerného oneskorenia OpenDDS implementácie, ktoré môžu byť v budúcnosti porovnané s výsledkami oneskorení ďalších DDS implementácií. Hlavnou myšlienkou bolo vytvoriť Publisher a Subscriber entitu, ktoré budú spoločne tvoriť DDS systém a prostredníctvom simulácií budú schopné prevádzať merania v rôznych konfiguráciách. Požadované bolo umožniť porovnanie jednosmerného oneskorenia:

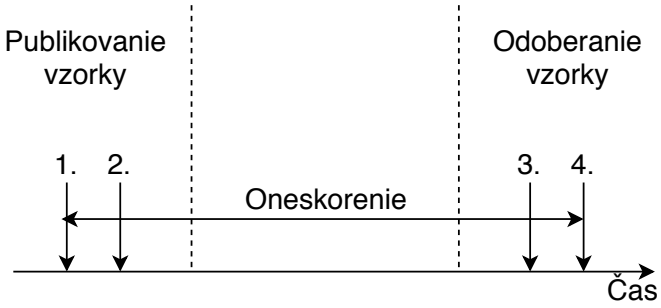
- Zabezpečeného a nezabezpečeného prenosu.
- Spoľahlivého a nespoľahlivého prenosu.
- Prenosov s rôznymi rýchlosťami odosielania vzoriek.

Pre simulovanie prenosu bol stanovený počet generovaných vzoriek entitou Publisher na 5000. Použité boli východiskové nastavenia QoS okrem RELIABILITY QoS parametru, pri ktorom boli využité obe jeho hodnoty. Možnosti konfigurácie simulácií prostredníctvom parametrov boli stanovené na nasledujúce:

- Možnosť nastavenia rýchlosti odosielania správ.
- Možnosť nastavenia spoľahlivosti prenosu.
- Možnosť nastavenia zabezpečeného prenosu.

2.2.1 Implementácia DDS systému

Základný princíp DDS systému schopného simulovať scenáre s meraním jednosmerného oneskorenia bol nasledujúci. Na začiatku prenosu bolo potrebné vložiť časové razítka do vzoriek, ktoré generuje Publisher entita. Pri následnom prímaní týchto vzoriek vygenerovanie ďalšieho časového razítka na strane Subscriber entity. Výsledné oneskorenie bolo stanovené ako rozdiel týchto 2 časových razítok. Graficky je tento princíp znázornený na obr. 2.4.



1. Odobratie časového razítka 1 a jeho vloženie do vzorky *n* pred jej odoslaním.
2. Odoslanie vzorky *n* entitou DataWriter.
3. Spracovanie vzorky *n* entitou DataReader.
4. Odobratie časového razítka 2 po spracovaní vzorky *n*.

Obr. 2.4: Princíp merania jednosmerného oneskorenia.

Pre DDS systém, ktorý bude fungovať na spomínanom princípe, bolo potrebné implementovať Publisher a Subscriber entitu podľa Messenger aplikácie, ktorej implementácia je popísaná v OpenDDS Developer's Guide [11]. Táto aplikácia bola testovaná v predchádzajúcej sekcii a nachádza sa v balíku OpenDDS implementácie. Ďalej bolo potrebné definovať Topic entitu, ktorej vzorky inštancií so štruktúrou znázornenou na obr. 2.5 budú prenášať premenné s nasledujúcim významom:

- Identifikátor inštancie.
- Sekvenčné číslo vzorky.
- Časové razítko vložené Publisher entitou.
- Sekvenciu bajtov s jej nastaviteľnou dĺžkou pre dosiahnutie požadovanej veľkosti prenášaných vzoriek.

Obr. 2.5: Struktúra Topic inštancie datového typu Payload.

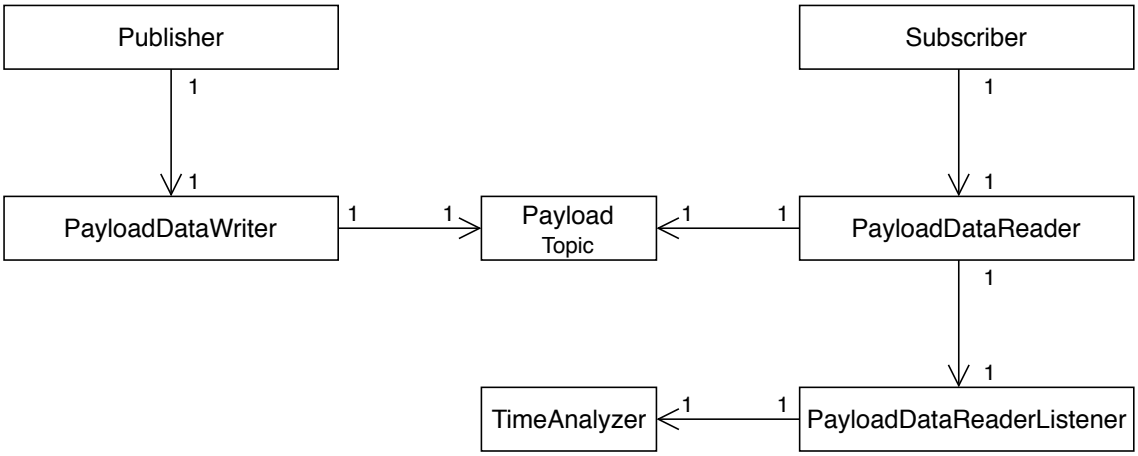
Po otestovaní funkčnosti implementácie popísaného základného DDS systému bolo ďalším cieľom rozšíriť predošlú implementáciu Publisher a Subscriber entít pre dosiahnutie požadovaných vlastností systému. K tomu bolo potrebné pridanie triedy s názvom *TimeAnalyzer*, ktorej funkcie využíva DataReader entita asociovaná k Subscriber entite. Vzťah entít vyskytujúcich sa v tomto systéme spolu s ich pomenovaním je priblížený graficky na obr. 2.6. Funkcie triedy *TimeAnalyzer* slúžia na:

- Výpočet okamžitého oneskorenia.
- Výpočet priemerného oneskorenia.
- Výpis informácií o vzorkách, ktoré daná DataReader entita spracovala.
- Výpis priemerného oneskorenia spolu s počtom spracovaných vzoriek.
- Uloženie výsledkov merania do súboru. V tejto funkcii je do súboru uložené oneskorenie všetkých prijatých vzoriek a ich počet. Ďalej je do tohto súboru uložené priemerné oneskorenie spolu s minimálnym a maximálnym oneskorením vypočítaným v tejto funkcii.

Spôsob implementácie odosielania vzoriek

Po inicializácii a asociácii entít DCPS architektúry sa v zdrojovom kóde Publisher entity nachádza vytvorenie Topic inštancie dátového typu *Payload*. Hneď za týmto krokom dochádza k plneniu identifikátora a sekvenčného čísla tejto inštancie na hodnotu 0. Dĺžka sekvencie bajtov je stanovená podľa parametra predaného aplikácii cez príkazový riadok. Taktiež je v tejto časti vypočítaná perióda odosielania vzoriek rovnako cez parameter predaný prostredníctvom príkazového riadku.

Za týmito krokmi nasleduje *for* cyklus, v ktorom je odčítané časové razítko následne predané do vzorky Topic inštancie prenášaného dátového typu *Payload*. Vzorka je následne odoslaná prostredníctvom metódy *write*, ktorú obsahuje DataWriter entita asociovaná s entitami Payload Topic a Publisher. Poslednými krokmi



Obr. 2.6: Diagram častí DDS systému s meraním oneskorenia.

tohto cyklu je zavolanie funkcie *usleep* na dĺžku periódy odosielania vzoriek vypočítanej pred týmto cyklom a inkrementácia sekvenčného čísla. Cyklus *for* popisaný v tejto sekcii obsahuje 5000 iterácii. To znamená, že Publisher entita vygeneruje celkom 5000 vzoriek.

Spôspob implementácie prijatia vzoriek

Rovnako ako v zdrojovom kóde Publisher entity, je v zdrojovom kóde Subscriber entity najskôr uskutočnená inicializácia a asociácia všetkých náležitých entít DCPS architektúry. Dôležitou časťou z hľadiska merania je asociácia PayloadDataReaderListener entity k DataReader entite, ktorú má daná Subscriber entita. Trieda PayloadDataReaderListener má z DataReaderListener rozhrania implementovanú metódu *on_data_available*. Táto metóda je vyvolaná vždy keď k nej asociovaná DataReader entita prijala vzorku.

Implementácia merania oneskorenia v tejto metóde spočíva v nasledujúcich krokoch. V jej tele je zavolaná funkcia *take_next_sample*, ktorá vykopíruje poslednú prijatú vzorku z vyrovnávacej pamäte DataReader entity a následne ju odtiaľ vymaže. Po prebratí danej vzorky vykopírovaním je odčítané časové rázítko a výsledné oneskorenie je spočítané ako rozdiel časového razítka odčítaného po vykopírovaní vzorky a po jej vytvorení. Okamžitá hodnota oneskorenia každej prijatej vzorky je spolu s ďalšími premennými prenášanými vo vzorkách vypísaná do konzoly. Rovnaká hodnota oneskorenia je uložená do vektoru objektu *TimeAnalyzer*, ktorý po ukončení prenosu vypočíta a vypíše do konzoly priemerné oneskorenie pomocou podielu súčtu jednotlivých oneskorení a dĺžky vektoru. Výsledky uložené do súboru po ukončení prenosu sú:

- Veľkosť prenášaných vzoriek.
- Veľkosť vektora okamžitých oneskorení.
- Hodnoty minimálneho, maximálneho a priemerného oneskorenia.
- Hodnoty vektora okamžitých oneskorení.

Implementácia zabezpečenia

Implementácia zabezpečenia bola uskutočnená podľa dokumentu Using DDS Security in OpenDDS [27]. Upravené boli dokumenty *Permissions.xml* a *Governance.xml* Messenger aplikácie. Šifrovanie prenosu bolo nastavené na úroveň podspráv Topic entity typu *Payload*. Prostredníctvom nástroja OpenSSL 1.1.1a bola vytvorená vlastná certifikačná autorita, ktorou boli podpísané identifikačné certifikáty participujúcich Publisher a Subscriber entít a *Permissions.xml* a *Governance.xml* dokumenty.

Otestovaná bola funkčnosť komunikácie s použitím RSA aj ECDSA certifikátov. Pri meraní so zabezpečením prenosu boli využité RSA certifikáty, keďže Cryptographic SPI a Authentication SPI sú od seba oddelené a na výsledné oneskorenie pri šifrovaní prenosu nemá spôsob autentifikácie vplyv. V zdrojových kódoch Publisher a Subscriber entity je cesta k týmto dokumentom nastavená na adresár v ktorom sa nachádzajú ich spustiteľné súbory.

2.2.2 Spúšťanie Publisher a Subscriber procesov

Základným spôsobom ako spustiť Publisher a Subscriber proces je prostredníctvom príkazového riadku. Oba procesy spoločne prímajú argument *-DCPSConfigFile* za ktorým nasleduje názov konfiguračného súboru. Tento konfiguračný súbor v sebe nesie nastavenie typu prenosu a či bude prenos zabezpečený. Multicast objavovanie a unicast prenos dát je uskutočňovaný prostredníctvom RTPS protokolu. Predanie tohto konfiguračného súboru Publisher a Subscriber procesom je potrebné pre uskutočnenie prenosu. V simuláciach boli použité 2 konfiguračné súbory s názvami:

- rtps_disc.ini - prenos bez zabezpečenia.
- rtps_disc_sec.ini - prenos so zabezpečením.

Pri spúšťaní Publisher procesu je možné prostredníctvom parametrov určiť veľkosť prenášaných vzoriek za argumentom *-b* a perióda ich odosielania za argumentom *-r*. Pokiaľ nie sú pri spúšťaní tieto parametre zadané, sú v zdrojovom kóde definované ich východiskové hodnoty na veľkosť vzorky 100 B a rýchlosť odosielania vzoriek na 100 za sekundu. Príklad spustenia Publisher procesu prostredníctvom príkazového riadku je nasledovný:

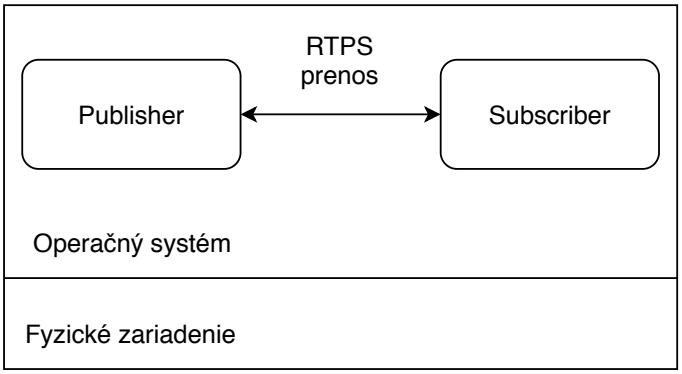
```
./publisher -DCPSConfigFile <názov konfiguračného súboru> -b <veľkosť
vzorky v B> -r <počet odoslaných správ za sekundu>
```

Pri spúšťaní Subscriber procesu je možné prostredníctvom parametru *-r* určiť, či bude prenos nespoľahlivý alebo spoľahlivý. Hodnota 1 za týmto parametrom nastaví RELIABILITY QoS na RELIABLE a hodnota 0 na BEST EFFORT. Východisková hodnota tohto parametru je v zdrojovom kóde definovaná na 0, keďže východisková hodnota RELIABILITY QoS parametru na entite DataReader je štandardom DDS definovaná na BEST EFFORT. Príklad spustenia Subscriber procesu prostredníctvom príkazového riadku je nasledovný:

```
./subscriber -DCPSConfigFile <názov konfiguračného súboru> -r <typ
spoľahlivosti prenosu>
```

2.2.3 Merania oneskorenia na 1 fyzickom zariadení

Meranie bolo uskutočnené na virtualizovanom OS Ubuntu 18.04 spúšťanom na zariadení Dell Latitude E7440 a OS Raspbian 9 spúšťanom na zariadení Raspberry Pi 3 B. Parametre týchto zariadení sú uvedené v tab. 2.1. Na virtualizovanom OS Ubuntu 18.04 bola veľkosť operačnej pamäte stanovená vedúcim práce na 3 GB. Pri jednotlivých simuláciach bol spustený Publisher a Subsriber proces tak ako je to graficky znázornené na obr. 2.7.



Obr. 2.7: Diagram merania oneskorenia na 1 fyzickom zariadení.

Skript vytvorený pre simulácie

Pre meranie oneskorenia na 1 zariadení bol vytvorený Perl skript, ktorý prostredníctvom príkazového riadku príma 3 parametre. Sú to parametre určujúce rýchlosť odosielania vzoriek, zabezpečenie a spoľahlivosť prenosu. V tomto skripte je definované pole veľkostí vzoriek, ktoré sa budú prenášať. Následne je prostredníctvom *for each* cyklu preiterované pole týchto veľkostí. V tele cyklu je spustený Publisher a Subsrciber proces spôsobom popísaným v kap. 2.2.2. Po ukončení spustených Publisher a Subscriber procesov sa príkazy v tele cyklu prevádzajú pre nasledujúcu

veľkosť vzorky definovanú v poli veľkostí vzoriek. Pri spúšťaní skriptu je potrebné dodržať poradie argumentov, kde argument s indexom 1 určuje rýchlosť odosielania vzoriek, s indexom 3 určuje zabezpečenie prenosu a s indexom 5 určuje spoľahlivosť prenosu. Rýchlosť odosielania vzoriek môže byť nastavená na 10 alebo 100 vzoriek za sekundu. Argumenty pre určenie spoľahlivosti a zabezpečenia prenosu prímajú hodnoty 0 alebo 1. Celý skript použitý v tejto kapitole je uvedený vo výpise A.1. Príklad spustenia skriptu je nasledovný:

```
./start_simulations.pl rate 100 sec 0 rel 0
```

Konfigurácie simulovaného prenosu

Merania boli prevádzané s rýchlosťou odosielania vzoriek 10 alebo 100 za sekundu. Pri týchto rýchlostiach boli kombinácie prenosu vzoriek nasledovné:

- Nespoľahlivý a nezabezpečený prenos.
- Nespoľahlivý a zabezpečený prenos.
- Spoľahlivý a nezabezpečený prenos.
- Spoľahlivý a zabezpečený prenos.

Veľkosti vzoriek v jednotlivých simuláciach bolo potrebné priebežne upravovať. Pôvodným zámerom bolo merať oneskorenie pri veľkosti vzoriek 100 kB až 1 MB s krokom 100 kB. Pri meraní prenosu so zabezpečením však bolo zistené, že pri veľkosti vzorky približne nad 64 kB vyhadzovala funkcia OpenDDS implementácie výnimku *preprocess_secure_submsg failed*. Táto výnimka vzniká v prípade problému spracovania zabezpečených podspráv DataReader entitou. Z toho dôvodu boli pre meranie oneskorania pri zabezpečenom prenose stanovené veľkosti vzoriek na 1 kB a 10 kB až 64 kB s krokom 10 kB.

S týmito hodnotami boli následne spravené aj merania pre spoľahlivý prenos na OS Ubuntu 18.04, keďže pri vyšších veľkostiach vzoriek pri spoľahlivom prenose správca pamäte zabíjal Publisher proces. Na zariadení Raspberry boli všetky merania z dôvodu obmedzených výkonnostných zdrojov prevedené rovnako pre veľkosti dát 1 kB a 10 kB až 64 kB s krokom 10 kB.

Finálne zvolené hodnoty veľkostí vzoriek pre merania na OS Ubuntu 18.04 a Raspbian 9 pri rôznych konfiguráciach prenosu boli nasledovné:

- 1 kB a 10 kB až 64 kB s krokom 10 kB pre všetky konfigurácie meraní na OS Ubuntu 18.04 a Raspbian 9.
- 100 kB až 1 MB s krokom 100 kB pre nezabezpečený a nespoľahlivý prenos na OS Ubuntu 18.04.

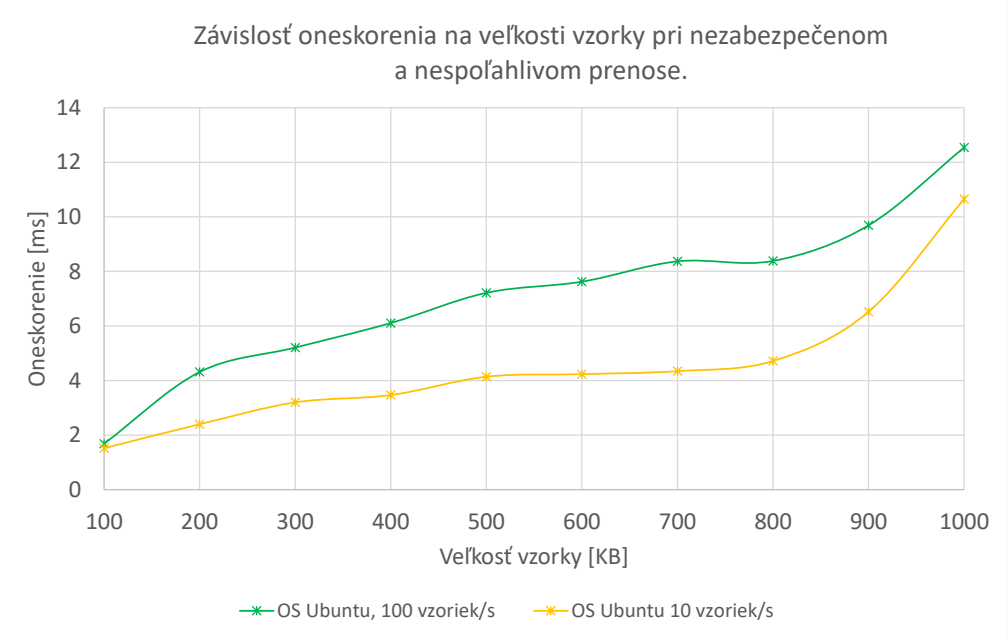
Výsledky simulácií

Po každej konfigurácií merania boli zaznamenané výsledky z jednotlivých vytvorených súborov pre konkrétnu veľkosť vzorky. Postupne boli do tabuliek .xlsx súboru vkladané hodnoty priemerného oneskorenia a veľkosť vektora s okamžitými oneskoreniami. Po vytvorení tabuliek boli vytvorené grafy so závislosťami oneskorenia na veľkosti vzorky pri rôznych konfiguráciách, ktoré bol záujem porovnať. Závislosti oneskorenia na veľkosti vzorky pri rôznych konfiguráciach prenosu pre OS Ubuntu 18.04 a Raspbian 9 sú vynesené do grafov a nachádzajú sa v prílohe C.1. Tabuľky s výsledkami oneskorení týchto simulácií sa nachádzajú v prílohách B.1 a B.2.

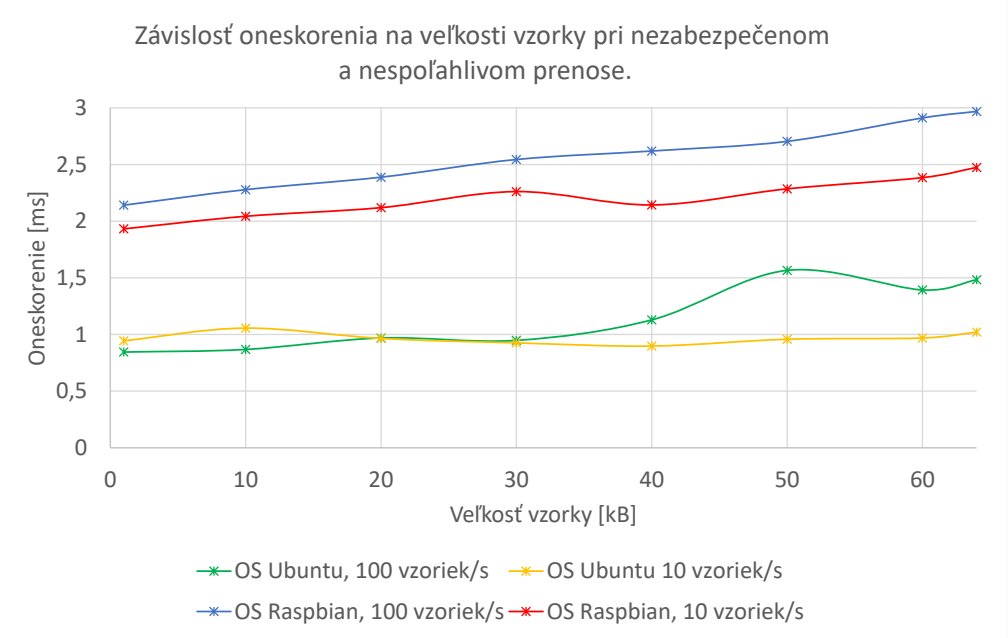
Závislosť oneskorenia nespoľahlivého a nezabezpečeného prenosu na OS Ubuntu 18.04 pre veľkosti vzoriek 100 kB až 1 MB je vynesená do grafu na obr. 2.8. Zo závislosti vynesenej v tomto grafe je zrejmé, že oneskorenie pri zvyšovaní veľkosti vzorky rastie. To je spôsobené kopírovaním prijatých dát z vyrovnávacej pamäte DataReader entity. Z grafu rovnako vyplýva, že pri vyššej rýchlosti odosielania vzoriek je oneskorenie vyššie, čo je spôsobené obmedzenou schopnosťou v rýchlosti spracovania vzorky DataReader entitou. Pri tejto konfigurácii prenosu s veľkosťami vzoriek 1 kB a 10 kB až 64 kB je závislosť oneskorenia na veľkosti vzorky pre OS Ubuntu 18.04 a Raspbian 9 vynesená do grafu na obr. 2.9. Pre krivky patriace prenosu na OS Ubuntu 18.04 je zrejmé, že pri daných veľkostiach vzorky je hodnota oneskorenia pri rýchlosti odosielania vzoriek 10 za sekundu takmer konštantná. Pri rýchlosti odosielania 100 vzoriek za sekundu je rozdiel oneskorení najnižšej a najvyššej vzorky 0,638 ms. Hodnoty oneskorení namerané na OS Raspbian 9 sú v porovnaní s hodnotami na OS Ubuntu 18.04 vyššie z dôvodu rozdielných výkonnostných parametrov zaradení, na ktorých sú tieto OS nainštalované.

Závislosť oneskorenia spoľahlivého a nespoľahlivého prenosu bez zabezpečenia pri rýchlosti odosielania vzoriek 10 za sekundu je porovnaná v grafe na obr. 2.10. Zo závislostí vynesených v tomto grafe je vidieť vplyv výkonnostného rozdielu medzi zariadeniami, na ktorých boli testy prevádzané. Vidieť tu výraznejší nárast oneskorenia pri zvyšujúcej sa veľkosti vzorky na OS Raspbian 9 v porovnaní s OS Ubuntu 18.04. Taktiež je z grafu zrejmý negatívny vplyv spoľahlivého prenosu na výsledné oneskorenie. Väčšie rozdiely medzi oneskorením spoľahlivého a nespoľahlivého prenosu boli zaznamenané opäť na zariadení s OS Raspbian 9.

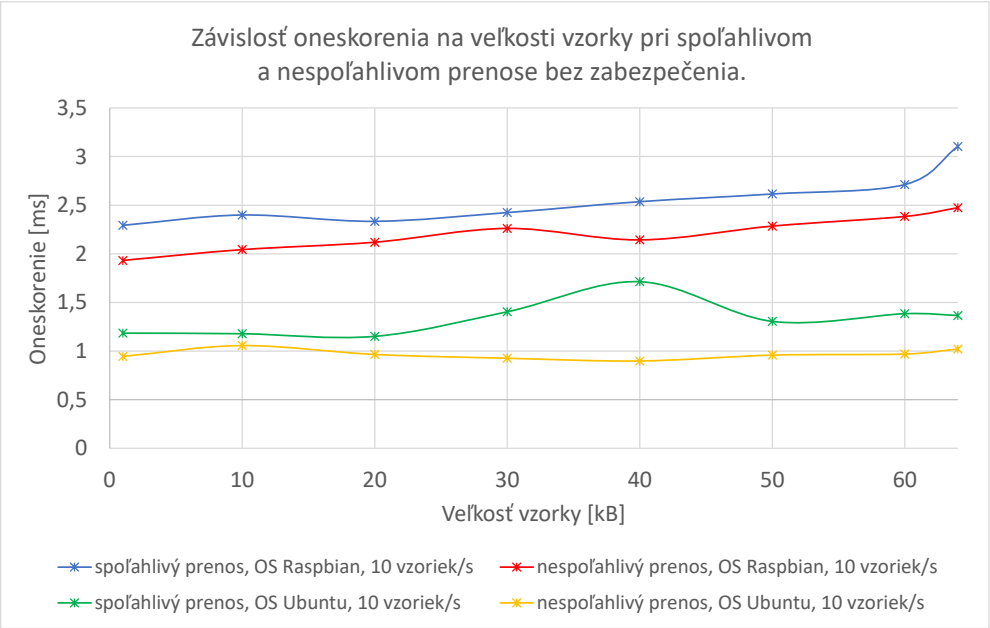
Vplyv zabezpečenia prenosu na oneskorenie je možné vidieť v grafe na obr. 2.11. V tomto grafe je vynesená závislosť oneskorenia na veľkosti vzorky pri nespoľahlivom prenose so zabezpečením a bez zabezpečenia s rýchlosťou odosielania vzoriek 10 za sekundu. Pri daných podmienkach sú rozdiely medzi zabezpečeným a nezabezpečeným prenosom na OS Ubuntu 18.04 minimálne. Naopak vplyv zabezpečenia prenosu na jeho oneskorenie sa výrazne prejavuje na zariadení s OS Raspbian 9.



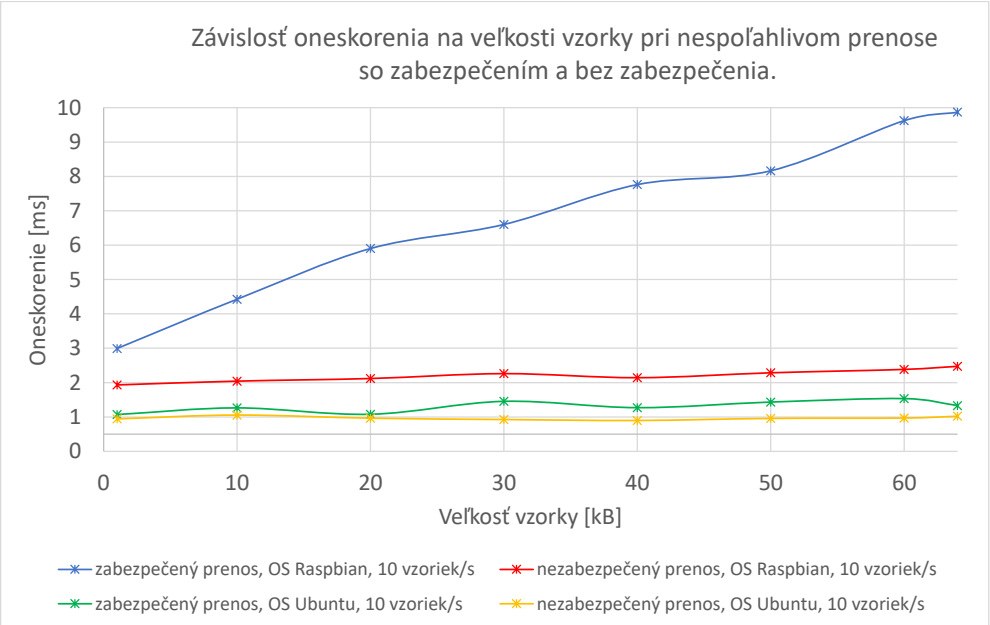
Obr. 2.8: Graf oneskorenia nespoľahlivého a nezabezpečeného prenosu.



Obr. 2.9: Graf oneskorenia nespoľahlivého a nezabezpečeného prenosu.



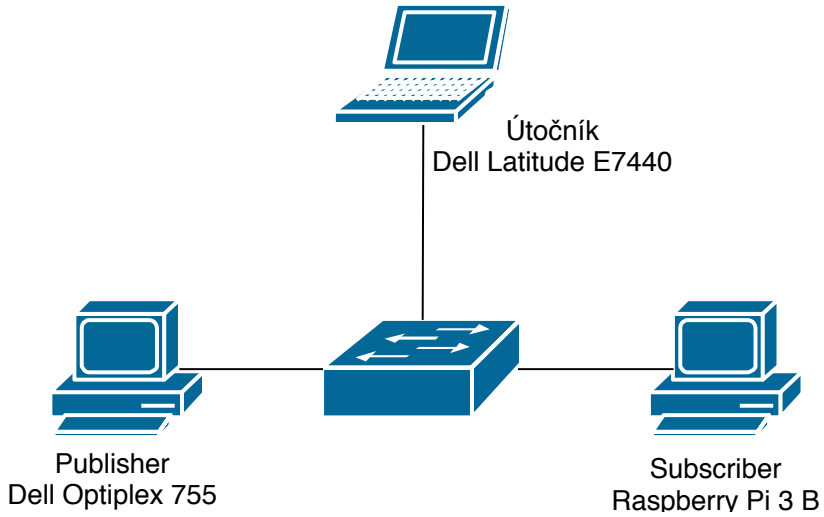
Obr. 2.10: Závislosť oneskorenia na veľkosti vzorky pri spoľahlivom a nespoľahlivom prenose bez zabezpečenia s rýchlosťou odosielania vzoriek 10 za sekundu.



Obr. 2.11: Závislosť oneskorenia na veľkosti vzorky pri nespoľahlivom prenose so zabezpečením a bez zabezpečenia s rýchlosťou odosielania vzoriek 10 za sekundu.

2.2.4 Merania oneskorenia medzi 2 fyzickými zariadeniami

Cieľom simulácií v tejto kapitole bolo zmerať oneskorenie RTPS prenosu medzi 2 fyzickými zariadeniami v lokálnej sieti s útočníkom a bez neho. V simuláciách boli použité OS Ubuntu 18.04 na zariadení Dell Optiplex 755 a Raspbian 9 na zariadení Raspberry Pi 3 B. Pre simulovanie RTPS prenosu bol využitý DDS systém, ktorého implementácia bola popísaná v kap. 2.2.1. Ako útočiace zariadenie bolo využitý Dell Latitude E7440 s virtualizovaným OS Ubuntu 18.04. Parametre zariadení sú uvedené v tab. 2.1. Zariadenia boli prepojené prepínačom Edimax ES-3305P. Graficky je prepojenie týchto zariadení znázornené na obr. 2.12.



Obr. 2.12: Schéma zapojenia zariadení s útočníkom.

V meraniach s útočníkom bol simulovaný MITM (man in the middle) útok. Pri tomto útoku bola zachytávaná komunikácia prebiehájúca medzi Publisher a Subscriber procesmi. Táto komunikácia nebola pre útočníka určená. Na útočiacom zariadení bol nainštalovaný nástroj Polymorph, prostredníctvom ktorého bol tento útok uskutočnený. Pre uskutočnenie útoku bola využitá technika nazývaná ARP (Address Resolution Protocol) Spoofing. Pre časovú synchronizáciu medzi zariadením Dell Optiplex 755 a Raspberry Pi 3 B bol využitý program Chrony. V simuláciách vystupoval ako NTP server OS Ubnuntu 18.04 zariadenia Dell Optiplex 755 [31].

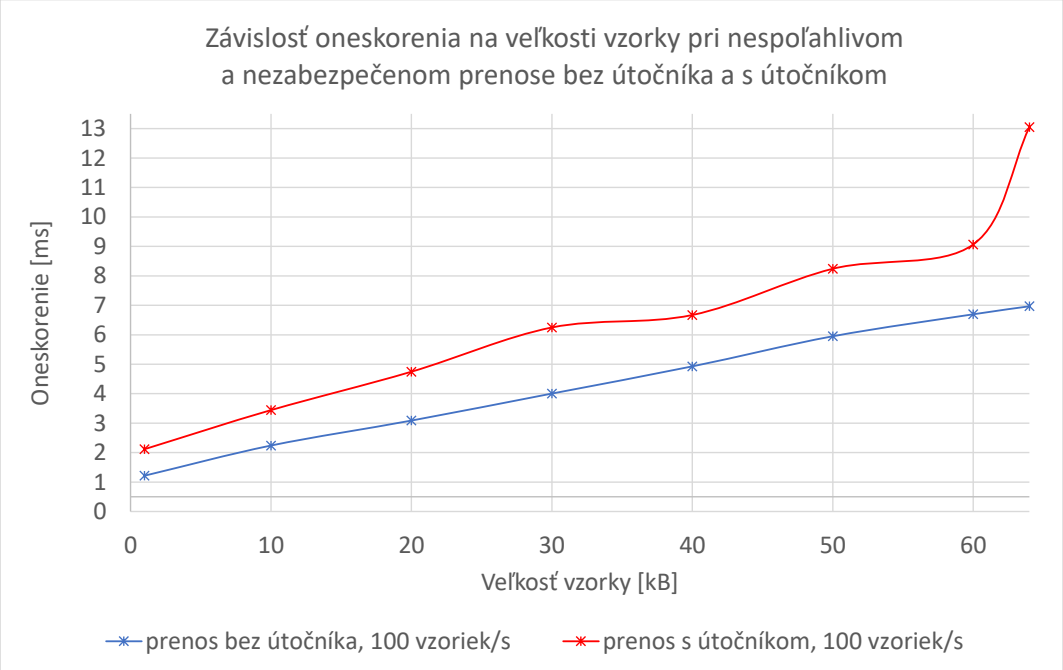
Uskutočnené boli 2 typy simulácií, s útočníkom a bez útočníka. Simulácie boli uskutočnené pre veľkosti vzoriek 1 kB a 10 kB až 64 kB s krokom 10 kB. Na OS Raspbian 9 bol spustený Subscriber proces a na OS Ubuntu 18.04 Publisher proces. Všetky merania boli uskutočnené s rýchlosťou odosielania vzoriek 100 za sekundu. Merania boli prevedené v nasledujúcich 4 konfiguráciách prenosu:

- Nespoľahlivý a nezabezpečený prenos.
- Nespoľahlivý a zabezpečený prenos.

- Spoľahlivý a nezabezpečený prenos.
- Spoľahlivý a zabezpečený prenos.

Výsledky simulácií

Výsledky simulácií sú uvedené v tabuľkách nachádzajúcich sa v prílohe B.3. Graficky boli porovnané hodnoty oneskorení prenosu s útočníkom a bez útočníka pre všetky uskutočnené konfigurácie simulácii. Grafy z týchto simulácií sa nachádzajú v prílohe C.2. Príklad vplyvu zachytávania prenosu útočníkom na oneskorenie prenosu je zobrazený v grafe na obr. 2.13. V tomto grafe je zobrazená závislosť oneskorenia na veľkosti vzorky pri nespoľahlivom a nezabezpečenom prenose s rýchlosťou odosielania vzoriek 100 za sekundu s útočníkom a bez neho. Zo závislosti oneskorenia s útočníkom je vidieť vplyv prechodu RTPS prenosu cez útočiace zariadenie. Vyššie hodnoty oneskorenia pri simulácií s útočníkom boli očakávané. Nárast oneskorenia pri simuláciách s útočníkom je spôsobený časom, ktorý potrebuje útočiace zariadenie pre spracovanie jednotlivých správ a ich následné odoslanie pravému príjemcovi.



Obr. 2.13: Porovnanie závislostí oneskorenia RTPS prenosu s útočníkom a bez neho.

3 Záver

V úvodnej časti práce zameranej na teoretické poznatky sú priblížené protokoly a štandardy middleware vrstvy distribuovaných systémov. Ďalšia časť je zameraná na vlastnosti štandardu OMG DDS a OMG DDS Security, z ktorých boli niektoré analyzované a overované v praktickej časti. V neposlednej rade je pasáž teoretickej časti venovaná RTPS protokolu, ktorý bol taktiež využitý v komunikačných scenároch praktickej časti. V závere teoretickej časti sú spomenuté niektoré DDS implementácie spolu s ich základnými črtami.

Praktická časť sa zameriava na niektoré z možností a vlastností implementácie OpenDDS 3.13, ktorá v nej bola využitá. V úvode praktickej časti sú spomenuté zariadenia a operačné systémy, na ktorých bola v práci táto implementácia nasadená a otestovaná. Uvedené sú tu konkrétne softvérové komponenty spolu s ich verziami použitými pri kompilácii implementácie na jednotlivých operačných systémoch.

Po kompilácii bolo úspešne otestované uskutočnenie komunikácie medzi zariadeniami v rozličných scenároch pomocou aplikácie nachádzajúcej sa v balíku implementácie. Konkrétne testovacie scenáre boli s 1 fyzickým zariadením, 3 fyzickými a 1 virtualizovaným zariadením a 2 fyzickými zariadeniami. V secnári s 1 fyzickým zariadením bolo vzájomné objavenie komunikujúcich entít pomocou procesu DCPSInfoRepo, ktorý je voliteľný centrálny prvok v OpenDDS 3.13 implementácii. V ďalších scenároch bol využitý RTPS protokol.

V ďalšej pasáži praktickej časti práce je priblížený DDS systém, ktorý bol vytvorený pre meranie oneskorenia prenosu pri rôznych nastaveniach z pohľadu zabezpečenia a spoľahlivosti prenosu. V tejto pasáži je popísaný použitý spôsob merania oneskorenia. Priblížený je postup vytvárania DDS systému, ktorý bol využitý pre simulácie prenosu vzoriek. Popísaná je tu aj implementácia jednotlivých entít DDS systému, ktorý bol pre meranie oneskorenia využitý.

Merania oneskorenia vytvoreného DDS systému boli uskutočnené na 1 zariadení a medzi 2 fyzickými zariadeniami. Pre simulácie prenosu na 1 zariadení bol vytvorený Perl skript, spúšťajúci procesy Publisher a Subscriber pri rôznom nastavení QoS a zabezpečenia. V skripte bola v poli definovaná množina veľkosti vzoriek pre ktoré sa mali simulácie uskutočniť.

Výsledky oneskorenia zo simulácií prenosu na 1 zariadení boli zaznamenané do tabuliek. Do grafov boli vynesené závislosti oneskorenia na veľkosti vzorky s rôznymi konfiguráciami rýchlosti odosielania, spoľahlivosti a zabezpečenia prenosu. Diskutované boli dosiahnuté rozdiely medzi zabezpečeným a nezabezpečeným a spoľahlivým a nespoľahlivým prenosom. V grafoch je vidieť vplyv veľkosti vzorky na oneskorenie najmä na zariadení Raspberry Pi 3 B. Na tomto zariadení boli tiež výraznejšie rozdiely oneskorenia pri zabezpečenej a nezabezpečenej komunikácií. Z pohľadu

spoľahlivosti prenosu je z grafov zrejmý rozdiel medzi spoľahlivým a nespoľahlivým prenosom. V prípade ak je prenos spoľahlivý a entita DataReader nemá priestor na uloženie vygenerovaných vzoriek, využie možnosť zablokovať prenos prostredníctvom metódy *write* pokiaľ nebudú uvoľnené pamäťové zdroje potrebné pre uloženie novej vzorky. Tento jav je zrejmý najmä pri rýchlosti odosielania vzoriek 100 za sekundu.

Pri meraní oneskorenia medzi 2 fyzickými zariadeniami bol do výsledného oneskorenia okrem času potrebného pre odoslanie a spracovanie vzorky entitami DDS systému započítaný aj čas potrebný na prenos vzorky sieťou. Výsledky meraní boli rovnako zaznamenané do tabuliek a závislosti oneskorenia na veľkosti vzorky pri rôznych konfiguráciách prenosu boli vynesené do grafov. Tieto merania boli uskutočnené 2 krát. V 2. prípade išiel prenos dát cez útočiace zariadenie zachytávajúce komunikáciu, ktorá mu nebola určená. V grafoch je vidieť vplyv útočníka na výsledné oneskorenie, ktoré je vyššie ako v prenose bez útočníka. Očakávaný nárast oneskorenia v meraniach s útočníkom v porovnaní s meraniami bez útočníka bol potvrdený a spôsobený časom potrebným pre spracovanie a preposlanie dát útočníkom. V grafoch je rovnako vidieť vplyv spoľahlivého prenosu na výsledné oneskorenie pri rýchlosti odosielania správ 100 za sekundu, ktorý bol diskutovaný pri meraniach oneskorenia na 1 zariadení.

Literatúra

[1] RAZZAQUE, Mohammad Abdur, Marija MILOJEVIC-JEVRIC, Andrei PALADE a Siobhan CLARKE. Middleware for Internet of Things: A Survey. *IEEE Internet of Things Journal* [online]. 2016, 3(1), 70-95 [cit. 2018-10-8]. DOI: 10.1109/JIOT.2015.2498900. ISSN 2327-4662. URL: http://ieeexplore.ieee.org/document/7322178/.

[2] Terminology for Constrained-Node Networks. *IETF Tools* [online]. Internet Engineering Task Force (IETF), 2014 [cit. 2018-10-8]. URL: https://tools.ietf.org/html/rfc7228.

[3] BADUGU, Narsimhmaswamy. Internet Of Things (IoT) Application Protocols. *IoT ONE* [online]. 12.1.2016 [cit. 2018-10-8]. URL: https://www.iotone.com/guide/internet-of-things-iot-application-protocols/g445.

[4] Gartner Says 8.4 Billion Connected Things Will Be in Use in 2017, Up 31 Percent From 2016. *Gartner* [online]. Egham: ©2018 Gartner, 2017, 7.2.2017 [cit. 2018-10-8]. URL: https://www.gartner.com/en/newsroom/press-releases/2017-02-07-gartner-says-8-billion-connected-things-will-be-in-use-in-2017-up-31-percent-from-2016.

[5] IoT Standards and Protocols. *Postscapes* [online]. © Postscapes, 2018, 20.8.2018 [cit. 2018-10-9]. URL: https://www.postscapes.com/internet-of-things-protocols/.

[6] NAIK, Nitin. *Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP* [online]. 2017 [cit. 2018-10-9]. DOI: 10.1109/SysEng.2017.8088251. ISBN 978-1-5386-3403-5. URL: http://ieeexplore.ieee.org/document/8088251/.

[7] Data Distribution Service (DDS). *OMG: Object Managment Group* [online]. © Object Management Group®, OMG®, 2014 [cit. 2018-10-14]. URL: https://www.omg.org/spec/DDS/1.4/PDF.

[8] Introduction to OpenDDS. *OpenDDS* [online]. © OCI [cit. 2018-10-14]. URL: http://opendds.org/about/articles/Article-Intro.html.

[9] KODALI, Ravi Kishore. An implementation of MQTT using CC3200. *2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)* [online]. IEEE, 2016, 2016, , 582-587 [cit. 2018-10-14]. DOI: 10.1109/ICCICCT.2016.7988017. ISBN 978-1-5090-5240-0. URL: http://ieeexplore.ieee.org/document/7988017/.

[10] PARDO-CASTELLOTE, G. OMG data-distribution service: architectural overview. *23rd International Conference on Distributed Computing Systems Workshops, 2003. Proceedings* [online]. IEEE, 2003, , 200-206 [cit. 2018-10-17]. DOI: 10.1109/ICDCSW.2003.1203555. ISBN 0-7695-1921-0. URL: http://ieeexplore.ieee.org/document/1203555/.

[11] OpenDDS Developer's Guide. *OpenDDS* [online]. © OCI [cit. 2018-10-15]. URL: http://download.objectcomputing.com/OpenDDS/OpenDDS-latest.pdf.

[12] CORSARO, Angelo. The DDS Tutorial. *LAAS: Laboratory for Analysis and Architecture of Systems* [online]. LAAS services [cit. 2018-10-15]. URL: http://www.laas.fr/files/SLides-A_Corsaro.pdf.

[13] CORSARO, Angelo. Applying the Data Distribution Service in an IoT Healthcare System. *ADLINK blog* [online]. © ADLINK Technology, 2014 [cit. 2018-10-15]. URL: https://istblog.adlinktech.com/tag/global-data-space/.

[14] DataWriter. *RTI community* [online]. © Real-Time Innovations, Inc. [cit. 2018-10-17]. URL: https://community.rti.com/glossary/datawriter.

[15] MEIER, Bill. Real-Time Publish-Subscribe (RTPS). *Wireshark Wiki* [online]. Wireshark Wiki, 2011 [cit. 2018-11-03]. URL: https://wiki.wireshark.org/Protocols/rtps.

[16] Real-Time Publish Subscribe (RTPS) Wire Protocol Specification. *IETF Tools* [online]. Internet Engineering Task Force (IETF), 2002 [cit. 2018-11-03]. URL: https://tools.ietf.org/html/draft-thiebaut-rtps-wps-00.

[17] ABOUT THE DDS INTEROPERABILITY WIRE PROTOCOL SPECIFICATION VERSION 2.2. *Object Managment Group* [online]. 2014 [cit. 2018-11-04]. URL: https://www.omg.org/spec/DDSI-RTPS/2.2.

[18] OpenSplice DDS C Tutorial Guide: Version 6.x. © 2014 PrismTech, 2014 [cit. 2019-03-20].

[19] The DDS Tutorial Release. *PrismTech* [online]. ADLINK Technology [cit. 2019-03-15]. URL: http://download.prismtech.com/docs/Vortex/pdfs/OpenSplice_DDSTutorial.pdf.

[20] CORSARO, Angelo. The DDS Tutorial ::Part I. *Object Managment Group* [online]. PrismTech, ©2009 [cit. 2019-03-19]. URL: http://www.omgwiki.org/dds/sites/default/files/Tutorial-Part.I.pdf.

[21] ABOUT THE DDS SECURITY SPECIFICATION VERSION 1.1. *Object Managment Group* [online]. 2018 [cit. 2019-03-15]. URL: https://www.omg.org/spec/DDSI-RTPS/2.2.

[22] REVOLUTIONIZING DATA DISTRIBUTION WITH AN OPEN AND SECURE DDS. *Object Computing* [online]. Object Computing, Inc. (OCI), ©2018 [cit. 2019-03-20]. URL: https://objectcomputing.com/index.php/download_file/view/2488.

[23] Support of DDS Security. *Welcome to Vortex Café's User Guide* [online]. ADLINK Technology Limited, ©2018 [cit. 2019-03-20]. URL: http://download.prismtech.com/docs/Vortex/html/cafe/user-guide/09-Security.html.

[24] DDS VENDOR DIRECTORY LISTING. *Object Managment Group* [online]. Object Management Group, ©2019 [cit. 2019-04-09]. URL: http://dds-directory.omg.org/vendor/list.htm.

[25] DDS Community. *ADLINK* [online]. ADLINK Technology, ©2018 [cit. 2019-04-09]. URL: https://www.adlinktech.com/en/data-distribution-service-dds-community.aspx.

[26] Secure Networking Configuration Release 6.x. *ADLINK knowledge base* [online]. ADLINK Technology Limited, ©2018 [cit. 2019-04-09]. URL: http://download.prismtech.com/docs/Vortex/pdfs/OpenSplice_SecureNetworkingGuide.pdf.

[27] Using DDS Security in OpenDDS. *OpenDDS* [online]. Object Computing, 2018 [cit. 2019-04-09]. URL: http://download.ociweb.com/OpenDDS/Using_DDS_Security_in_OpenDDS_3_13.pdf.

[28] RTI Connext DDS Professional. *RTI* [online]. RTI, ©2019 [cit. 2019-04-10]. URL: https://info.rti.com/hubfs/Datasheets/RTI_Datasheet_10017_Connext-DDS-Professional_V29_Web_0718.pdf.

[29] RTI Connext DDS Secure. *RTI* [online]. RTI, ©2019 [cit. 2019-04-10]. URL: https://info.rti.com/hubfs/Datasheets/RTI_Datasheet_10018_Connext-DDS-Secure_V7_Web_0718.pdf.

[30] Building. *OpenDDS* [online]. © OCI [cit. 2018-11-20]. URL: http://opendds.org/documents/building.html.

[31] RAMOS, Santiago Hernandez. Polymorph: A Real-Time Network Packet Manipulation Framework. *Exploit Database* [online]. Exploit Database, ©2019, April 2018 [cit. 2019-05-10]. URL: `https://www.exploit-db.com/docs/english/44457-polymorph-a-real-time-network-packet-manipulation-framework.pdf`.

Zoznam symbolov, veličín a skratiek

AES	Advanced Encryption Standard
AMQP	Advanced Message Queuing Protocol
ARP	Address Resolution Protocol
CoAP	Constrained Application Protocol
CoRE	Force Constrained RESTful Environments
CST	Composite State Transfer
DCPS	Data-Centric Publish-Subscribe
DDS	Data Distribution Service
DH	Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
XML	Extensible Markup Language
GDS	Global Data Space
GUID	Globally Unique Identifier
HTTP	Hypertext Transfer Protocol
HW	Hardware
IETF	Internet Engineering Task Force
IoT	internet vecí – Internet of Things
IP	Internet Protocol
MITM	man in the middle
MQTT	Message Queuing Telemetry Transport
M2M	Machine to Machine
OASIS	Organization for the Advancement of Structured Information Standards
OCI	Object Computing, Inc
OMG	Object Managment Group
OS	operačný systém
PIM	Platform Independent Model
PKI-DH	Public Key Infrastructure-Diffie-Hellman
PS	Publish-Subscribe
QoS	Quality of Service
RSA	Rivest–Shamir–Adleman
RTI	Real-Time Innovations
RTPS	Real-Time Publish-Subscribe
SPI	Service Plugin Interface
SW	Software
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

Zoznam príloh

A Skripty

A.1 Perl skript pre simulácie na 1 zariadení

Výpis A.1: Skript start_simulations.pl

```
1  #!/usr/bin/perl
2
3  use POSIX ":sys_wait_h";
4  use strict;
5  use warnings;
6
7  #Premenna rychlosti odosielania
8  #prebrana z prikazoveho riadku s indexom argumentu 1
9  my $rate=$ARGV[1];
10
11 #Premenna zabezpecenia prenosu
12 #prebrana z prikazoveho riadku s indexom argumentu 3
13 my $security=$ARGV[3];
14
15 #Premenna spolahlivosti prenosu
16 #prebrana z prikazoveho riadku s indexom argumentu 5
17 my $reliable=$ARGV[5];
18
19 #Premenna urcujuca pocet sekund po ktore
20 #ma $pid cakat pred pokracovanim vo vykonavani kodu
21 my $sleepPeriod;
22
23 my @sizesArray;  #Pole hodnot velkosti vzoriek v B
24 my $cfgFile;     #Premenna konfiguracneho suboru prenosu
25
26 #Premenna predana do parametru funkcie
27 #system() ktora spusti Subscriber proces
28 my $subCmd;
29
30 #Premenna predana do parametru funkcie
31 #system() ktora spusti Publisher proces
32 my $pubCmd;
33
34 #Overovanie hodnoty $security
35 #a definicia hodnot pola @sizesArray
36 if ($security==1){
```

```
37    $cfgFile="rtps_disc_sec.ini";
38    @sizesArray = (1000, 10000, 20000, 30000, 40000,
39    50000, 60000, 64000);
40  }
41  elsif ($security == 0){
42    $cfgFile="rtps_disc.ini";
43    @sizesArray = (100000, 200000, 300000, 400000,
44    500000, 600000, 700000, 800000, 900000, 1000000);
45  }
46  else{
47    print "Wrong␣security␣argument␣entered.
48  ␣␣Possible␣arguments:␣0;␣1.\n";
49    exit;
50  }
51  #Overovanie hodnoty $rate a definicia $sleepPeriod
52  if($rate == 100){
53    $sleepPeriod = (5000/$rate) * 3;
54  }
55  elsif ($rate == 10){
56    $sleepPeriod = 600;
57  }
58  else{
59    print "Wrong␣rate␣argument␣entered.
60  ␣␣Possible␣arguments:␣10;␣100.\n";
61    exit;
62  }
63  #Overovanie hodnoty $reliable a definicia premenniej $subCmd
64  if($reliable == 1){
65    $subCmd = "./subscriber␣-DCPSConfigFile␣$cfgFile␣-r␣1␣&";
66  }
67  elsif ($reliable == 0){
68    $subCmd = "./subscriber␣-DCPSConfigFile␣$cfgFile␣&";
69  }
70  else{
71    print "Wrong␣reliable␣argument␣entered.
72  ␣␣Possible␣arguments:␣0;␣1.\n";
73    exit;
74  }
75  #definicia @sizesArray pre spolahlivy a nezabespeceny prenos
76  if ($reliable == 1 && $security == 0){
77    @sizesArray = (1000, 10000, 20000, 30000, 40000,
```

```
78    50000, 60000, 64000);
79  }
80  #uprava hodnoty $sleepPeriod pri splneni podmienky
81  if ($reliable == 1 && $rate == 100){
82    $sleepPeriod = $sleepPeriod * 1.5;
83  }
84  #uprava hodnoty $sleepPeriod pri splneni podmienky
85  if ($reliable == 1 && $rate == 10){
86    $sleepPeriod = $sleepPeriod * 1.1;
87  }
88  #priradenie hodnoty do premennej $pubCmd
89  $pubCmd = "./publisher -DCPSConfigFile $cfgFile -r $rate";
90
91  #cyklus iterujuci cez pole @sizesArray
92  foreach my $messageSize (@sizesArray){
93    $pubCmd .= " -b $messageSize &"; #priradenie velkosti vzorky
94    my $pid = fork; #vytvorenie procesu potomka
95    if ($pid == 0){
96    my $retCode2 = system($pubCmd); #spustenie Publsiher
97    my $retCode1 = system($subCmd); #spustenie Subscriber
98    sleep($sleepPeriod);
99    exit(); #ukonecnie potomka
100   }
101   else {
102   waitpid $pid,0; #cakanie rodica na ukoncenie potomka
103   }
104   #priradenie hodnoty $pubCmd na vychodiskovu hodnotu
105   $pubCmd = "./publisher -DCPSConfigFile $cfgFile -r $rate";
106 }
107 print "Simulations finsished!\n";
```

B Tabuľky s výsledkami oneskorení

B.1 Výsledky oneskorení z OS Ubuntu 18.04

Tab. B.1: Oneskorenie nezabezpečeného a nespoľahlivého prenosu na OS Ubuntu.

Odoslaných vzoriek za sekundu: 10.			Odoslaných vzoriek za sekundu: 100.	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	0,945	5000	0,845	5000
10	1,056	5000	0,868	5000
20	0,965	5000	0,969	5000
30	0,926	5000	0,948	5000
40	0,898	5000	1,129	5000
50	0,958	5000	1,566	5000
60	0,969	5000	1,393	5000
64	1,021	5000	1,483	5000
100	1,518	5000	1,682	4997
200	2,397	963	4,313	1069
300	3,199	444	5,207	932
400	3,467	184	6,107	151
500	4,138	243	7,213	73
600	4,232	232	7,622	111
700	4,343	223	8,368	81
800	4,714	100	8,382	144
900	6,52	126	9,688	106
1000	10,65	128	12,54	96

Tab. B.2: Oneskorenie nezabezpečeného a spoľahlivého prenosu na OS Ubuntu.

Odoslaných vzoriek za sekundu: 10.			Odoslaných vzoriek za sekundu: 100.	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	1,184	5000	1,351	5000
10	1,178	5000	1,117	5000
20	1,151	5000	0,852	5000
30	1,404	5000	0,843	5000
40	1,713	5000	4,07	5000
50	1,305	5000	0,853	5000
60	1,384	5000	8,48	5000
64	1,366	5000	1,113	5000

Tab. B.3: Oneskorenie zabezpečeného a nespoľahlivého prenosu na OS Ubuntu.

Odoslaných vzoriek za sekundu: 10.			Odoslaných vzoriek za sekundu: 100.	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	1,071	5000	0,929	5000
10	1,264	5000	1,347	5000
20	1,079	5000	1,101	5000
30	1,455	5000	1,081	5000
40	1,271	5000	1,169	5000
50	1,434	5000	1,127	4999
60	1,535	5000	1,932	4999
64	1,332	5000	1,453	5000

Tab. B.4: Oneskorenie zabezpečeného a spoľahlivého prenosu na OS Ubuntu.

Odoslaných vzoriek za sekundu: 10.			Odoslaných vzoriek za sekundu: 100.	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	1,063	5000	1,025	5000
10	1,111	5000	1,373	5000
20	1,214	5000	1,274	5000
30	1,962	5000	1,322	5000
40	2,027	5000	10,61	5000
50	2,278	5000	1,355	5000
60	2,981	5000	1,362	5000
64	2,982	5000	1,246	5000

B.2 Výsledky oneskorení z OS Raspbian 9

Tab. B.5: Oneskorenie nezabezpečeného a nespoľahlivého prenosu na OS Raspbian.

Odoslaných vzoriek za sekundu: 10.			Odoslaných vzoriek za sekundu: 100.	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	1,932	5000	2,141	5000
10	2,043	5000	2,278	5000
20	2,119	5000	2,388	4999
30	2,261	5000	2,544	4999
40	2,143	5000	2,619	4997
50	2,285	5000	2,704	4996
60	2,384	5000	2,911	4939
64	2,474	5000	2,968	4931

Tab. B.6: Oneskorenie nezabezpečeného a spoľahlivého prenosu na OS Raspbian.

Odoslaných vzoriek za sekundu: 10.			Odoslaných vzoriek za sekundu: 100.	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	2,293	5000	2,385	5000
10	2,399	5000	2,685	5000
20	2,334	5000	2,771	5000
30	2,424	5000	20,37	5000
40	2,536	5000	93,82	5000
50	2,615	5000	96,74	5000
60	2,712	5000	95,64	5000
64	3,104	5000	96,96	5000

Tab. B.7: Oneskorenie zabezpečeného a nespoľahlivého prenosu na OS Raspbian.

Odoslaných vzoriek za sekundu: 10.			Odoslaných vzoriek za sekundu: 100.	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	2,992	5000	3,173	5000
10	4,426	5000	4,265	5000
20	5,901	5000	5,814	5000
30	6,601	5000	7,253	5000
40	7,761	5000	8,949	4998
50	8,163	5000	10,19	5000
60	9,622	5000	11,79	4999
64	9,867	5000	12,29	5000

Tab. B.8: Oneskorenie zabezpečeného a spoľahlivého prenosu na OS Raspbian.

Odoslaných vzoriek za sekundu: 10.			Odoslaných vzoriek za sekundu: 100.	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	3,503	5000	2,946	5000
10	4,902	5000	4,284	5000
20	6,554	5000	5,811	5000
30	8,119	5000	6,924	5000
40	9,485	5000	8,352	5000
50	10,51	5000	103,1	5000
60	12,34	5000	102,3	5000
64	13,16	5000	102,7	5000

B.3 Výsledky oneskorení medzi 2 zariadeniami

Tab. B.9: Oneskorenie nezabezpečeného a nespoľahlivého prenosu medzi 2 zariadeniami s útočníkom a bez neho.

Meranie bez útoku:			Meranie s útokom:	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	1,216	5000	2,116	5000
10	2,239	5000	3,442	5000
20	3,087	4999	4,746	4999
30	4,004	4996	6,248	4996
40	4,927	4995	6,671	4996
50	5.949	4995	8,241	4992
60	6,695	4992	9,066	4992
64	6,967	4992	13,05	4990

Tab. B.10: Oneskorenie nezabezpečeného a spoľahlivého prenosu medzi 2 zariadeniami s útočníkom a bez neho.

Meranie bez útoku:			Meranie s útokom:	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	1,412	5000	2,334	5000
10	2,421	5000	3,473	5000
20	5,468	5000	6,557	5000
30	96,94	5000	100,3	5000
40	101,1	5000	314,4	5000
50	104,3	5000	320,1	5000
60	318,6	5000	331,7	5000
64	325,6	5000	411,2	5000

Tab. B.11: Oneskorenie zabezpečeného a nespoľahlivého prenosu medzi 2 zariadeniami s útočníkom a bez neho.

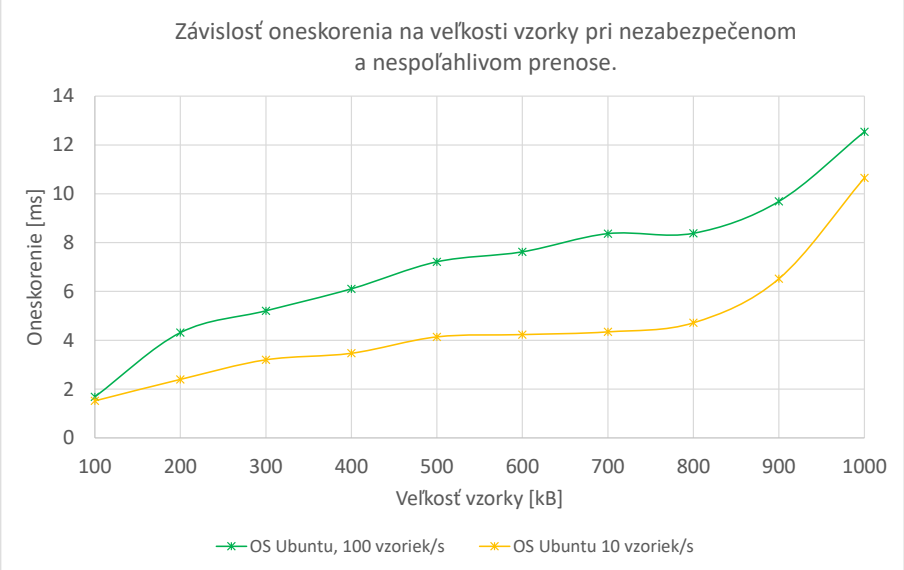
Meranie bez útoku:			Meranie s útokom:	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	2,026	5000	2,735	5000
10	3,551	5000	4,286	5000
20	5,213	4999	6,895	4999
30	6,671	5000	8,671	4998
40	7,116	4999	9,594	4995
50	8,605	4999	10,852	4998
60	9,905	4999	13,282	4997
64	10,23	4996	18,08	4989

Tab. B.12: Oneskorenie zabezpečeného a spoľahlivého prenosu medzi 2 zariadeniami s útočníkom a bez neho.

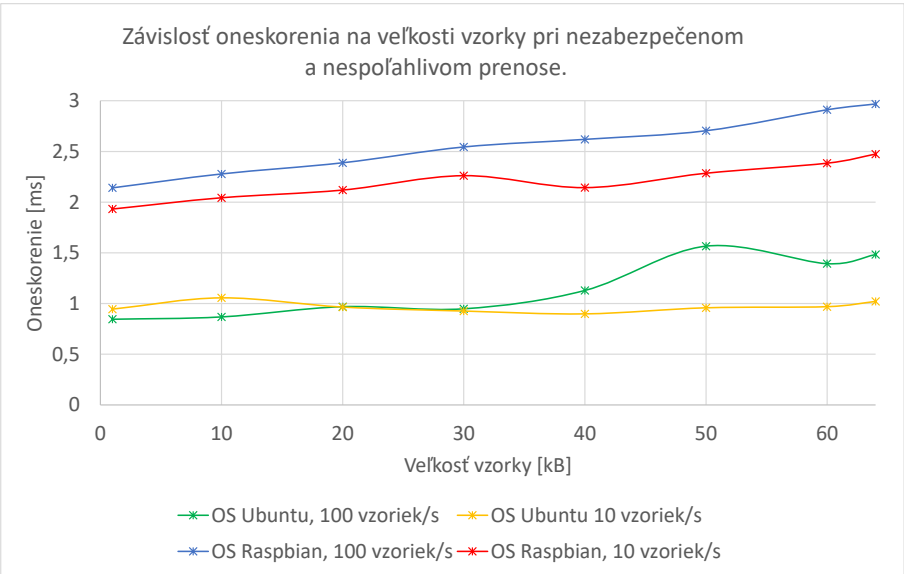
Meranie bez útoku:			Meranie s útokom:	
Veľkosť vzorky [kB]	Priemerné oneskorenie [ms]	Počet prijatých vzoriek	Priemerné oneskorenie [ms]	Počet prijatých vzoriek
1	2,213	5000	3,161	5000
10	3,882	5000	5,035	5000
20	6,331	5000	7,629	5000
30	97,3	5000	108,5	5000
40	101,9	5000	274,9	5000
50	102,8	5000	321,6	5000
60	311,4	5000	368,9	5000
64	437,8	5000	489,5	5000

C Grafy

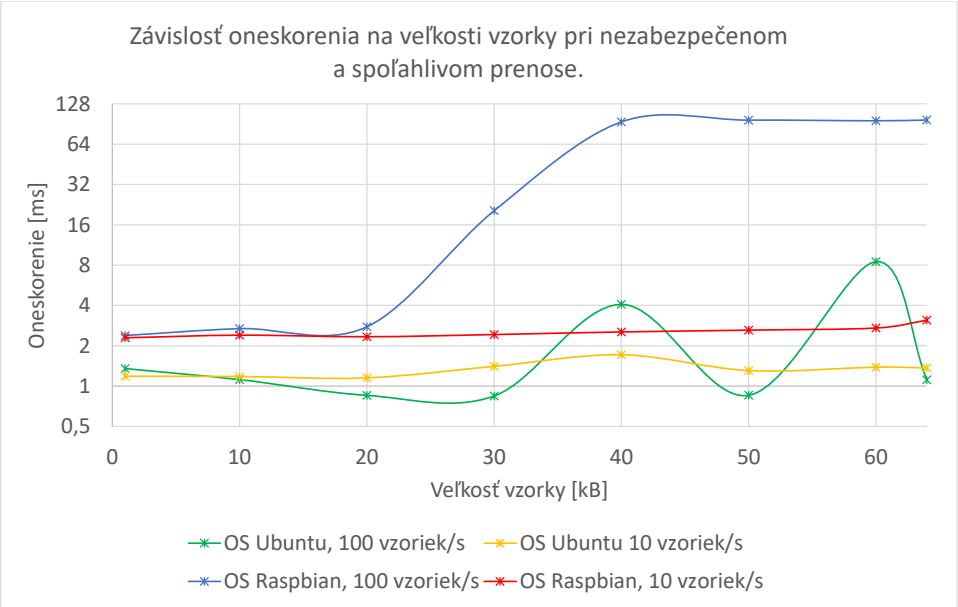
C.1 Grafy z meraní na 1 zariadení



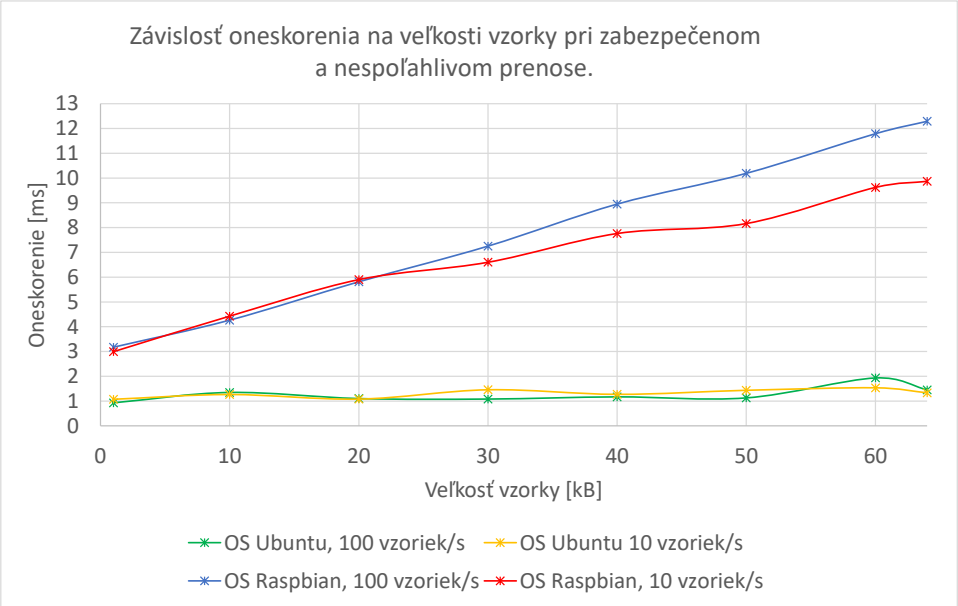
Obr. C.1: Závislosť oneskorenia na veľkosti vzorky nezabezpečeného a nespoľahlivého prenosu.



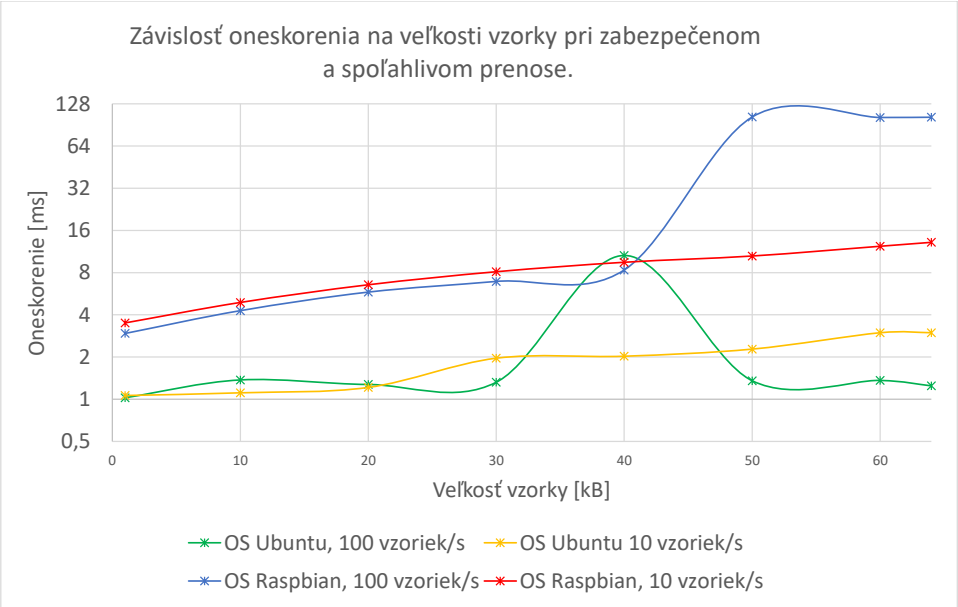
Obr. C.2: Závislosť oneskorenia na veľkosti vzorky nezabezpečeného a nespoľahlivého prenosu.



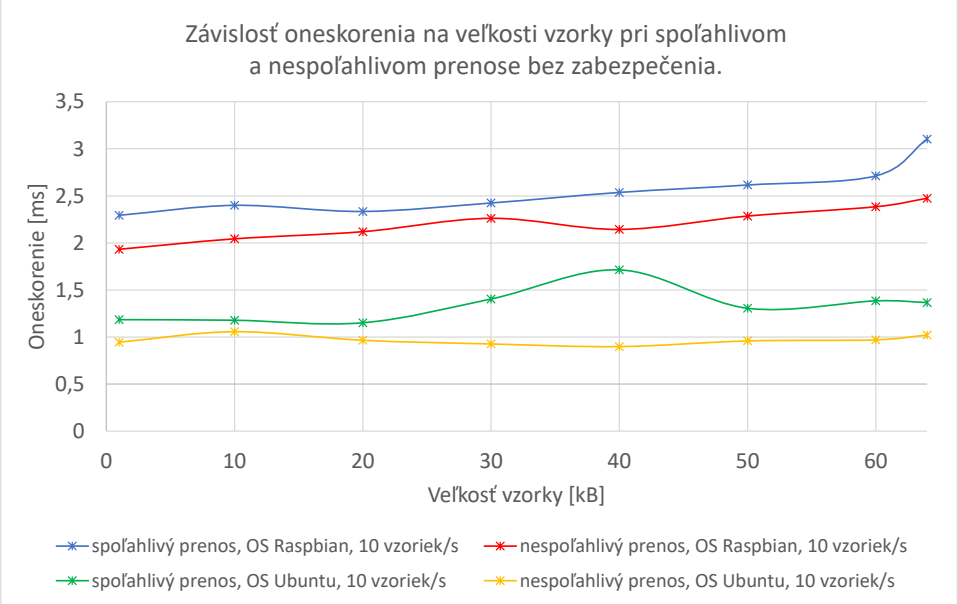
Obr. C.3: Závislosť oneskorenia na veľkosti vzorky pri nezabezpečenom a spoľahlivom prenose.



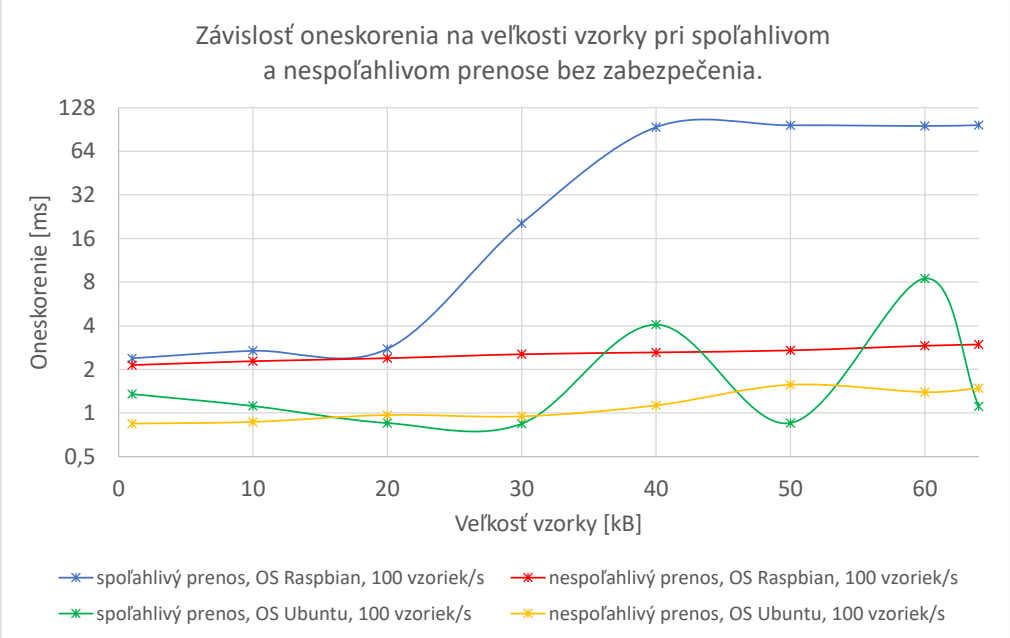
Obr. C.4: Závislosť oneskorenia na veľkosti vzorky pri zabezpečenom a nespoľahlivom prenose.



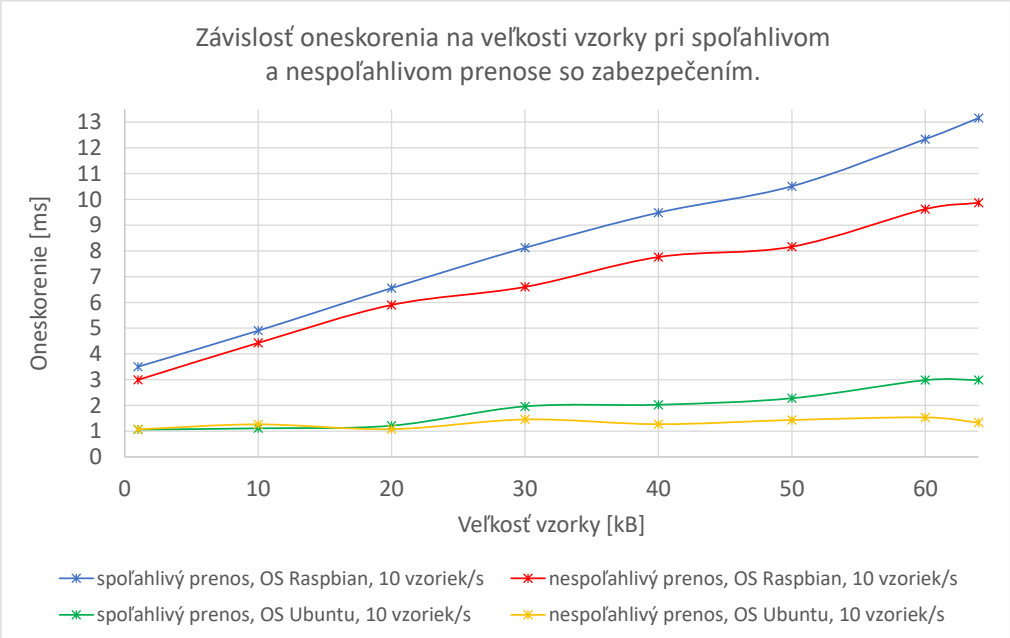
Obr. C.5: Závislosť oneskorenia na veľkosti vzorky pri zabezpečenom a spoľahlivom prenose.



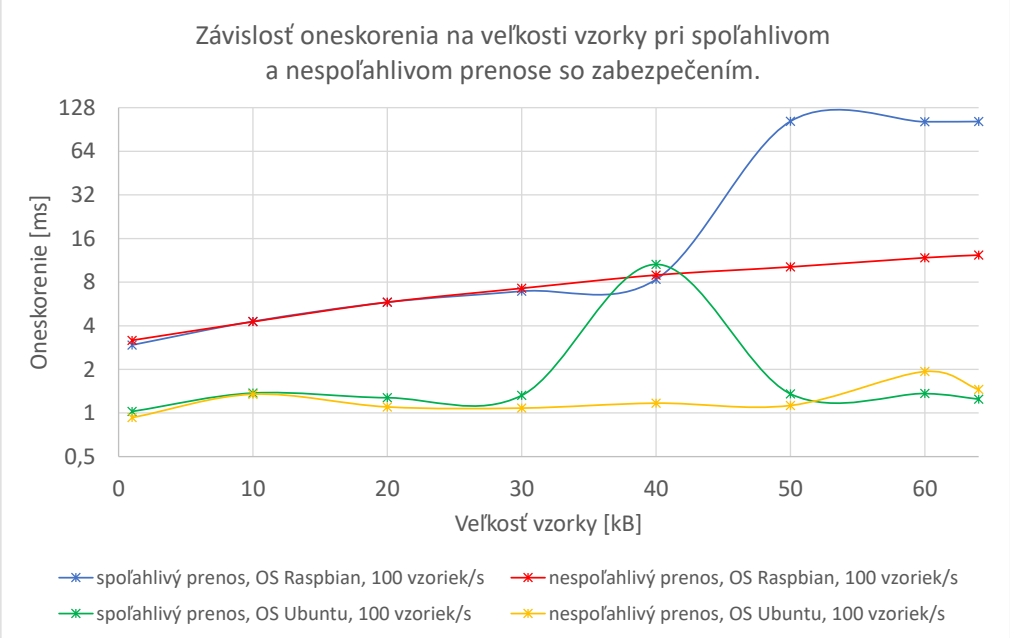
Obr. C.6: Závislosť oneskorenia na veľkosti vzorky pri spoľahlivom a nespoľahlivom prenose bez zabezpečenia.



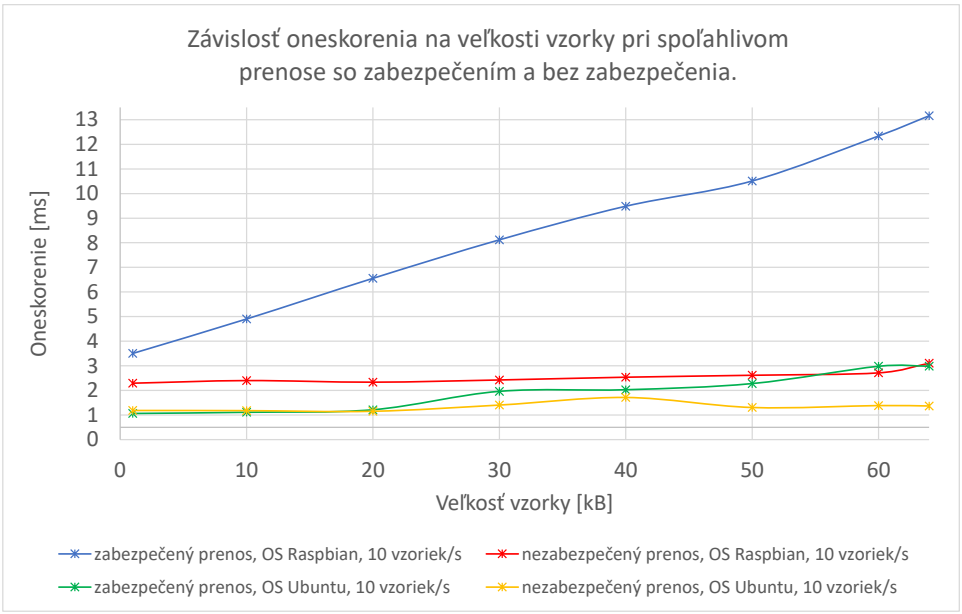
Obr. C.7: Závislosť oneskorenia na veľkosti vzorky pri spoľahlivom a nespoľahlivom prenose bez zabezpečenia.



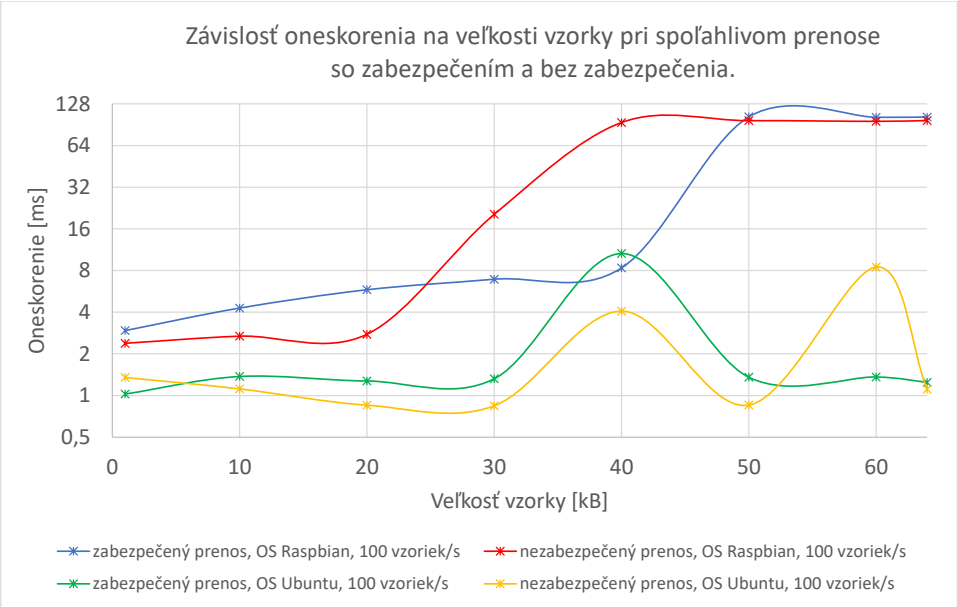
Obr. C.8: Závislosť oneskorenia na veľkosti vzorky pri spoľahlivom a nespoľahlivom prenose so zabezpečením.



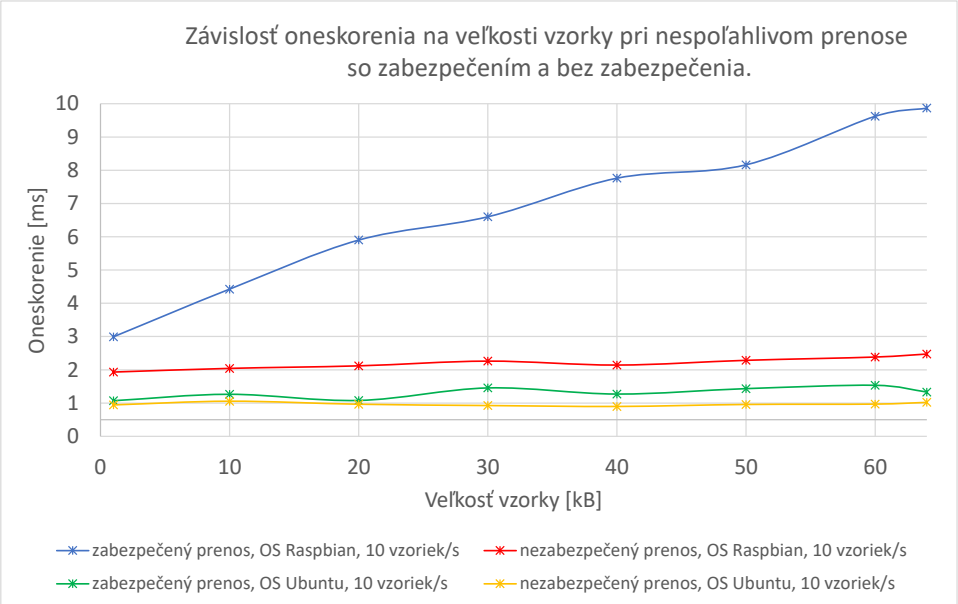
Obr. C.9: Závislosť oneskorenia na veľkosti vzorky pri spoľahlivom a nespoľahlivom prenose so zabezpečením.



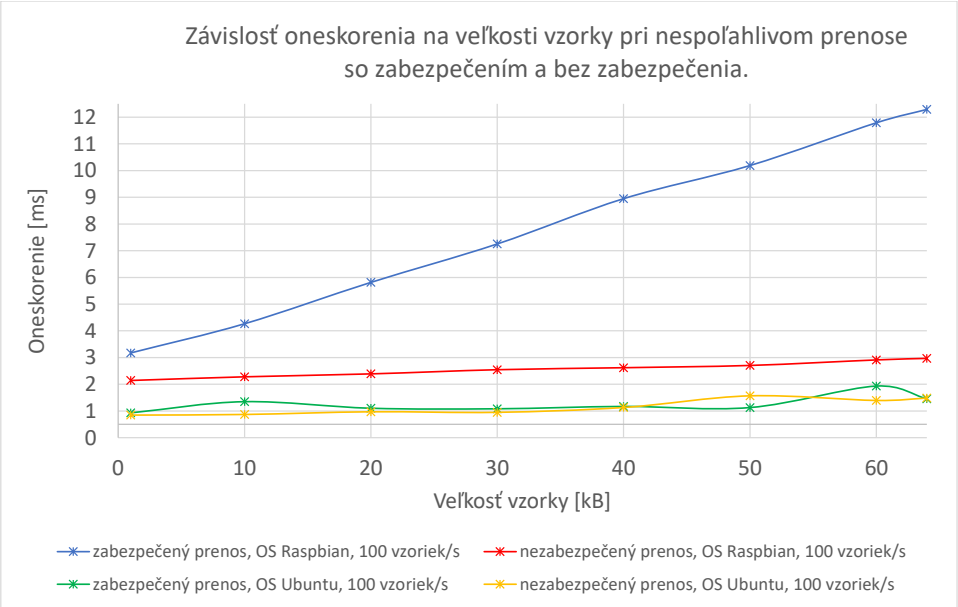
Obr. C.10: Závislosť oneskorenia na veľkosti vzorky pri spoľahlivom prenose so zabezpečením a bez zabezpečenia.



Obr. C.11: Závislosť oneskorenia na veľkosti vzorky pri spoľahlivom prenose so zabezpečením a bez zabezpečenia.

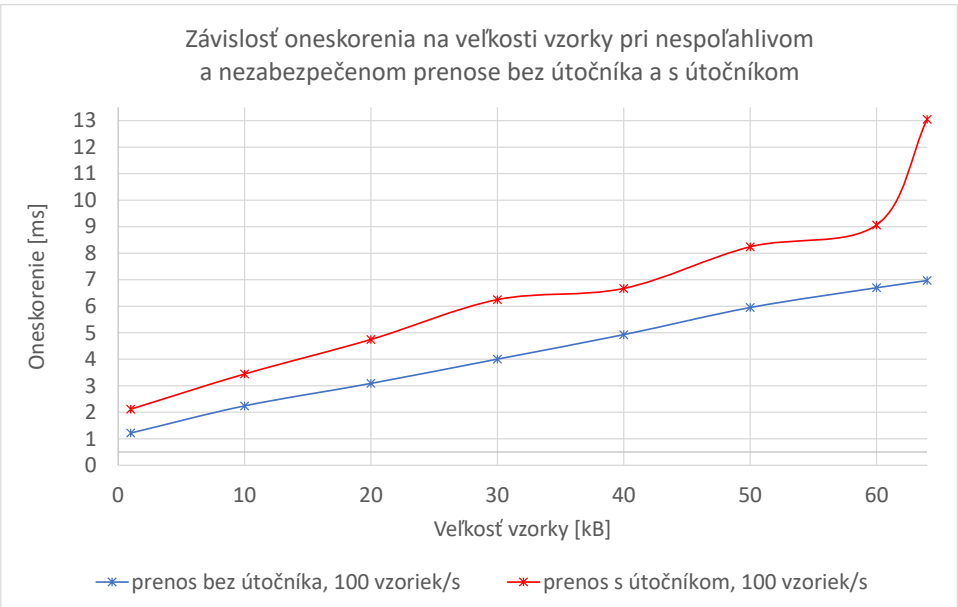


Obr. C.12: Závislosť oneskorenia na veľkosti vzorky pri nespoľahlivom prenose so zabezpečením a bez zabezpečenia.

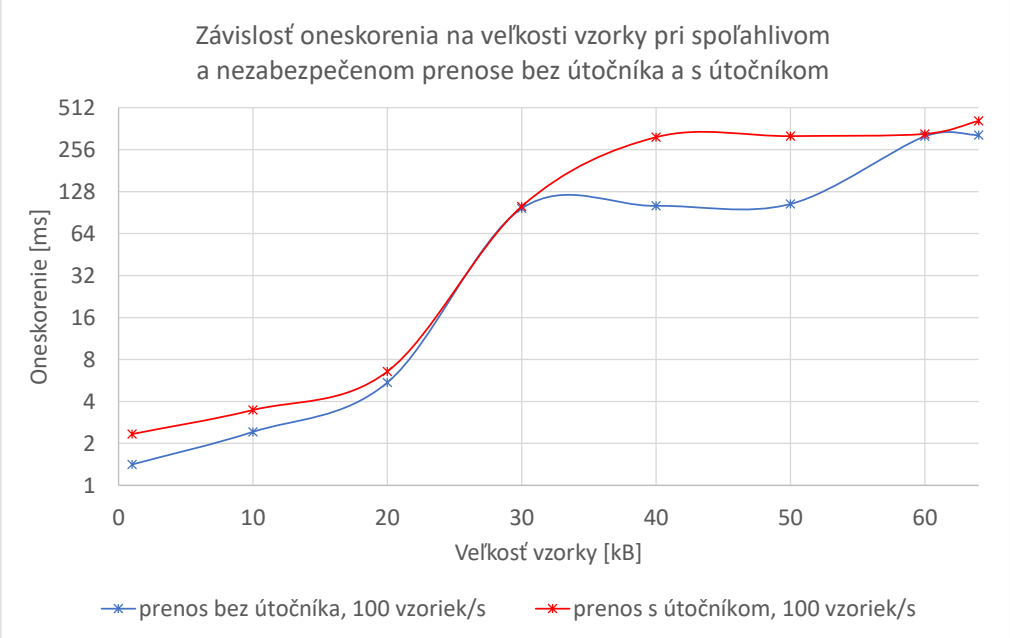


Obr. C.13: Závislosť oneskorenia na veľkosti vzorky pri nespoľahlivom prenose so zabezpečením a bez zabezpečenia.

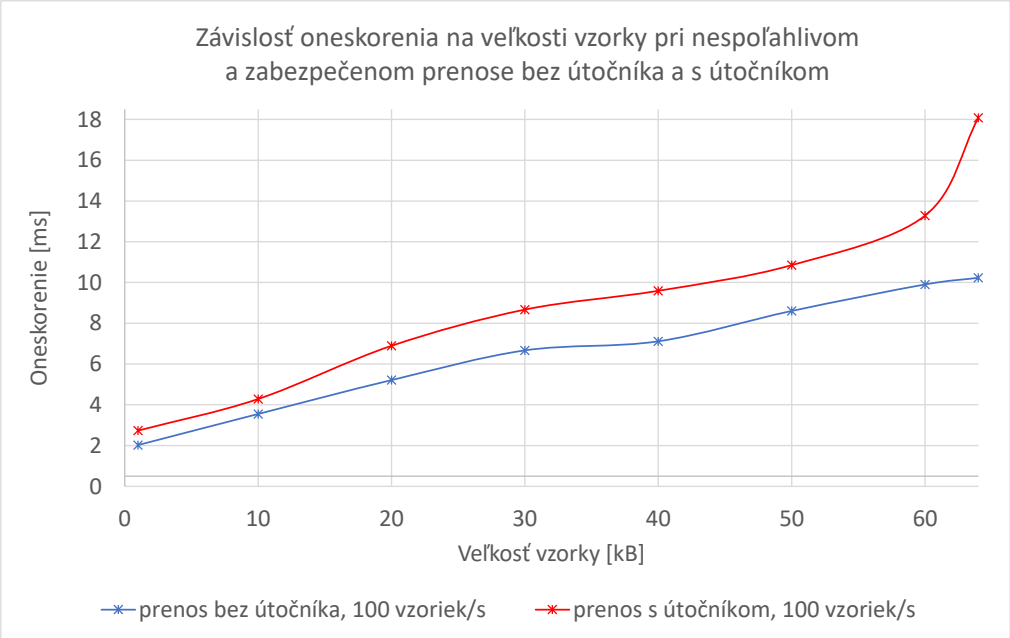
C.2 Grafy z meraní medzi 2 zariadeniami



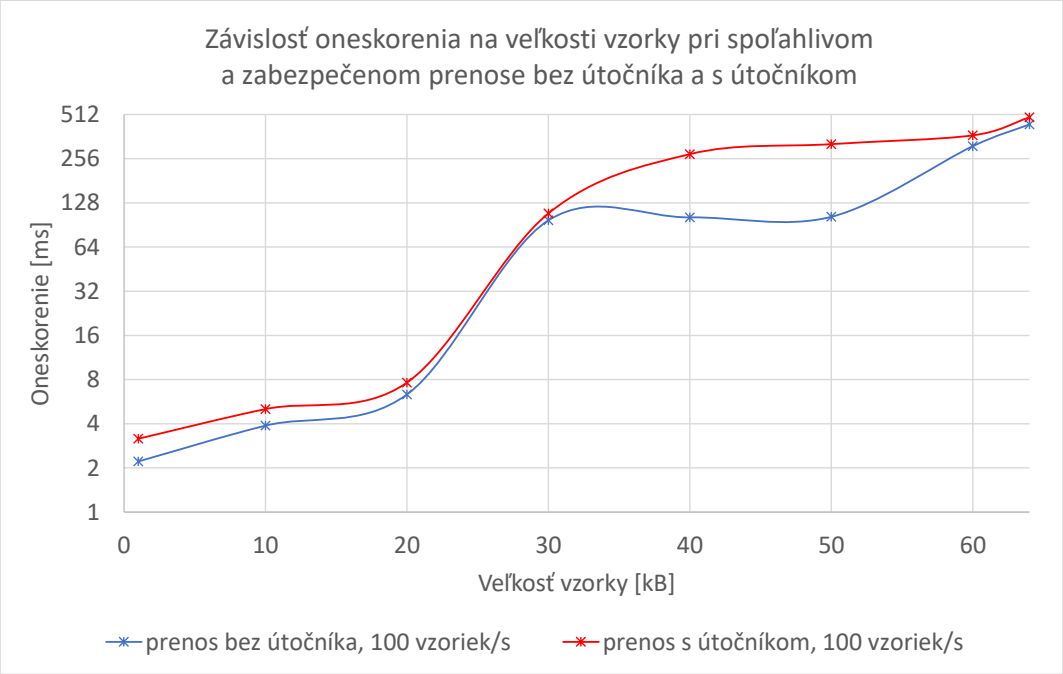
Obr. C.14: Závislosť oneskorenia na veľkosti vzorky pri nespoľahlivom a nezabezpečenom prenose bez útočníka a s útočníkom.



Obr. C.15: Závislosť oneskorenia na veľkosti vzorky pri spoľahlivom a nezabezpečenom prenose bez útočníka a s útočníkom.



Obr. C.16: Závislosť oneskorenia na veľkosti vzorky pri nespoľahlivom a zabezpečenom prenose bez útočníka a s útočníkom.



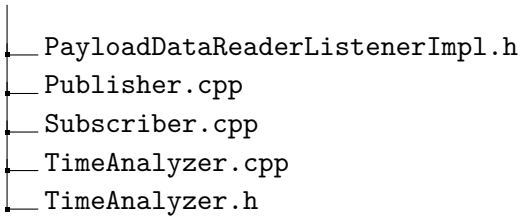
Obr. C.17: Závislosť oneskorenia na veľkosti vzorky pri spoľahlivom a zabezpečenom prenose bez útočníka a s útočníkom.

D Obsah priloženého CD

- Elektronická verzia práce vo formáte PDF.
- Adresár s certifikátmi, konfiguračnými súbormi, skriptom, výsledkami meraní a zdrojovými kódmy DDS systému vytvoreného pre meranie oneskorenia zabalený vo formáte ZIP.

Adresárová štruktúra priloženého CD:

```
/ ..... koreňový adresár priloženého CD
├── pdf ..... Elektronická verzia práce
└── prihloha_D_DDS_system.zip ..... Súbor DDS systému pre meranie oneskorenia
    ├── napoveda ..... Inštrukcie pre kompiláciu a spustenie DDS systému
    ├── certifikaty ... Certifikáty a podpísané Governance a Permissions dokumenty
    │   ├── ECDSA_certifikaty ..... ECDSA certifikáty a podpísané dokumenty
    │   │   ├── ca_cert.pem
    │   │   ├── ca_key.pem
    │   │   ├── cert_1.pem
    │   │   ├── cert_1_key.pem
    │   │   ├── cert_2.pem
    │   │   ├── cert_2_key.pem
    │   │   ├── governance_signed_SBMSG.p7s
    │   │   ├── permissions_1_signed.p7s
    │   │   └── permissions_2_signed.p7s
    │   └── RSA_certifikaty ..... RSA certifikáty a podpísané dokumenty
    │       ├── ca_cert.pem
    │       ├── ca_key.pem
    │       ├── cert_1.pem
    │       ├── cert_1_key.pem
    │       ├── cert_2.pem
    │       ├── cert_2_key.pem
    │       ├── governance_signed_SBMSG.p7s
    │       ├── permissions_1_signed.p7s
    │       └── permissions_2_signed.p7s
    ├── konfiguracne_subory ..... Konfiguračné súbory DDS systému
    │   ├── rtps_disc.ini
    │   └── rtps_disc_sec.ini
    ├── skripty ..... Skript použitý pri meraniach
    │   ├── start_simulations.pl
    │   └── start_simulations_win.pl
    ├── vysledky_merani ..... Tabuľky a grafy z meraní oneskorenia
    │   ├── vysledky_medzi_2_zariadeniami.xlsx
    │   └── vysledky_z_1_zariadenia.xlsx
    └── zdrojove_kody ..... Zdrojové kódy DDS systému pre meranie oneskorenia
        ├── Benchmark.idl
        ├── Benchmark.mpc
        └── PayloadDataReaderListenerImpl.cpp
```


PayloadDataReaderListenerImpl.h
Publisher.cpp
Subscriber.cpp
TimeAnalyzer.cpp
TimeAnalyzer.h